

<https://www.runoob.com/html/html-tutorial.html>

HTML 教程- (HTML5 标准)

超文本标记语言（英语：HyperText Markup Language，简称：HTML）是一种用于创建网页的标准标记语言。

您可以使用 HTML 来建立自己的 WEB 站点，HTML 运行在浏览器上，由浏览器来解析。

在本教程中，您将学习如何使用 HTML 来创建站点。

HTML 很容易学习！相信您能很快学会它！

HTML 实例

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>菜鸟教程(runoob.com)</title>
```

```
</head>
```

```
<body>
```

```
  <h1>我的第一个标题</h1>
```

<p>我的第一个段落。</p>

</body>

</html>

本教程包含了数百个 HTML 实例。

注意：对于中文网页需要使用 **<meta charset="utf-8">** 声明编码，否则会出现乱码。有些浏览器(如 360 浏览器)会设置 GBK 为默认编码，则需要设置为 **<meta charset="gbk">**。

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body> <h1>我的  
第一个标题</h1> <p>我的第一个段落。</p> </body> </html>
```

HTML 文档的后缀名

- **.html**
- **.htm**

以上两种后缀名没有区别，都可以使用。

[开始学习 HTML!](#)

HTML 实例

在 HTML 手册中包含了数百个在线实例，您可以在线编辑并查看运行结果。

[查看 HTML 实例!](#)

HTML 参考手册

在菜鸟教程中，我们提供了完整的 HTML 参考手册，其中包括标签、属性、颜色、实体等等。

[HTML 参考手册](#)

HTML/CSS/JS 在线工具

HTML/CSS/JS 在线工具可以在线编辑 HTML、CSS、JS 代码，并实时查看效果，你也可以将优质代码保存分享：

<https://www.jyshare.com/front-end/61>HTML/CSS/JS 在线工具可以在线编辑 HTML、CSS、JS 代码，并实时查看效果，你也可以将优质代码保存分享：<https://www.jyshare.com/front-end/61>

HTML 简介

HTML 实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body> <h1>我的  
第一个标题</h1> <p>我的第一个段落。</p> </body> </html>
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>菜鸟教程(runoob.com)</title>
</head>
<body>

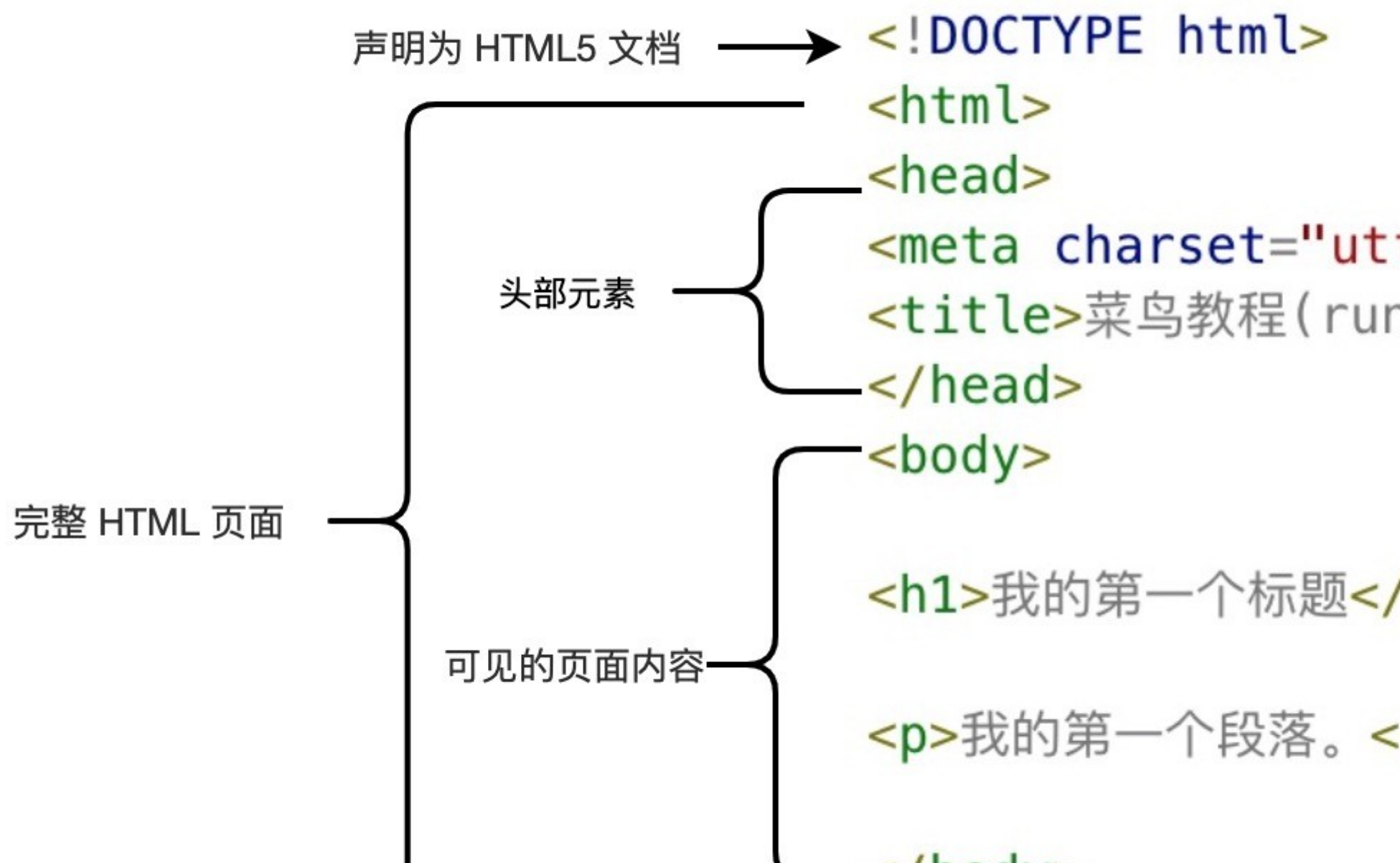
<h1>我的第一个标题</h1>

<p>我的第一个段落。</p>

</body>
</html>
```

尝试一下 »

实例解析



- **<!DOCTYPE html>** 声明为 HTML5 文档
- **<html>** 元素是 HTML 页面的根元素
- **<head>** 元素包含了文档的元（meta）数据，如 **<meta charset="utf-8">** 定义网页编码格式为 **utf-8**。
- **<title>** 元素描述了文档的标题
- **<body>** 元素包含了可见的页面内容
- **<h1>** 元素定义一个大标题
- **<p>** 元素定义一个段落

注：在浏览器的页面上使用键盘上的 **F12** 按键开启调试模式，就可以看到组成标签。

Elements Recorder Console Sources Performance insights

```
<!DOCTYPE html>
<html>
... ▶ <head> ... </head> == $0
▼ <body style>
  <!-- 头部 -->
  ▶ <div class="container logo-search"> ... </div>
  <!-- 导航栏 -->
  <!-- 导航栏 -->
  ▶ <div class="container navigation"> ... </div>
  ▶ <div class="container sub-navigation sub-navigation-articles" style="display
  <!-- 内容 -->
  ▼ <div class="container main">
    <!-- 中间 -->
    ▼ <div class="row">
```

html head

什么是 HTML?

HTML 是用来描述网页的一种语言。

- HTML 指的是超文本标记语言: **HyperText Markup Language**
- HTML 不是一种编程语言，而是一种**标记语言**
- 标记语言是一套**标记标签** (markup tag)
- HTML 使用标记标签来**描述**网页
- HTML 文档包含了 HTML **标签**及**文本**内容
- HTML 文档也叫做 **web 页面**

HTML 标签

HTML 标记标签通常被称为 HTML 标签 (HTML tag)。

- HTML 标签是由尖括号包围的关键词，比如 <html>
- HTML 标签通常是**成对出现**的，比如 和
- 标签对中的第一个标签是**开始标签**，第二个标签是**结束标签**
- 开始和结束标签也被称为**开放标签**和**闭合标签**

<标签>内容</标签>

HTML 元素

"HTML 标签" 和 "HTML 元素" 通常都是描述同样的意思.

但是严格来讲, 一个 HTML 元素包含了开始标签与结束标签, 如下实例:

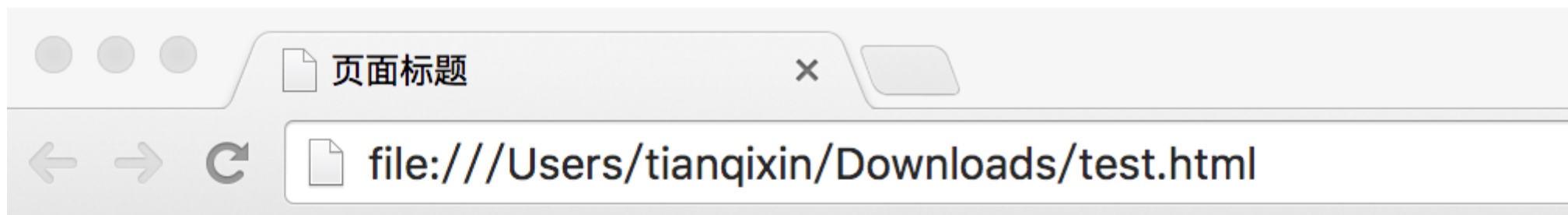
HTML 元素:

`<p>`这是一个段落。`</p>`

Web 浏览器

Web 浏览器（如谷歌浏览器，Internet Explorer，Firefox，Safari）是用于读取 HTML 文件，并将其作为网页显示。

浏览器并不是直接显示的 HTML 标签，但可以使用标签来决定如何展现 HTML 页面的内容给用户：



我的第一个标题

我的第一个段落。

HTML 网页结构

下面是一个可视化的 HTML 页面结构：

```
<html>
<head>
<title>页面标题</title>
</head>
<body>
<h1>这是一个标题</h1>
<p>这是一个段落。</p>
<p>这是另外一个段落。</p>
</body>
</html>
```



只有 `<body>` 区域 (白色部分) 才会在浏览器中显示。

HTML 版本

从初期的网络诞生后，已经出现了许多 HTML 版本：

版本	发布时间
HTML	1991

HTML+	1993
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML 1.0	2000
HTML5	2012
XHTML5	2013

<!DOCTYPE> 声明

<!DOCTYPE>声明有助于浏览器中正确显示网页。

网络上有很多不同的文件，如果能够正确声明 **HTML** 的版本，浏览器就能正确显示网页内容。

doctype 声明是不区分大小写的，以下方式均可：

```
<!DOCTYPE html>
```

```
<!DOCTYPE HTML>
```

```
<!doctype html>
```

```
<!Doctype Html>
```

通用声明

HTML5

```
<!DOCTYPE html>
```

HTML 4.01

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

XHTML 1.0

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

查看完整网页声明类型 [DOCTYPE 参考手册](#)。

中文编码

目前在大部分浏览器中，直接输出中文会出现中文乱码的情况，这时候我们就需要在头部将字符声明为 UTF-8 或 GBK。

HTML 实例

```
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> <title> 页面标题</title> </head> <body> <h1>我的第一个标题</h1> <p>我的第一个段落。</p> </body> </html>
```

尝试一下 »

[HTML 教程](#)

[HTML 编辑器](#)

2 篇笔记 写笔记

1. TnT
322***8263@qq.com

6575

1. doctype 声明是不区分大小写的，用来告知 Web 浏览器页面使用了哪种 HTML 版本。

在 HTML 4.01 中，<!DOCTYPE> 声明需引用 DTD（文档类型声明），因为 HTML 4.01 是基于 SGML（Standard Generalized Markup Language 标准通用标记语言）。

HTML 4.01 规定了三种不同的 <!DOCTYPE> 声明，分别是：Strict、Transitional 和 Frameset。

HTML5 不是基于 SGML，因此不要求引用 DTD。

2. 对于中文网页需要使用 <meta charset="utf-8"> 声明编码，否则会出现乱码。有些浏览器(如 360 浏览器)会设置 GBK 为默认编码，则你需要设置为 <meta charset="gbk">。

目前在大部分浏览器中，直接输出中文会出现中文乱码的情况，这时候需要在头部将字符声明为 UTF-8。

TnT

TnT
322***8263@qq.com

5 年前 (2018-11-16)

2. 东方不败
ren***i@qq.com

1337

对于设置 `<meta charset="utf-8" />` 后出现网页乱码问题，其实归根到底就是：你通过 meta 标签设置的编码和网页文件在保存时所使用的文档编码不相同造成的！

至于有的人说什么 360 浏览器默认 GBK 会造成乱码，我只想说的是，

只要你在 html 文件里写了 `<!doctype hmtl>`和 `<meta charset="utf-8" />`，浏览器就绝对会按照 **utf-8** 编码解析网页，没有第二种可能！再次重点说明：保存 html 文件时，文档编码和 meta 设置的编码，一定要相同，只要不相同，就一定会出现乱码！

之所以一定要写上 `<!doctype html>`，就是为了防止浏览器的怪异模式，强制浏览器按照标准模式渲染网页！

东方不败

东方不败

ren**i@qq.com

2 年前 (2022-09-05)

HTML 编辑器

HTML 编辑器推荐

可以使用专业的 HTML 编辑器来编辑 HTML，菜鸟教程为大家推荐几款常用的编辑器：

- VS Code: <https://code.visualstudio.com/>
- Sublime Text: <http://www.sublimetext.com/>
- 在线编辑器: <https://www.jyshare.com/front-end/61/>

你可以从以上软件的官网中下载对应的软件，按步骤安装即可。

接下来我们将为大家演示如何使用 VS Code 工具来创建 HTML 文件。

VS Code

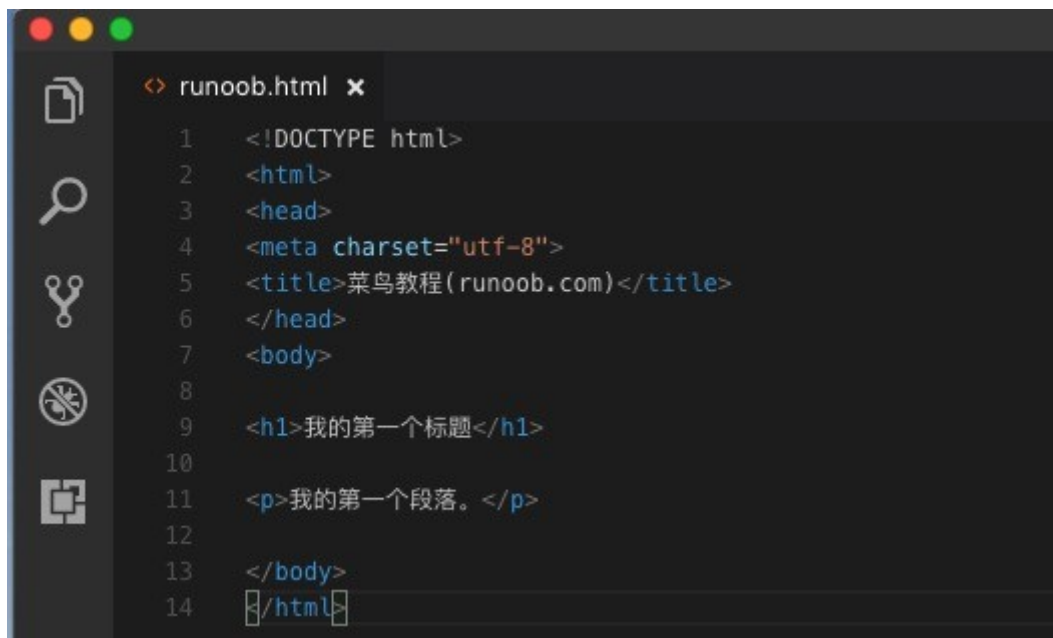
Visual Studio Code（简称 VS Code）是一个由微软开发，同时支持 Windows、Linux 和 macOS 等操作系统且开放源代码的代码编辑器，编辑器中内置了扩展程序管理的功能。

VS Code 安装教程参考: <https://www.runoob.com/w3cnote/vscode-tutorial.html>

步骤 1: 新建 HTML 文件

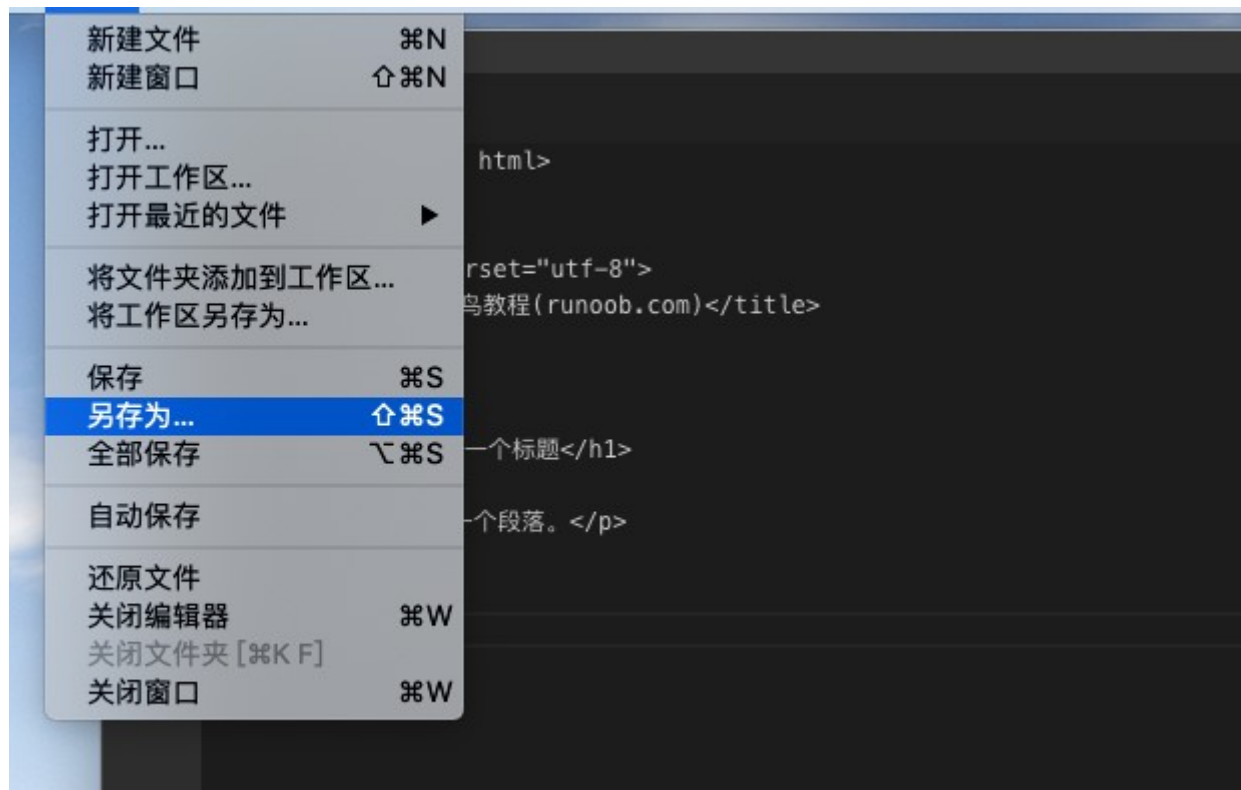
在 VS Code 安装完成后, 选择" 文件(F)->新建文件(N) ", 在新建的文件中输入以下代码:

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body> <h1>我的  
第一个标题</h1> <p>我的第一个段落。</p> </body> </html>
```



步骤 2: 另存为 HTML 文件

然后选择" 文件(F)->另存为(A) ", 文件名为 runoob.html:



当您保存 HTML 文件时，既可以使用 `.htm` 也可以使用 `.html` 扩展名。两者没有区别，完全根据您的喜好，我建议统一用 `.html`。在一个容易记忆的文件夹中保存这个文件，比如 `runoob`

