# 第1章 CSS 教程

CSS 是一种描述 HTML 文档样式的语言。

CSS 描述应该如何显示 HTML 元素。

本教程将从零起点的基础教程开始，一直到 CSS3 高级教程，为您提供全面系统地讲解。

每一章中的实例

本 CSS 教程包含数百个 CSS 实例。

通过使用我们的在线编辑器（W3Schoo TIY），您可以编辑 CSS，然后单击运行按钮来查看结果。

CSS 实例

<!DOCTYPE html>

<html>

<head>

<style>

body {

background-color: lightblue;

}

h1 {

color: white;

text-align: center;

}

p {

font-family: verdana;

font-size: 20px;

}

</style>

</head>

<body>

<h1>我的第一个 CSS 实例</h1>

<p>这是一个段落。</p>

</body>

</html>

# 第2章 CSS 简介

## 2.1 什么是 CSS？

CSS 指的是层叠样式表\* (Cascading Style Sheets)

CSS 描述了如何在屏幕、纸张或其他媒体上显示 HTML 元素

CSS 节省了大量工作。它可以同时控制多张网页的布局

外部样式表存储在 CSS 文件中

\*：也称级联样式表。

## 2.2 为何使用 CSS？

CSS 用于定义网页的样式，包括针对不同设备和屏幕尺寸的设计和布局。

body {

background-color: lightblue;

}

h1 {

color: white;

text-align: center;

}

p {

font-family: verdana;

font-size: 20px;

}

## 2.3 CSS 解决了一个大问题

HTML 从未打算包含用于格式化网页的标签！

创建 HTML 的目的是描述网页的内容，例如：

<h1>这是一个标题。</h1>

<p>这是一个段落。</p>

将 <font> 之类的标签和 color 属性添加到 HTML 3.2 规范后，Web 开发人员的噩梦开始了。大型网站的开发将字体和颜色信息添加到每个页面中，这演变为一个漫长而昂贵的过程。

为了解决这个问题，万维网联盟（W3C）创建了 CSS。

CSS 从 HTML 页面中删除了样式格式！

如果您不知道 HTML 是什么，建议您阅读 HTML 教程。

## 2.4 CSS 节省了大量工作！

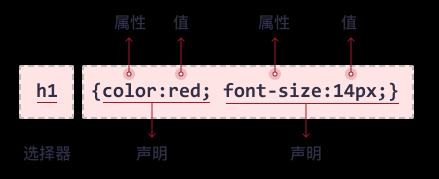
样式定义通常保存在外部 .css 文件中。

通过使用外部样式表文件，您只需更改一个文件即可更改整个网站的外观！

# 第3章 CSS 语法

## 3.1 CSS 语法

CSS 规则集（rule-set）由选择器和声明块组成：



选择器指向您需要设置样式的 HTML 元素。

声明块包含一条或多条用分号分隔的声明。

每条声明都包含一个 CSS 属性名称和一个值，以冒号分隔。

多条 CSS 声明用分号分隔，声明块用花括号括起来。

实例

在此例中，所有 <p> 元素都将居中对齐，并带有红色文本颜色：

p {

color: red;

text-align: center;

}

## 3.2 例子解释

p 是 CSS 中的选择器（它指向要设置样式的 HTML 元素：<p>）。

color 是属性，red 是属性值

text-align 是属性，center 是属性值

在下一章中，您将学到更多关于 CSS 选择器和 CSS 属性的知识。

# 第4章 CSS 选择器

CSS 选择器用于“查找”（或选取）要设置样式的 HTML 元素。

我们可以将 CSS 选择器分为五类：

1.简单选择器（根据名称、id、类来选取元素）

2.组合器选择器（根据它们之间的特定关系来选取元素）

3.伪类选择器（根据特定状态选取元素）

4.伪元素选择器（选取元素的一部分并设置其样式）

5.属性选择器（根据属性或属性值来选取元素）

此页会讲解最基本的 CSS 选择器。

## 4.1 CSS 元素选择器

元素选择器根据元素名称来选择 HTML 元素。

实例

在这里，页面上的所有 <p> 元素都将居中对齐，并带有红色文本颜色：

p {

text-align: center;

color: red;

}

## 4.2 CSS id 选择器

id 选择器使用 HTML 元素的 id 属性来选择特定元素。

元素的 id 在页面中是唯一的，因此 id 选择器用于选择一个唯一的元素！

要选择具有特定 id 的元素，请写一个井号（＃），后跟该元素的 id。

实例

这条 CSS 规则将应用于 id="para1" 的 HTML 元素：

#para1 {

text-align: center;

color: red;

}

注意：id 名称不能以数字开头。

## 4.3 CSS 类选择器

类选择器选择有特定 class 属性的 HTML 元素。

如需选择拥有特定 class 的元素，请写一个句点（.）字符，后面跟类名。

实例

在此例中，所有带有 class="center" 的 HTML 元素将为红色且居中对齐：

.center {

text-align: center;

color: red;

}

您还可以指定只有特定的 HTML 元素会受类的影响。

实例

在这个例子中，只有具有 class="center" 的 <p> 元素会居中对齐：

p.center {

text-align: center;

color: red;

}

HTML 元素也可以引用多个类。

实例

在这个例子中，<p> 元素将根据 class="center" 和 class="large" 进行样式设置：

<p class="center large">这个段落引用两个类。</p>

注意：类名不能以数字开头！

## 4.4 CSS 通用选择器

通用选择器（\*）选择页面上的所有的 HTML 元素。

实例

下面的 CSS 规则会影响页面上的每个 HTML 元素：

\* {

text-align: center;

color: blue;

}

## 4.5 CSS 分组选择器

分组选择器选取所有具有相同样式定义的 HTML 元素。

请看下面的 CSS 代码（h1、h2 和 p 元素具有相同的样式定义）：

h1 {

text-align: center;

color: red;

}

h2 {

text-align: center;

color: red;

}

p {

text-align: center;

color: red;

}

最好对选择器进行分组，以最大程度地缩减代码。

如需对选择器进行分组，请用逗号来分隔每个选择器。

实例

在这个例子中，我们对上述代码中的选择器进行分组：

h1, h2, p {

text-align: center;

color: red;

}

## 4.6 所有简单的 CSS 选择器

|  |  |  |
| --- | --- | --- |
| 选择器 | 例子 | 例子描述 |
| .class | .intro | 选取所有 class="intro" 的元素。 |
| #id | #firstname | 选取 id="firstname" 的那个元素。 |
| \* | \* | 选取所有元素。 |
| element | p | 选取所有 <p> 元素。 |
| element,element,.. | div, p | 选取所有 <div> 元素和所有 <p> 元素。 |

# 第5章 如何添加 CSS

当浏览器读到样式表时，它将根据样式表中的信息来格式化 HTML 文档。

## 5.1 三种使用 CSS 的方法

有三种插入样式表的方法：

外部 CSS

内部 CSS

行内 CSS

## 5.2 外部 CSS

通过使用外部样式表，您只需修改一个文件即可改变整个网站的外观！

每张 HTML 页面必须在 head 部分的 <link> 元素内包含对外部样式表文件的引用。

实例

外部样式在 HTML 页面 <head> 部分内的 <link> 元素中进行定义：

<!DOCTYPE html>

<html>

<head>

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

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

外部样式表可以在任何文本编辑器中编写，并且必须以 .css 扩展名保存。

外部 .css 文件不应包含任何 HTML 标签。

"mystyle.css" 是这样的：

"mystyle.css"

body {

background-color: lightblue;

}

h1 {

color: navy;

margin-left: 20px;

}

注意：请勿在属性值和单位之间添加空格（例如 margin-left: 20 px;）。正确的写法是：margin-left: 20px;

## 5.3 内部 CSS

如果一张 HTML 页面拥有唯一的样式，那么可以使用内部样式表。

内部样式是在 head 部分的 <style> 元素中进行定义。

实例

内部样式在 HTML 页面的 <head> 部分内的 <style> 元素中进行定义：

<!DOCTYPE html>

<html>

<head>

<style>

body {

background-color: linen;

}

h1 {

color: maroon;

margin-left: 40px;

}

</style>

</head>

<body>

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

## 5.4 行内 CSS

行内样式（也称内联样式）可用于为单个元素应用唯一的样式。

如需使用行内样式，请将 style 属性添加到相关元素。style 属性可包含任何 CSS 属性。

实例

行内样式在相关元素的 "style" 属性中定义：

<!DOCTYPE html>

<html>

<body>

<h1 style="color:blue;text-align:center;">This is a heading</h1>

<p style="color:red;">This is a paragraph.</p>

</body>

</html>

提示：行内样式失去了样式表的许多优点（通过将内容与呈现混合在一起）。请谨慎使用此方法。

## 5.5 多个样式表

如果在不同样式表中为同一选择器（元素）定义了一些属性，则将使用最后读取的样式表中的值。

假设某个外部样式表为 <h1> 元素设定的如下样式：

h1 {

color: navy;

}

然后，假设某个内部样式表也为 <h1> 元素设置了如下样式：

h1 {

color: orange;

}

实例

如果内部样式是在链接到外部样式表之后定义的，则 <h1> 元素将是橙色：

<head>

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

<style>

h1 {

color: orange;

