> 此页面由社区从英文翻译而来。了解更多并加入 MDN Web Docs 社区。

# CSS 的值与单位

CSS 中使用的每个属性都允许拥有一个或一组值，查看 MDN 上的任何属性页将帮助你理解对任何特定属性有效的值。在本节课中，我们将学习一些最常用的值和单位。

| | |
|---|---|
| **前提：** | 基本的计算机知识、安装基本的软件、文件处理基本知识、HTML 基础知识（学习 HTML 入门），以及 CSS 如何工作的基本常识（如果完全不了解 CSS，请移步学习 CSS 第一步。） |
| **目标：** | 了解 CSS 属性中使用的不同类型的值和单位。 |

## 什么是 CSS 的值?

在 CSS 规范和 MDN 的属性页上，你将能够发现值的存在，因为它们将被尖括号包围，如 `<color>` 或 `<length>`。当你看到值 `<color>` 对特定属性有效时，这意味着你可以使用任何有效的颜色作为该属性的值，如 `<color>` 参考页面所列。

> **备注：** 你还将看到被称为数据类型的 CSS 值。这些术语基本上是可以互换的——当你在 CSS 中看到一些被称为数据类型的东西时，它实际上只是一种表示值的奇特方式。

> **备注：** 是的，CSS 值倾向于使用尖括号表示，以区别于 CSS 属性（例如 `color` 属性和 `<color>` 数据类型）。你可能还会混淆 CSS 数据类型和 HTML 元素，因为它们都使用尖括号，但这不太可能——它们在完全不一样的上下文中使用。

在下面的例子中，我们使用关键字设置标题的颜色，使用 `rgb()` 函数设置背景：

```css
h1 {
  color: black;
  background-color: rgb(197, 93, 161);
}
```

CSS 中的值类型是一种定义了一些可使用的值的集合的方式。这意味着如果你看到的 `<color>` 是有效的，那么你就不需要考虑可以使用哪种类型——不管是关键字、十六进制值还是 `rgb()` 函数等都是有效的。如果浏览器支持这些可用的 `<color>` 值，则可以使用它们当中的任意一个。MDN 上针对每个值类型的页面将提供有关浏览器支持的信息。例如，如果你查看 `<color>` 的页面，你将看到浏览器兼容性部分列出了不同类型的颜色值以及对它们的支持。

让我们来看看你可能经常遇到的一些值和单位类型，并提供一些示例，以便你尝试使用各种值的可能性。

## 数字、长度和百分比

你可能会发现自己在 CSS 中使用了各种数值数据类型。以下全部归类为数值：

| 数值类型 | 描述 |
| --- | --- |
| `<integer>` | `<integer>` 是一个整数，比如 `1024` 或 `-55`。 |
| `<number>` | `<number>` 表示一个小数——它可能有小数点后面的部分，也可能没有，例如 `0.255`、`128` 或 `-1.2`。 |
| `<dimension>` | `<dimension>` 是一个 `<number>` 它有一个附加的单位，例如 `45deg`、`5s` 或 `10px`。`<dimension>` 是一个伞形类别，包括 `<length>`、`<angle>`、`<time>` 和 `<resolution>` 类型。 |
| `<percentage>` | `<percentage>` 表示一些其他值的一部分，例如 `50%`。百分比值总是相对于另一个量。例如，一个元素的长度相对于其父元素的长度。 |

### 长度

最常见的数字类型是 `<length>`，例如 `10px`（像素）或 `30em`。CSS 中有两种类型的长度——相对长度和绝对长度。重要的是要知道它们之间的区别，以便理解他们控制的元素将变得有多大。

### 绝对长度单位

以下都是**绝对**长度单位——它们与其他任何东西都没有关系，通常被认为总是相同的大小。

| 单位 | 名称 | 等价换算 |
| --- | --- | --- |
| `cm` | 厘米 | 1cm = 37.8px = 25.2/64in |
| `mm` | 毫米 | 1mm = 1/10th of 1cm |
| `Q` | 四分之一毫米 | 1Q = 1/40th of 1cm |
| `in` | 英寸 | 1in = 2.54cm = 96px |
| `pc` | 派卡 | 1pc = 1/6th of 1in |
| `pt` | 点 | 1pt = 1/72th of 1in |
| `px` | 像素 | 1px = 1/96th of 1in |