步骤 3: 在浏览器中运行这个 HTML 文件

然后鼠标右击编辑器上的文件名，选择在默认浏览器打开（也可以其他的浏览器）：

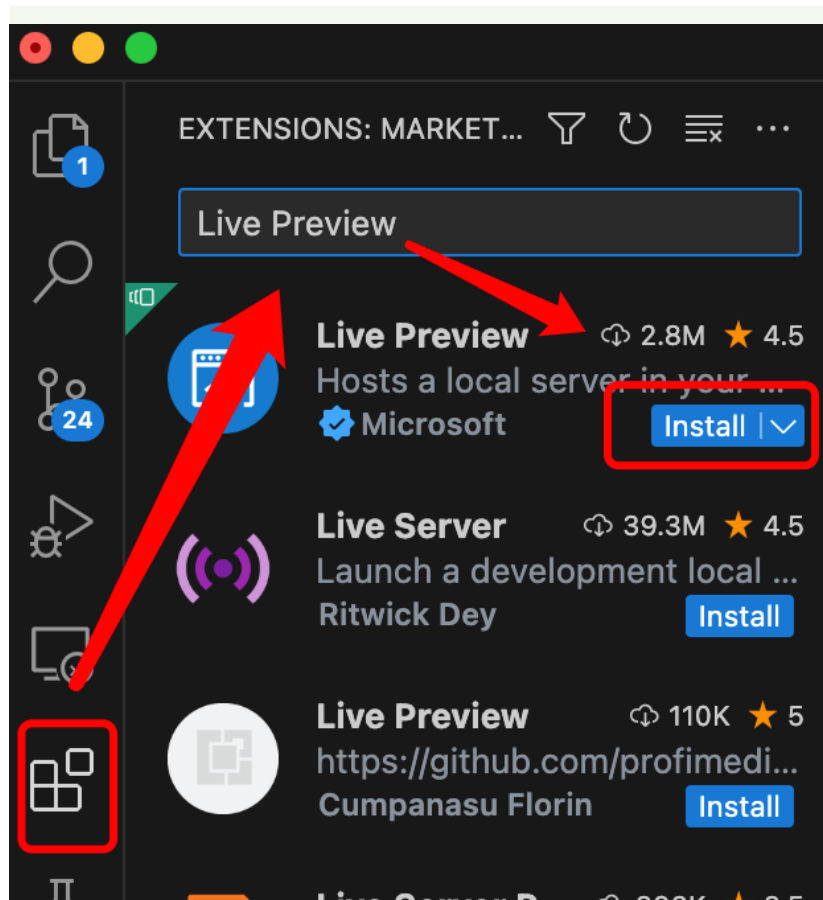


注：vscode 中使用浏览器打开 html 文件需要 安装 "**open in browser**" 扩展。

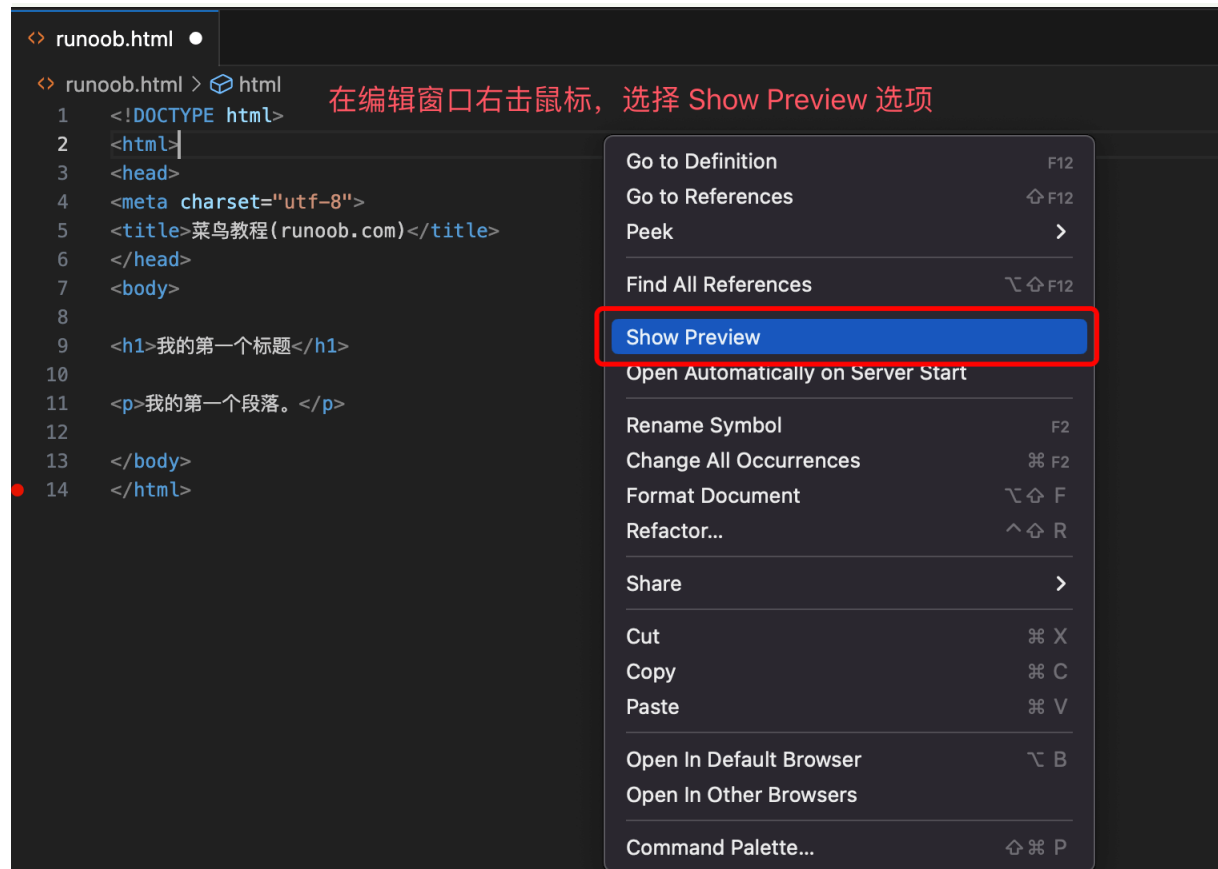
运行显示结果类似如下：



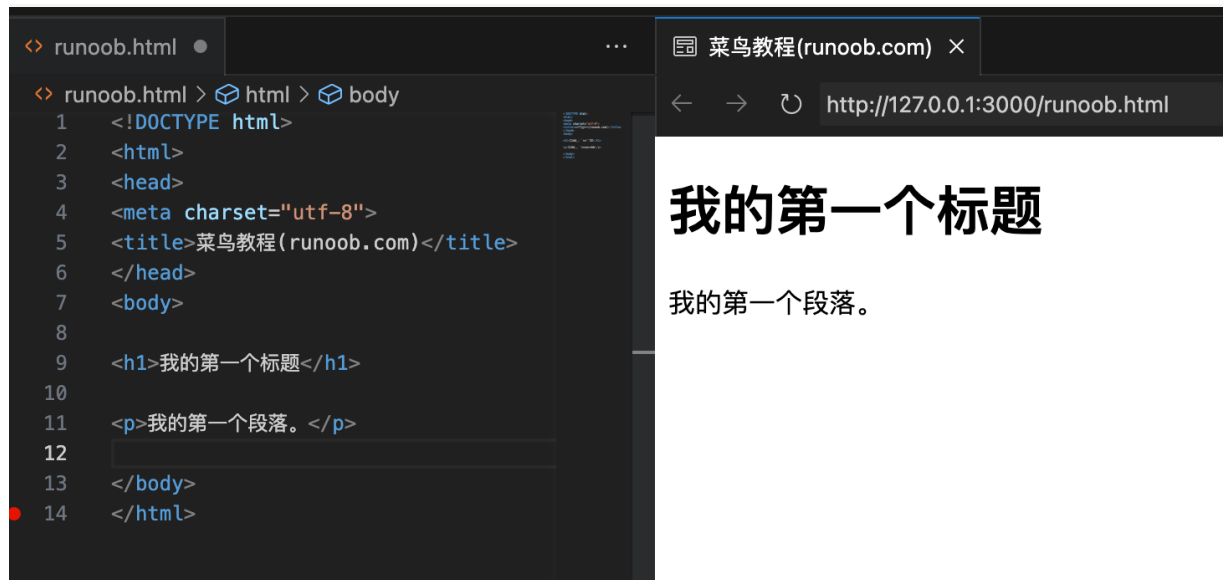
VS Code 可以安装 **Live Preview** 插件来实时预览编写的代码:



在编辑窗口右击鼠标，选择 **Show Preview** 选项：



显示结果:



这样我们就可以在编辑代码过程中实时预览到效果了。

[HTML 简介](#)

[HTML 基础](#)

2 篇笔记 写笔记

1. 祖骁先生 TM
703***363@qq.com

2448

每一种操作系统都带有简单的文本编辑器：

- o Windows 用户可以使用记事本；

- o Linux 用户可以选择几种不同的文本编辑器, 如 vi、vim 或者 emacs ;
- o Mac 用户可以使用 OS X 预装的 TextEdit。

祖骁先生 TM

祖骁先生 TM

703***363@qq.com

7 年前 (2017-03-12)

2. 姒虞

TR1***8401919@163.com

1536

VS Code 怎样改成中文版?

首先点击扩展 (也就是四个小方格), 在搜索栏内输入 **Chinese**。

点击 install, 下载安装简体中文, 在右下方会出现 restart 点击就好。

再重启 vs 后就会变成简体中文。



姒虞

姒虞

TR1***8401919@163.com

3 年前 (2021-03-28)

HTML 基础- 4 个实例

不要担心本章中您还没有学过的例子。

您将在下面的章节中学到它们。

HTML 标题

HTML 标题（Heading）是通过<h1> - <h6> 标签来定义的。

实例

```
<h1>这是一个标题</h1> <h2>这是一个标题</h2> <h3>这是一个标题</h3>
```

尝试一下 »

HTML 段落

HTML 段落是通过标签 <p> 来定义的。

实例

```
<p>这是一个段落。</p> <p>这是另外一个段落。</p>
```

尝试一下 »

HTML 链接

HTML 链接是通过标签 <a> 来定义的。

实例

```
<a href="https://www.runoob.com">这是一个链接</a>
```

[尝试一下 »](#)

提示:在 href 属性中指定链接的地址。

(您将在本教程稍后的章节中学习更多有关属性的知识)。

HTML 图像

HTML 图像是通过标签 来定义的.

实例

```

```

[尝试一下 »](#)

注意： 图像的名称和尺寸是以属性的形式提供的。

[HTML 编辑器](#)

[HTML 元素](#)

HTML 元素

HTML 文档由 HTML 元素定义。

HTML 元素

<="" p="">

开始标签 *	元素内容	结束标签 *
<p>	这是一个段落	</p>
	这是一个链接	
 	换行	

*开始标签常被称为**起始标签（opening tag）**，结束标签常称为**闭合标签（closing tag）**。

HTML 元素语法

- HTML 元素以**开始标签**起始
- HTML 元素以**结束标签**终止
- **元素的内容**是开始标签与结束标签之间的内容
- 某些 HTML 元素具有**空内容（empty content）**
- 空元素在**开始标签中进行关闭**（以开始标签的结束而结束）
- 大多数 HTML 元素可拥有**属性**

注释: 您将在本教程的下一章中学习更多有关属性的内容。

嵌套的 HTML 元素

大多数 HTML 元素可以嵌套（HTML 元素可以包含其他 HTML 元素）。

HTML 文档由相互嵌套的 HTML 元素构成。

HTML 文档实例

```
<!DOCTYPE html>
<html>

<body>
<p>这是第一个段落。</p>
</body>

</html>
```

以上实例包含了三个 HTML 元素。

HTML 实例解析

<p> 元素:

```
<p>这是第一个段落。</p>
```

这个 <p> 元素定义了 HTML 文档中的一个段落。

这个元素拥有一个开始标签 <p> 以及一个结束标签 </p>.

元素内容是: 这是第一个段落。

<body> 元素:

```
<body>
<p>这是第一个段落。</p>
</body>
```

`<body>` 元素定义了 HTML 文档的主体。

这个元素拥有一个开始标签 `<body>` 以及一个结束标签 `</body>`。

元素内容是另一个 HTML 元素（p 元素）。

`<html>` 元素：

```
<html>
```

```
<body>
```

```
<p>这是第一个段落。</p>
```

```
</body>
```

```
</html>
```

`<html>` 元素定义了整个 HTML 文档。

这个元素拥有一个开始标签 `<html>`，以及一个结束标签 `</html>`。

元素内容是另一个 HTML 元素（body 元素）。

不要忘记结束标签

即使您忘记了使用结束标签，大多数浏览器也会正确地显示 HTML：

```
<p>这是一个段落
```

```
<p>这是一个段落
```

以上实例在浏览器中也能正常显示，因为关闭标签是可选的。

但不要依赖这种做法。忘记使用结束标签会产生不可预料的结果或错误。

HTML 空元素

没有内容的 HTML 元素被称为空元素。空元素是在开始标签中关闭的。

`
` 就是没有关闭标签的空元素（`
` 标签定义换行）。

在 XHTML、XML 以及未来版本的 HTML 中，所有元素都必须被关闭。

在开始标签中添加斜杠，比如 `
`，是关闭空元素的正确方法，HTML、XHTML 和 XML 都接受这种方式。

即使 `
` 在所有浏览器中都是有效的，但使用 `
` 其实是更长远的保障。

HTML 提示：使用小写标签

HTML 标签对大小写不敏感：`<P>` 等同于 `<p>`。许多网站都使用大写的 HTML 标签。

菜鸟教程使用的是小写标签，因为万维网联盟（W3C）在 HTML 4 中**推荐**使用小写，而在未来 (X)HTML 版本中**强制**使用小写。

[HTML 基础](#)

[HTML 属性](#)

1 篇笔记 写笔记

```
1. cover
905***968@qq.com
```

3262

1. 一些标签的使用，切记所有标签都需要闭合，不管是单体标签还是成对标签。（尽管目前浏览器是识别有些标签不闭合的情况，但是取的最好的保证兼容性，使用闭合）

2. 标签写法要用小写字母（有些版本对大小写可认为相同，而 xhtml 中强制使用小写）

cover

```
cover
905***968@qq.com
```

7 年前 (2017-04-21)

HTML 属性

属性是 HTML 元素提供的附加信息。

HTML 属性

- HTML 元素可以设置属性
 - 属性可以在元素中添加附加信息
 - 属性一般描述于开始标签
 - 属性总是以名称/值对的形式出现，比如：**name="value"**。
-

属性实例


HTML 链接由 <a> 标签定义。链接的地址在 **href 属性**中指定：

实例

```
<a href="http://www.runoob.com">这是一个链接</a>
```

[尝试一下 »](#)

HTML 属性常用引用属性值

属性值应该始终被包括在引号内。
双引号是最常用的，不过使用单引号也没有问题。
 **提示:** 在某些个别的情况下，比如属性值本身就含有双引号，那么您必须使用单引号，例如：
`name='John "ShotGun" Nelson'`

HTML 提示：使用小写属性

属性和属性值对大小写不敏感。
不过，万维网联盟在其 HTML 4 推荐标准中推荐小写的属性/属性值。
而新版本的 (X)HTML 要求使用小写属性。

HTML 属性参考手册

查看完整的 HTML 属性列表: [HTML 标签参考手册](#)。
下面列出了适用于大多数 HTML 元素的属性：

属性	描述
class	为 html 元素定义一个或多个类名（classname）(类名从样式文件引入)
id	定义元素的唯一 id
style	规定元素的行内样式（inline style）
title	描述了元素的额外信息 (作为工具条使用)

更多标准属性说明：[HTML 标准属性参考手册](#).

[HTML 元素](#)

HTML 标题

1 篇笔记 写笔记

1. gouer
wyl***@qq.com

3229

1. 属性和属性值，尽量小写，本来这样做也方便些。
2. class 属性可以多用 **class=" "**（引号里面可以填入多个 class 属性）
3. id 属性只能单独设置 **id=" "**（只能填写一个，多个无效）

gouer

gouer
wyl***@qq.com

7 年前 (2017-06-14)

HTML 标题

在 HTML 文档中，标题很重要。

HTML 标题

标题（Heading）是通过 `<h1>` - `<h6>` 标签进行定义的。

`<h1>` 定义最大的标题。`<h6>` 定义最小的标题。

实例

```
<h1>这是一个标题。</h1> <h2>这是一个标题。</h2> <h3>这是一个标题。</h3>
```

尝试一下 »

注释: 浏览器会自动地在标题的前后添加空行。

标题很重要

请确保将 HTML 标题 标签只用于标题。不要仅仅是为了生成**粗体**或**大号**的文本而使用标题。

搜索引擎使用标题为您的网页的结构和内容编制索引。

因为用户可以通过标题来快速浏览您的网页，所以用标题来呈现文档结构是很重要的。

应该将 h1 用作主标题（最重要的），其后是 h2（次重要的），再其次是 h3，以此类推。

HTML 水平线

<hr> 标签在 HTML 页面中创建水平线。

hr 元素可用于分隔内容。

实例

```
<p>这是一个段落。</p> <hr> <p>这是一个段落。</p> <hr> <p>这是一个段落。</p>
```

尝试一下 »

HTML 注释

可以将注释插入 HTML 代码中，这样可以提高其可读性，使代码更易被人理解。浏览器会忽略注释，也不会显示它们。

注释写法如下：

实例

```
<!-- 这是一个注释 -->
```

[尝试一下 »](#)

注释: 开始括号之后（左边的括号）需要紧跟一个叹号 **!** (英文标点符号), 结束括号之前（右边的括号）不需要, 合理地使用注释可以对未来的代码编辑工作产生帮助。

HTML 提示 - 如何查看源代码

你是否看过一些网页然后惊叹它是如何实现的。

如果您想找到其中的奥秘, 只需要单击右键, 然后选择"查看源文件" (IE) 或"查看页面源代码" (Firefox), 其他浏览器的做法也是类似的。这么做会打开一个包含页面 HTML 代码的窗口。



本站实例

[标题](#)

如何在 HTML 文档中显示标题。

[隐藏注释](#)

如何在 HTML 源代码中插入注释。

[水平线](#)

如何插入水平线。

HTML 标签参考手册

菜鸟教程的标签参考手册提供了有关这些标题及其属性的更多信息。
您将在本教程下面的章节中学到更多有关 HTML 标签和属性的知识。

标签	描述
<u><html></u>	定义 HTML 文档
<u><body></u>	定义文档的主体
<u><h1></u> - <u><h6></u>	定义 HTML 标题
<u><hr></u>	定义水平线
<u><!--...--></u>	定义注释

[HTML 属性](#)

[HTML 段落](#)

1 篇笔记 写笔记

1. stinkaroo
190***276@qq.com

2627

标题大小与字体大小的关系

1 到 6 号标题与 1 到 6 号字体逆序对应，比如 1 号字体对应 6 号标题，2 号字体对应 5 号标题。

```
<h1>这是 1 号标题</h1>
<font size="6">这是 6 号字体文本</font>

<h2>这是 2 号标题</h2>
<font size="5">这是 5 号字体文本</font>

<h3>这是 3 号标题</h3>
<font size="4">这是 4 号字体文本</font>

<h4>这是 4 号标题</h4>
<font size="3">这是 3 号字体文本</font>

<h5>这是 5 号标题</h5>
<font size="2">这是 2 号字体文本</font>

<h6>这是 6 号标题</h6>
<font size="1">这是 1 号字体文本</font>
```

尝试一下 »

[stinkaroo](#)

stinkaroo

190***276@qq.com

7 年前 (2017-05-03)

HTML 段落

HTML 可以将文档分割为若干段落。

HTML 段落

段落是通过 `<p>` 标签定义的。

实例

```
<p>这是一个段落 </p> <p>这是另一个段落</p>
```

尝试一下 »

注意：浏览器会自动地在段落的前后添加空行。（`</p>` 是块级元素）

不要忘记结束标签

即使忘了使用结束标签，大多数浏览器也会正确地将 HTML 显示出来：

实例

```
<p>这是一个段落 <p>这是另一个段落
```

尝试一下 »

上面的例子在大多数浏览器中都没问题，但不要依赖这种做法。忘记使用结束标签会产生意想不到的结果和错误。

注释：在未来的 HTML 版本中，不允许省略结束标签。

HTML 折行

如果您希望在不产生一个新段落的情况下进行换行（新行），请使用 `
` 标签：

实例

```
<p>这个<br>段落<br>演示了分行的效果</p>
```

[尝试一下 »](#)

`
` 元素是一个空的 HTML 元素。由于关闭标签没有任何意义，因此它没有结束标签。

HTML 输出- 使用提醒

我们无法确定 HTML 被显示的确切效果。屏幕的大小，以及对窗口的调整都可能导致不同的结果。

对于 HTML，您无法通过在 HTML 代码中添加额外的空格或换行来改变输出的效果。

当显示页面时，浏览器会移除源代码中多余的空格和空行。所有连续的空格或空行都会被算作一个空格。需要注意的是，HTML 代码中的所有连续的空行（换行）也被显示为一个空格。

[尝试一下](#)

（这个例子演示了一些 HTML 格式化方面的问题）



本站实例

[HTML 段落](#)

如何在浏览器中显示 HTML 段落。

[换行](#)

在 HTML 文档中使用换行。

[在 HTML 代码中的排版一首唐诗](#)

浏览器在显示 HTML 时，会省略源代码中多余的空白字符（空格或回车等）。

更多实例

[更多段落](#)

段落的默认行为。

HTML 标签参考手册

菜鸟教程的标签参考手册提供了有关 HTML 元素及其属性的更多信息。

标签	描述
<p>	定义一个段落

	插入单个折行（换行）

[HTML 标题](#)

[HTML 文本格式化](#)

1 篇笔记 写笔记

- phantom_47
122***2803@qq.com

2470

区分一下：**
**, **
** 以及 **
**（带有空格）

**
** 是 HTML 写法。

是 XHTML1.1 的写法, 也是 XML 写法。**
** 是 XHTML 为兼容 HTML 的写法,也是 XML 写法。HTML5 因为兼容 XHTML, 所以三种写法都可以使用。

早期发布的 HTML 规范当中，
 与 等元素是不用封闭自身的，但是这种元素造成了 HTML 规范的不严谨，于是在之后发布的 XHTML 语言中，参考了更为严谨的 XML 规范，在这些不用自身封闭的元素后加 / 来表示自行封闭，在逻辑上来讲等同于
....</br>（但是没有 </br> 这种写法），这样一来保证了较少的代码量，二来保证了规范的严谨。

[phantom_47](#)

phantom_47

122***2803@qq.com

6 年前 (2018-05-03)

HTML 文本格式化

HTML 文本格式化

加粗文本

斜体文本

电脑自动输出

这是_{下标}和^{上标}

尝试一下»

HTML 格式化标签

HTML 使用标签 ``("bold") 与 `<i>`("italic") 对输出的文本进行格式, 如: **粗体** or *斜体*
这些 HTML 标签被称为格式化标签 (请查看底部完整标签参考手册)。

通常标签 `` 替换加粗标签 `` 来使用, `` 替换 `<i>` 标签使用。



然而, 这些标签的含义是不同的:

`` 与 `<i>` 定义粗体或斜体文本。

`` 或者 `` 意味着你要呈现的文本是重要的, 所以要突出显示。现今所有主要浏览器都能渲染各种效果的字体。不过, 未来浏览器可能会支持更好的渲染效果。



在线实例

[文本格式化](#)

此例演示如何在一个 HTML 文件中对文本进行格式化

[预格式文本](#)

此例演示如何使用 `pre` 标签对空行和空格进行控制。

["计算机输出"标签](#)

此例演示不同的"计算机输出"标签的显示效果。

地址

此例演示如何在 HTML 文件中写地址。

缩写和首字母缩写

此例演示如何实现缩写或首字母缩写。

文字方向

此例演示如何改变文字的方向。

块引用

此例演示如何实现长短不一的引用语。

删除字效果和插入字效果

此例演示如何标记删除文本和插入文本。

HTML 文本格式化标签

标签	描述
<u></u>	定义粗体文本
<u></u>	定义着重文字
<u><i></u>	定义斜体字
<u><small></u>	定义小号字
<u></u>	定义加重语气
<u><sub></u>	定义下标字
<u><sup></u>	定义上标字

<u><ins></u>	定义插入字
<u></u>	定义删除字

HTML "计算机输出" 标签

标签	描述
<u><code></u>	定义计算机代码
<u><kbd></u>	定义键盘码
<u><samp></u>	定义计算机代码样本
<u><var></u>	定义变量
<u><pre></u>	定义预格式文本

HTML 引文, 引用, 及标签定义

标签	描述
<u><abbr></u>	定义缩写
<u><address></u>	定义地址
<u><bdo></u>	定义文字方向
<u><blockquote></u>	定义长的引用
<u><q></u>	定义短的引用语

<u><cite></u>	定义引用、引证
<u><dfn></u>	定义一个定义项目。

HTML 链接

HTML 使用超级链接与网络上的另一个文档相连。

HTML 中的链接是一种用于在不同网页之间导航的元素。

链接通常用于将一个网页与另一个网页或资源（如文档、图像、音频文件等）相关联。

链接允许用户在浏览网页时单击文本或图像来跳转到其他位置，从而实现网页之间的互联。



尝试一下 - 实例

[HTML 链接](#)

如何在 HTML 文档中创建链接。

(可以在本页底端找到更多实例)

HTML 超链接（链接）

HTML 使用标签 `<a>` 来设置超文本链接。

超链接可以是一个字，一个词，或者一组词，也可以是一幅图像，您可以点击这些内容来跳转到新的文档或者当前文档中的某个部分。

当您把鼠标指针移动到网页中的某个链接上时，箭头会变为一只小手。

在标签 `<a>` 中使用了 `href` 属性来描述链接的地址。

默认情况下，链接将以以下形式出现在浏览器中：

- 一个未访问过的链接显示为蓝色字体并带有下划线。
- 访问过的链接显示为紫色并带有下划线。
- 点击链接时，链接显示为红色并带有下划线。

注意：如果为这些超链接设置了 CSS 样式，展示样式会根据 CSS 的设定而显示。

HTML 链接语法

以下是 HTML 中创建链接的基本语法和属性：<a> 元素：创建链接的主要 HTML 元素是<a>（锚）元素。<a>元素具有以下属性：

- href：指定链接目标的 URL，这是链接的最重要属性。可以是另一个网页的 URL、文件的 URL 或其他资源的 URL。
- target（可选）：指定链接如何在浏览器中打开。常见的值包括 _blank（在新标签或窗口中打开链接）和 _self（在当前标签或窗口中打开链接）。
- title（可选）：提供链接的额外信息，通常在鼠标悬停在链接上时显示为工具提示。
- rel（可选）：指定与链接目标的关系，如 nofollow、noopener 等。

链接的 HTML 代码很简单，它类似这样：

```
<a href="url">链接文本</a>
```

href 属性描述了链接的目标。

实例

```
<a href="https://www.runoob.com/">访问菜鸟教程</a>
```

上面这行代码显示为：[访问菜鸟教程](#)