}

</style>

</head>

实例

不过，如果在链接到外部样式表之前定义了内部样式，则 <h1> 元素将是深蓝色：

<head>

<style>

h1 {

color: orange;

}

</style>

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

</head>

## 5.6 层叠顺序

当为某个 HTML 元素指定了多个样式时，会使用哪种样式呢？

页面中的所有样式将按照以下规则“层叠”为新的“虚拟”样式表，其中第一优先级最高：

行内样式（在 HTML 元素中）

外部和内部样式表（在 head 部分）

浏览器默认样式

因此，行内样式具有最高优先级，并且将覆盖外部和内部样式以及浏览器默认样式。

# 第6章 CSS 注释

## 6.1 CSS 注释

注释用于解释代码，以后在您编辑源代码时可能会有所帮助。

浏览器会忽略注释。

位于 <style> 元素内的 CSS 注释，以 /\* 开始，以 \*/ 结束：

实例

/\* 这是一条单行注释 \*/

p {

color: red;

}

您可以在代码中的任何位置添加注释：

实例

p {

color: red; /\* 把文本设置为红色 \*/

}

注释能横跨多行：

实例

/\* 这是

一条多行的

注释 \*/

p {

color: red;

}

## 6.2 HTML 和 CSS 注释

从 HTML 教程中，您学习到可以使用 <!--...--> 语法在 HTML 源代码中添加注释。

在下面的例子中，我们结合使用了 HTML 和 CSS 注释：

实例

<!DOCTYPE html>

<html>

<head>

<style>

p {

color: red; /\* 将文字颜色设置为红色 \*/

}

</style>

</head>

<body>

<h2>My Heading</h2>

<!-- 这些段落将是红色的 -->

<p>Hello World!</p>

<p>这段文本由 CSS 设置样式。</p>

<p>CSS 注释不会在输出中显示。</p>

</body>

</html>

# 第7章 CSS 颜色

指定颜色是通过使用预定义的颜色名称，或 RGB、HEX、HSL、RGBA、HSLA 值。

## 7.1 CSS 背景色

您可以为 HTML 元素设置背景色：

实例

<h1 style="background-color:DodgerBlue;">China</h1>

<p style="background-color:Tomato;">China is a great country!</p>

## 7.2 CSS 文本颜色

您可以设置文本的颜色：

China

China is a great country!

China, officially the People's Republic of China, is a country in East Asia.

实例

<h1 style="color:Tomato;">China</h1>

<p style="color:DodgerBlue;">China is a great country!</p>

<p style="color:MediumSeaGreen;">China, officially the People's Republic of China...</p>

## 7.3 CSS 边框颜色

您可以设置边框的颜色：

实例

<h1 style="border:2px solid Tomato;">Hello World</h1>

<h1 style="border:2px solid DodgerBlue;">Hello World</h1>

<h1 style="border:2px solid Violet;">Hello World</h1>

## 7.4 CSS 颜色值

在 CSS 中，还可以使用 RGB 值、HEX 值、HSL 值、RGBA 值或者 HSLA 值来指定颜色：

与颜色名 "Tomato" 等效：

<h3>实例</h3>

<h1 style="background-color:rgb(255, 99, 71);">...</h1>

<h1 style="background-color:#ff6347;">...</h1>

<h1 style="background-color:hsl(9, 100%, 64%);">...</h1>

<h1 style="background-color:rgba(255, 99, 71, 0.5);">...</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.5);">...</h1>

# 第8章 CSS RGB 颜色

## 8.1 RGB 值

在 CSS 中，可以使用下面的公式将颜色指定为 RGB 值：

rgb(red, green, blue)

每个参数 (red、green 以及 blue) 定义了 0 到 255 之间的颜色强度。

例如，rgb(255, 0, 0) 显示为红色，因为红色设置为最大值（255），其他设置为 0。

要显示黑色，请将所有颜色参数设置为 0，如下所示：rgb(0, 0, 0)。

要显示白色，请将所有颜色参数设置为 255，如下所示：rgb(255, 255, 255)。

请通过混合以下 RGB 值来进行实验：

## 8.2 RGBA 值

RGBA 颜色值是具有 alpha 通道的 RGB 颜色值的扩展 - 它指定了颜色的不透明度。

RGBA 颜色值指定为：

rgba(red, green, blue, alpha)

alpha 参数是介于 0.0（完全透明）和 1.0（完全不透明）之间的数字：

请通过混合以下 RGBA 值来进行实验：

# 第9章 HEX 值

## 9.1 HEX 值

在 CSS 中，可以使用以下格式的十六进制值指定颜色：

#rrggbb

其中 rr（红色）、gg（绿色）和 bb（蓝色）是介于 00 和 ff 之间的十六进制值（与十进制 0-255 相同）。

例如，#ff0000 显示为红色，因为红色设置为最大值（ff），其他设置为最小值（00）。

请通过混合以下十六进制值来进行实验：

# 第10章 CSS HSL 颜色

## 10.1 HSL 值

在 CSS 中，可以使用色相、饱和度和明度（HSL）来指定颜色，格式如下：

hsla(hue, saturation, lightness)

色相（hue）是色轮上从 0 到 360 的度数。0 是红色，120 是绿色，240 是蓝色。

饱和度（saturation）是一个百分比值，0％ 表示灰色阴影，而 100％ 是全色。

亮度（lightness）也是百分比，0％ 是黑色，50％ 是既不明也不暗，100％是白色。

请通过混合以下 HSL 值来进行实验：

## 10.2 饱和度

饱和度可以描述为颜色的强度。

100％ 是纯色，没有灰色阴影

50％ 是 50％ 灰色，但是您仍然可以看到颜色。

0％ 是完全灰色，您无法再看到颜色。

## 10.3 亮度

颜色的明暗度可以描述为要赋予颜色多少光，其中 0％ 表示不亮（黑色），50％ 表示 50％ 亮（既不暗也不亮），100％ 表示全明（白）。

实例

<h1 style="background-color:hsl(0, 100%, 0%);">hsl(0, 100%, 0%)</h1>

<h1 style="background-color:hsl(0, 100%, 25%);">hsl(0, 100%, 25%)</h1>

<h1 style="background-color:hsl(0, 100%, 50%);">hsl(0, 100%, 50%)</h1>

<h1 style="background-color:hsl(0, 100%, 75%);">hsl(0, 100%, 75%)</h1>

<h1 style="background-color:hsl(0, 100%, 90%);">hsl(0, 100%, 90%)</h1>

<h1 style="background-color:hsl(0, 100%, 100%);">hsl(0, 100%, 100%)</h1>

<p>通过 HSL 颜色，亮度 0% 显示黑色，亮度 100 显示白色。</p>

</body>

</html>

## 10.4 HSLA 值

HSLA 颜色值是带有 Alpha 通道的 HSL 颜色值的扩展 - 它指定了颜色的不透明度。

HSLA 颜色值指定为：

hsla(hue, saturation, lightness, alpha)

alpha 参数是介于 0.0（完全透明）和 1.0（完全不透明）之间的数字：

请通过混合以下 HSLA 值进行实验：

<!DOCTYPE html>

<html>

<body>

<h1 style="background-color:hsla(9, 100%, 64%, 0);">hsla(9, 100%, 64%, 0)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.2);">hsla(9, 100%, 64%, 0.2)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.4);">hsla(9, 100%, 64%, 0.4)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.6);">hsla(9, 100%, 64%, 0.6)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 0.8);">hsla(9, 100%, 64%, 0.8)</h1>

<h1 style="background-color:hsla(9, 100%, 64%, 1);">hsla(9, 100%, 64%, 1)</h1>

<p>您可使用 HSLA 颜色值生成透明色。</p>

</body>

</html>

# 第11章 CSS 背景

CSS 背景属性用于定义元素的背景效果。

在这些章节中，您将学习如下 CSS 背景属性：

background-color

background-image

background-repeat

background-attachment

background-position

## 11.1 CSS background-color

background-color 属性指定元素的背景色。

实例

页面的背景色设置如下：

body {

background-color: lightblue;

}

通过 CSS，颜色通常由以下方式指定：

有效的颜色名称 - 比如 "red"

十六进制值 - 比如 "#ff0000"

RGB 值 - 比如 "rgb(255,0,0)"

## 11.2 其他元素

您可以为任何 HTML 元素设置背景颜色：

实例

在这里，<h1>、<p> 和 <div> 元素将拥有不同的背景色：

h1 {

background-color: green;

}

div {

background-color: lightblue;

}

p {

background-color: yellow;

}

## 11.3 不透明度 / 透明度

opacity 属性指定元素的不透明度/透明度。取值范围为 0.0 - 1.0。值越低，越透明：

div {

background-color: green;

opacity: 0.3;

}

注意：使用 opacity 属性为元素的背景添加透明度时，其所有子元素都继承相同的透明度。这可能会使完全透明的元素内的文本难以阅读。

## 11.4 使用 RGBA 的透明度

如果您不希望对子元素应用不透明度，例如上面的例子，请使用 RGBA 颜色值。下面的例子设置背景色而不是文本的不透明度：

您从我们的 CSS 颜色 章节中学到了可以将 RGB 用作颜色值。除 RGB 外，还可以将 RGB 颜色值与 alpha 通道一起使用（RGBA） - 该通道指定颜色的不透明度。