这些值中的大多数在用于打印时比用于屏幕输出时更有用。例如，我们通常不会在屏幕上使用 `cm`（厘米）。惟一一个你经常使用的值，估计就是 `px`（像素）。

## 相对长度单位

相对长度单位是相对于其他某些东西的。例如：

- `em` 和 `rem` 分别相对于父元素和根元素的字体大小。
- `vh` 和 `vw` 分别相对于视口的高度和宽度。

使用相对单位的好处是，通过一些精心的规划，你可以使文本或其他元素的大小相对于页面上的任何指定的东西进行缩放。要获取可用的相对单位的完整列表，请参阅 `<length>` 类型的参考页面。

在本节中，我们将探讨一些最常见的相对单位。

## 探索一个例子

在下面的示例中，你可以看到一些相对长度单位和绝对长度单位的行为。第一个框以像素为单位设置 `width`。作为一个绝对单位，这个宽度将保持不变，无论其他如何变化。

第二个框的宽度设置为 `vw`（视口宽度）单位。这个值相对于视口宽度，所以 `10vw` 是视口宽度的 10%。如果你更改浏览器窗口的宽度，那么框的大小应该会更改，但是这个示例使用 `<iframe>` 嵌入到页面中，所以这将不起作用。要查看实际情况，你必须在打开示例的浏览器选项卡后尝试该示例 。

第三个盒子使用 `em` 单位。这些是相对于字体大小的。我在包含 `<div>` 的元素上设置了一个 `1em` 的字体大小，它有一个 `.wrapper` 类。将这个值更改为 `1.5em` ，你将看到所有元素的字体大小都增加了，但是只有最后一项会变宽，因为宽度与字体大小有关。

按照上面的说明操作之后，尝试以其他方式处理这些值，看看你将收获什么。

I am 200px wide

I am 10vw wide

I am 10em wide

### Interactive editor

```
.wrapper {
  font-size: 1em;
}

.px {
  width: 200px;
}

.vw {
  width: 10vw;
}

.em {
  width: 10em;
}
```

```
<div class="wrapper">
  <div class="box px">I am 200px wide</div>
  <div class="box vw">I am 10vw wide</div>
  <div class="box em">I am 10em wide</div>
</div>
```

Reset

## em 和 rem

`em` 和 `rem` 是你在从框到文本调整大小时最常遇到的两个相对长度。了解这些方法是如何工作的以及它们之间的区别是很有意义的，尤其是当你开始学习更复杂的主题时，比如样式化文本或 CSS 布局。下面的示例提供了一个演示。

HTML 是一组嵌套的列表——我们总共有三个列表，并且两个示例都有相同的 HTML。唯一的区别是第一个类具有 *em*，第二个类具有 *rem*。

首先，我们将 16px 设置为 `<html>` 元素的字体大小。

**概括地说，在排版属性中 em 单位的意思是“父元素的字体大小”。**带有 `ems` 类的 `<ul>` 内的 `<li>` 元素从它们的父元素中获取大小。因此，每一个连续的嵌套级别都会逐渐变大，因为每个嵌套的字体大小都被设置为 `1.3em` ——是其父嵌套字体大小的 1.3 倍。

**概括地说，rem 单位的意思是“根元素的字体大小”。**（“根 em”的 rem 标准。）`<ul>` 内的 `<li>` 元素和一个 rems 类从根元素（`<html>`）中获取它们的大小。这意味着每一个连续的嵌套层都不会不断变大。

但是，如果你在 CSS 中更改 `<html>` 字体大小，你将看到所有其他相关内容都发生了更改，包括 `rem` 和 `em` 大小的文本。

## 百分比

在许多情况下，百分比与长度的处理方法是一样的。百分比的问题在于，它们总是相对于其他值设置的。例如，如果将元素的字体大小设置为百分比，那么它将是元素父元素字体大小的百分比。如果使用百分比作为宽度值，那么它将是父值宽度的百分比。

在下面的示例中，两个百分比大小的框和两个像素大小的框具有相同的类名。分别为 200px 和 40% 宽。

不同之处在于，第二组两个框位于一个 400px 宽的包装器中。第二个 200px 宽的盒子和第一个一样宽，但是第二个 40% 的盒子现在是 400px 的 40%——比第一个窄多了！

**尝试更改包装器的宽度或百分比值，看看这是如何工作的。**

下一个示例以百分比设置字体大小。每个 `<li>` 都有 80% 的字体大小，因此嵌套列表项在从父级继承其大小时将逐渐变小。

注意，虽然许多值接受长度或百分比，但也有一些值只接受长度。你可以在 MDN 属性引用页面上看到它能接受哪些值。如果允许的值包括 `<length-percentage>`，则可以使用长度或百分比。如果允许的值只包含 `<length>`，则不可能使用百分比。