点击这个超链接会把用户带到菜鸟教程的首页。

提示："链接文本"不必一定是文本。图片或其他 HTML 元素都可以成为链接。

文本链接：最常见的链接类型是文本链接，它使用 <a> 元素将一段文本转化为可点击的链接，例如：

```
<a href="https://www.example.com">访问示例网站</a>
```

图像链接：您还可以使用图像作为链接。在这种情况下，<a> 元素包围着 元素。例如：

```
<a href="https://www.example.com">
  
</a>
```

锚点链接：除了链接到其他网页外，您还可以在同一页面内创建内部链接，这称为锚点链接。要创建锚点链接，需要在目标位置使用 <a> 元素定义一个标记，并使用#符号引用该标记。例如：

```
<a href="#section2">跳转到第二部分</a>
```

```
<!-- 在页面中的某个位置 -->
```

```
<a name="section2"></a>
```

下载链接：如果您希望链接用于下载文件而不是导航到另一个网页，可以使用 download 属性。例如：

```
<a href="document.pdf" download>下载文档</a>
```

HTML 链接 - target 属性

使用 target 属性，你可以定义被链接的文档在何处显示。

下面的这行会在新窗口打开文档：

实例

```
<a href="https://www.runoob.com/" target="_blank" rel="noopener noreferrer">访问菜鸟教程!</a>
```

尝试一下 »

HTML 链接- id 属性

id 属性可用于创建一个 HTML 文档书签。

提示: 书签不会以任何特殊方式显示，即在 HTML 页面中是不显示的，所以对于读者来说是隐藏的。

实例

在 HTML 文档中插入 ID:

```
<a id="tips">有用的提示部分</a>
```

在 HTML 文档中创建一个链接到"有用的提示部分(id="tips") ":

```
<a href="#tips">访问有用的提示部分</a>
```

或者，从另一个页面创建一个链接到"有用的提示部分(id="tips") ":

```
<a href="https://www.runoob.com/html/html-links.html#tips">
```

```
访问有用的提示部分</a>
```

基本的注意事项 - 有用的提示

注释: 请始终将正斜杠添加到子文件夹。假如这样书写链接: href="https://www.runoob.com/html", 就会向服务器产生两次 HTTP 请求。这是因为服务器会添加正斜杠到这个地址, 然后创建一个新的请求, 就像这样: href="https://www.runoob.com/html/"。



更多实例

[图片链接](#)

如何使用图片链接。

[在当前页面链接到指定位置](#)

如何使用书签

[跳出框架](#)

本例演示如何跳出框架，假如你的页面被固定在框架之内。

[创建电子邮件链接](#)

本例演示如何链接到一个邮件。（本例在安装邮件客户端程序后才能工作。）

[创建电子邮件链接 2](#)

本例演示更加复杂的邮件链接。

HTML 链接标签

标签	描述
<code><a></code>	定义一个超级链接

HTML `<head>`



查看在线实例

[<title> - 定义了 HTML 文档的标题](#)

使用 `<title>` 标签定义 HTML 文档的标题

[<base> - 定义了所有链接的 URL](#)

使用 <base> 定义页面中所有链接默认的连接目标地址。

[<meta> - 提供了 HTML 文档的 meta 标记](#)

使用 <meta> 元素来描述 HTML 文档的描述，关键词，作者，字符集等。

HTML <head> 元素

<head> 元素包含了所有的头部标签元素。在 <head>元素中你可以插入脚本（scripts），样式文件（CSS），及各种 meta 信息。可以添加在头部区域的元素标签为: <title>, <style>, <meta>, <link>, <script>, <noscript> 和 <base>。

HTML <title> 元素

<title> 标签定义了不同文档的标题。

<title> 在 HTML/XHTML 文档中是必需的。

<title> 元素:

- 定义了浏览器工具栏的标题
- 当网页添加到收藏夹时，显示在收藏夹中的标题
- 显示在搜索引擎结果页面的标题

一个简单的 HTML 文档:

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容..... </body> </html>
```

HTML <base> 元素

<base> 标签描述了基本的链接地址/链接目标，该标签作为 HTML 文档中所有的链接标签的默认链接：

实例

```
<head> <base href="http://www.runoob.com/images/" target="_blank"> </head>
```

HTML <link> 元素

<link> 标签定义了文档与外部资源之间的关系。

<link> 标签通常用于链接到样式表：

实例

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

HTML <style> 元素

<style> 标签定义了 HTML 文档的样式文件引用地址。

在<style> 元素中你也可以直接添加样式来渲染 HTML 文档：

实例

```
<head> <style type="text/css"> body { background-color:yellow; } p { color:blue } </style> </head>
```

HTML <meta> 元素

meta 标签描述了一些基本的元数据。

<meta> 标签提供了元数据.元数据也不显示在页面上，但会被浏览器解析。

META 元素通常用于指定网页的描述，关键词，文件的最后修改时间，作者，和其他元数据。

元数据可以使用于浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或其他 Web 服务。

<meta> 一般放置于 <head> 区域

<meta> 标签- 使用实例

为搜索引擎定义关键词:

```
<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript">
```

为网页定义描述内容:

```
<meta name="description" content="免费 Web & 编程 教程">
```

定义网页作者:

```
<meta name="author" content="Runoob">
```

每 30 秒钟刷新当前页面:

```
<meta http-equiv="refresh" content="30">
```

HTML <script> 元素

<script> 标签用于加载脚本文件，如：JavaScript。

<script> 元素在以后的章节中会详细描述。

HTML head 元素

| 标签 | 描述 |
|-------------------------------------|----------|
| <u><head></u> | 定义了文档的信息 |

| | |
|---------------------------------------|-------------------|
| <u><title></u> | 定义了文档的标题 |
| <u><base></u> | 定义了页面链接标签的默认链接地址 |
| <u><link></u> | 定义了一个文档和外部资源之间的关系 |
| <u><meta></u> | 定义了 HTML 文档中的元数据 |
| <u><script></u> | 定义了客户端的脚本文件 |
| <u><style></u> | 定义了 HTML 文档的样式文件 |

[HTML 链接](#)

[HTML CSS](#)

2 篇笔记 写笔记

1. 坚持就是胜利
175***4696@qq.com

2216

HTML<title>元素不仅可以显示文本，也可以在左侧显示 logo 等图片。

显示时，要将<link>标签放入<head>里。

举例：

```
<!doctype HTML>
<html>
<head>
<link rel="shortcut icon" href="图片url">
```

```
<title>这是一个带图片的标签</title>
</head>
<body>
.....
.....
.....
</body>
</html>
```

坚持就是胜利

坚持就是胜利

175***4696@qq.com

6 年前 (2018-07-29)

2. CHINA 陈

252***4754@qq.com

1101

head 标签和 header 标签的不同

head 标签用于定义文档头部，它是所有头部元素的容器。<head> 中的元素可以引用脚本、指示浏览器在哪里找到样式表、提供元信息等等。

如：

```
<html>
  <head>
    <title>文档标题</title>
  </head>
</html>
```

header 标签用于定义文档的页眉（介绍信息）。

如:

```
<html>
  <body>
    <header>
      <p>段落</p>
      <h1>一级标题</h1>
    </header>
  </body>
</html>
```

注意千万不要弄混。

[CHINA 陈](#)

CHINA 陈

252***4754@qq.com

5 年前 (2019-09-22)

HTML 样式- CSS

CSS (Cascading Style Sheets) 用于渲染 HTML 元素标签的样式。

Look! Styles and colors

M a n i p u l a t e T e x t

C o l o r s , **B o x e s**

and more...

尝试一下 »



尝试一下 - 实例

[HTML 使用样式](#)

本例演示如何使用添加到 <head> 部分的样式信息对 HTML 进行格式化。

[本例演示如何使用样式属性做一个没有下划线的链接。](#)

如何使用 style 属性制作一个没有下划线的链接。

[链接到一个外部样式表](#)

本例演示如何 标签链接到一个外部样式表。

如何使用 CSS

CSS 是在 HTML 4 开始使用的,是为了更好的渲染 HTML 元素而引入的.

CSS 可以通过以下方式添加到 HTML 中:

- 内联样式- 在 HTML 元素中使用"style" 属性
- 内部样式表 -在 HTML 文档头部 <head> 区域使用<style> 元素 来包含 CSS
- 外部引用 - 使用外部 **CSS 文件**

最好的方式是通过外部引用 CSS 文件.

在本站的 HTML 教程中我们使用了内联 CSS 样式来介绍实例, 这是为了简化的例子, 也使得你能更容易在线编辑代码并在线运行实例。

你可以通过本站的 [CSS 教程](#) 学习更多的 CSS 知识。

内联样式

当特殊的样式需要应用到个别元素时，就可以使用内联样式。使用内联样式的方法是在相关的标签中使用样式属性。样式属性可以包含任何 CSS 属性。以下实例显示出如何改变段落的颜色和左外边距。

```
<p style="color:blue;margin-left:20px;">这是一个段落。</p>
```

学习更多样式，请访问 [CSS 教程](#)。

HTML 样式实例 - 背景颜色

背景色属性（background-color）定义一个元素的背景颜色：

实例

```
<body style="background-color:yellow;"> <h2 style="background-color:red;">这是一个标题</h2> <p style="background-color:green;">这是一个段落。</p> </body>
```

[尝试一下 »](#)

早期背景色属性（background-color）是使用 bgcolor 属性定义。

[尝试一下: 旧版 HTML 来设置背景方式](#)

HTML 样式实例 - 字体, 字体颜色， 字体大小

我们可以使用 font-family（字体）， color（颜色）， 和 font-size（字体大小）属性来定义字体的样式：

实例

```
<h1 style="font-family:verdana;">一个标题</h1> <p style="font-family:arial;color:red;font-size:20px;">一个段落。</p>
```

[尝试一下 »](#)

现在通常使用 font-family（字体）， color（颜色）， 和 font-size（字体大小）属性来定义文本样式，而不是使用标签。

HTML 样式实例 - 文本对齐方式

使用 text-align（文字对齐）属性指定文本的水平与垂直对齐方式：

实例

```
<h1 style="text-align:center;">居中对齐的标题</h1> <p>这是一个段落。</p>
```

尝试一下 »

文本对齐属性 text-align 取代了旧标签 <center> 。

[尝试一下](#)

内部样式表

当单个文件需要特别样式时，就可以使用内部样式表。你可以在<head> 部分通过 <style>标签定义内部样式表：

```
<head>
<style type="text/css">
body {background-color:yellow;}
p {color:blue;}
</style>
</head>
```

外部样式表

当样式需要被应用到很多页面的时候，外部样式表将是理想的选择。使用外部样式表，你可以通过更改一个文件来改变整个站点的外观。

```
<head>
```

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

HTML 样式标签

标签	描述
<style>	定义文本样式
<link>	定义资源引用地址

已弃用的标签和属性

在 HTML 4, 原来支持定义 HTML 元素样式的标签和属性已被弃用。这些标签将不支持新版本的 HTML 标签。

不建议使用的标签有: , <center>, <strike>

不建议使用的属性: color 和 bgcolor.

[HTML 头部](#)

[HTML 图像](#)

2 篇笔记 写笔记

1. LPW5806
289***1083@qq.com

CSS 修饰标签的样式，有 "内联" 和 "外引" 两种方式。

对于大部分标签，以上两种方法均可，且修改父级标签，子级标签特性也会改变。但某些标签确无法通过修改父级标签来改变子级标签特性，如 a 标签，修改其颜色特性，必须直接修改 a 标签的特性才可。

实例：

```
<a href="#" style="color:red;" rel="nofollow ugc">只能使用"内联"方式</a>
```

更多内容参考：[HTML 中引入 CSS 的方式](#)

[LPW5806](#)

LPW5806

289***1083@qq.com

6 年前 (2017-11-10)

2. 际起航 jiqih

498***456@qq.com

CSS 引入外部标签

第一种方式：使用最多，稳定

```
<link rel="stylesheet" href="标签路径">
```

第二种方式：

```
<style>
@import url("标签路径")
</style>
```

扩展知识点：link 和 import 之间的区别？

差别 1：

本质的差别：**link** 属于 XHTML 标签，而 **@import** 完全是 CSS 提供的一种方式。

差别 2：

加载顺序的差别： 当一个页面被加载的时候（就是被浏览者浏览的时候），link 引用的 CSS 会同时被加载，而 **@import** 引用的 CSS 会等到页面全部被下载完再被加载。所以有时候浏览 **@import** 加载 CSS 的页面时开始会没有样式(就是闪烁)，网速慢的时候还挺明显。

差别 3：

兼容性的差别： **@import** 是 CSS2.1 提出的，所以老的浏览器不支持，**@import** 只有在 IE5 以上的才能识别，而 **link** 标签无此问题。

差别 4：

使用 dom(document object model 文档对象模型)控制样式时的差别：当使用 javascript 控制 dom 去改变样式的时候，只能使用 link 标签，因为**@import** 不是 dom 可以控制的。

[际起航 jiqih](#)

际起航 jiqih

498***456@qq.com

2 年前 (2022-04-30)

HTML 图像

实例

Norwegian Mountain Trip



尝试一下 »



在线实例

插入图像

本例演示如何在网页中显示图像。

从不同的位置插入图片

本例演示如何将其他文件夹或服务器的图片显示到网页中。

（可以在本页底端找到更多实例。）

HTML 图像- 图像标签（）和源属性（Src）

在 HTML 中，图像由 标签定义。

 是空标签，意思是说，它只包含属性，并且没有闭合标签。

要在页面上显示图像，你需要使用源属性（src）。src 指 "source"。源属性的值是图像的 URL 地址。

定义图像的语法是：

```

```

URL 指存储图像的位置。如果名为 "pulpit.jpg" 的图像位于 www.runoob.com 的 images 目录中，那么其 URL 为 <http://www.runoob.com/images/pulpit.jpg>。

浏览器将图像显示在文档中图像标签出现的地方。如果你将图像标签置于两个段落之间，那么浏览器会首先显示第一个段落，然后显示图片，最后显示第二段。

HTML 图像- Alt 属性

alt 属性用来为图像定义一串预备的可替换的文本。

替换文本属性的值是用户定义的。

```

```

在浏览器无法载入图像时，替换文本属性告诉读者她们失去的信息。此时，浏览器将显示这个替代性的文本而不是图像。为页面上的图像都加上替换文本属性是个好习惯，这样有助于更好的显示信息，并且对于那些使用纯文本浏览器的人来说是非常有用的。

HTML 图像- 设置图像的高度与宽度

height（高度）与 width（宽度）属性用于设置图像的高度与宽度。

属性值默认单位为像素：

```

```

提示：指定图像的高度和宽度是一个很好的习惯。如果图像指定了高度宽度，页面加载时就会保留指定的尺寸。如果没有指定图片的大小，加载页面时有可能会破坏 HTML 页面的整体布局。

基本的注意事项 - 有用的提示：

注意: 假如某个 HTML 文件包含十个图像，那么为了正确显示这个页面，需要加载 11 个文件。加载图片是需要时间的，所以我们的建议是：慎用图片。

注意: 加载页面时，要注意插入页面图像的路径，如果不能正确设置图像的位置，浏览器无法加载图片，图像标签就会显示一个破碎的图片。



更多实例

[排列图片](#)

本例演示如何在文字中排列图像。

[浮动图像](#)

本例演示如何使图片浮动至段落的左边或右边。

[设置图像链接](#)

本例演示如何将图像作为一个链接使用。

[创建图像映射](#)

本例显示如何创建带有可供点击区域的图像地图。其中的每个区域都是一个超级链接。

HTML 图像标签

标签	描述
	定义图像
<map>	定义图像地图

<area>	定义图像地图中的可点击区域
------------------------------	---------------

[HTML CSS](#)

[HTML 表格](#)

2 篇笔记 写笔记

1. Jan
jkl***8@qq.com

1440

示例"创建图像映射"中的代码:

```
<map name="planetmap">  
  <area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm">  
  <area shape="circle" coords="90,58,3" alt="Mercury" href="mercur.htm">  
  <area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm">  
</map>
```

该段代码中的 **shape** 指的是点击区域的形状, **coords** 指的应该是链接区域在图片中的坐标 (像素为单位)

[Jan](#)

Jan
jkl***8@qq.com

7 年前 (2017-06-03)

2. 乖乖游文武
869***014@qq.com
[参考地址](#)

1、矩形: (左上角顶点坐标为(x1,y1), 右下角顶点坐标为(x2,y2))

```
<area shape="rect" coords="x1,y1,x2,y2" href=url>
```

2、圆形: (圆心坐标为(X1,y1), 半径为 r)

```
<area shape="circle" coords="x1,y1,r" href=url>
```

3、多边形: (各顶点坐标依次为(x1,y1)、(x2,y2)、(x3,y3))

```
<area shape="poly" coords="x1,y1,x2,y2 ..... " href=url>
```

[乖乖游文武](#)

乖乖游文武

869***014@qq.com

[参考地址](#)

7 年前 (2017-08-01)

HTML 表格

HTML 表格由 `<table>` 标签来定义。

HTML 表格是一种用于展示结构化数据的标记语言元素。

每个表格均有若干行（由 `<tr>` 标签定义），每行被分割为若干单元格（由 `<td>` 标签定义），表格可以包含标题行（`<th>`）用于定义列的标题。

- **tr**: tr 是 table row 的缩写，表示表格的一行。
- **td**: td 是 table data 的缩写，表示表格的数据单元格。
- **th**: th 是 table header 的缩写，表示表格的表头单元格。

数据单元格可以包含文本、图片、列表、段落、表单、水平线、表格等等。

HTML 表格生成器: <https://www.jyshare.com/front-end/7688/>。

以下是一个简单的 HTML 表格实例:

实例

```
<table>
  <thead>
    <tr>
      <th>列标题 1</th>
      <th>列标题 2</th>
      <th>列标题 3</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>行 1, 列 1</td>
      <td>行 1, 列 2</td>
      <td>行 1, 列 3</td>
    </tr>
    <tr>
      <td>行 2, 列 1</td>
      <td>行 2, 列 2</td>
      <td>行 2, 列 3</td>
    </tr>
  </tbody>
</table>
```

以上的表格实例代码中, <table> 元素表示整个表格, 它包含两个主要部分: <thead> 和 <tbody>。

- **<thead> 用于定义表格的标题部分:** 在 `<thead>` 中，使用 `<th>` 元素定义列的标题，以上实例中列标题分别为"列标题 1"，"列标题 2"和"列标题 3"。
- **<tbody> 用于定义表格的主体部分:** 在 `<tbody>` 中，使用 `<tr>` 元素定义行，并在每行中使用 `<td>` 元素定义单元格数据，以上实例中有两行数据，每行包含三个单元格。

通过使用 `<th>` 元素定义列标题，可以使其在表格中以粗体显示，与普通单元格区分开来。

HTML 表格还可以具有其他部分，如 `<tfoot>`（表格页脚）和 `<caption>`（表格标题），`<tfoot>` 可用于在表格的底部定义摘要、统计信息等内容。`<caption>` 可用于为整个表格定义标题。

HTML 表格还支持合并单元格和跨行/跨列的操作，以及其他样式和属性的应用，以满足各种需求。

我们也可以使用 CSS 来进一步自定义表格的样式和外观。



在线实例

实例

```
<h4>一列:</h4> <table border="1"> <tr> <td>100</td> </tr> </table> <h4>一行三列:</h4> <table border="1"> <tr>
<td>100</td> <td>200</td> <td>300</td> </tr> </table> <h4>两行三列:</h4> <table border="1"> <tr> <td>100</td>
<td>200</td> <td>300</td> </tr> <tr> <td>400</td> <td>500</td> <td>600</td> </tr> </table>
```

[尝试一下 »](#)

（可以在本页底端找到更多实例。）

表格实例

实例

```
<table border="1"> <tr> <td>row 1, cell 1</td> <td>row 1, cell 2</td> </tr> <tr> <td>row 2, cell 1</td> <td>row 2, cell 2</td> </tr> </table>
```

在浏览器显示如下：

row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2

HTML 表格和边框属性

如果不定义边框属性，表格将不显示边框。有时这很有用，但是大多数时候，我们希望显示边框。

使用边框属性来显示一个带有边框的表格：

实例

```
<table border="1"> <tr> <td>Row 1, cell 1</td> <td>Row 1, cell 2</td> </tr> </table>
```

HTML 表格表头

表格的表头使用 <th> 标签进行定义。

大多数浏览器会把表头显示为粗体居中的文本：

实例

```
<table border="1"> <tr> <th>Header 1</th> <th>Header 2</th> </tr> <tr> <td>row 1, cell 1</td> <td>row 1, cell 2</td> </tr> <tr> <td>row 2, cell 1</td> <td>row 2, cell 2</td> </tr> </table>
```

在浏览器显示如下：

Header 1	Header 2
row 1, cell 1	row 1, cell 2
row 2, cell 1	row 2, cell 2



更多实例

没有边框的表格

本例演示一个没有边框的表格。

表格中的表头(Heading)

本例演示如何显示表格表头。

带有标题的表格

本例演示一个带标题 (caption) 的表格

跨行或跨列的表格单元格

本例演示如何定义跨行或跨列的表格单元格。

表格内的标签

本例演示如何在不同的元素内显示元素。

单元格边距(Cell padding)

本例演示如何使用 Cell padding 来创建单元格内容与其边框之间的空白。

单元格间距(Cell spacing)

本例演示如何使用 Cell spacing 增加单元格之间的距离。

漂亮的表格

HTML 表格标签

标签	描述
<code><table></code>	定义表格
<code><th></code>	定义表格的表头
<code><tr></code>	定义表格的行
<code><td></code>	定义表格单元
<code><caption></code>	定义表格标题
<code><colgroup></code>	定义表格列的组
<code><col></code>	定义用于表格列的属性
<code><thead></code>	定义表格的页眉
<code><tbody></code>	定义表格的主体
<code><tfoot></code>	定义表格的页脚

HTML 列表

HTML 支持有序、无序和定义列表:

HTML 列表

有序列表

1. 第一个列表项
2. 第二个列表项
3. 第三个列表项

无序列表

- 列表项
- 列表项
- 列表项



在线实例

[无序列表](#)

本例演示无序列表。

[有序列表](#)

本例演示有序列表。

(可以在本页底端找到更多实例。)

HTML 无序列表

无序列表是一个项目的列表，此列项目使用粗体圆点（典型的小黑圆圈）进行标记。

无序列表使用 `` 标签

```
<ul>  
<li>Coffee</li>  
<li>Milk</li>  
</ul>
```

浏览器显示如下：

- Coffee

- Milk

HTML 有序列表

同样，有序列表也是一列项目，列表项目使用数字进行标记。有序列表始于 `` 标签。每个列表项始于 `` 标签。列表项使用数字来标记。

```
<ol>
<li>Coffee</li>
<li>Milk</li>
</ol>
```

浏览器中显示如下：

1. Coffee
 2. Milk
-

HTML 自定义列表

自定义列表不仅仅是一列项目，而是项目及其注释的组合。

自定义列表以 `<dl>` 标签开始。每个自定义列表项以 `<dt>` 开始。每个自定义列表项的定义以 `<dd>` 开始。