RGBA 颜色值指定为：rgba(red, green, blue, alpha)。alpha 参数是介于 0.0（完全透明）和 1.0（完全不透明）之间的数字。

提示：您可在我们的 CSS 颜色 一章中学到有关 RGBA 颜色的更多知识。

实例

div {

background: rgba(0, 128, 0, 0.3) /\* 30% 不透明度的绿色背景 \*/

}

# 第12章 CSS 背景图像

## 12.1 CSS 背景图像

background-image 属性指定用作元素背景的图像。

默认情况下，图像会重复，以覆盖整个元素。

实例

页面的背景图像可以像这样设置：

body {

background-image: url("paper.gif");

}

实例

本例展示了文本和背景图像的错误组合。文字难以阅读：

body {

background-image: url("bgdesert.jpg");

}

注意：使用背景图像时，请使用不会干扰文本的图像。

还可以为特定元素设置背景图像，例如 <p> 元素：

实例

p {

background-image: url("paper.gif");

}

# 第13章 CSS 背景重复

## 13.1 CSS background-repeat

默认情况下，background-image 属性在水平和垂直方向上都重复图像。

某些图像应只适合水平或垂直方向上重复，否则它们看起来会很奇怪，如下所示：

实例

body {

background-image: url("gradient\_bg.png");

}

如果上面的图像仅在水平方向重复 (background-repeat: repeat-x;)，则背景看起来会更好：

实例

body {

background-image: url("gradient\_bg.png");

background-repeat: repeat-x;

}

提示：如需垂直重复图像，请设置 background-repeat: repeat-y;。

## 13.2 CSS background-repeat: no-repeat

background-repeat 属性还可指定只显示一次背景图像：

实例

背景图像仅显示一次：

body {

background-image: url("tree.png");

background-repeat: no-repeat;

}

在上例中，背景图像与文本放置在同一位置。我们想要更改图像的位置，以免图像过多干扰文本。

## 13.3 CSS background-position

background-position 属性用于指定背景图像的位置。

实例

把背景图片放在右上角：

body {

background-image: url("tree.png");

background-repeat: no-repeat;

background-position: right top;

}

# 第14章 CSS 背景附着

## 14.1 CSS background-attachment

background-attachment 属性指定背景图像是应该滚动还是固定的（不会随页面的其余部分一起滚动）：

实例

指定应该固定背景图像：

body {

background-image: url("tree.png");

background-repeat: no-repeat;

background-position: right top;

background-attachment: fixed;

}

实例

指定背景图像应随页面的其余部分一起滚动：

body {

background-image: url("tree.png");

background-repeat: no-repeat;

background-position: right top;

background-attachment: scroll;

}

# 第15章 CSS 背景简写

## 15.1 CSS background - 简写属性

如需缩短代码，也可以在一个属性中指定所有背景属性。它被称为简写属性。

而不是这样写：

body {

background-color: #ffffff;

background-image: url("tree.png");

background-repeat: no-repeat;

background-position: right top;

}

您能够使用简写属性 background：

实例

使用简写属性在一条声明中设置背景属性：

body {

background: #ffffff url("tree.png") no-repeat right top;

}

在使用简写属性时，属性值的顺序为：

background-color

background-image

background-repeat

background-attachment

background-position

属性值之一缺失并不要紧，只要按照此顺序设置其他值即可。请注意，在上面的例子中，我们没有使用 background-attachment 属性，因为它没有值。

## 15.2 所有 CSS 背景属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| background | 在一条声明中设置所有背景属性的简写属性。 |
| background-attachment | 设置背景图像是固定的还是与页面的其余部分一起滚动。 |
| background-clip | 规定背景的绘制区域。 |
| background-color | 设置元素的背景色。 |
| background-image | 设置元素的背景图像。 |
| background-origin | 规定在何处放置背景图像。 |
| background-position | 设置背景图像的开始位置。 |
| background-repeat | 设置背景图像是否及如何重复。 |
| background-size | 规定背景图像的尺寸。 |

# 第16章 CSS 边框

## 16.1 CSS 边框属性

CSS border 属性允许您指定元素边框的样式、宽度和颜色。

我的所有边都有边框。

我有一条红色的下边框。

我有圆角边框。

我有一条蓝色的左边框。

## 16.2 CSS 边框样式

border-style 属性指定要显示的边框类型。

允许以下值：

dotted - 定义点线边框

dashed - 定义虚线边框

solid - 定义实线边框

double - 定义双边框

groove - 定义 3D 坡口边框。效果取决于 border-color 值

ridge - 定义 3D 脊线边框。效果取决于 border-color 值

inset - 定义 3D inset 边框。效果取决于 border-color 值

outset - 定义 3D outset 边框。效果取决于 border-color 值

none - 定义无边框

hidden - 定义隐藏边框

border-style 属性可以设置一到四个值（用于上边框、右边框、下边框和左边框）。

实例

演示不同的边框样式：

p.dotted {border-style: dotted;}

p.dashed {border-style: dashed;}

p.solid {border-style: solid;}

p.double {border-style: double;}

p.groove {border-style: groove;}

p.ridge {border-style: ridge;}

p.inset {border-style: inset;}

p.outset {border-style: outset;}

p.none {border-style: none;}

p.hidden {border-style: hidden;}

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

注意：除非设置了 border-style 属性，否则其他 CSS 边框属性（将在下一章中详细讲解）都不会有任何作用！

# 第17章 CSS 边框宽度

## 17.1 CSS 边框宽度

border-width 属性指定四个边框的宽度。

可以将宽度设置为特定大小（以 px、pt、cm、em 计），也可以使用以下三个预定义值之一：thin、medium 或 thick：

实例

演示不同的边框宽度：

p.one {

border-style: solid;

border-width: 5px;

}

p.two {

border-style: solid;

border-width: medium;

}

p.three {

border-style: dotted;

border-width: 2px;

}

p.four {

border-style: dotted;

border-width: thick;

}

## 17.2 特定边的宽度

border-width 属性可以设置一到四个值（用于上边框、右边框、下边框和左边框）：

实例

p.one {

border-style: solid;

border-width: 5px 20px; /\* 上边框和下边框为 5px，其他边为 20px \*/

}

p.two {

border-style: solid;

border-width: 20px 5px; /\* 上边框和下边框为 20px，其他边为 5px \*/

}

p.three {

border-style: solid;

border-width: 25px 10px 4px 35px; /\* 上边框 25px，右边框 10px，下边框 4px，左边框 35px \*/

}

# 第18章 CSS 边框颜色

## 18.1 CSS 边框颜色

border-color 属性用于设置四个边框的颜色。

可以通过以下方式设置颜色：

name - 指定颜色名，比如 "red"

HEX - 指定十六进制值，比如 "#ff0000"

RGB - 指定 RGB 值，比如 "rgb(255,0,0)"

HSL - 指定 HSL 值，比如 "hsl(0, 100%, 50%)"

transparent

注释：如果未设置 border-color，则它将继承元素的颜色。

实例

演示不同的边框颜色：

p.one {

border-style: solid;

border-color: red;

}

p.two {

border-style: solid;

border-color: green;

}

p.three {

border-style: dotted;

border-color: blue;

}

## 18.2 特定边框的颜色

border-color 属性可以设置一到四个值（用于上边框、右边框、下边框和左边框）。

实例

p.one {

border-style: solid;

border-color: red green blue yellow; /\* 上红、右绿、下蓝、左黄 \*/

}

## 18.3 HEX 值

边框的颜色也可以使用十六进制值（HEX）来指定：

实例

p.one {

border-style: solid;

border-color: #ff0000; /\* 红色 \*/

}

## 18.4 RGB 值

或者使用 RGB 值：

实例

p.one {

border-style: solid;

border-color: rgb(255, 0, 0); /\* 红色 \*/

}

## 18.5 HSL 值

也可以使用 HSL 值：

实例

p.one {

border-style: solid;

border-color: hsl(0, 100%, 50%); /\* 红色 \*/

}

# 第19章 CSS 边框各边

## 19.1 CSS 边框 - 单独的边

从上一章的例子中，您已经看到可以为每一侧指定不同的边框。

在 CSS 中，还有一些属性可用于指定每个边框（顶部、右侧、底部和左侧）：

实例

p {

border-top-style: dotted;

border-right-style: solid;

border-bottom-style: dotted;

border-left-style: solid;

}

## 19.2 不同的边框样式

上例的结果与此相同：

实例

p {

border-style: dotted solid;

}

它的工作原理是这样的：

如果 border-style 属性设置四个值：

border-style: dotted solid double dashed;

上边框是虚线

右边框是实线

下边框是双线

左边框是虚线

如果 border-style 属性设置三个值：

border-style: dotted solid double;

上边框是虚线

右和左边框是实线

下边框是双线

如果 border-style 属性设置两个值：

border-style: dotted solid;

上和下边框是虚线

右和左边框是实线

如果 border-style 属性设置一个值：

border-style: dotted;

四条边均为虚线

实例

/\* 四个值 \*/

p {

border-style: dotted solid double dashed;

}

/\* 三个值 \*/

p {

border-style: dotted solid double;

}

/\* 两个值 \*/

p {

border-style: dotted solid;

}

/\* 一个值 \*/

p {

border-style: dotted;