## 数字

有些值接受数字，不添加任何单位。接受无单位数字的属性的一个例子是不透明度属性（`opacity`），它控制元素的不透明度（它的透明程度）。此属性接受 `0`（完全透明）和 `1`（完全不透明）之间的数字。

**在下面的示例中，尝试将不透明度值更改为 `0` 到 `1` 之间的各种小数值，并查看框及其内容是如何变得透明或者不透明的。**

> **备注：** 当你在 CSS 中使用数字作为值时，它不应该用引号括起来。

## 颜色

在 CSS 中指定颜色的方法有很多，其中一些是最近才实现的。在 CSS 中，相同的颜色值可以在任何地方使用，无论你指定的是文本颜色、背景颜色还是其他颜色。

现代计算机的标准颜色系统是 24 位的，它允许通过不同的红、绿、蓝通道的组合显示大约 1670 万种不同的颜色，每个通道有 256 个不同的值（256 x 256 x 256 = 16,777,216）。让我们来看看在 CSS 中指定颜色的一些方法。

> **备注：** 在本教程中，我们将研究具有良好浏览器支持的常用指定颜色的方法；虽然还有其他的，但是他们没有很好的支持，也不太常见。

## 颜色关键词

在这学习示例或 MDN 上的其他示例中，你经常会看到使用的颜色关键字，因为它们是一种指定颜色的简单易懂的方式。有一些关键词，其中一些有相当有趣的名字！你可以在页面上看到 `<color>` 值的完整列表。

**在下面的示例中尝试使用不同的颜色值，以了解它们是如何工作的。**

## 十六进制 RGB 值

你可能遇到的下一种颜色值类型是十六进制代码。每个十六进制值由一个散列/磅符号（#）和六个十六进制数字组成，每个十六进制数字都可以取 0 到 f（代表 15）之间的 16 个值中的一个——所以是 `0123456789abcdef` 。每对值表示一个通道——红色、绿色和蓝色——并允许我们为每个通道指定 256 个可用值中的任意一个（16 x 16 = 256）。

这些值有点复杂，不太容易理解，但是它们比关键字更通用——你可以使用十六进制值来表示你想在配色方案中使用的任何颜色。

**同样，大胆尝试更改值，看看颜色如何变化吧！**

## RGB 和 RGBA 的值

我们将在这里讨论的第三种方案是 RGB。RGB 值是一个函数——`rgb()`——它有三个参数，表示颜色的红色、绿色和蓝色通道值，与十六进制值的方法非常相似。RGB 的不同之处在于，每个通道不是由两个十六进制数字表示的，而是由一个介于 0 到 255 之间的十进制数字表示的——这有点容易理解。

让我们重写上一个例子，使用 RGB 颜色：

你可以向 `rgb()` 传递第四个参数，它代表颜色的 alpha 通道，控制不透明度。如果你把这个值设置为 `0`，它将使颜色完全透明，而 `1` 将使它完全不透明。介于两者之间的值会给你带来不同级别的透明度。

> **备注：** 在颜色上设置 alpha 通道与使用我们前面看到的 `opacity` 属性有一个关键区别。当你使用不透明度时，你让元素和它里面的所有东西都不透明，而使用 RGB 与 alpha 参数的颜色只让你指定的颜色不透明。

在下面的例子中，我添加了一个背景图片到我们的彩色方块的包含块中。然后我设置了不同的不透明度值——注意当 alpha 通道值较小时，背景如何显示的。

**在本例中，尝试更改 alpha 通道值，看看它如何影响颜色输出。**

> **备注：** 在旧版本的 CSS 中，`rgb()` 语法不支持 alpha 参数——你需要使用另一个叫 `rgba()` 的函数来实现。如今，你可以向 `rgb()` 传递一个 alpha 参数，但为了向后兼容旧网站，`rgba()` 语法仍然被支持，并且具有与 `rgb()` 完全相同的行为。

## HSL 和 HSLA 的值

另一种指定颜色的方法是 HSL 颜色模型。`hsl()` 函数不接受红、绿、蓝值，而是接受色相、饱和度和亮度值，这些值用于区分 1670 万种颜色，但方式不同：

- **色调**：颜色的底色。这个值在 0 和 360 之间，表示 color wheel (en-US) 周围的角度。
- **饱和度**：颜色有多饱和？它的值为 0–100%，其中 0 为无颜色（它将显示为灰色阴影），100% 为全色饱和度
- **亮度**：颜色有多亮？它从 0–100% 中获取一个值，其中 0 表示没有光（它将完全显示为黑色），100% 表示完全亮（它将完全显示为白色）