```
<dl>
<dt>Coffee</dt>
<dd>- black hot drink</dd>
<dt>Milk</dt>
```

```
<dd>- white cold drink</dd>
</dl>
```

浏览器显示如下:

Coffee

- black hot drink

Milk

- white cold drink

注意事项 - 有用提示

提示: 列表项内部可以使用段落、换行符、图片、链接以及其他列表等等。



更多实例

[不同类型的有序列表](#)

本例演示不同类型的有序列表。

[不同类型的无序列表](#)

本例演示不同类型的无序列表。

[嵌套列表](#)

本例演示如何嵌套列表。

[嵌套列表 2](#)

本例演示更复杂的嵌套列表。

[自定义列表](#)

本例演示一个定义列表。

HTML 列表标签

标签	描述
<u></u>	定义有序列表
<u></u>	定义无序列表
<u></u>	定义列表项
<u><dl></u>	定义列表
<u><dt></u>	自定义列表项目
<u><dd></u>	定义自定义列表项的描述

[HTML 表格](#)

[HTML 区块](#)

2 篇笔记 写笔记

1. Slient
che***oka@qq.com

675

HTML: My 列表 实例:<https://c.runoob.com/codedemo/2915>

[Slient](#)

Slient
che***oka@qq.com

7 年前 (2017-05-24)

2. bkgst
119***7990@qq.com

1837

ul 是 unordered lists 的缩写 (无序列表)
li 是 list item 的缩写 (列表项目)
ol 是 ordered lists 的缩写 (有序列表)
dl 是 definition lists 的英文缩写 (自定义列表)
dt 是 definition term 的缩写 (自定义列表组)
dd 是 definition description 的缩写 (自定义列表描述)
nl 是 navigation lists 的英文缩写 (导航列表)
tr 是 table row 的缩写 (表格中的一行)
th 是 table header cell 的缩写 (表格中的表头)
td 是 table data cell 的缩写 (表格中的一个单元格)

bkgst

bkgst
119***7990@qq.com

3 年前 (2021-02-18)

HTML `<div>` 和 ``

HTML 可以通过 `<div>` 和 `` 将元素组合起来。

HTML 区块元素

大多数 HTML 元素被定义为**块级元素**或**内联元素**。

块级元素在浏览器显示时，通常会以新行来开始（和结束）。

实例: <h1>, <p>, , <table>

HTML 内联元素

内联元素在显示时通常不会以新行开始。

实例: , <td>, <a>,

HTML <div> 元素

HTML <div> 元素是块级元素，它可用于组合其他 HTML 元素的容器。

<div> 元素没有特定的含义。除此之外，由于它属于块级元素，浏览器会在其前后显示折行。

如果与 CSS 一同使用，<div> 元素可用于对大的内容块设置样式属性。

<div> 元素的另一个常见的用途是文档布局。它取代了使用表格定义布局的老式方法。使用 <table> 元素进行文档布局不是表格的正确用法。

<table> 元素的作用是显示表格化的数据。

HTML 元素

HTML 元素是内联元素，可用作文本的容器

 元素也没有特定的含义。

当与 CSS 一同使用时， 元素可用于为部分文本设置样式属性。

HTML 分组标签

标签	描述
<code><div></code>	定义了文档的区域，块级 (block-level)
<code></code>	用来组合文档中的行内元素，内联元素 (inline)

[HTML 列表](#)

[HTML 布局](#)

1 篇笔记 写笔记

1. 老刀把子
594***084@qq.com

188

当涉及到 `<div>` 和 `` 的区别时，以下几点是值得进一步明确的：

块级元素 vs. 行内元素：

- o `<div>` 是块级元素，它独占一行，可以设置宽度、高度以及边距等样式属性。它适合用于创建页面的大块结构，例如页面的主体区域、容器、布局等。
- o `` 是行内元素，它不会独占一行，宽度默认由其内容决定。它适合用于对文本或其他行内元素进行样式化、标记或包裹。

默认样式和布局：

- o `<div>` 元素的默认样式为块级显示，会以块的形式占据可用空间。

- o `` 元素的默认样式为行内显示，它不会独占一行，只占据其内容的宽度。

嵌套关系：

- o `<div>` 可以容纳其他块级元素和行内元素，包括其他的 `<div>` 和 `` 元素。
- o `` 通常被用来包裹文本或其他行内元素，比如用来设置特定文本的样式。

总之，`<div>` 用于创建页面结构和布局，而 `` 用于对文本或行内元素进行样式化或包裹。它们在页面设计和样式设置中有不同的用途和作用。



HTML 布局

网页布局对改善网站的外观非常重要。
请慎重设计您的网页布局。



在线实例

[使用 `<div>` 元素的网页布局](#)

如何使用 `<div>` 元素添加布局。

[使用 `<table>` 元素的网页布局](#)

如何使用 `<table>` 元素添加布局。

网站布局

大多数网站会把内容安排到多个列中（就像杂志或报纸那样）。

大多数网站可以使用 <div> 或者 <table> 元素来创建多列。CSS 用于对元素进行定位，或者为页面创建背景以及色彩丰富的外观。



虽然我们可以使用 HTML table 标签来设计出漂亮的布局，但是 table 标签是不建议作为布局工具使用的 - 表格不是布局工具。

HTML 布局 - 使用<div> 元素

div 元素是用于分组 HTML 元素的块级元素。

下面的例子使用五个 div 元素来创建多列布局：

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body> <div id="container" style="width:500px"> <div id="header" style="background-color:#FFA500;"> <h1 style="margin-bottom:0;">主要的网页标题</h1></div> <div id="menu" style="background-color:#FFD700;height:200px;width:100px;float:left;"> <b>菜单</b><br> HTML<br> CSS<br> JavaScript</div> <div id="content" style="background-color:#EEEEEE;height:200px;width:400px;float:left;"> 内容在这里</div> <div id="footer" style="background-color:#FFA500;clear:both;text-align:center;"> 版权© runoob.com</div> </div> </body> </html>
```

尝试一下 »

上面的 HTML 代码会产生如下结果：

主要的网页标题

菜单

HTML

CSS

JavaScript

内容在这里

版权 © runoob.com

HTML 布局 - 使用表格

使用 HTML `<table>` 标签是创建布局的一种简单的方式。

大多数站点可以使用 `<div>` 或者 `<table>` 元素来创建多列。CSS 用于对元素进行定位，或者为页面创建背景以及色彩丰富的外观。



即使可以使用 HTML 表格来创建漂亮的布局，但设计表格的目的是呈现表格化数据 - 表格不是布局工具！

下面的例子使用三行两列的表格 - 第一和最后一行使用 `colspan` 属性来横跨两列：

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body>
<table width="500" border="0"> <tr> <td colspan="2" style="background-color:#FFA500;"> <h1>主要的网页标题</h1>
</td> </tr> <tr> <td style="background-color:#FFD700;width:100px;"> <b>菜单</b><br> HTML<br> CSS<br>
JavaScript </td> <td style="background-color:#eeeeee;height:200px;width:400px;"> 内容在这里</td> </tr> <tr> <td
colspan="2" style="background-color:#FFA500;text-align:center;"> 版权© runoob.com</td> </tr> </table> </body>
</html>
```

尝试一下 »

上面的 HTML 代码会产生以下结果：

主要的网页标题

菜单

HTML

CSS

JavaScript

内容在这里

版权 © runoob.com

HTML 布局 - 有用的提示

Tip: 使用 CSS 最大的好处是，如果把 CSS 代码存放到外部样式表中，那么站点会更易于维护。通过编辑单一的文件，就可以改变所有页面的布局。如需学习更多有关 CSS 的知识，请访问我们的 [CSS 教程](#)。

Tip: 由于创建高级的布局非常耗时，使用模板是一个快速的选项。通过搜索引擎可以找到很多免费的网站模板（您可以使用这些预先构建好的网站布局，并优化它们）。

HTML 布局标签

标签	描述
<div>	定义文档区块，块级(block-level)
	定义 span，用来组合文档中的行内元素。

[HTML 区块](#)

[HTML 表单](#)

2 篇笔记 写笔记

1. Slient
che***oka@qq.com

480

Div 布局实例:

<https://c.runoob.com/codedemo/2917>

[Slient](#)

Slient
che***oka@qq.com

7 年前 (2017-05-24)

2. zhang
272***9838@qq.com
[参考地址](#)

466

在 css 里面定义:

```
p {margin:0; padding:0}
```

即可, 用全局样式把 p 的边距给归零。

不过这样的副作用是网页所有的 p 都被重置了, 所以你也可以指定专门标签下的 p。例如, 如果这个 p 是在一个 class 为 content 的 div 下的, 想去掉它的边距, 就这么定义:

```
.content p {margin:0; padding:0;}
```

[zhang](#)

zhang
272***9838@qq.com
[参考地址](#)

4 年前 (2019-11-05)

HTML 表单和输入

HTML 表单用于收集用户的输入信息。

HTML 表单表示文档中的一个区域, 此区域包含交互控件, 将用户收集到的信息发送到 Web 服务器。

HTML 表单通常包含各种输入字段、复选框、单选按钮、下拉列表等元素。

以下是一个简单的 HTML 表单的例子:

- <form> 元素用于创建表单，action 属性定义了表单数据提交的目标 URL，method 属性定义了提交数据的 HTTP 方法（这里使用的是 "post"）。
- <label> 元素用于为表单元素添加标签，提高可访问性。
- <input> 元素是最常用的表单元素之一，它可以创建文本输入框、密码框、单选按钮、复选框等。type 属性定义了输入框的类型，id 属性用于关联 <label> 元素，name 属性用于标识表单字段。
- <select> 元素用于创建下拉列表，而 <option> 元素用于定义下拉列表中的选项。

实例

```
<form action="/" method="post">
  <!-- 文本输入框 -->
  <label for="name">用户名:</label>
  <input type="text" id="name" name="name" required>

  <br>

  <!-- 密码输入框 -->
  <label for="password">密码:</label>
  <input type="password" id="password" name="password" required>

  <br>

  <!-- 单选按钮 -->
```

```
<label>性别:</label>
<input type="radio" id="male" name="gender" value="male" checked>
<label for="male">男</label>
<input type="radio" id="female" name="gender" value="female">
<label for="female">女</label>

<br>

<!-- 复选框 -->
<input type="checkbox" id="subscribe" name="subscribe" checked>
<label for="subscribe">订阅推送信息</label>

<br>

<!-- 下拉列表 -->
<label for="country">国家:</label>
<select id="country" name="country">
  <option value="cn">CN</option>
  <option value="usa">USA</option>
  <option value="uk">UK</option>
</select>

<br>

<!-- 提交按钮 -->
<input type="submit" value="提交">
</form>
```

[尝试一下 »](#)



在线实例

实例

以下实例创建了一个表单，包含两个输入框：

```
<form action="">
First name: <input type="text" name="firstname"><br>
Last name: <input type="text" name="lastname">
</form>
```

[尝试一下 »](#)

实例

以下实例创建了一个表单，包含一个普通输入框和一个密码输入框：

```
<form action="">
Username: <input type="text" name="user"><br>
Password: <input type="password" name="password">
</form>
```

[尝试一下 »](#)

（在本页底端可以找到更多实例。）

HTML 表单

表单是一个包含表单元素的区域。

表单元素是允许用户在表单中输入内容，比如：文本域（`textarea`）、下拉列表（`select`）、单选框（`radio-buttons`）、复选框（`checkbox`）等等。

我们可以使用 `<form>` 标签来创建表单：

实例

```
<form>
.
input 元素
.
</form>
```

HTML 表单 - 输入元素

多数情况下被用到的表单标签是输入标签 `<input>`。

输入类型是由 `type` 属性定义。

接下来我们介绍几种常用的输入类型。

文本域（Text Fields）

文本域通过 `<input type="text">` 标签来设定，当用户要在表单中键入字母、数字等内容时，就会用到文本域。

实例

```
<form>
First name: <input type="text" name="firstname"><br>
Last name: <input type="text" name="lastname">
</form>
```

浏览器显示如下:

First name:

Last name:

注意:表单本身并不可见。同时，在大多数浏览器中，文本框的默认宽度是 20 个字符。

密码字段

密码字段通过标签 `<input type="password">` 来定义:

实例

```
<form>
Password: <input type="password" name="pwd">
</form>
```

浏览器显示效果如下:

Password:

注意: 密码字段字符不会明文显示，而是以星号 * 或圆点 . 替代。

单选按钮（Radio Buttons）

`<input type="radio">` 标签定义了表单的单选框选项:

实例

```
<form action="">
<input type="radio" name="sex" value="male">男<br>
<input type="radio" name="sex" value="female">女
</form>
```

浏览器显示效果如下:

☐ 男

☐ 女

复选框（Checkboxes）

`<input type="checkbox">` 定义了复选框。

复选框可以选取一个或多个选项:

实例

```
<form>
<input type="checkbox" name="vehicle" value="Bike">我喜欢自行车<br>
<input type="checkbox" name="vehicle" value="Car">我喜欢小汽车
</form>
```

浏览器显示效果如下:

☐ 我喜欢自行车

☐ 我喜欢小汽车

提交按钮(Submit)

`<input type="submit">` 定义了提交按钮。

当用户单击确认按钮时，表单的内容会被传送到服务器。表单的动作属性 **action** 定义了服务端的文件名。

action 属性会对接收到的用户输入数据进行相关的处理：

实例

```
<form name="input" action="html_form_action.php" method="get">
Username: <input type="text" name="user">
<input type="submit" value="Submit">
</form>
```

浏览器显示效果如下：

Username:

Submit

假如您在上面的文本框内键入几个字母，然后点击确认按钮，那么输入数据会传送到 **html_form_action.php** 文件，该页面将显示出输入的结果。

以上实例中有一个 **method** 属性，它用于定义表单数据的提交方式，可以是以下值：

- **post**：指的是 HTTP POST 方法，表单数据会包含在表单体内然后发送给服务器，用于提交敏感数据，如用户名与密码等。
- **get**：默认值，指的是 HTTP GET 方法，表单数据会附加在 **action** 属性的 URL 中，并以 **?** 作为分隔符，一般用于不敏感信息，如分页等。例如：<https://www.runoob.com/?page=1>，这里的 **page=1** 就是 **get** 方法提交的数据。

实例

```
<!-- 以下表单使用 GET 请求发送数据到当前的 URL，method 默认位 GET。 -->
<form>
  <label>Name:
    <input name="submitted-name" autocomplete="name">
  </label>
```

```
<button>Save</button>
</form>

<!-- 以下表单使用 POST 请求发送数据到当前的 URL。 -->
<form method="post">
  <label>Name:
    <input name="submitted-name" autocomplete="name">
  </label>
  <button>Save</button>
</form>

<!-- 表单使用 fieldset, legend, 和 label 标签 -->
<form method="post">
  <fieldset>
    <legend>Title</legend>
    <label><input type="radio" name="radio"> Select me</label>
  </fieldset>
</form>
```



更多实例

[单选按钮\(Radio buttons\)](#)

本例演示如何在 HTML 中创建单选按钮。

[复选框\(Checkboxes\)](#)

本例演示如何在 HTML 页中创建复选框。用户可以选中或取消选取复选框。

[简单的下拉列表](#)

本例演示如何在 HTML 页面中创建简单的下拉列表框。下拉列表框是一个可选列表。

[预选下拉列表](#)

本例演示如何创建一个简单的带有预选值的下拉列表。

[文本域\(Textarea\)](#)

本例演示如何创建文本域（多行文本输入控件）。用户可在文本域中写入文本。可写入字符的字数不受限制。

[创建按钮](#)

本例演示如何创建按钮。你可以对按钮上的文字进行自定义。



表单实例

[带边框的表单](#)

本例演示如何在数据周围绘制一个带标题的框。

[带有输入框和确认按钮的表单](#)

本例演示如何向页面添加表单。此表单包含两个输入框和一个确认按钮。

[带有复选框的表单](#)

此表单包含两个复选框和一个确认按钮。

[带有单选按钮的表单](#)

此表单包含两个单选框和一个确认按钮。

[从表单发送电子邮件](#)

此例演示如何从表单发送电子邮件。

HTML 表单标签

New : HTML5 新标签

标签	描述
----	----

<u><form></u>	定义供用户输入的表单
<u><input></u>	定义输入域
<u><textarea></u>	定义文本域 (一个多行的输入控件)
<u><label></u>	定义了 <input> 元素的标签, 一般为输入标题
<u><fieldset></u>	定义了一组相关的表单元素, 并使用外框包含起来
<u><legend></u>	定义了 <fieldset> 元素的标题
<u><select></u>	定义了下拉选项列表
<u><optgroup></u>	定义选项组
<u><option></u>	定义下拉列表中的选项
<u><button></u>	定义一个点击按钮
<u><datalist></u> New	指定一个预先定义的输入控件选项列表
<u><keygen></u> New	定义了表单的密钥对生成器字段
<u><output></u> New	定义一个计算结果

[HTML 布局](#)

[HTML 框架](#)

HTML 框架

通过使用框架，你可以在同一个浏览器窗口中显示不止一个页面。

iframe 语法:

```
<iframe src="URL"></iframe>
```

该 URL 指向不同的网页。

iframe - 设置高度与宽度

height 和 width 属性用来定义 iframe 标签的高度与宽度。

属性默认以像素为单位，但是你可以指定其按比例显示 (如: "80%")。

实例

```
<iframe src="demo_iframe.htm" width="200" height="200"></iframe>
```

[尝试一下 »](#)

iframe - 移除边框

frameborder 属性用于定义 iframe 表示是否显示边框。

设置属性值为 "0" 移除 iframe 的边框:

实例

```
<iframe src="demo_iframe.htm" frameborder="0"></iframe>
```

尝试一下 »

使用 iframe 来显示目标链接页面

iframe 可以显示一个目标链接的页面
目标链接的属性必须使用 iframe 的属性，如下实例：

实例

```
<iframe src="demo_iframe.htm" name="iframe_a"></iframe> <p><a href="https://www.runoob.com" target="iframe_a" rel="noopener">RUNOOB.COM</a></p>
```

尝试一下 »

HTML iframe 标签

标签	说明
<u><iframe></u>	定义一个内联的 iframe

[HTML 表单](#)

[HTML 颜色](#)

1 篇笔记 写笔记

1. tony
ton***g@163.com
[参考地址](#)

717

出于有些网页不希望被嵌套, 响应头中有一选项

X-Frame-Options

他有三个可配置值

DENY: 表示该网站页面不允许被嵌套, 即便是在自己的域名的页面中也不能进行嵌套。

SAMEORIGIN: 表示该页面可以在相同域名页面中被嵌套展示。

ALLOW-FROM uri: 表示该页面可以在指定来源页面中进行嵌套展示。

[tony](#)

tony
ton***g@163.com
[参考地址](#)

3 年前 (2021-03-13)

HTML 颜色

HTML 颜色由红色、绿色、蓝色混合而成。

颜色值

HTML 颜色由一个十六进制符号来定义，这个符号由红色、绿色和蓝色的值组成（RGB）。
每种颜色的最小值是 0（十六进制：#00）。最大值是 255（十六进制：#FF）。
这个表格给出了由三种颜色混合而成的具体效果：

颜色值

颜色(Color)	颜色十六进制(Color HEX)	颜色 RGB(Color RGB)
	#000000	rgb(0,0,0)
	#FF0000	rgb(255,0,0)
	#00FF00	rgb(0,255,0)
	#0000FF	rgb(0,0,255)
	#FFFF00	rgb(255,255,0)
	#00FFFF	rgb(0,255,255)
	#FF00FF	rgb(255,0,255)
	#C0C0C0	rgb(192,192,192)
	#FFFFFF	rgb(255,255,255)

[尝试一下 »](#)

1600 万种不同颜色

三种颜色 红，绿，蓝的组合从 0 到 255，一共有 1600 万种不同颜色(256 x 256 x 256)。
在下面的颜色表中你会看到不同的结果，从 0 到 255 的红色，同时设置绿色和蓝色的值为 0,随着红色的值变化，不同的值都显示了不同的颜色。

Red Light	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#080000	rgb(8,0,0)
	#100000	rgb(16,0,0)
	#180000	rgb(24,0,0)
	#200000	rgb(32,0,0)
	#280000	rgb(40,0,0)
	#300000	rgb(48,0,0)
	#380000	rgb(56,0,0)
	#400000	rgb(64,0,0)
	#480000	rgb(72,0,0)
	#500000	rgb(80,0,0)
	#580000	rgb(88,0,0)
	#600000	rgb(96,0,0)

	#680000	rgb(104,0,0)
	#700000	rgb(112,0,0)
	#780000	rgb(120,0,0)
	#800000	rgb(128,0,0)
	#880000	rgb(136,0,0)
	#900000	rgb(144,0,0)
	#980000	rgb(152,0,0)
	#A00000	rgb(160,0,0)
	#A80000	rgb(168,0,0)
	#B00000	rgb(176,0,0)
	#B80000	rgb(184,0,0)
	#C00000	rgb(192,0,0)
	#C80000	rgb(200,0,0)
	#D00000	rgb(208,0,0)
	#D80000	rgb(216,0,0)
	#E00000	rgb(224,0,0)

	#E80000	rgb(232,0,0)
	#F00000	rgb(240,0,0)
	#F80000	rgb(248,0,0)
	#FF0000	rgb(255,0,0)

灰暗色调

以下展示了黑色到灰色的渐变

Gray Shades	Color HEX	Color RGB
	#000000	rgb(0,0,0)
	#080808	rgb(8,8,8)
	#101010	rgb(16,16,16)
	#181818	rgb(24,24,24)
	#202020	rgb(32,32,32)
	#282828	rgb(40,40,40)
	#303030	rgb(48,48,48)
	#383838	rgb(56,56,56)

	#404040	rgb(64,64,64)
	#484848	rgb(72,72,72)
	#505050	rgb(80,80,80)
	#585858	rgb(88,88,88)
	#606060	rgb(96,96,96)
	#686868	rgb(104,104,104)
	#707070	rgb(112,112,112)
	#787878	rgb(120,120,120)
	#808080	rgb(128,128,128)
	#888888	rgb(136,136,136)
	#909090	rgb(144,144,144)
	#989898	rgb(152,152,152)
	#A0A0A0	rgb(160,160,160)
	#A8A8A8	rgb(168,168,168)
	#B0B0B0	rgb(176,176,176)
	#B8B8B8	rgb(184,184,184)

	#C0C0C0	rgb(192,192,192)
	#C8C8C8	rgb(200,200,200)
	#D0D0D0	rgb(208,208,208)
	#D8D8D8	rgb(216,216,216)
	#E0E0E0	rgb(224,224,224)
	#E8E8E8	rgb(232,232,232)
	#F0F0F0	rgb(240,240,240)
	#F8F8F8	rgb(248,248,248)
	#FFFFFF	rgb(255,255,255)

Web 安全色？

数年以前，当大多数计算机仅支持 256 种颜色的时候，一系列 216 种 Web 安全色作为 Web 标准被建议使用。其中的原因是，微软和 Mac 操作系统使用了 40 种不同的保留的固定系统颜色（双方大约各使用 20 种）。

我们不确定如今这么做的意义有多大，因为越来越多的计算机有能力处理数百万种颜色，不过做选择还是你自己。

最初，216 跨平台 web 安全色被用来确保：当计算机使用 256 色调色板时，所有的计算机能够正确地显示所有的颜色。

000000	000033	000066	000099	0000CC	0000FF
003300	003333	003366	003399	0033CC	0033FF
006600	006633	006666	006699	0066CC	0066FF
009900	009933	009966	009999	0099CC	0099FF

00CC00	00CC33	00CC66	00CC99	00CCCC	00CCFF
00FF00	00FF33	00FF66	00FF99	00FFCC	00FFFF
330000	330033	330066	330099	3300CC	3300FF
333300	333333	333366	333399	3333CC	3333FF
336600	336633	336666	336699	3366CC	3366FF
339900	339933	339966	339999	3399CC	3399FF
33CC00	33CC33	33CC66	33CC99	33CCCC	33CCFF
33FF00	33FF33	33FF66	33FF99	33FFCC	33FFFF
660000	660033	660066	660099	6600CC	6600FF
663300	663333	663366	663399	6633CC	6633FF
666600	666633	666666	666699	6666CC	6666FF
669900	669933	669966	669999	6699CC	6699FF
66CC00	66CC33	66CC66	66CC99	66CCCC	66CCFF
66FF00	66FF33	66FF66	66FF99	66FFCC	66FFFF
990000	990033	990066	990099	9900CC	9900FF
993300	993333	993366	993399	9933CC	9933FF
996600	996633	996666	996699	9966CC	9966FF
999900	999933	999966	999999	9999CC	9999FF
99CC00	99CC33	99CC66	99CC99	99CCCC	99CCFF
99FF00	99FF33	99FF66	99FF99	99FFCC	99FFFF
CC0000	CC0033	CC0066	CC0099	CC00CC	CC00FF
CC3300	CC3333	CC3366	CC3399	CC33CC	CC33FF
CC6600	CC6633	CC6666	CC6699	CC66CC	CC66FF
CC9900	CC9933	CC9966	CC9999	CC99CC	CC99FF
CCCC00	CCCC33	CCCC66	CCCC99	CCCCCC	CCCCFF
CCFF00	CCFF33	CCFF66	CCFF99	CCFFCC	CCFFFF
FF0000	FF0033	FF0066	FF0099	FF00CC	FF00FF
FF3300	FF3333	FF3366	FF3399	FF33CC	FF33FF

FF6600	FF6633	FF6666	FF6699	FF66CC	FF66FF
FF9900	FF9933	FF9966	FF9999	FF99CC	FF99FF
FFCC00	FFCC33	FFCC66	FFCC99	FFCCCC	FFCCFF
FFFF00	FFFF33	FFFF66	FFFF99	FFFFCC	FFFFFF

[HTML 框架](#)

[HTML 颜色名](#)

1 篇笔记 写笔记

1. naozhi
786***078@qq.com

692

RGBA 的意思是（Red-Green-Blue-Alpha）它是在 RGB 上扩展包括了 “**alpha**” 通道，运行对颜色值设置透明度。