}

上例中使用的是 border-style 属性。但 border-width 和 border-color 也同样适用。

# 第20章 CSS 简写边框属性

## 20.1 CSS Border - 简写属性

就像您在上一章中所见，处理边框时要考虑许多属性。

为了缩减代码，也可以在一个属性中指定所有单独的边框属性。

border 属性是以下各个边框属性的简写属性：

border-width

border-style（必需）

border-color

实例

p {

border: 5px solid red;

}

您还可以只为一个边指定所有单个边框属性：

左边框

p {

border-left: 6px solid red;

background-color: lightgrey;

}

下边框

p {

border-bottom: 6px solid red;

background-color: lightgrey;

}

# 第21章 CSS 圆角边框

## 21.1 CSS 圆角边框

border-radius 属性用于向元素添加圆角边框：

实例

p {

border: 2px solid red;

border-radius: 5px;

}

## 21.2 一个声明中的所有上边框属性

<!DOCTYPE html>

<html>

<head>

<style>

p {

border-style: solid;

border-top: thick double #ff0000;

}

</style>

</head>

<body>

<p>这是段落中的一些文本。</p>

</body>

</html>

## 21.3 设置下边框的样式

<!DOCTYPE html>

<html>

<head>

<style>

p {border-style: solid;}

p.none {border-bottom-style: none;}

p.dotted {border-bottom-style: dotted;}

p.dashed {border-bottom-style: dashed;}

p.solid {border-bottom-style: solid;}

p.double {border-bottom-style: double;}

p.groove {border-bottom-style: groove;}

p.ridge {border-bottom-style: ridge;}

p.inset {border-bottom-style: inset;}

p.outset {border-bottom-style: outset;}

</style>

</head>

<body>

<p class="none">无下边框。</p>

<p class="dotted">点状下边框。</p>

<p class="dashed">虚线下边框。</p>

<p class="solid">实线下边框。</p>

<p class="double">双线下边框。</p>

<p class="groove">凹槽下边框。</p>

<p class="ridge">垄状下边框。</p>

<p class="inset">内陷下边框。</p>

<p class="outset">凸出下边框。</p>

</body>

</html>

## 21.4 设置左边框的宽度

<!DOCTYPE html>

<html>

<head>

<style>

p {

border-style: solid;

border-left-width: 15px;

}

</style>

</head>

<body>

<p><b>注释：</b>如果单独使用，"border-left-width" 属性不起作用。请首先使用 "border-style" 属性来设置边框。</p>

</body>

</html>

## 21.5 设置四条边框的颜色

<!DOCTYPE html>

<html>

<head>

<style>

p.one {

border-style: solid;

border-color: #0000ff;

}

p.two {

border-style: solid;

border-color: #ff0000 #0000ff;

}

p.three {

border-style: solid;

border-color: #ff0000 #00ff00 #0000ff;

}

p.four {

border-style: solid;

border-color: #ff0000 #00ff00 #0000ff rgb(250,0,255);

}

</style>

</head>

<body>

<p class="one">一种颜色的边框！</p>

<p class="two">两种颜色的边框！</p>

<p class="three">三种颜色的边框！</p>

<p class="four">四种颜色的边框！</p>

<p><b>注释：</b>如果单独使用，"border-color" 属性不起作用。请首先使用 "border-style" 属性来设置边框。</p>

</body>

</html>

## 21.6 设置右边框的颜色

<!DOCTYPE html>

<html>

<head>

<style>

p {

border-style: solid;

border-right-color: #ff0000;

}

</style>

</head>

<body>

<p>这是段落中的一些文本。</p>

</body>

</html>

## 21.7 所有 CSS 边框属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| border | 简写属性，在一条声明中设置所有边框属性。 |
| border-color | 简写属性，设置四条边框的颜色。 |
| border-radius | 简写属性，可设置圆角的所有四个 border-\*-radius 属性。 |
| border-style | 简写属性，设置四条边框的样式。 |
| border-width | 简写属性，设置四条边框的宽度。 |
| border-bottom | 简写属性，在一条声明中设置所有下边框属性。 |
| border-bottom-color | 设置下边框的颜色。 |
| border-bottom-style | 设置下边框的样式。 |
| border-bottom-width | 设置下边框的宽度。 |
| border-left | 简写属性，在一条声明中设置所有左边框属性。 |
| border-left-color | 设置左边框的颜色。 |
| border-left-style | 设置左边框的样式。 |
| border-left-width | 设置左边框的宽度。 |
| border-right | 简写属性，在一条声明中设置所有右边框属性。 |
| border-right-color | 设置右边框的颜色。 |
| border-right-style | 设置右边框的样式。 |
| border-right-width | 设置右边框的宽度。 |
| border-top | 简写属性，在一条声明中设置所有上边框属性。 |
| border-top-color | 设置上边框的颜色。 |
| border-top-style | 设置上边框的样式。 |
| border-top-width | 设置上边框的宽度。 |

# 第22章 CSS 外边距

## 22.1 CSS 外边距

CSS margin 属性用于在任何定义的边框之外，为元素周围创建空间。

通过 CSS，您可以完全控制外边距。有一些属性可用于设置元素每侧（上、右、下和左）的外边距。

## 22.2 Margin - 单独的边

CSS 拥有用于为元素的每一侧指定外边距的属性：

margin-top

margin-right

margin-bottom

margin-left

所有外边距属性都可以设置以下值：

auto - 浏览器来计算外边距

length - 以 px、pt、cm 等单位指定外边距

% - 指定以包含元素宽度的百分比计的外边距

inherit - 指定应从父元素继承外边距

提示：允许负值。

实例

为 <p> 元素的所有四个边设置不同的外边距：

p {

margin-top: 100px;

margin-bottom: 100px;

margin-right: 150px;

margin-left: 80px;

}

## 22.3 Margin - 简写属性

为了缩减代码，可以在一个属性中指定所有外边距属性。

margin 属性是以下各外边距属性的简写属性：

margin-top

margin-right

margin-bottom

margin-left

工作原理是这样的：

如果 margin 属性有四个值：

margin: 25px 50px 75px 100px;

上外边距是 25px

右外边距是 50px

下外边距是 75px

左外边距是 100px

实例

margin 简写属性设置四个值：

p {

margin: 25px 50px 75px 100px;

}

如果 margin 属性设置三个值：

margin: 25px 50px 75px;

上外边距是 25px

右和左外边距是 50px

下外边距是 75px

实例

使用已设置三个值的 margin 简写属性：

p {

margin: 25px 50px 75px;

}

如果 margin 属性设置两个值：

margin: 25px 50px;

上和下外边距是 25px

右和左外边距是 50px

实例

使用设置了两个值的 margin 简写属性：

p {

margin: 25px 50px;

}

如果 margin 属性设置了一个值：

margin: 25px;

所有四个外边距都是 25px

实例

使用设置一个值的 margin 简写属性：

p {

margin: 25px;

}

## 22.4 auto 值

您可以将 margin 属性设置为 auto，以使元素在其容器中水平居中。

然后，该元素将占据指定的宽度，并且剩余空间将在左右边界之间平均分配。

实例

使用 margin: auto：

div {

width: 300px;

margin: auto;

border: 1px solid red;

}

## 22.5 inherit 值

本例使 <p class="ex1"> 元素的左外边距继承自父元素（<div>）：

实例

使用 inherit 值：

div {

border: 1px solid red;

margin-left: 100px;

}

p.ex1 {

margin-left: inherit;

}

# 第23章 CSS 外边距合并

外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。

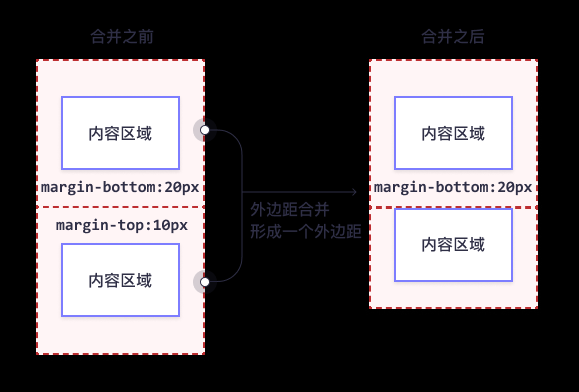
合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

## 23.1 外边距合并

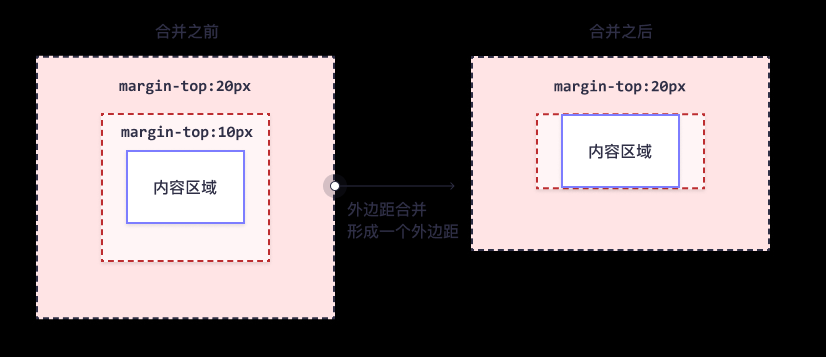
外边距合并（叠加）是一个相当简单的概念。但是，在实践中对网页进行布局时，它会造成许多混淆。