我们可以更新 RGB 的例子来使用 HSL 颜色，就像这样：

就像 `rgb()` 一样，你可以向 `hsl()` 传递一个 alpha 参数来指定不透明度。

**备注：** 在旧版本的 CSS 中， `hsl()` 语法不支持 alpha 参数——你需要使用一个叫做 `hsla()` 的不同函数来实现。现在你可以向 `hsl()` 传递一个 alpha 参数，但为了向后兼

> 容老网站， `hsla()` 语法仍然被支持，并且具有与 `hsl()` 完全相同的行为。

你可以在项目中使用这些颜色值中的任何一个。对于大多数项目，你可能会选择一个调色板，然后在整个项目中使用这些颜色——以及你所选择的定义这些颜色的方法。你可以混合使用不同的颜色模型，但是为了一致性，通常最好是你的整个项目使用相同的一个！

## 图片

`<image>` 数据类型用于图像为有效值的任何地方。它可以是一个通过 `url()` 函数指向的实际图像文件，也可以是一个渐变。

在下面的例子中，我们演示了一个图像和一个渐变作为 CSS `background-image` 属性的值。

**备注：** `<image>` 还有一些其他可能的值，但是这些都是较新的，并且目前对浏览器的支

> 持很差。如果你想了解 `<image>` 数据类型，请查看 MDN 页面。

## 位置

`<position>` 数据类型表示一组 2D 坐标，用于定位一个元素，如背景图像（通过 `background-position`）。它可以使用关键字（如 `top`、`left`、`bottom`、`right` 以及 `center`）将元素与 2D 框的特定边界对齐，以及表示框的顶部和左侧边缘偏移量的长度。

一个典型的位置值由两个值组成——第一个值水平地设置位置，第二个值垂直地设置位置。如果只指定一个轴的值，另一个轴将默认为 `center`。

在下面的示例中，我们使用关键字将背景图像从容器的顶部到右侧放置了 40px。

**尝试使用这些值，看看如何把这些图像移来移去。**

## 字符串和标识符

在上面的示例中，我们看到关键字被用作值的地方（例如 `<color>` 关键字，如 `red` 、 `black` 、 `rebeccapurple` 和 `goldenrod` ）。这些关键字被更准确地描述为标识符，一个 CSS 可以理解的特

殊值。因此它们没有使用引号括起来——它们不被当作字符串。

在某些地方可以使用 CSS 中的字符串，例如在指定生成的内容时。在本例中，引用该值以证明它是一个字符串。在下面的示例中，我们使用非引号括起来的颜色关键字和引号括起来的内容字符串。

## 函数

我们将查看的最后一种类型的值是一组称为函数的值。在编程中，函数是一段可重用的代码，可以多次运行，以完成重复的任务，对开发人员和计算机都是如此。函数通常与 JavaScript、Python 或 C++ 等语言相关联，但它们也以属性值的形式存在于 CSS 中。我们已经在颜色部分看到了函数的作用——`rgb()`、`hsl()` 等。用于从文件返回图像的值——`url()`——也是一个函数。

`calc()` CSS 函数的行为更像你在传统编程语言中可能找到的东西。这个函数使你能够在 CSS 中进行简单的计算。如果你想计算一些你在编写项目的 CSS 时无法定义的数值，并且需要浏览器在运行时为你计算，那么它特别有用。

例如，下面我们使用 `calc()` 使框宽为 `20% + 100px`。20% 是根据父容器 `.wrapper` 的宽度来计算的，因此如果宽度改变，它也会改变。我们不能事先做这个计算，因为我们不知道父类的 20% 是多少，所以我们使用 `calc()` 来告诉浏览器为我们做这个计算。



## 技能测试！

你已经到了本文的结尾，但你能记住其中重要的信息吗？你可以在继续前进之前进行一些测试来验证你是否记住了这些内容——技能测试：值和单位 (en-US)。

## 总结

本文简要介绍了你可能会遇到的最常见的值和单位类型。你可以看看所有不同类型的 CSS 的值和单位参考页面；当你学习这些课程时，你将会遇到很多这样的情况。

需要记住的关键一点是，每个属性都有一个已定义的允许值列表，每个值都有一个定义来解释子值是什么。然后你可以在 MDN 上查看详细信息。

例如，理解 `<image>` 还允许你创建一个颜色渐变有意义的，但也许这个章节并不会提供太多明显的相关知识！