```
div {  
  
background: rgba(255,0,0,0.5);  
  
}
```

相对于使用 `rgb(255,255,0)`，使用 `rgba(255,255,0,0.5)` 可以实现设置颜色透明度的功能，这里的 0.5 表示透明度，范围 0~1，0 表示全透明。

通常我们为了省略一个 0:

```
div {  
    background: rgba(255,0,0,.5);  
}
```

实例：

```
<p style="background-color:rgb(255,255,0)">
通过 rgb 值设置背景颜色
</p>
<p style="background-color:rgba(255,255,0,0.25)">
通过 rgb 值设置背景颜色
</p>
<p style="background-color:rgba(255,255,0,0.5)">
通过 rgb 值设置背景颜色
</p>
<p style="background-color:rgba(255,255,0,0.75)">
通过 rgb 值设置背景颜色
</p>
```

[尝试一下 »](#)

[naozzhi](#)

naozzhi

786***078@qq.com

7 年前 (2017-06-08)

HTML 颜色名

目前所有浏览器都支持以下颜色名。

141 个颜色名称是在 HTML 和 CSS 颜色规范定义的（17 标准颜色，再加 124）。下表列出了所有颜色的值，包括十六进制值。




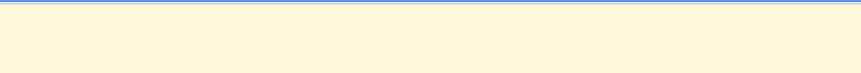




💡 **提示:** 17 标准颜色：黑色，蓝色，水，紫红色，灰色，绿色，石灰，栗色，海军，橄榄，橙，紫，红，白，银，蓝绿色，黄色。点击其中一个颜色名称（或一个十六进制值）就可以查看与不同文字颜色搭配的背景颜色。


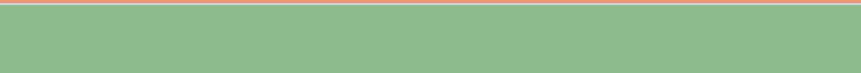
按颜色名排序

[按十六进制的值排序](#)


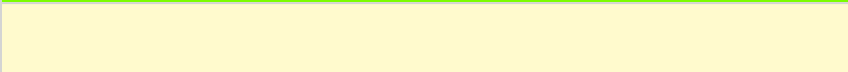


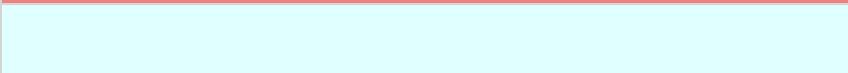
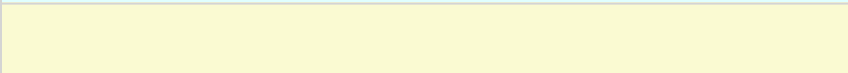

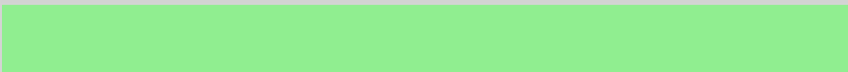
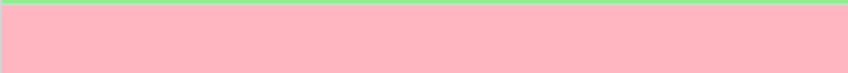




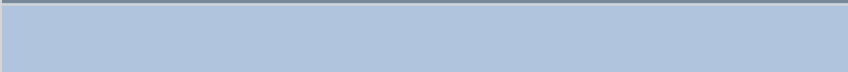
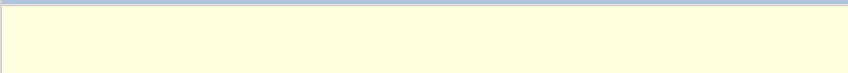

单击一个颜色名或者 16 进制值，就可以查看与不同文字颜色搭配的背景颜色。


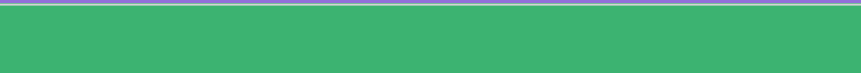

颜色名	HEX	Color
AliceBlue	#F0F8FF	
AntiqueWhite	#FAEBD7	
Aqua	#00FFFF	
Aquamarine	#7FFFD4	
Azure	#F0FFFF	
Beige	#F5F5DC	
Bisque	#FFE4C4	
Black	#000000	
BlanchedAlmond	#FFEBCD	
Blue	#0000FF	
BlueViolet	#8A2BE2	
Brown	#A52A2A	
BurlyWood	#DEB887	



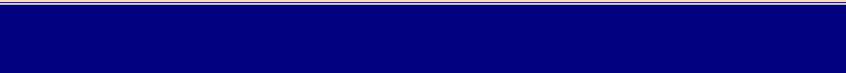






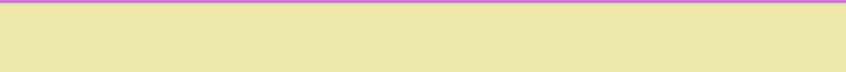
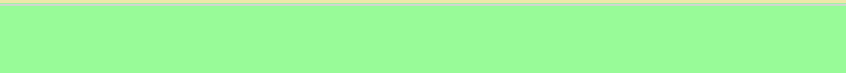
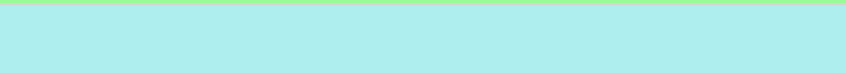




CadetBlue	#5F9EA0	
Chartreuse	#7FFF00	
Chocolate	#D2691E	
Coral	#FF7F50	
CornflowerBlue	#6495ED	
Cornsilk	#FFF8DC	
Crimson	#DC143C	
Cyan	#00FFFF	
DarkBlue	#00008B	
DarkCyan	#008B8B	
DarkGoldenRod	#B8860B	
DarkGray	#A9A9A9	
DarkGreen	#006400	
DarkKhaki	#BDB76B	
DarkMagenta	#8B008B	
DarkOliveGreen	#556B2F	




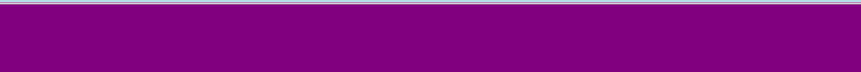


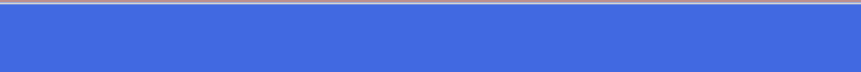




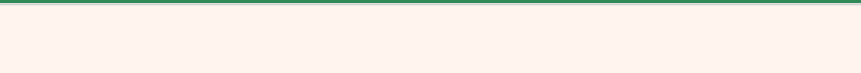

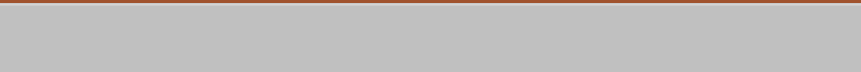

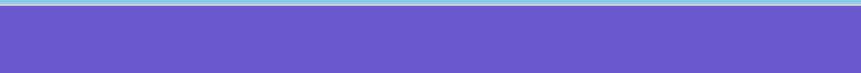
DarkOrange	#FF8C00	
DarkOrchid	#9932CC	
DarkRed	#8B0000	
DarkSalmon	#E9967A	
DarkSeaGreen	#8FBC8F	
DarkSlateBlue	#483D8B	
DarkSlateGray	#2F4F4F	
DarkTurquoise	#00CED1	
DarkViolet	#9400D3	
DeepPink	#FF1493	
DeepSkyBlue	#00BFFF	
DimGray	#696969	
DodgerBlue	#1E90FF	
FireBrick	#B22222	
FloralWhite	#FFFAF0	
ForestGreen	#228B22	






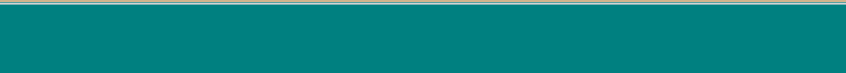


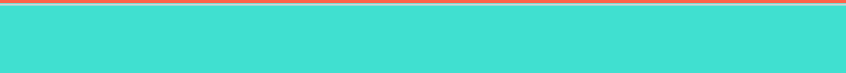

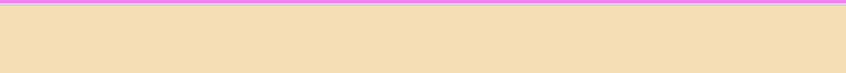
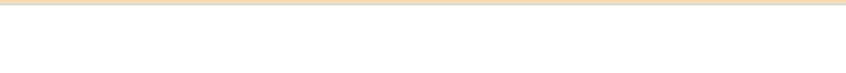
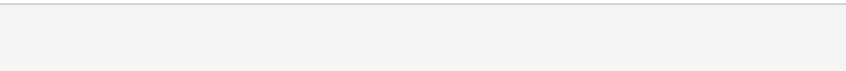


Fuchsia	#FF00FF	
Gainsboro	#DCDCDC	
GhostWhite	#F8F8FF	
Gold	#FFD700	
GoldenRod	#DAA520	
Gray	#808080	
Green	#008000	
GreenYellow	#ADFF2F	
HoneyDew	#F0FFF0	
HotPink	#FF69B4	
IndianRed	#CD5C5C	
Indigo	#4B0082	
Ivory	#FFFFFF0	
Khaki	#F0E68C	
Lavender	#E6E6FA	
LavenderBlush	#FFF0F5	

LawnGreen	#7CFC00	
LemonChiffon	#FFFACD	
LightBlue	#ADD8E6	
LightCoral	#F08080	
LightCyan	#E0FFFF	
LightGoldenRodYellow	#FAFAD2	
LightGray	#D3D3D3	
LightGreen	#90EE90	
LightPink	#FFB6C1	
LightSalmon	#FFA07A	
LightSeaGreen	#20B2AA	
LightSkyBlue	#87CEFA	
LightSlateGray	#778899	
LightSteelBlue	#B0C4DE	
LightYellow	#FFFFE0	
Lime	#00FF00	

LimeGreen	#32CD32	
Linen	#FAF0E6	
Magenta	#FF00FF	
Maroon	#800000	
MediumAquaMarine	#66CDAA	
MediumBlue	#0000CD	
MediumOrchid	#BA55D3	
MediumPurple	#9370DB	
MediumSeaGreen	#3CB371	
MediumSlateBlue	#7B68EE	
MediumSpringGreen	#00FA9A	
MediumTurquoise	#48D1CC	
MediumVioletRed	#C71585	
MidnightBlue	#191970	
MintCream	#F5FFFA	
MistyRose	#FFE4E1	

Moccasin	#FFE4B5	
NavajoWhite	#FFDEAD	
Navy	#000080	
OldLace	#FDF5E6	
Olive	#808000	
OliveDrab	#6B8E23	
Orange	#FFA500	
OrangeRed	#FF4500	
Orchid	#DA70D6	
PaleGoldenRod	#EEE8AA	
PaleGreen	#98FB98	
PaleTurquoise	#AFEEEE	
PaleVioletRed	#DB7093	
PapayaWhip	#FFEFD5	
PeachPuff	#FFDAB9	
Peru	#CD853F	

Pink	#FFC0CB	
Plum	#DDA0DD	
PowderBlue	#B0E0E6	
Purple	#800080	
Red	#FF0000	
RosyBrown	#BC8F8F	
RoyalBlue	#4169E1	
SaddleBrown	#8B4513	
Salmon	#FA8072	
SandyBrown	#F4A460	
SeaGreen	#2E8B57	
SeaShell	#FFF5EE	
Sienna	#A0522D	
Silver	#C0C0C0	
SkyBlue	#87CEEB	
SlateBlue	#6A5ACD	

SlateGray	#708090	
Snow	#FFFAFA	
SpringGreen	#00FF7F	
SteelBlue	#4682B4	
Tan	#D2B48C	
Teal	#008080	
Thistle	#D8BFD8	
Tomato	#FF6347	
Turquoise	#40E0D0	
Violet	#EE82EE	
Wheat	#F5DEB3	
White	#FFFFFF	
WhiteSmoke	#F5F5F5	
Yellow	#FFFF00	
YellowGreen	#9ACD32	

HTML 颜色值

颜色由红(R)、绿(G)、蓝(B)组成。

颜色值

颜色值由十六进制来表示红、绿、蓝（RGB）。
每个颜色的最低值为 0(十六进制为 00)，最高值为 255(十六进制为 FF)。
十六进制值的写法为 # 号后跟三个或六个十六进制字符。
三位数表示法为：#RGB，转换为 6 位数表示为：#RRGGBB。

颜色实例




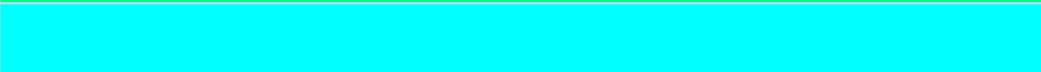

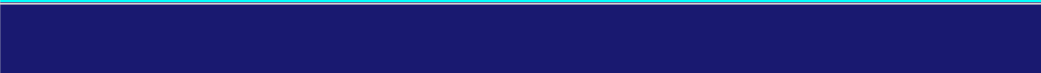
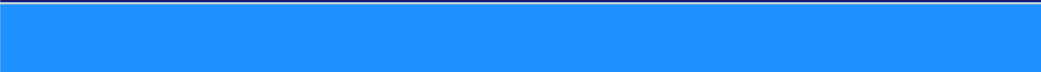
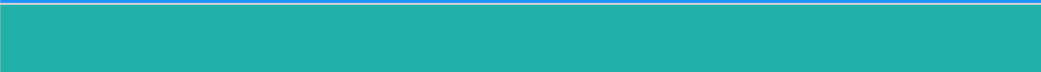



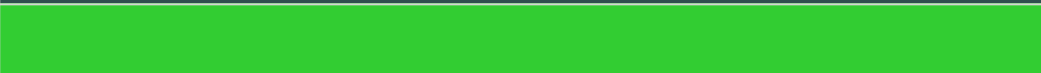




颜色	3 位十六进制颜色值	6 位十六进制颜色值	RGB
	#000	#000000	rgb(0,0,0)
	#F00	#FF0000	rgb(255,0,0)
	#0F0	#00FF00	rgb(0,255,0)
	#00F	#0000FF	rgb(0,0,255)
	#FF0	#FFFF00	rgb(255,255,0)
	#0FF	#00FFFF	rgb(0,255,255)
	#F0F	#FF00FF	rgb(255,0,255)
	#888	#888888	rgb(136,136,136)
	#FFF	#FFFFFF	rgb(255,255,255)

尝试一下 »


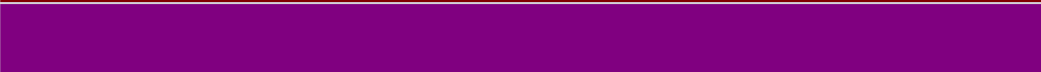
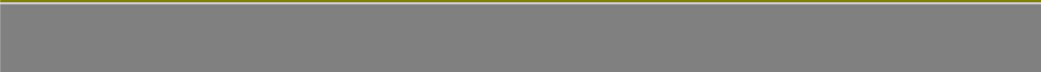

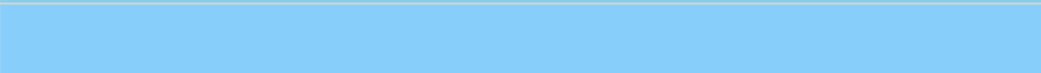
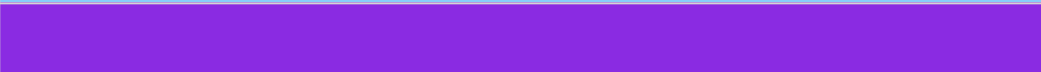






通过十六进制（Hex）的颜色值排序

[查看以颜色名称排序的颜色列表](#)

颜色名	HEX	Color
Black	#000000	
Navy	#000080	
DarkBlue	#00008B	
MediumBlue	#0000CD	
Blue	#0000FF	
DarkGreen	#006400	
Green	#008000	
Teal	#008080	
DarkCyan	#008B8B	
DeepSkyBlue	#00BFFF	
DarkTurquoise	#00CED1	

<u>MediumSpringGreen</u>	<u>#00FA9A</u>	
<u>Lime</u>	<u>#00FF00</u>	
<u>SpringGreen</u>	<u>#00FF7F</u>	
<u>Aqua</u>	<u>#00FFFF</u>	
<u>Cyan</u>	<u>#00FFFF</u>	
<u>MidnightBlue</u>	<u>#191970</u>	
<u>DodgerBlue</u>	<u>#1E90FF</u>	
<u>LightSeaGreen</u>	<u>#20B2AA</u>	
<u>ForestGreen</u>	<u>#228B22</u>	
<u>SeaGreen</u>	<u>#2E8B57</u>	
<u>DarkSlateGray</u>	<u>#2F4F4F</u>	
<u>LimeGreen</u>	<u>#32CD32</u>	
<u>MediumSeaGreen</u>	<u>#3CB371</u>	
<u>Turquoise</u>	<u>#40E0D0</u>	
<u>RoyalBlue</u>	<u>#4169E1</u>	
<u>SteelBlue</u>	<u>#4682B4</u>	

<u>DarkSlateBlue</u>	<u>#483D8B</u>	
<u>MediumTurquoise</u>	<u>#48D1CC</u>	
<u>Indigo</u>	<u>#4B0082</u>	
<u>DarkOliveGreen</u>	<u>#556B2F</u>	
<u>CadetBlue</u>	<u>#5F9EA0</u>	
<u>CornflowerBlue</u>	<u>#6495ED</u>	
<u>MediumAquaMarine</u>	<u>#66CDAA</u>	
<u>DimGray</u>	<u>#696969</u>	
<u>SlateBlue</u>	<u>#6A5ACD</u>	
<u>OliveDrab</u>	<u>#6B8E23</u>	
<u>SlateGray</u>	<u>#708090</u>	
<u>LightSlateGray</u>	<u>#778899</u>	
<u>MediumSlateBlue</u>	<u>#7B68EE</u>	
<u>LawnGreen</u>	<u>#7CFC00</u>	
<u>Chartreuse</u>	<u>#7FFF00</u>	
<u>Aquamarine</u>	<u>#7FFFD4</u>	

<u>Maroon</u>	<u>#800000</u>	
<u>Purple</u>	<u>#800080</u>	
<u>Olive</u>	<u>#808000</u>	
<u>Gray</u>	<u>#808080</u>	
<u>SkyBlue</u>	<u>#87CEEB</u>	
<u>LightSkyBlue</u>	<u>#87CEFA</u>	
<u>BlueViolet</u>	<u>#8A2BE2</u>	
<u>DarkRed</u>	<u>#8B0000</u>	
<u>DarkMagenta</u>	<u>#8B008B</u>	
<u>SaddleBrown</u>	<u>#8B4513</u>	
<u>DarkSeaGreen</u>	<u>#8FBC8F</u>	
<u>LightGreen</u>	<u>#90EE90</u>	
<u>MediumPurple</u>	<u>#9370DB</u>	
<u>DarkViolet</u>	<u>#9400D3</u>	
<u>PaleGreen</u>	<u>#98FB98</u>	
<u>DarkOrchid</u>	<u>#9932CC</u>	

<u>YellowGreen</u>	<u>#9ACD32</u>	
<u>Sienna</u>	<u>#A0522D</u>	
<u>Brown</u>	<u>#A52A2A</u>	
<u>DarkGray</u>	<u>#A9A9A9</u>	
<u>LightBlue</u>	<u>#ADD8E6</u>	
<u>GreenYellow</u>	<u>#ADFF2F</u>	
<u>PaleTurquoise</u>	<u>#AFEEEE</u>	
<u>LightSteelBlue</u>	<u>#B0C4DE</u>	
<u>PowderBlue</u>	<u>#B0E0E6</u>	
<u>FireBrick</u>	<u>#B22222</u>	
<u>DarkGoldenRod</u>	<u>#B8860B</u>	
<u>MediumOrchid</u>	<u>#BA55D3</u>	
<u>RosyBrown</u>	<u>#BC8F8F</u>	
<u>DarkKhaki</u>	<u>#BDB76B</u>	
<u>Silver</u>	<u>#C0C0C0</u>	
<u>MediumVioletRed</u>	<u>#C71585</u>	

<u>IndianRed</u>	<u>#CD5C5C</u>	
<u>Peru</u>	<u>#CD853F</u>	
<u>Chocolate</u>	<u>#D2691E</u>	
<u>Tan</u>	<u>#D2B48C</u>	
<u>LightGray</u>	<u>#D3D3D3</u>	
<u>Thistle</u>	<u>#D8BFD8</u>	
<u>Orchid</u>	<u>#DA70D6</u>	
<u>GoldenRod</u>	<u>#DAA520</u>	
<u>PaleVioletRed</u>	<u>#DB7093</u>	
<u>Crimson</u>	<u>#DC143C</u>	
<u>Gainsboro</u>	<u>#DCDCDC</u>	
<u>Plum</u>	<u>#DDA0DD</u>	
<u>BurlyWood</u>	<u>#DEB887</u>	
<u>LightCyan</u>	<u>#E0FFFF</u>	
<u>Lavender</u>	<u>#E6E6FA</u>	
<u>DarkSalmon</u>	<u>#E9967A</u>	

<u>Violet</u>	<u>#EE82EE</u>	
<u>PaleGoldenRod</u>	<u>#EEE8AA</u>	
<u>LightCoral</u>	<u>#F08080</u>	
<u>Khaki</u>	<u>#F0E68C</u>	
<u>AliceBlue</u>	<u>#F0F8FF</u>	
<u>HoneyDew</u>	<u>#F0FFF0</u>	
<u>Azure</u>	<u>#F0FFFF</u>	
<u>SandyBrown</u>	<u>#F4A460</u>	
<u>Wheat</u>	<u>#F5DEB3</u>	
<u>Beige</u>	<u>#F5F5DC</u>	
<u>WhiteSmoke</u>	<u>#F5F5F5</u>	
<u>MintCream</u>	<u>#F5FFFA</u>	
<u>GhostWhite</u>	<u>#F8F8FF</u>	
<u>Salmon</u>	<u>#FA8072</u>	
<u>AntiqueWhite</u>	<u>#FAEBD7</u>	
<u>Linen</u>	<u>#FAF0E6</u>	

<u>LightGoldenRodYellow</u>	<u>#FAFAD2</u>	
<u>OldLace</u>	<u>#FDF5E6</u>	
<u>Red</u>	<u>#FF0000</u>	
<u>Fuchsia</u>	<u>#FF00FF</u>	
<u>Magenta</u>	<u>#FF00FF</u>	
<u>DeepPink</u>	<u>#FF1493</u>	
<u>OrangeRed</u>	<u>#FF4500</u>	
<u>Tomato</u>	<u>#FF6347</u>	
<u>HotPink</u>	<u>#FF69B4</u>	
<u>Coral</u>	<u>#FF7F50</u>	
<u>DarkOrange</u>	<u>#FF8C00</u>	
<u>LightSalmon</u>	<u>#FFA07A</u>	
<u>Orange</u>	<u>#FFA500</u>	
<u>LightPink</u>	<u>#FFB6C1</u>	
<u>Pink</u>	<u>#FFC0CB</u>	
<u>Gold</u>	<u>#FFD700</u>	

<u>PeachPuff</u>	<u>#FFDAB9</u>	
<u>NavajoWhite</u>	<u>#FFDEAD</u>	
<u>Moccasin</u>	<u>#FFE4B5</u>	
<u>Bisque</u>	<u>#FFE4C4</u>	
<u>MistyRose</u>	<u>#FFE4E1</u>	
<u>BlanchedAlmond</u>	<u>#FFEBCD</u>	
<u>PapayaWhip</u>	<u>#FFEFD5</u>	
<u>LavenderBlush</u>	<u>#FFF0F5</u>	
<u>SeaShell</u>	<u>#FFF5EE</u>	
<u>Cornsilk</u>	<u>#FFF8DC</u>	
<u>LemonChiffon</u>	<u>#FFFACD</u>	
<u>FloralWhite</u>	<u>#FFFAF0</u>	
<u>Snow</u>	<u>#FFFAFA</u>	
<u>Yellow</u>	<u>#FFFF00</u>	
<u>LightYellow</u>	<u>#FFFFE0</u>	
<u>Ivory</u>	<u>#FFFFFF</u>	

White	#FFFFFF	
-----------------------	-------------------------	--

[HTML 颜色名](#)

[HTML 脚本](#)

点我分享笔记

HTML 脚本

JavaScript 使 HTML 页面具有更强的动态和交互性。



在线实例

[插入一段脚本](#)

如何将脚本插入 HTML 文档。

[使用 <noscript> 标签](#)

如何应对不支持脚本或禁用脚本的浏览器。

HTML <script> 标签

<script> 标签用于定义客户端脚本，比如 JavaScript。

<script> 元素既可包含脚本语句，也可通过 src 属性指向外部脚本文件。

JavaScript 最常用于图片操作、表单验证以及内容动态更新。

下面的脚本会向浏览器输出"Hello World!":

实例