简单地说，外边距合并指的是，当两个垂直外边距相遇时，它们将形成一个外边距。合并后的外边距的高度等于两个发生合并的外边距的高度中的较大者。

当一个元素出现在另一个元素上面时，第一个元素的下外边距与第二个元素的上外边距会发生合并。请看下图：

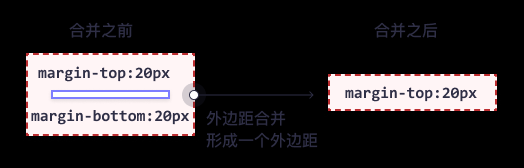


当一个元素包含在另一个元素中时（假设没有内边距或边框把外边距分隔开），它们的上和/或下外边距也会发生合并。请看下图：

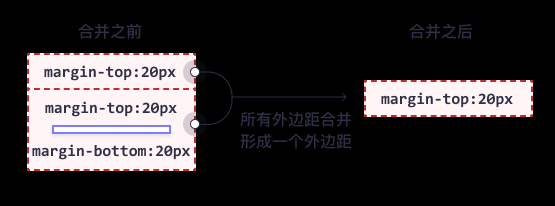


尽管看上去有些奇怪，但是外边距甚至可以与自身发生合并。

假设有一个空元素，它有外边距，但是没有边框或填充。在这种情况下，上外边距与下外边距就碰到了一起，它们会发生合并：

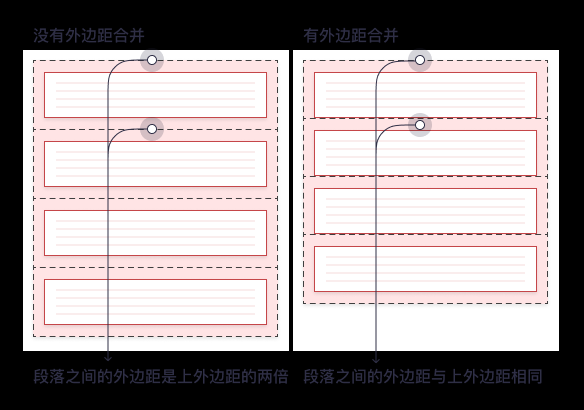


如果这个外边距遇到另一个元素的外边距，它还会发生合并：



这就是一系列的段落元素占用空间非常小的原因，因为它们的所有外边距都合并到一起，形成了一个小的外边距。

外边距合并初看上去可能有点奇怪，但是实际上，它是有意义的。以由几个段落组成的典型文本页面为例。第一个段落上面的空间等于段落的上外边距。如果没有外边距合并，后续所有段落之间的外边距都将是相邻上外边距和下外边距的和。这意味着段落之间的空间是页面顶部的两倍。如果发生外边距合并，段落之间的上外边距和下外边距就合并在一起，这样各处的距离就一致了。



注释：只有普通文档流中块框的垂直外边距才会发生外边距合并。行内框、浮动框或绝对定位之间的外边距不会合并。

## 23.2 所有 CSS 外边距属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| margin | 用于在一条声明中设置外边距属性的简写属性。 |
| margin-bottom | 设置元素的下外边距。 |
| margin-left | 设置元素的左外边距。 |
| margin-right | 设置元素的右外边距。 |
| margin-top | 设置元素的上外边距。 |

# 第24章 CSS 内边距

## 24.1 CSS 内边距

CSS padding 属性用于在任何定义的边界内的元素内容周围生成空间。

通过 CSS，您可以完全控制内边距（填充）。有一些属性可以为元素的每一侧（上、右、下和左侧）设置内边距。

## 24.2 Padding - 单独的边

CSS 拥有用于为元素的每一侧指定内边距的属性：

padding-top

padding-right

padding-bottom

padding-left

所有内边距属性都可以设置以下值：

length - 以 px、pt、cm 等单位指定内边距

% - 指定以包含元素宽度的百分比计的内边距

inherit - 指定应从父元素继承内边距

提示：不允许负值。

实例

为 <div> 元素的所有四个边设置不同的内边距：

div {

padding-top: 50px;

padding-right: 30px;

padding-bottom: 50px;

padding-left: 80px;

}

## 24.3 Padding - 简写属性

为了缩减代码，可以在一个属性中指定所有内边距属性。

padding 属性是以下各内边距属性的简写属性：

padding-top

padding-right

padding-bottom

padding-left

工作原理是这样的：

如果 padding 属性有四个值：

padding: 25px 50px 75px 100px;

上内边距是 25px

右内边距是 50px

下内边距是 75px

左内边距是 100px

实例

使用设置了四个值的 padding 简写属性：

div {

padding: 25px 50px 75px 100px;

}

如果 padding 属性设置了三个值：

padding: 25px 50px 75px;

上内边距是 25px

右和左内边距是 50px

下内边距是 75px

实例

使用设置了三个值的 padding 简写属性：

div {

padding: 25px 50px 75px;

}

如果 padding 属性设置了两个值：

padding: 25px 50px;

上和下内边距是 25px

右和左内边距是 50px

实例

使用设置了两个值的 padding 简写属性：

div {

padding: 25px 50px;

}

如果 padding 属性设置了一个值：

padding: 25px;

所有四个内边距都是 25px

实例

使用设置了一个值的 padding 简写属性：

div {

padding: 25px;

}

## 24.4 内边距和元素宽度

CSS width 属性指定元素内容区域的宽度。内容区域是元素（盒模型）的内边距、边框和外边距内的部分。

因此，如果元素拥有指定的宽度，则添加到该元素的内边距会添加到元素的总宽度中。这通常是不希望的结果。

实例

在这里，<div> 元素的宽度为 300px。但是，<div> 元素的实际宽度将是 350px（300px + 左内边距 25px + 右内边距 25px）：

div {

width: 300px;

padding: 25px;

}

若要将宽度保持为 300px，无论填充量如何，那么您可以使用 box-sizing 属性。这将导致元素保持其宽度。如果增加内边距，则可用的内容空间会减少。

实例

使用 box-sizing 属性将宽度保持为 300px，无论填充量如何：

div {

width: 300px;

padding: 25px;

box-sizing: border-box;

}

## 24.5 所有 CSS 内边距属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| padding | 用于在一条声明中设置所有内边距属性的简写属性。 |
| padding-bottom | 设置元素的下内边距。 |
| padding-left | 设置元素的左内边距。 |
| padding-right | 设置元素的右内边距。 |
| padding-top | 设置元素的上内边距。 |

# 第25章 CSS 高度和宽度

## 25.1 CSS 设置高度和宽度

height 和 width 属性用于设置元素的高度和宽度。

height 和 width 属性不包括内边距、边框或外边距。它设置的是元素内边距、边框以及外边距内的区域的高度或宽度。

## 25.2 CSS 高度和宽度值

height 和 width 属性可设置如下值：

auto - 默认。浏览器计算高度和宽度。

length - 以 px、cm 等定义高度/宽度。

% - 以包含块的百分比定义高度/宽度。

initial - 将高度/宽度设置为默认值。

inherit - 从其父值继承高度/宽度。

## 25.3 CSS 高度和宽度实例

实例

设置 <div> 元素的高度和宽度：

div {

height: 200px;

width: 50%;

background-color: powderblue;

}

实例

设置另一个 <div> 元素的高度和宽度：

div {

height: 100px;

width: 500px;

background-color: powderblue;

}

注意：请记住，height 和 width 属性不包括内边距、边框或外边距！它们设置的是元素的内边距、边框和外边距内的区域的高度/宽度！

## 25.4 设置 max-width

max-width 属性用于设置元素的最大宽度。

可以用长度值（例如 px、cm 等）或包含块的百分比（％）来指定 max-width（最大宽度），也可以将其设置为 none（默认值。意味着没有最大宽度）。

当浏览器窗口小于元素的宽度（500px）时，会发生之前那个 <div> 的问题。然后，浏览器会将水平滚动条添加到页面。

在这种情况下，使用 max-width 能够改善浏览器对小窗口的处理。

提示：将浏览器窗口拖动到小于500px的宽度，以查看两个 div 之间的区别！

此元素的高度为 100 像素，最大宽度为 500 像素。

此元素的高度为 100 像素，最大宽度为 500 像素。

注释：max-width 属性的值将覆盖 width（宽度）。

实例

这个 <div> 元素的高度为 100 像素，最大宽度为 500 像素：

div {

max-width: 500px;

height: 100px;

background-color: powderblue;

}

## 25.5 设置 CSS 尺寸属性

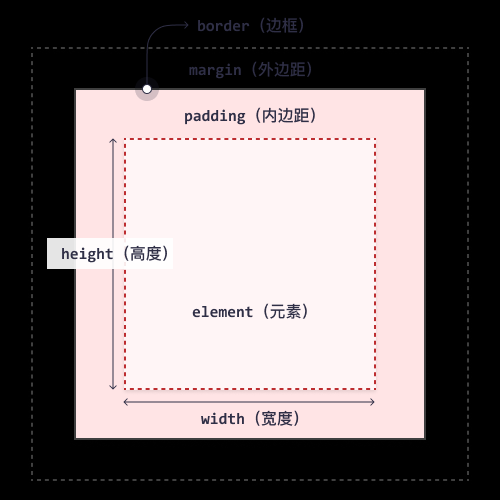
|  |  |
| --- | --- |
| 属性 | 描述 |
| height | 设置元素的高度。 |
| max-height | 设置元素的最大高度。 |
| max-width | 设置元素的最大宽度。 |
| min-height | 设置元素的最小高度。 |
| min-width | 设置元素的最小宽度。 |
| width | 设置元素的宽度。 |