```
<script> document.write("Hello World!"); </script>
```

尝试一下 »

💡**Tip:** 学习更多关于 Javascript 教程, 请查看 [JavaScript 教程!](#)

HTML<noscript> 标签

<noscript> 标签提供无法使用脚本时的替代内容, 比方在浏览器禁用脚本时, 或浏览器不支持客户端脚本时。

<noscript>元素可包含普通 HTML 页面的 body 元素中能够找到的所有元素。

只有在浏览器不支持脚本或者禁用脚本时, 才会显示 <noscript> 元素中的内容:

实例

```
<script> document.write("Hello World!") </script> <noscript>抱歉, 你的浏览器不支持 JavaScript!</noscript>
```

尝试一下 »

JavaScript 体验(来自本站 javascript 教程)

JavaScript 实例代码:

JavaScript 可以直接在 HTML 输出:

```
document.write("<p>这是一个段落。</p>");
```


尝试一下 »

JavaScript 事件响应:

```
<button type="button" onclick="myFunction()">点我! </button>
```

尝试一下 »

JavaScript 处理 HTML 样式:

```
document.getElementById("demo").style.color="#ff0000";
```

尝试一下 »

HTML 脚本标签

| 标签 | 描述 |
|------------|------------------|
| <script> | 定义了客户端脚本 |
| <noscript> | 定义了不支持脚本浏览器输出的文本 |

[HTML 颜色值](#)

[HTML 字符实体](#)

点我分享笔记

HTML5 简介

HTML5 是 HTML 最新的修订版本，2014 年 10 月由万维网联盟（W3C）完成标准制定。

HTML5 的设计目的是为了在移动设备上支持多媒体。

HTML5 简单易学。

什么是 HTML5?

HTML5 是下一代 HTML 标准。

HTML , HTML 4.01 的上一个版本诞生于 1999 年。自从那以后，Web 世界已经经历了巨变。

HTML5 仍处于完善之中。然而，大部分现代浏览器已经具备了某些 HTML5 支持。

HTML5 是如何起步的?

HTML5 是 W3C 与 WHATWG 合作的结果,WHATWG 指 Web Hypertext Application Technology Working Group。

WHATWG 致力于 web 表单和应用程序，而 W3C 专注于 XHTML 2.0。在 2006 年，双方决定进行合作，来创建一个新版本的 HTML。

HTML5 中的一些有趣的新特性：

- 用于绘画的 canvas 元素
- 用于媒介回放的 video 和 audio 元素
- 对本地离线存储的更好的支持
- 新的特殊内容元素，比如 article、footer、header、nav、section
- 新的表单控件，比如 calendar、date、time、email、url、search

HTML5 <!DOCTYPE>

<!doctype> 声明必须位于 HTML5 文档中的第一行,使用非常简单:

```
<!DOCTYPE html>
```

最小的 HTML5 文档

下面是一个简单的 HTML5 文档:

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容..... </body> </html>
```

注意: 对于中文网页需要使用 **<meta charset="utf-8">** 声明编码, 否则会出现乱码。

HTML5 的改进

- 新元素
- 新属性
- 完全支持 CSS3
- Video 和 Audio
- 2D/3D 制图
- 本地存储
- 本地 SQL 数据

- Web 应用
-

HTML5 多媒体

使用 HTML5 你可以简单的在网页中播放 视频(video)与音频 (audio) 。

- HTML5 `<video>`
 - HTML5 `<audio>`
-

HTML5 应用

使用 HTML5 你可以简单地开发应用

- 本地数据存储
 - 访问本地文件
 - 本地 SQL 数据
 - 缓存引用
 - Javascript 工作者
 - XMLHttpRequest 2
-

HTML5 图形

使用 HTML5 你可以简单的绘制图形:

- 使用 `<canvas>` 元素。

- 使用内联 SVG。
 - 使用 CSS3 2D 转换、CSS3 3D 转换。
-

HTML5 使用 CSS3

- 新选择器
- 新属性
- 动画
- 2D/3D 转换
- 圆角
- 阴影效果
- 可下载的字體

了解更多 CSS3 知识请查看本站的 [CSS3 教程](#)。

语义元素

HTML5 添加了很多语义元素如下所示：

| 标签 | 描述 |
|-----------|--------------|
| <article> | 定义页面独立的内容区域。 |
| <aside> | 定义页面的侧边栏内容。 |

| | |
|--------------|---------------------------------------|
| <bdi> | 允许您设置一段文本，使其脱离其父元素的文本方向设置。 |
| <command> | 定义命令按钮，比如单选按钮、复选框或按钮 |
| <details> | 用于描述文档或文档某个部分的细节 |
| <dialog> | 定义对话框，比如提示框 |
| <summary> | 标签包含 details 元素的标题 |
| <figure> | 规定独立的流内容（图像、图表、照片、代码等等）。 |
| <figcaption> | 定义 <figure> 元素的标题 |
| <footer> | 定义 section 或 document 的页脚。 |
| <header> | 定义了文档的头部区域 |
| <mark> | 定义带有记号的文本。 |
| <meter> | 定义度量衡。仅用于已知最大和最小值的度量。 |
| <nav> | 定义导航链接的部分。 |
| <progress> | 定义任何类型的任务的进度。 |
| <ruby> | 定义 ruby 注释（中文注音或字符）。 |
| <rt> | 定义字符（中文注音或字符）的解释或发音。 |
| <rp> | 在 ruby 注释中使用，定义不支持 ruby 元素的浏览器所显示的内容。 |

| | |
|-----------|----------------------|
| <section> | 定义文档中的节（section、区段）。 |
| <time> | 定义日期或时间。 |
| <wbr> | 规定在文本中的何处适合添加换行符。 |

HTML5 表单

新表单元素, 新属性, 新输入类型, 自动验证。

已移除元素

以下的 HTML 4.01 元素在 HTML5 中已经被删除:

- <acronym>
- <applet>
- <basefont>
- <big>
- <center>
- <dir>
-
- <frame>
- <frameset>

- <noframes>

- <strike>

每一章中的实例

通过我们的 HTML 编辑器，您能够编辑 HTML，然后点击按钮来查看结果。

实例

```
<!DOCTYPE HTML> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body>
<video width="320" height="240" controls> <source src="movie.mp4" type="video/mp4"> <source src="movie.ogg"
type="video/ogg"> 你的浏览器不支持 video 标签。 </video> </body> </html>
```

尝试一下 »

点击 "尝试一下" 按钮查看在线运行结果。

HTML5 浏览器支持

最新版本的 Safari、Chrome、Firefox 以及 Opera 支持某些 HTML5 特性。Internet Explorer 9 将支持某些 HTML5 特性。



IE9 以下版本浏览器兼容 HTML5 的方法，使用本站的静态资源的 html5shiv 包：

```
<!--[if lt IE 9]> <script src="http://cdn.static.runoob.com/libs/html5shiv/3.7/html5shiv.min.js"></script> <![endif]-->
```

载入后，初始化新标签的 CSS：

```
/*html5*/ article,aside,dialog,footer,header,section,nav,figure,menu{display:block}
```

HTML5 参考手册

在本站中你可以找到关于 HTML5 的标签及属性描述，详细请点击 [HTML5 参考手册](#)。

[XHTML 简介](#)

[HTML5 浏览器支持](#)

[点我分享笔记](#)

HTML5 浏览器支持

你可以让一些较早的浏览器（不支持 HTML5）支持 HTML5。

HTML5 浏览器支持

现代的浏览器都支持 HTML5。

此外，所有浏览器，包括旧的和最新的，对无法识别的元素会作为内联元素自动处理。

正因为如此，你可以 "教会" 浏览器处理 "未知" 的 HTML 元素。



甚至你可以教会 IE6 (Windows XP 2001) 浏览器处理未知的 HTML 元素。

将 HTML5 元素定义为块元素

HTML5 定了 8 个新的 HTML 语义 (**semantic**) 元素。所有这些元素都是 **块级** 元素。

为了能让旧版本的浏览器正确显示这些元素，你可以设置 CSS 的 **display** 属性值为 **block**:

实例

```
header, section, footer, aside, nav, main, article, figure { display: block; }
```

为 HTML 添加新元素

你可以为 HTML 添加新的元素。

该实例向 HTML 添加的新的元素，并为该元素定义样式，元素名为 **<myHero>**：

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>为 HTML 添加新元素</title> <script>
document.createElement("myHero")
</script> <style>
myHero { display: block; background-color: #ddd; padding: 50px; font-size: 30px; }
</style> </head> <body> <h1>我的第一个标题</h1> <p>我的第一个段落。</p> <myHero>我的第一个新元素</myHero> </body>
</html>
```

尝试一下 »

JavaScript 语句 `document.createElement("myHero")` 是为 IE 浏览器添加新的元素。

Internet Explorer 浏览器问题

你可以使用以上的方法来为 IE 浏览器添加 HTML5 元素，但是：



Internet Explorer 8 及更早 IE 版本的浏览器不支持以上的方式。

我们可以使用 Sjoerd Visscher 创建的 "HTML5 Enabling JavaScript", " **shiv**" 来解决该问题:

```
<!--[if lt IE 9]>
  <script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
<![endif]-->
```

以上代码是一个注释，作用是在 IE 浏览器的版本小于 IE9 时将读取 html5.js 文件，并解析它。

注意：国内用户请使用本站静态资源库（Google 资源库在国内不稳定）：

```
<!--[if lt IE 9]>
  <script src="https://lf6-cdn-tos.bytecdntp.com/cdn/expire-1-M/html5shiv/3.7.3/html5shiv.min.js"></script>
<![endif]-->
```

针对 IE 浏览器 html5shiv 是比较好的解决方案。html5shiv 主要解决 HTML5 提出的新的元素不被 IE6-8 识别，这些新元素不能作为父节点包裹子元素，并且不能应用 CSS 样式。

完美的 Shiv 解决方案

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>渲染 HTML5</title> <!--[if lt IE 9]> <script
src="https://lf6-cdn-tos.bytecdntp.com/cdn/expire-1-M/html5shiv/3.7.3/html5shiv.min.js"></script> <![endif]-->
</head> <body> <h1>我的第一篇文章</h1> <article> 菜鸟教程 —— 学的不仅是技术，更是梦想!!! </article> </body> </html>
```

尝试一下 »

html5shiv.js 引用代码必须放在 <head> 元素中，因为 IE 浏览器在解析 HTML5 新元素时需要先加载该文件。

[HTML5 教程](#)

[HTML5 新元素](#)

点我分享笔记

HTML5 新元素

HTML5 新元素

自 1999 年以后 HTML 4.01 已经改变了很多,今天, 在 HTML 4.01 中的几个已经被废弃, 这些元素在 HTML5 中已经被删除或重新定义。为了更好地处理今天的互联网应用, HTML5 添加了很多新元素及功能, 比如: 图形的绘制, 多媒体内容, 更好的页面结构, 更好的形式 处理, 和几个 api 拖放元素, 定位, 包括网页 应用程序缓存, 存储, 网络工作者, 等。

<canvas> 新元素

标签	描述
<canvas>	标签定义图形, 比如图表和其他图像。该标签基于 JavaScript 的绘图 API

新多媒体元素

标签	描述
<audio>	定义音频内容
<video>	定义视频 (video 或者 movie)
<source>	定义多媒体资源 <video> 和 <audio>
<embed>	定义嵌入的内容, 比如插件。

<track>	为诸如 <video> 和 <audio> 元素之类的媒介规定外部文本轨道。
---------	--

新表单元素

标签	描述
<datalist>	定义选项列表。请与 input 元素配合使用该元素，来定义 input 可能的值。
<keygen>	规定用于表单的密钥对生成器字段。
<output>	定义不同类型的输出，比如脚本的输出。

新的语义和结构元素

HTML5 提供了新的元素来创建更好的页面结构：

标签	描述
<article>	定义页面独立的内容区域。
<aside>	定义页面的侧边栏内容。
<bdi>	允许您设置一段文本，使其脱离其父元素的文本方向设置。
<command>	定义命令按钮，比如单选按钮、复选框或按钮
<details>	用于描述文档或文档某个部分的细节

<dialog>	定义对话框，比如提示框
<summary>	标签包含 details 元素的标题
<figure>	规定独立的流内容（图像、图表、照片、代码等等）。
<figcaption>	定义 <figure> 元素的标题
<footer>	定义 section 或 document 的页脚。
<header>	定义了文档的头部区域
<mark>	定义带有记号的文本。
<meter>	定义度量衡。仅用于已知最大和最小值的度量。
<nav>	定义导航链接的部分。
<progress>	定义任何类型的任务的进度。
<ruby>	定义 ruby 注释（中文注音或字符）。
<rt>	定义字符（中文注音或字符）的解释或发音。
<rp>	在 ruby 注释中使用，定义不支持 ruby 元素的浏览器所显示的内容。
<section>	定义文档中的节（section、区段）。
<time>	定义日期或时间。
<wbr>	规定在文本中的何处适合添加换行符。

已移除的元素

以下的 HTML 4.01 元素在 HTML5 中已经被删除:

- <acronym>
- <applet>
- <basefont>
- <big>
- <center>
- <dir>
-
- <frame>
- <frameset>
- <noframes>
- <strike>
- <tt>

HTML5 浏览器支持

HTML5 Canvas

点我分享笔记

HTML5 Canvas

<canvas> 标签定义图形，比如图表和其他图像，您必须使用脚本来绘制图形。
在画布上（Canvas）画一个红色矩形，渐变矩形，彩色矩形，和一些彩色的文字。

什么是 canvas?

HTML5 <canvas> 元素用于图形的绘制，通过脚本 (通常是 JavaScript)来完成。
<canvas> 标签只是图形容器，您必须使用脚本来绘制图形。
你可以通过多种方法使用 canvas 绘制路径,盒、圆、字符以及添加图像。

浏览器支持

表格中的数字表示支持 <canvas> 元素的第一个浏览器版本号。

元素					
<canvas>	4.0	9.0	2.0	3.1	9.0

创建一个画布（Canvas）

一个画布在网页中是一个矩形框，通过 `<canvas>` 元素来绘制。

注意：默认情况下 `<canvas>` 元素没有边框和内容。

`<canvas>` 简单实例如下：

```
<canvas id="myCanvas" width="200" height="100"></canvas>
```

注意：标签通常需要指定一个 `id` 属性（脚本中经常引用），`width` 和 `height` 属性定义的画布的大小。

提示：你可以在 HTML 页面中使用多个 `<canvas>` 元素。

使用 `style` 属性来添加边框：

实例

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid #000000;"> </canvas>
```

[尝试一下 »](#)

使用 JavaScript 来绘制图像

`canvas` 元素本身是没有绘图能力的。所有的绘制工作必须在 JavaScript 内部完成：

实例

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); ctx.fillStyle="#FF0000";  
ctx.fillRect(0,0,150,75);
```

[尝试一下 »](#)

实例解析：

首先，找到 `<canvas>` 元素：

```
var c=document.getElementById("myCanvas");
```

然后，创建 context 对象：

```
var ctx=c.getContext("2d");
```

getContext("2d") 对象是内建的 HTML5 对象，拥有多种绘制路径、矩形、圆形、字符以及添加图像的方法。

下面的两行代码绘制一个红色的矩形：

```
ctx.fillStyle="#FF0000";
```

```
ctx.fillRect(0,0,150,75);
```

设置 fillStyle 属性可以是 CSS 颜色，渐变，或图案。fillStyle 默认设置是#000000（黑色）。

fillRect(x,y,width,height) 方法定义了矩形当前的填充方式。

Canvas 坐标

canvas 是一个二维网格。

canvas 的左上角坐标为 (0,0)

上面的 fillRect 方法拥有参数 (0,0,150,75)。

意思是：在画布上绘制 150x75 的矩形，从左上角开始 (0,0)。

坐标实例

如下图所示，画布的 X 和 Y 坐标用于在画布上对绘画进行定位。鼠标移动的矩形框上，显示定位坐标。

X

Y

Canvas - 路径

在 Canvas 上画线，我们将使用以下两种方法：

- moveTo(x,y) 定义线条开始坐标
- lineTo(x,y) 定义线条结束坐标

绘制线条我们必须使用到 "ink" 的方法, 就像 stroke().

实例

定义开始坐标(0,0), 和结束坐标 (200,100)。然后使用 stroke() 方法来绘制线条:

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); ctx.moveTo(0,0); ctx.lineTo(200,100);  
ctx.stroke();
```

[尝试一下 »](#)

在 canvas 中绘制圆形, 我们将使用以下方法:

arc(x,y,r,start,stop)

实际上我们在绘制圆形时使用了 "ink" 的方法, 比如 stroke() 或者 fill()。

实例

使用 arc() 方法 绘制一个圆:

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); ctx.beginPath();  
ctx.arc(95,50,40,0,2*Math.PI); ctx.stroke();
```

[尝试一下 »](#)

Canvas - 文本

使用 canvas 绘制文本, 重要的属性和方法如下:

- font - 定义字体

- `fillText(text,x,y)` - 在 canvas 上绘制实心的文本
- `strokeText(text,x,y)` - 在 canvas 上绘制空心的文本

使用 `fillText()`:

实例

使用 "Arial" 字体在画布上绘制一个高 30px 的文字（实心）：

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); ctx.font="30px Arial";  
ctx.fillText("Hello World",10,50);
```

[尝试一下 »](#)

使用 `strokeText()`:

实例

使用 "Arial" 字体在画布上绘制一个高 30px 的文字（空心）：

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); ctx.font="30px Arial";  
ctx.strokeText("Hello World",10,50);
```

[尝试一下 »](#)

Canvas - 渐变

渐变可以填充在矩形, 圆形, 线条, 文本等等, 各种形状可以自己定义不同的颜色。

以下有两种不同的方式来设置 Canvas 渐变:

- createLinearGradient(x,y,x1,y1) - 创建线条渐变
- createRadialGradient(x,y,r,x1,y1,r1) - 创建一个径向/圆渐变

当我们使用渐变对象，必须使用两种或两种以上的停止颜色。

addColorStop()方法指定颜色停止，参数使用坐标来描述，可以是 0 至 1.

使用渐变，设置 fillStyle 或 strokeStyle 的值为 渐变，然后绘制形状，如矩形，文本，或一条线。

使用 createLinearGradient():

实例

创建一个线性渐变。使用渐变填充矩形:

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); // 创建渐变 var  
grd=ctx.createLinearGradient(0,0,200,0); grd.addColorStop(0,"red"); grd.addColorStop(1,"white"); // 填充渐变  
ctx.fillStyle=grd; ctx.fillRect(10,10,150,80);
```

[尝试一下 »](#)

使用 createRadialGradient():

实例

创建一个径向/圆渐变。使用渐变填充矩形:

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); // 创建渐变 var  
grd=ctx.createRadialGradient(75,50,5,90,60,100); grd.addColorStop(0,"red"); grd.addColorStop(1,"white"); // 填  
充渐变 ctx.fillStyle=grd; ctx.fillRect(10,10,150,80);
```

[尝试一下 »](#)

Canvas - 图像

把一幅图像放置到画布上, 使用以下方法:

- `drawImage(image,x,y)`

使用图像:



实例

把一幅图像放置到画布上:

JavaScript:

```
var c=document.getElementById("myCanvas"); var ctx=c.getContext("2d"); var  
img=document.getElementById("scream"); ctx.drawImage(img,10,10);
```

尝试一下 »

HTML Canvas 参考手册

标签的完整属性可以参考 [Canvas 参考手册](#).

HTML <canvas> 标签

Tag	描述
<canvas>	HTML5 的 canvas 元素使用 JavaScript 在网页上绘制图像。

更多内容可参考: [学习 HTML5 Canvas 这一篇文章就够了](#)

[HTML5 新元素](#)

[HTML5 SVG](#)

8 篇笔记

HTML5 SVG

SVG 定义为可缩放矢量图形。

HTML5 支持内联 SVG。

HTML `<svg>` 元素是 SVG 图形的容器。

SVG 有多种绘制路径、框、圆、文本和图形图像的方法。

什么是 SVG?

- SVG 指可伸缩矢量图形 (Scalable Vector Graphics)
- SVG 用于定义用于网络的基于矢量的图形
- SVG 使用 XML 格式定义图形
- SVG 图像在放大或改变尺寸的情况下其图形质量不会有损失
- SVG 是万维网联盟的标准

SVG 优势

与其他图像格式相比（比如 JPEG 和 GIF），使用 SVG 的优势在于：

- SVG 图像可通过文本编辑器来创建和修改
 - SVG 图像可被搜索、索引、脚本化或压缩
 - SVG 是可伸缩的
 - SVG 图像可在任何的分辨率下被高质量地打印
 - SVG 可在图像质量不下降的情况下被放大
-

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

元素					
<svg>	4.0	9.0	3.0	3.2	10.1

把 SVG 直接嵌入 HTML 页面

在 HTML5 中，您能够将 SVG 元素直接嵌入 HTML 页面中。

SVG 圆形

实例

```
<svg xmlns="http://www.w3.org/2000/svg" version="1.1"> <circle cx="100" cy="50" r="40" stroke="black" stroke-width="2" fill="red" /> </svg>
```

[尝试一下 »](#)

结果：

SVG 五角星

实例

```
<!DOCTYPE html> <html> <body> <svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="190"> <polygon
points="100,10 40,180 190,60 10,60 160,180" style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;">
</svg> </body> </html>
```

尝试一下 »

结果：
学习更多关于 SVG 教程, 请访问 [SVG 教程](#)。

SVG 与 Canvas 两者间的区别

SVG 是一种使用 XML 描述 2D 图形的语言。
Canvas 通过 JavaScript 来绘制 2D 图形。
SVG 基于 XML，这意味着 SVG DOM 中的每个元素都是可用的。您可以为某个元素附加 JavaScript 事件处理器。
在 SVG 中，每个被绘制的图形均被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。
Canvas 是逐像素进行渲染的。在 canvas 中，一旦图形被绘制完成，它就不会继续得到浏览器的关注。如果其位置发生变化，那么整个场景也需要重新绘制，包括任何或许已被图形覆盖的对象。

Canvas 与 SVG 的比较

下表列出了 canvas 与 SVG 之间的一些不同之处。

Canvas	SVG
<ul style="list-style-type: none">• 依赖分辨率• 不支持事件处理器• 弱的文本渲染能力	<ul style="list-style-type: none">• 不依赖分辨率• 支持事件处理器• 最适合带有大型渲染区域的应用程序（比如谷歌地图）

- | | |
|--|--|
| <ul style="list-style-type: none">•能够以 .png 或 .jpg 格式保存结果图像•最适合图像密集型的游戏，其中的许多对象会被频繁重绘 | <ul style="list-style-type: none">•复杂度高会减慢渲染速度（任何过度使用 DOM 的应用都不快）•不适合游戏应用 |
|--|--|

[HTML5 Canvas](#)

[HTML5 MathML](#)

点我分享笔记

HTML5 MathML

HTML5 可以在文档中使用 MathML 元素，对应的标签是 `$...$`。

MathML 是数学标记语言，是一种基于 XML（标准通用标记语言的子集）的标准，用来在互联网上书写数学符号和公式的置标语言。

注意：目前只有 Firefox 或 Safari 浏览器支持，大部分浏览器还不支持 MathML 标签，如果你的浏览器不支持该标签，可以使用最新版的 Firefox 或 Safari 浏览器查看。

另外我们可以使用第三方的样式库来支持数学标记，本章节使用的样式库可以点击下面按钮下载：

[下载 mathml.zip](#)

MathML 实例

以下是一个简单的 MathML 实例：

实例

```
<!DOCTYPE html>
<html>
```

```
<head>
  <meta charset="UTF-8">
  <title>菜鸟教程(runoob.com)</title>
</head>

<body>

  <math xmlns="http://www.w3.org/1998/Math/MathML">

    <mrow>
      <msup><mi>a</mi><mn>2</mn></msup>
      <mo>+</mo>

      <msup><mi>b</mi><mn>2</mn></msup>
      <mo>=</mo>

      <msup><mi>c</mi><mn>2</mn></msup>
    </mrow>

  </math>

</body>
</html>
```

尝试一下 »

使用第三方库来支持：

实例

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>菜鸟教程(runoob.com)</title>
    <script type="text/javascript" src="https://static.jyshare.com/assets/js/mathml/mspace.js"></script>
  </head>

  <body>

    <math xmlns="http://www.w3.org/1998/Math/MathML">

      <mrow>
        <msup><mi>a</mi><mn>2</mn></msup>
        <mo>+</mo>

        <msup><mi>b</mi><mn>2</mn></msup>
        <mo>=</mo>

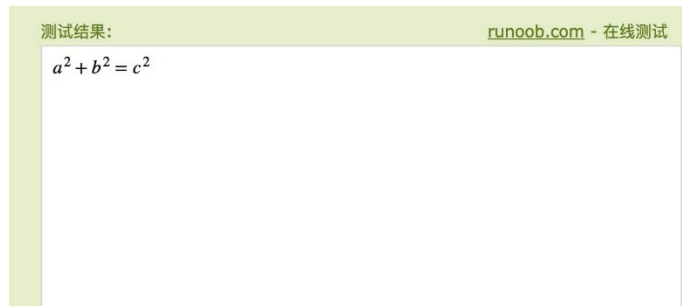
        <msup><mi>c</mi><mn>2</mn></msup>
      </mrow>

    </math>

  </body>
</html>
```

[尝试一下 »](#)

运行结果图，如下所示：



以下实例添加了一些运算符：

实例

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>菜鸟教程 (runoob.com)</title>
  </head>

  <body>

    <math xmlns="http://www.w3.org/1998/Math/MathML">

      <mrow>
        <mrow>

          <msup>
```

```
        <mi>x</mi>
        <mn>2</mn>
    </msup>

    <mo>+</mo>

    <mrow>
        <mn>4</mn>
        <mo></mo>
        <mi>x</mi>
    </mrow>

    <mo>+</mo>
    <mn>4</mn>

</mrow>

    <mo>=</mo>
    <mn>0</mn>

</mrow>
</math>

</body>
</html>
```

[尝试一下 »](#)

使用第三方库来支持：

实例

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>菜鸟教程(runoob.com)</title>
    <script type="text/javascript" src="https://static.jyshare.com/assets/js/mathml/mspace.js"></script>
  </head>

  <body>

    <math xmlns="http://www.w3.org/1998/Math/MathML">

      <mrow>
        <mrow>

          <msup>
            <mi>x</mi>
            <mn>2</mn>
          </msup>

          <mo>+</mo>

          <mrow>
            <mn>4</mn>
            <mi>x</mi>
          </mrow>
        </mrow>
      </math>
    </body>
</html>
```

```
        </mrow>

        <mo>+</mo>
        <mn>4</mn>

    </mrow>

    <mo>=</mo>
    <mn>0</mn>

</mrow>
</math>

</body>
</html>
```

尝试一下 »

运行结果图，如下所示：



以下实例是一个 2×2 矩阵，可以在 Firefox 3.5 以上版本查看到效果：

实例

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>菜鸟教程(runoob.com)</title>
  </head>

  <body>
    <math xmlns="http://www.w3.org/1998/Math/MathML">

      <mrow>
        <mi>A</mi>
        <mo>=</mo>

        <mfenced open="[" close="]">

          <mtable>
            <mtr>
              <mtd><mi>x</mi></mtd>
              <mtd><mi>y</mi></mtd>
            </mtr>

            <mtr>
              <mtd><mi>z</mi></mtd>
              <mtd><mi>w</mi></mtd>
            </mtr>
          </mtable>
        </mfenced>
      </mrow>
    </math>
```

```
        </mtr>
      </mtable>

    </mfenced>
  </mrow>
</math>

</body>
</html>
```

[尝试一下 »](#)

使用第三方库来支持：

实例

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>菜鸟教程(runoob.com)</title>
    <script type="text/javascript" src="https://static.jyshare.com/assets/js/mathml/mspace.js"></script>
  </head>

  <body>
    <math xmlns="http://www.w3.org/1998/Math/MathML">

      <mrow>
        <mi>A</mi>
```

```
<mo>=</mo>

<mfenced open="[" close="]">

  <table>
    <mtr>
      <td><mi>x</mi></td>
      <td><mi>y</mi></td>
    </mtr>

    <mtr>
      <td><mi>z</mi></td>
      <td><mi>w</mi></td>
    </mtr>
  </table>

</mfenced>
</mrow>
</math>

</body>
</html>
```

[尝试一下 »](#)

运行结果图，如下所示：

测试结果:

runoob.com - 在线测试

$$A = \begin{bmatrix} x & y \\ z & w \end{bmatrix}$$

[HTML5 SVG](#)

[HTML5 拖放](#)

点我分享笔记

HTML5 拖放 (Drag 和 Drop)

拖放 (Drag 和 drop) 是 HTML5 标准的组成部分。

RUNOOB.COM

将 **RUNOOB.COM** 图标拖动到矩形框中。

拖放

拖放是一种常见的特性，即抓取对象以后拖到另一个位置。
在 HTML5 中，拖放是标准的一部分，任何元素都能够拖放。

浏览器支持

Internet Explorer 9+, Firefox, Opera, Chrome, 和 Safari 支持拖动。
注意:Safari 5.1.2 不支持拖动。

HTML5 拖放实例

下面的例子是一个简单的拖放实例：

实例

```
<!DOCTYPE HTML> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> <style
type="text/css"> #div1 {width:350px;height:70px;padding:10px;border:1px solid #aaaaaa;} </style> <script>
function allowDrop(ev) { ev.preventDefault(); } function drag(ev)
{ ev.dataTransfer.setData("Text",ev.target.id); } function drop(ev) { ev.preventDefault(); var
data=ev.dataTransfer.getData("Text"); ev.target.appendChild(document.getElementById(data)); } </script>
</head> <body> <p>拖动 RUNOOB.COM 图片到矩形框中:</p> <div id="div1" ondrop="drop(event)"
ondragover="allowDrop(event)"></div> <br>  </body> </html>
```

尝试一下 »

它看上去也许有些复杂，不过我们可以分别研究拖放事件的不同部分。

设置元素为可拖放

首先，为了使元素可拖动，把 `draggable` 属性设置为 `true`：

```
<img draggable="true">
```

拖动什么 - `ondragstart` 和 `setData()`

然后，规定当元素被拖动时，会发生什么。

在上面的例子中，`ondragstart` 属性调用了函数 `drag(event)`，它规定了被拖动的数据。

`dataTransfer.setData()` 方法设置被拖数据的数据类型和值：

```
function drag(ev)
{
    ev.dataTransfer.setData("Text",ev.target.id);
}
```

`Text` 是一个 `DOMString` 表示要添加到 `drag object` 的拖动数据的类型。值是可拖动元素的 `id` (`"drag1"`)。

放到何处 - `ondragover`

`ondragover` 事件规定在何处放置被拖动的数据。

默认地，无法将数据/元素放置到其他元素中。如果需要设置允许放置，我们必须阻止对元素的默认处理方式。

这要通过调用 `ondragover` 事件的 `event.preventDefault()` 方法：

```
event.preventDefault()
```

进行放置 - `ondrop`

当放置被拖数据时，会发生 `drop` 事件。

在上面的例子中，`ondrop` 属性调用了函数 `drop(event)`：


```
function drop(ev)
{
    ev.preventDefault();
    var data=ev.dataTransfer.getData("Text");
    ev.target.appendChild(document.getElementById(data));
}
```

代码解释：

- 调用 `preventDefault()` 来避免浏览器对数据的默认处理（`drop` 事件的默认行为是以链接形式打开）
- 通过 `dataTransfer.getData("Text")` 方法获得被拖的数据。该方法将返回在 `setData()` 方法中设置为相同类型的任何数据。
- 被拖数据是被拖元素的 `id` ("drag1")
- 把被拖元素追加到放置元素（目标元素）中

更多实例

[来回拖放图片](#)

如何在两个 `<div>` 元素之间拖放图像。

[HTML5 MathML](#)

[HTML5 地理定位](#)

HTML5 Geolocation（地理定位）

HTML5 Geolocation（地理定位）用于定位用户的位置。

定位用户的位置

HTML5 Geolocation API 用于获得用户的地理位置。

鉴于该特性可能侵犯用户的隐私，除非用户同意，否则用户位置信息是不可用的。

Internet Explorer 9+, Firefox, Chrome, Safari 和 Opera 支持 Geolocation（地理定位）。

注意: Geolocation（地理定位）对于拥有 GPS 的设备，比如 iPhone，地理定位更加精确。

HTML5 - 使用地理定位

请使用 `getCurrentPosition()` 方法来获得用户的位置。

下例是一个简单的地理定位实例，可返回用户位置的经度和纬度：

实例

```
var x=document.getElementById("demo"); function getLocation() { if (navigator.geolocation) {  
navigator.geolocation.getCurrentPosition(showPosition); } else { x.innerHTML="该浏览器不支持获取地理位置。"; } }  
function showPosition(position) { x.innerHTML="纬度： " + position.coords.latitude + "<br>经度： " +  
position.coords.longitude; }
```

[尝试一下 »](#)

实例解析：

- 检测是否支持地理定位
- 如果支持，则运行 `getCurrentPosition()` 方法。如果不支持，则向用户显示一段消息。

- 如果 `getCurrentPosition()` 运行成功，则向参数 `showPosition` 中规定的函数返回一个 `coordinates` 对象
- `showPosition()` 函数获得并显示经度和纬度

上面的例子是一个非常基础的地理定位脚本，不含错误处理。

处理错误和拒绝

`getCurrentPosition()` 方法的第二个参数用于处理错误。它规定当获取用户位置失败时运行的函数：

实例

```
function showError(error) { switch(error.code) { case error.PERMISSION_DENIED: x.innerHTML="用户拒绝对获取地理位置的请求。" break; case error.POSITION_UNAVAILABLE: x.innerHTML="位置信息是不可用的。" break; case error.TIMEOUT: x.innerHTML="请求用户地理位置超时。" break; case error.UNKNOWN_ERROR: x.innerHTML="未知错误。" break; } }
```

尝试一下 »

错误代码：

- Permission denied - 用户不允许地理定位
- Position unavailable - 无法获取当前位置
- Timeout - 操作超时

在地图中显示结果

如需在地图中显示结果，您需要访问可使用经纬度的地图服务，比如谷歌地图或百度地图：

实例

```
function showPosition(position) { var latlon=position.coords.latitude+","+position.coords.longitude; var  
img_url="http://maps.googleapis.com/maps/api/staticmap?center=" +latlon+"&zoom=14&size=400x300&sensor=false";  
document.getElementById("mapholder").innerHTML="<img src='"+img_url+"'>"; }
```

尝试一下 »

在上例中，我们使用返回的经纬度数据在谷歌地图中显示位置（使用静态图像）。

[Google 地图脚本](#)

上面的链接向您演示如何使用脚本来显示带有标记、缩放和拖曳选项的交互式地图。

给定位置的信息

本页演示的是如何在地图上显示用户的位置。不过，地理定位对于给定位置的信息同样很有用处。

可用于：

- 更新本地信息
- 显示用户周围的兴趣点
- 交互式车载导航系统 (GPS)

getCurrentPosition() 方法 - 返回数据

T 若成功，则 `getCurrentPosition()` 方法返回对象。始终会返回 `latitude`、`longitude` 以及 `accuracy` 属性。如果可用，则会返回其他下面的属性。

属性	描述
<code>coords.latitude</code>	十进制数的纬度

coords.longitude	十进制数的经度
coords.accuracy	位置精度
coords.altitude	海拔，海平面以上以米计
coords.altitudeAccuracy	位置的海拔精度
coords.heading	方向，从正北开始以度计
coords.speed	速度，以米/每秒计
timestamp	响应的日期/时间

Geolocation 对象 - 其他有趣的方法

watchPosition() - 返回用户的当前位置，并继续返回用户移动时的更新位置（就像汽车上的 GPS）。

clearWatch() - 停止 watchPosition() 方法

下面的例子展示 watchPosition() 方法。您需要一台精确的 GPS 设备来测试该例（比如 iPhone）：

实例

```
var x=document.getElementById("demo"); function getLocation() { if (navigator.geolocation) {
navigator.geolocation.watchPosition(showPosition); } else { x.innerHTML="该浏览器不支持获取地理位置。"; } }
function showPosition(position) { x.innerHTML="纬度： " + position.coords.latitude + "<br>经度： " +
position.coords.longitude; }
```

[HTML5 拖放](#)

1 篇笔记 写笔记

1. Kai
790***286@qq.com
参考地址

265

百度获取经纬度的例子（各浏览器适用，含 IE5）：

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
  <title></title>
  <!-- 引入百度 API, "ak=" 后面一串码是密钥, 最好自己申请 -->
  <script type="text/javascript" src="https://api.map.baidu.com/api?v=2.0&ak=7a6QKaIilZftIMmKGAF LG7QT1GLfInc g"></script>
</head>
<body>
  <input type="button" onclick="getLocation()" value="确认" />
  <div id="position"></div>
  <script type="text/javascript">
var x = document.getElementById('position');
function getLocation() {
  // 创建百度地理位置实例, 代替 navigator.geolocation
  var geolocation = new BMap.Geolocation();
```

```
geolocation.getCurrentPosition(function(e) {  
    if(this.getStatus() == BMAP_STATUS_SUCCESS){  
        // 百度 geolocation 的经纬度属性不同, 此处是 point.lat 而不是 coords.latitude  
        x.innerHTML = '纬度: ' + e.point.lat + '<br/>经度: ' + e.point.lng;  
    } else {  
        x.innerHTML = 'failed' + this.getStatus();  
    }  
});  
}  
</script>  
</body>  
</html>
```

[Kai](#)

HTML5 Video(视频)

很多站点都会使用到视频. HTML5 提供了展示视频的标准。

检测您的浏览器是否支持 HTML5 视频：

检测

Web 站点上的视频

直到现在，仍然不存在一项旨在网页上显示视频的标准。

今天，大多数视频是通过插件（比如 Flash）来显示的。然而，并非所有浏览器都拥有同样的插件。

HTML5 规定了一种通过 video 元素来包含视频的标准方法。

浏览器支持

Internet Explorer 9+, Firefox, Opera, Chrome, 和 Safari 支持 <video> 元素.

注意: Internet Explorer 8 或者更早的 IE 版本不支持 <video> 元素。

HTML5 (视频)- 如何工作

如需在 HTML5 中显示视频，您所有需要的是：

实例

```
<video width="320" height="240" controls> <source src="movie.mp4" type="video/mp4"> <source src="movie.ogg" type="video/ogg"> 您的浏览器不支持 Video 标签。 </video>
```

尝试一下 »

<video> 元素提供了 播放、暂停和音量控件来控制视频。

同时 <video> 元素也提供了 width 和 height 属性控制视频的尺寸.如果设置的高度和宽度，所需的视频空间会在页面加载时保留。如果没有设置这些属性，浏览器不知道大小的视频，浏览器就不能再加载时保留特定的空间，页面就会根据原始视频的大小而改变。

<video> 与</video> 标签之间插入的内容是提供给不支持 video 元素的浏览器显示的。

<video> 元素支持多个 <source> 元素. <source> 元素可以链接不同的视频文件。浏览器将使用第一个可识别的格式：

视频格式与浏览器的支持

当前， <video> 元素支持三种视频格式： MP4, WebM, 和 Ogg:

浏览器	MP4	WebM	Ogg
-----	-----	------	-----

Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	NO	NO
Opera	YES (从 Opera 25 起)	YES	YES

- MP4 = 带有 H.264 视频编码和 AAC 音频编码的 MPEG 4 文件
- WebM = 带有 VP8 视频编码和 Vorbis 音频编码的 WebM 文件
- Ogg = 带有 Theora 视频编码和 Vorbis 音频编码的 Ogg 文件

视频格式

格式	MIME-type
MP4	video/mp4
WebM	video/webm
Ogg	video/ogg

HTML5 <video> - 使用 DOM 进行控制

HTML5 <video> 和 <audio> 元素同样拥有方法、属性和事件。
<video> 和 <audio>元素的方法、属性和事件可以使用 JavaScript 进行控制。
其中的方法用于播放、暂停以及加载等。其中的属性（比如时长、音量等）可以被读取或设置。其中的 DOM 事件能够通知您，比方说，<video> 元素开始播放、已暂停，已停止，等等。
例中简单的方法，向我们演示了如何使用 <video> 元素，读取并设置属性，以及如何调用方法。

实例 1

为视频创建简单的播放/暂停以及调整尺寸控件：

播放/暂停 放大 缩小 普通

上面的例子调用了两个方法：play() 和 pause()。它同时使用了两个属性：paused 和 width。
尝试一下 »

更多参考请查看 [HTML5 Audio/Video DOM 参考手册](#)。

HTML5 Video 标签

标签	描述
<video>	定义一个视频
<source>	定义多种媒体资源,比如 <video> 和<audio>
<track>	定义在媒体播放器文本轨迹

[HTML5 地理定位](#)

HTML5 Audio(音频)

点我分享笔记

HTML5 Audio(音频)

HTML5 提供了播放音频文件的标准。

互联网上的音频

直到现在，仍然不存在一项旨在网页上播放音频的标准。

今天，大多数音频是通过插件（比如 Flash）来播放的。然而，并非所有浏览器都拥有同样的插件。

HTML5 规定了在网页上嵌入音频元素的标准，即使用 `<audio>` 元素。

Internet Explorer 9+, Firefox, Opera, Chrome, 和 Safari 都支持 `<audio>` 元素。

注意: Internet Explorer 8 及更早 IE 版本不支持 `<audio>` 元素。

HTML5 Audio - 如何工作

如需在 HTML5 中播放音频，你需要使用以下代码：

实例

```
<audio controls> <source src="horse.ogg" type="audio/ogg"> <source src="horse.mp3" type="audio/mpeg"> 您的浏览器不支持 audio 元素。</audio>
```

尝试一下 »

control 属性供添加播放、暂停和音量控件。

在<audio> 与 </audio> 之间你需要插入浏览器不支持的<audio>元素的提示文本 。

<audio> 元素允许使用多个 <source> 元素. <source> 元素可以链接不同的音频文件， 浏览器将使用第一个支持的音频文件

音频格式及浏览器支持

目前, <audio>元素支持三种音频格式文件: MP3, Wav, 和 Ogg:

浏览器	MP3	Wav	Ogg
Internet Explorer 9+	YES	NO	NO
Chrome 6+	YES	YES	YES
Firefox 3.6+	YES	YES	YES
Safari 5+	YES	YES	NO
Opera 10+	YES	YES	YES

音频格式的 MIME 类型

Format	MIME-type
MP3	audio/mpeg
Ogg	audio/ogg
Wav	audio/wav

HTML5 Audio 标签

标签	描述
<audio>	定义了声音内容
<source>	规定了多媒体资源, 可以是多个, 在 <video> 与 <audio>标签中使用

[HTML5 Video\(视频\)](#)