# 第26章 CSS 框模型

## 26.1 CSS 框模型

所有 HTML 元素都可以视为方框。在 CSS 中，在谈论设计和布局时，会使用术语“盒模型”或“框模型”。

CSS 框模型实质上是一个包围每个 HTML 元素的框。它包括：外边距、边框、内边距以及实际的内容。下图展示了框模型：



对不同部分的说明：

1.内容 - 框的内容，其中显示文本和图像。

2.内边距 - 清除内容周围的区域。内边距是透明的。

3.边框 - 围绕内边距和内容的边框。

4.外边距 - 清除边界外的区域。外边距是透明的。

框模型允许我们在元素周围添加边框，并定义元素之间的空间。

元素框的最内部分是实际的内容，直接包围内容的是内边距。内边距呈现了元素的背景。内边距的边缘是边框。边框以外是外边距，外边距默认是透明的，因此不会遮挡其后的任何元素。

**提示**：背景应用于由内容和内边距、边框组成的区域。

内边距、边框和外边距都是可选的，默认值是零。但是，许多元素将由用户代理样式表设置外边距和内边距。可以通过将元素的 margin 和 padding 设置为零来覆盖这些浏览器样式。这可以分别进行，也可以使用通用选择器对所有元素进行设置：

\* {

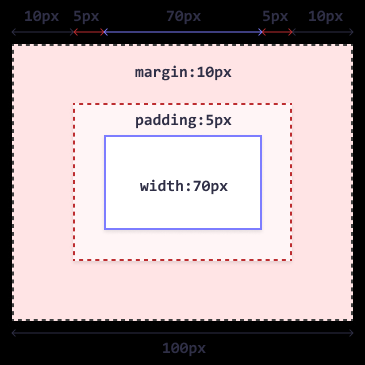
margin: 0;

padding: 0;

}

在 CSS 中，width 和 height 指的是内容区域的宽度和高度。增加内边距、边框和外边距不会影响内容区域的尺寸，但是会增加元素框的总尺寸。

假设框的每个边上有 10 个像素的外边距和 5 个像素的内边距。如果希望这个元素框达到 100 个像素，就需要将内容的宽度设置为 70 像素，请看下图：



#box {

width: 70px;

margin: 10px;

padding: 5px;

}

提示：内边距、边框和外边距可以应用于一个元素的所有边，也可以应用于单独的边。

提示：外边距可以是负值，而且在很多情况下都要使用负值的外边距。

实例

演示框模型：

div {

width: 300px;

border: 15px solid green;

padding: 50px;

margin: 20px;

}

## 26.2 元素的宽度和高度

为了在所有浏览器中正确设置元素的宽度和高度，您需要了解框模型如何工作。

重要提示：使用 CSS 设置元素的 width 和 height 属性时，只需设置内容区域的宽度和高度。要计算元素的完整大小，还必须把内边距、边框和外边距加起来。

实例

<div> 元素的总宽度将是 350px：

div {

width: 320px;

padding: 10px;

border: 5px solid gray;

margin: 0;

}

## 26.3 计算如下：

320px(宽度)

+ 20px（左+右内边距）

+ 10px（左+右边框）

+ 0px（左+右外边距）

= 350px

元素的总宽度应该这样计算：

元素总宽度 = 宽度 + 左内边距 + 右内边距 + 左边框 + 右边框 + 左外边距 + 右外边距

元素的总高度应该这样计算：

元素总高度 = 高度 + 上内边距 + 下内边距 + 上边框 + 下边框 + 上外边距 + 下外边距

# 第27章 CSS 轮廓

## 27.1 CSS 轮廓

轮廓是在元素周围绘制的一条线，在边框之外，以凸显元素。

CSS 拥有如下轮廓属性：

outline-style

outline-color

outline-width

outline-offset

outline

注意：轮廓与边框不同！不同之处在于：轮廓是在元素边框之外绘制的，并且可能与其他内容重叠。同样，轮廓也不是元素尺寸的一部分；元素的总宽度和高度不受轮廓线宽度的影响。

## 27.2 CSS 轮廓样式

outline-style 属性指定轮廓的样式，并可设置如下值：

dotted - 定义点状的轮廓。

dashed - 定义虚线的轮廓。

solid - 定义实线的轮廓。

double - 定义双线的轮廓。

groove - 定义 3D 凹槽轮廓。

ridge - 定义 3D 凸槽轮廓。

inset - 定义 3D 凹边轮廓。

outset - 定义 3D 凸边轮廓。

none - 定义无轮廓。

hidden - 定义隐藏的轮廓。

下例展示了不同的 outline-style 值：

实例

演示不同的轮廓样式：

p.dotted {outline-style: dotted;}

p.dashed {outline-style: dashed;}

p.solid {outline-style: solid;}

p.double {outline-style: double;}

p.groove {outline-style: groove;}

p.ridge {outline-style: ridge;}

p.inset {outline-style: inset;}

p.outset {outline-style: outset;}

注意：除非设置了 outline-style 属性，否则其他轮廓属性（在下一章中将详细介绍）都不会有任何作用！

# 第28章 CSS 轮廓宽度

## 28.1 CSS 轮廓宽度

outline-width 属性指定轮廓的宽度，并可设置如下值之一：

thin（通常为 1px）

medium（通常为 3px）

thick （通常为 5px）

特定尺寸（以 px、pt、cm、em 计）

下例展示了一些不同宽度的轮廓：

实例

p.ex1 {

border: 1px solid black;

outline-style: solid;

outline-color: red;

outline-width: thin;

}

p.ex2 {

border: 1px solid black;

outline-style: solid;

outline-color: red;

outline-width: medium;

}

p.ex3 {

border: 1px solid black;

outline-style: solid;

outline-color: red;

outline-width: thick;

}

p.ex4 {

border: 1px solid black;

outline-style: solid;

outline-color: red;

outline-width: 4px;

}

# 第29章 CSS 轮廓颜色

## 29.1 CSS 轮廓颜色

outline-color 属性用于设置轮廓的颜色。

可以通过以下方式设置颜色：

name - 指定颜色名，比如 "red"

HEX - 指定十六进制值，比如 "#ff0000"

RGB - 指定 RGB 值，比如 "rgb(255,0,0)"

HSL - 指定 HSL 值，比如 "hsl(0, 100%, 50%)"

invert - 执行颜色反转（确保轮廓可见，无论是什么颜色背景）

下例展示了一些不同颜色的不同轮廓样式。请注意，这些元素在轮廓内还有黑色细边框：

实例

p.ex1 {

border: 2px solid black;

outline-style: solid;

outline-color: red;

}

p.ex2 {

border: 2px solid black;

outline-style: dotted;

outline-color: blue;

}

p.ex3 {

border: 2px solid black;

outline-style: outset;

outline-color: grey;

}

HEX 值

您也可以使用十六进制值（HEX）指定轮廓颜色：

实例

p.ex1 {

outline-style: solid;

outline-color: #ff0000; /\* 红色 \*/

}

RGB 值

或者通过使用 RGB 值：

实例

p.ex1 {

outline-style: solid;

outline-color: rgb(255, 0, 0); /\* 红色 \*/

}

HSL 值

您还可以使用 HSL 值：

实例

p.ex1 {

outline-style: solid;

outline-color: hsl(0, 100%, 50%); /\* 红色 \*/

}

## 29.2 反转颜色

下例使用 outline-color: invert，执行了颜色反转。这样可以确保无论颜色背景如何，轮廓都是可见的：

p.ex1 {

border: 1px solid yellow;

outline-style: solid;

outline-color: invert;

}

# 第30章 CSS 轮廓简写

## 30.1 CSS Outline - 简写属性

outline 属性是用于设置以下各个轮廓属性的简写属性：

outline-width

outline-style（必需）

outline-color

从上面的列表中，outline 属性可指定一个、两个或三个值。值的顺序无关紧要。

下例展示了用简写的 outline 属性指定的一些轮廓：

实例

p.ex1 {outline: dashed;}

p.ex2 {outline: dotted red;}

p.ex3 {outline: 5px solid yellow;}

p.ex4 {outline: thick ridge pink;}

# 第31章 CSS 轮廓偏移

## 31.1 CSS 轮廓偏移

outline-offset 属性在元素的轮廓与边框之间添加空间。元素及其轮廓之间的空间是透明的。

下例指定边框边缘外 25px 的轮廓：

实例

p {

margin: 50px;

border: 1px solid black;

outline: 1px solid red;

outline-offset: 25px;

}

下例显示元素与其轮廓之间的空间是透明的：

实例

p {

margin: 30px;

background: yellow;

border: 1px solid black;

outline: 1px solid red;

outline-offset: 25px;

}

## 31.2 所有 CSS 轮廓属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| outline | 简写属性，在一条声明中设置 outline-width、outline-style 以及 outline-color。 |
| outline-color | 设置轮廓的颜色。 |
| outline-offset | 指定轮廓与元素的边缘或边框之间的空间。 |
| outline-style | 设置轮廓的样式。 |
| outline-width | 设置轮廓的宽度。 |

# 第32章 CSS 文本

## 32.1 文本颜色

color 属性用于设置文本的颜色。颜色由以下值指定：

颜色名 - 比如 "red"

十六进制值 - 比如 "#ff0000"

RGB 值 - 比如 "rgb(255,0,0)"

查看 CSS 颜色值，以获取可能颜色值的完整列表。

页面的默认文本颜色是在 body 选择器中定义的。

实例

body {

color: blue;

}

h1 {

color: green;

}

提示：对于 W3C compliant CSS：如果您定义了 color 属性，则还必须定义 background-color 属性。

## 32.2 文本颜色和背景色

在本例中，我们定义了 background-color 属性和 color 属性：

实例

body {

background-color: lightgrey;

color: blue;

}

h1 {

background-color: black;

color: white;

}

# 第33章 CSS 文本对齐

## 33.1 文本对齐

text-align 属性用于设置文本的水平对齐方式。

文本可以左对齐或右对齐，或居中对齐。

下例展示了居中对齐以及左右对齐的文本（如果文本方向是从左到右，则默认为左对齐；如果文本方向是从右到左，则默认是右对齐）：

实例

h1 {

text-align: center;

}

h2 {

text-align: left;

}

h3 {

text-align: right;

}

当 text-align 属性设置为 "justify" 后，将拉伸每一行，以使每一行具有相等的宽度，并且左右边距是直的（就像在杂志和报纸中）：

实例

div {

text-align: justify;

}

## 33.2 文本方向

direction 和 unicode-bidi 属性可用于更改元素的文本方向：

实例

p {

direction: rtl;

unicode-bidi: bidi-override;

}

## 33.3 垂直对齐

vertical-align 属性设置元素的垂直对齐方式。

本例演示如何设置文本中图像的垂直对齐方式：

实例

img.top {

vertical-align: top;

}

img.middle {

vertical-align: middle;

}

img.bottom {

vertical-align: bottom;

}

# 第34章 CSS 文字装饰

## 34.1 文字装饰

text-decoration 属性用于设置或删除文本装饰。

text-decoration: none; 通常用于从链接上删除下划线：

实例

a {

text-decoration: none;

}

其他 text-decoration 值用于装饰文本：

实例

h1 {

text-decoration: overline;

}

h2 {

text-decoration: line-through;

}

h3 {

text-decoration: underline;

}

注释：建议不要在非链接文本加下划线，因为这经常会使读者感到困惑。

# 第35章 CSS 文本转换

文本转换

text-transform 属性用于指定文本中的大写和小写字母。

它可用于将所有内容转换为大写或小写字母，或将每个单词的首字母大写：

实例

p.uppercase {

text-transform: uppercase;

}

p.lowercase {

text-transform: lowercase;

}

p.capitalize {

text-transform: capitalize;

}

# 第36章 CSS 文字间距

## 36.1 文字缩进

text-indent 属性用于指定文本第一行的缩进：

实例

p {

text-indent: 50px;

}

## 36.2 字母间距

letter-spacing 属性用于指定文本中字符之间的间距。

下例演示如何增加或减少字符之间的间距：

实例

h1 {

letter-spacing: 3px;

}

h2 {

letter-spacing: -3px;

}

## 36.3 行高

line-height 属性用于指定行之间的间距：

实例

p.small {

line-height: 0.8;

}

p.big {

line-height: 1.8;

}

## 36.4 字间距

word-spacing 属性用于指定文本中单词之间的间距。

下例演示如何增加或减少单词之间的间距：

实例

h1 {

word-spacing: 10px;

}

h2 {

word-spacing: -5px;

}

## 36.5 空白

white-space 属性指定元素内部空白的处理方式。

此例演示如何禁用元素内的文本换行：

实例

p {

white-space: nowrap;

}

# 第37章 CSS 文本阴影

## 37.1 文本阴影

text-shadow 属性为文本添加阴影。

最简单的用法是只指定水平阴影（2px）和垂直阴影（2px）：

文字阴影效果！

实例

h1 {

text-shadow: 2px 2px;

}

实例

h1 {

text-shadow: 2px 2px red;

}

实例

h1 {

text-shadow: 2px 2px 5px red;

}

## 37.2 所有 CSS 文本属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| color | 设置文本颜色。 |
| direction | 指定文本的方向 / 书写方向。 |
| letter-spacing | 设置字符间距。 |
| line-height | 设置行高。 |
| text-align | 指定文本的水平对齐方式。 |
| text-decoration | 指定添加到文本的装饰效果。 |
| text-indent | 指定文本块中首行的缩进。 |
| text-shadow | 指定添加到文本的阴影效果。 |
| text-transform | 控制文本的大小写。 |
| text-overflow | 指定应如何向用户示意未显示的溢出内容。 |
| unicode-bidi | 与 direction 属性一起使用，设置或返回是否应重写文本来支持同一文档中的多种语言。 |
| vertical-align | 指定文本的垂直对齐方式。 |
| white-space | 指定如何处理元素内的空白。 |
| word-spacing | 设置单词间距。 |

# 第38章 CSS 字体

## 38.1 字体选择很重要

选择正确的字体会对网站的用户体验产生巨大影响。

正确的字体可以为您的品牌创造强有力的形象。

使用易于阅读的字体很重要。字体为您的文本增加了价值。为字体选择正确的颜色和文本大小也很重要。

## 38.2 通用字体族

在 CSS 中，有五个通用字体族：

衬线字体（Serif）- 在每个字母的边缘都有一个小的笔触。它们营造出一种形式感和优雅感。

无衬线字体（Sans-serif）- 字体线条简洁（没有小笔画）。它们营造出现代而简约的外观。

等宽字体（Monospace）- 这里所有字母都有相同的固定宽度。它们创造出机械式的外观。

草书字体（Cursive）- 模仿了人类的笔迹。

幻想字体（Fantasy）- 是装饰性/俏皮的字体。

所有不同的字体名称都属于这五个通用字体系列之一。

提示：在计算机屏幕上，无衬线字体被认为比衬线字体更易于阅读。

## 38.3 CSS font-family 属性

在 CSS 中，我们使用 font-family 属性规定文本的字体。

font-family 属性应包含多个字体名称作为“后备”系统，以确保浏览器/操作系统之间的最大兼容性。请以您需要的字体开始，并以通用系列结束（如果没有其他可用字体，则让浏览器选择通用系列中的相似字体）。字体名称应以逗号分隔。

注释：如果字体名称不止一个单词，则必须用引号引起来，例如："Times New Roman"。

实例

为三个段落规定不同的字体：

.p1 {

font-family: "Times New Roman", Times, serif;

}

.p2 {

font-family: Arial, Helvetica, sans-serif;

}

.p3 {

font-family: "Lucida Console", "Courier New", monospace;

}

# 第39章 CSS 字体样式

## 39.1 字体样式

font-style 属性主要用于指定斜体文本。

此属性可设置三个值：

normal - 文字正常显示

italic - 文本以斜体显示

oblique - 文本为“倾斜”（倾斜与斜体非常相似，但支持较少）

实例

p.normal {

font-style: normal;

}

p.italic {

font-style: italic;

}

p.oblique {

font-style: oblique;

}

## 39.2 字体粗细

font-weight 属性指定字体的粗细：

实例

p.normal {

font-weight: normal;

}

p.thick {

font-weight: bold;

}

## 39.3 字体变体

font-variant 属性指定是否以 small-caps 字体（小型大写字母）显示文本。

在 small-caps 字体中，所有小写字母都将转换为大写字母。但是，转换后的大写字母的字体大小小于文本中原始大写字母的字体大小。

实例

p.normal {

font-variant: normal;

}

p.small {

font-variant: small-caps;

}

# 第40章 CSS 字体大小

## 40.1 字体大小

font-size 属性设置文本的大小。

在网页设计中，能够管理文本大小很重要。但是，不应使用调整字体大小来使段落看起来像标题，或是使标题看起来像段落。

请始终使用正确的 HTML 标签，例如 <h1> - <h6> 用于标题，而 <p> 仅用于段落。

font-size 值可以是绝对或相对大小。

绝对尺寸：

将文本设置为指定大小

不允许用户在所有浏览器中更改文本大小（可访问性不佳）

当输出的物理尺寸已知时，绝对尺寸很有用

相对尺寸：

设置相对于周围元素的大小

允许用户在浏览器中更改文本大小

注释：如果您没有指定字体大小，则普通文本（如段落）的默认大小为 16px（16px = 1em）。

## 40.2 以像素设置字体大小

使用像素设置文本大小可以完全控制文本大小：

实例

h1 {

font-size: 40px;

}

h2 {

font-size: 30px;

}

p {

font-size: 14px;

}

提示：如果您使用了像素，则仍然可以使用缩放工具来调整整个页面的大小。

## 40.3 用 em 设置字体大小

为了允许用户调整文本大小（在浏览器菜单中），许多开发人员使用 em 而不是像素。

W3C 建议使用 em 尺寸单位。

1em 等于当前字体大小。浏览器中的默认文本大小为 16px。因此，默认大小 1em 为 16px。

可以使用这个公式从像素到 em 来计算大小：pixels/16=em。

实例

h1 {

font-size: 2.5em; /\* 40px/16=2.5em \*/

}

h2 {

font-size: 1.875em; /\* 30px/16=1.875em \*/

}

p {

font-size: 0.875em; /\* 14px/16=0.875em \*/

}

在上例中，em 单位的文本大小与上一个例子中的像素大小相同。但是，若使用 em 尺寸，则可以在所有浏览器中调整文本大小。

不幸的是，旧版本的 Internet Explorer 仍然存在问题。放大文本时它比应该大的尺寸更大，缩小文本时会更小。

## 40.4 使用百分比和 Em 的组合

实例

body {

font-size: 100%;

}

h1 {

font-size: 2.5em;

}

h2 {

font-size: 1.875em;

}

p {

font-size: 0.875em;

}

我们的代码目前运行良好！它在所有浏览器中显示相同的文本大小，并允许所有浏览器缩放或调整文本大小！

## 40.5 响应式字体大小

可以使用 vw 单位设置文本大小，它的意思是“视口宽度”（"viewport width"）。

这样，文本大小将遵循浏览器窗口的大小，请调整浏览器窗口的大小，以查看字体大小如何缩放：

实例

<h1 style="font-size:10vw">Hello World</h1>

视口（Viewport）是浏览器窗口的大小。 1vw = 视口宽度的 1％。如果视口为 50 厘米宽，则 1vw 为 0.5 厘米。

# 第41章 CSS 谷歌字体

## 41.1 谷歌字体

如果您不想使用 HTML 中的任何标准字体，则可以使用 Google Fonts API 向页面添加数百种其他字体。

只需添加一个样式表链接并引用您选择的字体系列：

实例

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Sofia">

<style>

body {

font-family: "Sofia";

font-size: 22px;

}

</style>

</head>

<body>

<h1>Sofia Font</h1>

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</p>

</body>

</html>

# 第42章 CSS 字体属性

## 42.1 字体属性

为了缩短代码，也可以在一个属性中指定所有单个字体属性。

font 属性是以下属性的简写属性：

font-style

font-variant

font-weight

font-size/line-height

font-family

实例

使用简写声明设置一些字体属性：

p.a {

font: 20px Arial, sans-serif;

}

p.b {

font: italic small-caps bold 12px/30px Georgia, serif;

}

注意：font-size 和 font-family 的值是必需的。如果缺少其他值之一，则会使用其默认值。

## 42.2 所有 CSS 字体属性

|  |  |
| --- | --- |
| 属性 | 描述 |
| font | 简写属性。在一条声明中设置所有字体属性。 |
| font-family | 规定文本的字体系列（字体族）。 |
| font-size | 规定文本的字体大小。 |
| font-style | 规定文本的字体样式。 |
| font-variant | 规定是否以小型大写字母的字体显示文本。 |
| font-weight | 规定字体的粗细。 |

# 第43章 CSS 图标

## 43.1 如何添加图标

向 HTML 页面添加图标的最简单方法是使用图标库，比如 Font Awesome。

将指定的图标类的名称添加到任何行内 HTML 元素（如 <i> 或 <span>）。

下面的图标库中的所有图标都是可缩放矢量，可以使用 CSS进行自定义（大小、颜色、阴影等）。

## 43.2 Font Awesome 图标

如需使用 Font Awesome 图标，请访问 fontawesome.com，登录并获取代码添加到 HTML 页面的 <head> 部分：

<script src="https://kit.fontawesome.com/yourcode.js"></script>

请在我们的 Font Awesome 5 教程中，阅读有关如何开始使用 Font Awesome 的更多内容。

提示：无需下载或安装！

实例

<!DOCTYPE html>

<html>

<head>

<script src="https://kit.fontawesome.com/a076d05399.js"></script>

</head>

<body>

<i class="fas fa-cloud"></i>

<i class="fas fa-heart"></i>

<i class="fas fa-car"></i>

<i class="fas fa-file"></i>

<i class="fas fa-bars"></i>

</body>

</html>

## 43.3 Bootstrap 图标

如需使用 Bootstrap glyphicons，请在 HTML 页面的 <head> 部分内添加这行：

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

提示：无需下载或安装！

实例

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

</head>

<body>

<i class="glyphicon glyphicon-cloud"></i>

<i class="glyphicon glyphicon-remove"></i>

<i class="glyphicon glyphicon-user"></i>

<i class="glyphicon glyphicon-envelope"></i>

<i class="glyphicon glyphicon-thumbs-up"></i>

</body>

</html>

## 43.4 Google 图标

如需使用 Google 图标，请在HTML页面的 <head> 部分中添加以下行：

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

提示：无需下载或安装！

实例

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

</head>

<body>

<i class="material-icons">cloud</i>

<i class="material-icons">favorite</i>

<i class="material-icons">attachment</i>

<i class="material-icons">computer</i>

<i class="material-icons">traffic</i>

</body>

</html>

# 第44章 CSS 链接

## 44.1 设置链接样式

链接可以使用任何 CSS 属性（例如 color、font-family、background 等）来设置样式。

实例

a {

color: hotpink;

}

此外，可以根据链接处于什么状态来设置链接的不同样式。

四种链接状态分别是：

a:link - 正常的，未访问的链接

a:visited - 用户访问过的链接

a:hover - 用户将鼠标悬停在链接上时

a:active - 链接被点击时

实例

/\* 未被访问的链接 \*/

a:link {

color: red;

}

/\* 已被访问的链接 \*/

a:visited {

color: green;

}

/\* 将鼠标悬停在链接上 \*/

a:hover {

color: hotpink;

}

/\* 被选择的链接 \*/

a:active {

color: blue;

}

如果为多个链接状态设置样式，请遵循如下顺序规则：

a:hover 必须 a:link 和 a:visited 之后

a:active 必须在 a:hover 之后

## 44.2 文本装饰

text-decoration 属性主要用于从链接中删除下划线：

实例

a:link {

text-decoration: none;

}

a:visited {

text-decoration: none;

}

a:hover {

text-decoration: underline;

}

a:active {

text-decoration: underline;

}

## 44.3 背景色

background-color 属性可用于指定链接的背景色：

实例

a:link {

background-color: yellow;

}

a:visited {

background-color: cyan;

}

a:hover {

background-color: lightgreen;

}

a:active {

background-color: hotpink;

}

## 44.4 链接按钮

本例演示了一个更高级的例子，其中我们组合了多个 CSS 属性，将链接显示为框/按钮：

实例

a:link, a:visited {

background-color: #f44336;

color: white;

padding: 14px 25px;

text-align: center;

text-decoration: none;

display: inline-block;

}

a:hover, a:active {

background-color: red;

}

## 44.5 为超链接添加不同的样式

<!DOCTYPE html>

<html>

<head>

<style>

a.one:link {color:#ff0000;}

a.one:visited {color:#0000ff;}

a.one:hover {color:#ffcc00;}

a.two:link {color:#ff0000;}

a.two:visited {color:#0000ff;}

a.two:hover {font-size:150%;}

a.three:link {color:#ff0000;}

a.three:visited {color:#0000ff;}

a.three:hover {background:#66ff66;}

a.four:link {color:#ff0000;}

a.four:visited {color:#0000ff;}

a.four:hover {font-family:monospace;}

a.five:link {color:#ff0000;text-decoration:none;}

a.five:visited {color:#0000ff;text-decoration:none;}

a.five:hover {text-decoration:underline;}

</style>

</head>

<body>

<p>请把鼠标移到链接上并观察样式的变化：</p>

<p><b><a class="one" href="default.asp" target="\_blank">此链接改变颜色</a></b></p>

<p><b><a class="two" href="default.asp" target="\_blank">此链接改变字体大小</a></b></p>

<p><b><a class="three" href="default.asp" target="\_blank">此链接改变背景色</a></b></p>

<p><b><a class="four" href="default.asp" target="\_blank">此链接改变字体族</a></b></p>

<p><b><a class="five" href="default.asp" target="\_blank">此链接改变文本装饰</a></b></p>

</body>

</html>

## 44.6 高级 - 创建带边框的链接按钮

<!DOCTYPE html>

<html>

<head>

<style>

a:link, a:visited {

background-color: white;

color: black;

border: 2px solid green;

padding: 10px 20px;

text-align: center;

text-decoration: none;

display: inline-block;

}

a:hover, a:active {

background-color: green;

color: white;

}

</style>

</head>

<body>

<a href="default.asp" target="\_blank">这是一个链接</a>

</body>

</html>

## 44.7 改变光标

<!DOCTYPE html>

<html>

<body>

<p>请把鼠标移动到单词上，以查看指针效果：</p>

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

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

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

<span style="cursor:e-resize">e-resize</span><br>

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

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

<span style="cursor:n-resize">n-resize</span><br>

<span style="cursor:ne-resize">ne-resize</span><br>

<span style="cursor:nw-resize">nw-resize</span><br>

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

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

<span style="cursor:s-resize">s-resize</span><br>

<span style="cursor:se-resize">se-resize</span><br>

<span style="cursor:sw-resize">sw-resize</span><br>

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

<span style="cursor:w-resize">w-resize</span><br>

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

</body>

</html>