[HTML5 Input 类型](#)

HTML5 新的 Input 类型

HTML5 拥有多个新的表单输入类型。这些新特性提供了更好的输入控制和验证。

本章全面介绍这些新的输入类型：

- color
- date
- datetime
- datetime-local
- email
- month

- number
- range
- search
- tel
- time
- url
- week

注意:并不是所有的主流浏览器都支持新的 input 类型，不过您已经可以在所有主流的浏览器中使用它们了。即使不被支持，仍然可以显示为常规的文本域。

Input 类型: color

color 类型用在 input 字段主要用于选取颜色，如下所示：

实例

从拾色器中选择一个颜色：

选择你喜欢的颜色：<input type="color" name="favcolor">

Input 类型: date

date 类型允许你从一个日期选择器选择一个日期。

实例

定义一个时间控制器:

生日: `<input type="date" name="bday">`

尝试一下 »

Input 类型: datetime

`datetime` 类型允许你选择一个日期 (UTC 时间)。

实例

定义一个日期和时间控制器 (本地时间):

生日 (日期和时间): `<input type="datetime" name="bdaytime">`

尝试一下 »

Input 类型: datetime-local

`datetime-local` 类型允许你选择一个日期和时间 (无时区)。

实例

定义一个日期和时间 (无时区):

生日 (日期和时间): `<input type="datetime-local" name="bdaytime">`

尝试一下 »

Input 类型: email

email 类型用于应该包含 e-mail 地址的输入域。

实例

在提交表单时，会自动验证 email 域的值是否合法有效:

E-mail: <input type="email" name="email">

尝试一下 »

提示: iPhone 中的 Safari 浏览器支持 email 输入类型，并通过改变触摸屏键盘来配合它（添加 @ 和 .com 选项）。

Input 类型: month

month 类型允许你选择一个月份。

实例

定义月与年 (无时区):

生日 (月和年): <input type="month" name="bdaymonth">

尝试一下 »

Input 类型: number

number 类型用于应该包含数值的输入域。

您还能够设定对所接受的数字的限定：

实例

定义一个数值输入域(限定)：

数量 (1 到 5 之间)：<input type="number" name="quantity" min="1" max="5">

尝试一下 »

使用下面的属性来规定对数字类型的限定：

属性	描述
disabled	规定输入字段是禁用的
max	规定允许的最大值
maxlength	规定输入字段的最大字符长度
min	规定允许的最小值
pattern	规定用于验证输入字段的模式
readonly	规定输入字段的值无法修改
required	规定输入字段的值是必需的
size	规定输入字段中的可见字符数

step	规定输入字段的合法数字间隔
value	规定输入字段的默认值

尝试一下带有所有限定属性的例子 [尝试一下](#)

Input 类型: range

range 类型用于应该包含一定范围内数字值的输入域。

range 类型显示为滑动条。



实例

定义一个不需要非常精确的数值（类似于滑块控制）：

```
<input type="range" name="points" min="1" max="10">
```

尝试一下 »

请使用下面的属性来规定对数字类型的限定：

- max - 规定允许的最大值
- min - 规定允许的最小值
- step - 规定合法的数字间隔
- value - 规定默认值

Input 类型: search

search 类型用于搜索域，比如站点搜索或 Google 搜索。



实例

定义一个搜索字段 (类似站点搜索或者 Google 搜索):

Search Google: `<input type="search" name="googlesearch">`

尝试一下 »

Input 类型: tel



实例

定义输入电话号码字段:

电话号码: `<input type="tel" name="usrtel">`

尝试一下 »

Input 类型: time

time 类型允许你选择一个时间。



实例

定义可输入时间控制器（无时区）：

选择时间：<input type="time" name="usr_time">

尝试一下 »

Input 类型: url

url 类型用于应该包含 URL 地址的输入域。

在提交表单时，会自动验证 url 域的值。



实例

定义输入 URL 字段:

添加您的主页：<input type="url" name="homepage">

尝试一下 »

提示: iPhone 中的 Safari 浏览器支持 url 输入类型，并通过改变触摸屏键盘来配合它（添加 .com 选项）。

Input 类型: week

week 类型允许你选择周和年。



实例

定义周和年 (无时区):

选择周：<input type="week" name="week_year">

尝试一下 »

HTML5 <input> 标签

标签	描述
<input>	描述 input 输入器

[HTML5 Audio\(音频\)](#)

[HTML5 表单元素](#)

点我分享笔记

HTML5 表单元素

HTML5 新的表单元素

HTML5 有以下新的表单元素:

- <datalist>
- <keygen>
- <output>

注意:不是所有的浏览器都支持 HTML5 新的表单元素，但是你可以在使用它们，即使浏览器不支持表单属性，仍然可以显示为常规的表单元素。

HTML5 <datalist> 元素

<datalist> 元素规定输入域的选项列表。

<datalist> 属性规定 form 或 input 域应该拥有自动完成功能。当用户在自动完成域中开始输入时，浏览器应该在该域中显示填写的选项：

使用 <input> 元素的列表属性与 <datalist> 元素绑定。



实例

<input> 元素使用<datalist>预定义值：

```
<input list="browsers"> <datalist id="browsers"> <option value="Internet Explorer"> <option value="Firefox">
<option value="Chrome"> <option value="Opera"> <option value="Safari"> </datalist>
```

尝试一下 »

HTML5 <keygen> 元素

<keygen> 元素的作用是提供一种验证用户的可靠方法。

<keygen>标签规定用于表单的密钥对生成器字段。

当提交表单时，会生成两个键，一个是私钥，一个公钥。

私钥（private key）存储于客户端，公钥（public key）则被发送到服务器。公钥可用于之后验证用户的客户端证书（client certificate）。



实例

带有 keygen 字段的表单：

```
<form action="demo_keygen.asp" method="get"> 用户名: <input type="text" name="usr_name"> 加密: <keygen  
name="security"> <input type="submit"> </form>
```

尝试一下 »

HTML5 <output> 元素

<output> 元素用于不同类型的输出，比如计算或脚本输出：

实例

将计算结果显示在 <output> 元素：

```
<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">0 <input type="range" id="a" value="50">100 +  
<input type="number" id="b" value="50">= <output name="x" for="a b"></output> </form>
```

尝试一下 »

HTML5 新表单元素

标签	描述
<datalist>	<input>标签定义选项列表。请与 input 元素配合使用该元素，来定义 input 可能的值。
<keygen>	<keygen> 标签规定用于表单的密钥对生成器字段。
<output>	<output> 标签定义不同类型的输出，比如脚本的输出。

HTML5 Input 类型

HTML5 表单属性

点我分享笔记

HTML5 表单属性

HTML5 新的表单属性

HTML5 的 <form> 和 <input> 标签添加了几个新属性.

<form>新属性:

- autocomplete
- novalidate

<input>新属性:

- autocomplete
- autofocus
- form
- formaction
- formenctype
- formmethod

- formnovalidate

- formtarget

- height 与 width

- list

- min 与 max

- multiple

- pattern (regexp)

- placeholder

- required

- step

<form> / <input> autocomplete 属性

autocomplete 属性规定 form 或 input 域应该拥有自动完成功能。

当用户在自动完成域中开始输入时，浏览器应该在该域中显示填写的选项。

提示: autocomplete 属性有可能在 form 元素中是开启的，而在 input 元素中是关闭的。

注意: autocomplete 适用于 <form> 标签，以及以下类型的 <input> 标签：text, search, url, telephone, email, password, datepickers, range 以及 color。

实例

HTML form 中开启 autocomplete (一个 input 字段关闭 autocomplete):

```
<form action="demo-form.php" autocomplete="on"> First name:<input type="text" name="fname"><br> Last name:
<input type="text" name="lname"><br> E-mail: <input type="email" name="email" autocomplete="off"><br> <input
type="submit"> </form>
```

[尝试一下 »](#)

提示:某些浏览器中,您可能需要启用自动完成功能,以使该属性生效。

<form> novalidate 属性

novalidate 是一个布尔 (true 或 false) 属性。

novalidate 属性是 HTML 表单元素的一个布尔属性,用于设置浏览器不对表单进行验证。

当该属性被添加到 <form> 元素上时,浏览器将不会执行默认的表单验证,不会检查输入字段是否符合指定的验证规则。

使用 novalidate 属性可以让开发者完全控制表单验证的逻辑,可以通过 JavaScript 或其他方式来自定义表单验证的行为。

实例

无需验证提交的表单数据

```
<form action="demo-form.php" novalidate> E-mail: <input type="email" name="user_email"> <input type="submit">
</form>
```

[尝试一下 »](#)

<input> autofocus 属性

autofocus 属性是一个布尔属性。

autofocus 属性规定在页面加载时,域自动地获得焦点。

实例

让 "First name" input 输入域在页面载入时自动聚焦:

```
First name:<input type="text" name="fname" autofocus>
```

[尝试一下 »](#)

<input> form 属性

form 属性规定输入域所属的一个或多个表单。

提示:如需引用一个以上的表单, 请使用空格分隔的列表。

实例

位于 form 表单外的 input 字段引用了 HTML form (该 input 表单仍然属于 form 表单的一部分):

```
<form action="demo-form.php" id="form1"> First name: <input type="text" name="fname"><br> <input type="submit" value="提交"> </form> Last name: <input type="text" name="lname" form="form1">
```

[尝试一下 »](#)

<input> formaction 属性

The formaction 属性用于描述表单提交的 URL 地址。

The formaction 属性会覆盖<form> 元素中的 action 属性。

注意: The formaction 属性用于 type="submit" 和 type="image".

实例

以下 HTMLform 表单包含了两个不同地址的提交按钮:

```
<form action="demo-form.php"> First name: <input type="text" name="fname"><br> Last name: <input type="text" name="lname"><br> <input type="submit" value="提交"><br> <input type="submit" formaction="demo-admin.php" value="提交"> </form>
```

尝试一下 »

<input> formenctype 属性

formenctype 属性描述了表单提交到服务器的数据编码 (只对 form 表单中 method="post" 表单)

formenctype 属性覆盖 form 元素的 enctype 属性。

主要: 该属性与 type="submit" 和 type="image" 配合使用。

实例

第一个提交按钮以默认编码发送表单数据, 第二个提交按钮以 "multipart/form-data" 编码格式发送表单数据:

```
<form action="demo-post_enctype.php" method="post"> First name: <input type="text" name="fname"><br> <input type="submit" value="提交"> <input type="submit" formenctype="multipart/form-data" value="以 Multipart/form-data 提交"> </form>
```

尝试一下 »

<input> formmethod 属性

formmethod 属性定义了表单提交的方式。

formmethod 属性覆盖了 <form> 元素的 method 属性。

注意: 该属性可以与 type="submit" 和 type="image" 配合使用。

实例

重新定义表单提交方式实例:

```
<form action="demo-form.php" method="get"> First name: <input type="text" name="fname"><br> Last name: <input type="text" name="lname"><br> <input type="submit" value="提交"> <input type="submit" formmethod="post" formaction="demo-post.php" value="使用 POST 提交"> </form>
```

[尝试一下 »](#)

<input> formnovalidate 属性

novalidate 属性是一个 boolean 属性。

novalidate 属性描述了 <input> 元素在表单提交时无需被验证。

formnovalidate 属性会覆盖 <form> 元素的 novalidate 属性。

注意: formnovalidate 属性与 **type="submit"** 一起使用

实例

两个提交按钮的表单(使用与不适用验证):

```
<form action="demo-form.php"> E-mail: <input type="email" name="userid"><br> <input type="submit" value="提交"><br> <input type="submit" formnovalidate value="不验证提交"> </form>
```

[尝试一下 »](#)

<input> formtarget 属性

formtarget 属性指定一个名称或一个关键字来指明表单提交数据接收后的展示。

formtarget 属性覆盖 <form>元素的 target 属性。

注意: formtarget 属性与 **type="submit"** 和 **type="image"** 配合使用。

实例

两个提交按钮的表单, 在不同窗口中显示:

```
<form action="demo-form.php"> First name: <input type="text" name="fname"><br> Last name: <input type="text" name="lname"><br> <input type="submit" value="正常提交"> <input type="submit" formtarget="_blank" value="提交到一个新的页面上"> </form>
```

[尝试一下 »](#)

<input> height 和 width 属性

height 和 width 属性规定用于 image 类型的 <input> 标签的图像高度和宽度。

注意: height 和 width 属性只适用于 image 类型的<input> 标签。

提示:图像通常会同时指定高度和宽度属性。如果图像设置高度和宽度, 图像所需的空间 在加载页时会被保留。如果没有这些属性, 浏览器不知道图像的大小, 并不能预留 适当的空间。图片在加载过程中会使页面布局效果改变 (尽管图片已加载)。

实例

定义了一个图像提交按钮, 使用了 height 和 width 属性:

```
<input type="image" src="img_submit.gif" alt="Submit" width="48" height="48">
```

尝试一下 »

<input> list 属性

list 属性规定输入域的 datalist。datalist 是输入域的选项列表。

实例

在 <datalist> 中预定义 <input> 值:

```
<input list="browsers">

<datalist id="browsers">
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
```

尝试一下 »

<input> min 和 max 属性

min、max 和 step 属性用于为包含数字或日期的 input 类型规定限定（约束）。

注意: min、max 和 step 属性适用于以下类型的 <input> 标签: date pickers、number 以及 range。



实例

<input> 元素最小值与最大值设置:

Enter a date before 1980-01-01:

```
<input type="date" name="bday" max="1979-12-31">
```

Enter a date after 2000-01-01:

```
<input type="date" name="bday" min="2000-01-02">
```

Quantity (between 1 and 5):

```
<input type="number" name="quantity" min="1" max="5">
```

尝试一下 »

<input> multiple 属性

multiple 属性是一个 boolean 属性。

multiple 属性规定<input> 元素中可选择多个值。

注意: multiple 属性适用于以下类型的 <input> 标签: email 和 file:



实例

上传多个文件:

```
Select images: <input type="file" name="img" multiple>
```


尝试一下 »

<input> pattern 属性

pattern 属性描述了一个正则表达式用于验证 <input> 元素的值。

注意:pattern 属性适用于以下类型的 <input> 标签: text, search, url, tel, email, 和 password。

提示: 是用来全局 [title](#) 属性来描述模式。

提示: 您可以在我们的 [JavaScript 教程](#) 中学习到有关正则表达式的内容。



实例

下面的例子显示了一个只能包含三个字母的文本域（不含数字及特殊字符）：

Country code: `<input type="text" name="country_code" pattern="[A-Za-z]{3}" title="Three letter country code">`

尝试一下 »

<input> placeholder 属性

placeholder 属性提供一种提示（hint），描述输入域所期待的值。

简短的提示在用户输入值前会显示在输入域上。

注意: placeholder 属性适用于以下类型的 <input> 标签: text, search, url, telephone, email 以及 password。



实例

input 字段提示文本 t:

```
<input type="text" name="fname" placeholder="First name">
```

尝试一下 »

<input> required 属性

required 属性是一个 boolean 属性.

required 属性规定必须在提交之前填写输入域（不能为空）。

注意:required 属性适用于以下类型的 <input> 标签: text, search, url, telephone, email, password, date pickers, number, checkbox, radio 以及 file。



实例

不能为空的 input 字段:

```
Username: <input type="text" name="username" required>
```

尝试一下 »

<input> step 属性

step 属性为输入域规定合法的数字间隔。

如果 step="3", 则合法的数是 -3,0,3,6 等

提示: step 属性可以与 max 和 min 属性创建一个区域值.

注意: step 属性与以下 type 类型一起使用: number, range, date, datetime, datetime-local, month, time 和 week.



实例

规定 input step 步长为 3:

```
<input type="number" name="points" step="3">
```

[尝试一下 »](#)

HTML5 <input> 标签

标签	描述
<form>	定义一个 form 表单
<input>	定义一个 input 域

[HTML5 表单元素](#)

[HTML5 语义元素](#)

点我分享笔记

HTML5 语义元素

语义= 意义

语义元素 = 有意义的元素

什么是语义元素？

一个语义元素能够清楚的描述其意义给浏览器和开发者。

无语义 元素实例: `<div>` 和 `` - 无需考虑内容.

语义元素实例: `<form>`, `<table>`, and `` - 清楚的定义了它的内容.

浏览器支持



Internet Explorer 9+, Firefox, Chrome, Safari 和 Opera 支持语义元素。

注意: Internet Explorer 8 及更早版本不支持该元素。但是文章底部提供了兼容的解决方法.

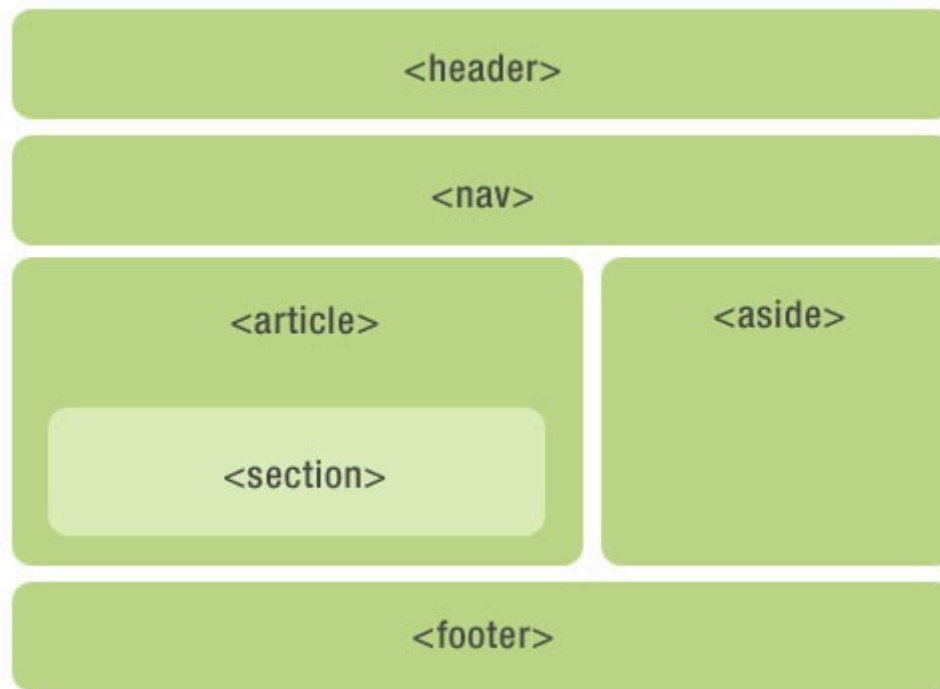
HTML5 中新的语义元素

许多现有网站都包含以下 HTML 代码: `<div id="nav">`, `<div class="header">`, 或者 `<div id="footer">`, 来指明导航链接, 头部, 以及尾部.

HTML5 提供了新的语义元素来明确一个 Web 页面的不同部分:

- `<header>`
- `<nav>`
- `<section>`
- `<article>`
- `<aside>`

- <figcaption>
- <figure>
- <footer>



HTML5 <section> 元素

<section> 标签定义文档中的节（section、区段）。比如章节、页眉、页脚或文档中的其他部分。

根据 W3C HTML5 文档: section 包含了一组内容及其标题。

实例

```
<section> <h1>WWF</h1> <p>The World Wide Fund for Nature (WWF) is...</p> </section>
```

尝试一下 »

HTML5 <article> 元素

<article> 标签定义独立的内容。.

<article> 元素使用实例:

- Forum post
- Blog post
- News story
- Comment

实例

```
<article> <h1>Internet Explorer 9</h1> <p>Windows Internet Explorer 9(缩写为 IE9 )在 2011 年 3 月 14 日 21:00 发布。</p> </article>
```

尝试一下 »

HTML5 <nav> 元素

<nav> 标签定义导航链接的部分。

<nav> 元素用于定义页面的导航链接部分区域，但是，不是所有的链接都需要包含在 <nav> 元素中!

实例

```
<nav> <a href="/html/">HTML</a> | <a href="/css/">CSS</a> | <a href="/js/">JavaScript</a> | <a href="/jquery/">jQuery</a> </nav>
```

尝试一下 »

HTML5 <aside> 元素

<aside> 标签定义页面主区域内容之外的内容（比如侧边栏）。

aside 标签的内容应与主区域的内容相关。

实例

```
<p>My family and I visited The Epcot center this summer.</p> <aside> <h4>Epcot Center</h4> <p>The Epcot Center is a theme park in Disney World, Florida.</p> </aside>
```

尝试一下 »

HTML5 <header> 元素

<header>元素描述了文档的头部区域

<header>元素主要用于定义内容的介绍展示区域。

在页面中你可以使用多个<header> 元素。

以下实例定义了文章的头部：

实例

```
<article> <header> <h1>Internet Explorer 9</h1> <p><time pubdate datetime="2011-03-15"></time></p> </header>
<p>Windows Internet Explorer 9(缩写为 IE9 )是在2011年3月14日21:00发布的</p> </article>
```

尝试一下 »

HTML5 <footer> 元素

<footer> 元素描述了文档的底部区域。

<footer> 元素应该包含它的包含元素

一个页脚通常包含文档的作者，著作权信息，链接的使用条款，联系信息等

文档中你可以使用多个 <footer>元素。

实例

```
<footer> <p>Posted by: Hege Refsnes</p> <p><time pubdate datetime="2012-03-01"></time></p> </footer>
```

尝试一下 »

HTML5 <figure> 和 <figcaption> 元素

<figure>标签规定独立的流内容（图像、图表、照片、代码等等）。

<figure> 元素的内容应该与主内容相关，但如果被删除，则不应影响文档流。

<figcaption> 标签定义 <figure> 元素的标题。

<figcaption> 元素应该被置于 "figure" 元素的第一个或最后一个子元素的位置。

实例

```
<figure>  <figcaption>Fig1. - The Pulpit Pock, Norway.</figcaption> </figure>
```

尝试一下 »

我们可以开始使用这些语义元素吗？

以上的元素都是块元素(除了<figcaption>)。

为了让这些块级元素在所有版本的浏览器中生效，你需要在样式表文件中设置一下属性 (以下样式代码可以让旧版本浏览器支持本章介绍的块级元素)：

```
header, section, footer, aside, nav, article, figure
{
    display: block;
}
```

Internet Explorer 8 及更早 IE 版本中的问题

IE8 及更早 IE 版本无法在这些元素中渲染 CSS 效果，以至于你不能使用 <header>, <section>, <footer>, <aside>, <nav>, <article>, <figure>, 或者其他的 HTML5 elements。

解决办法: 你可以使用 HTML5 Shiv Javascript 脚本来解决 IE 的兼容问题。HTML5 Shiv 下载地址：

<https://lf26-cdn-tos.bytecdntp.com/cdn/expire-1-M/html5shiv/3.7.3/html5shiv.min.js>

下载后，将以下代码放入到网页中：

```
<!--[if lt IE 9]>
<script src="html5shiv.js"></script>
<![endif]-->
```

以上代码在浏览器小于 IE9 版本时会加载 html5shiv.js 文件. 你必须将其放置于<head> 元素中, 因为 IE 浏览器需要在头部加载后渲染这些 HTML5 的新元素

[HTML5 表单属性](#)

[HTML5 Web 存储](#)

[点我分享笔记](#)

HTML5 Web 存储

HTML5 web 存储, 一个比 cookie 更好的本地存储方式。

什么是 HTML5 Web 存储?

使用 HTML5 可以在本地存储用户的浏览数据。

早些时候,本地存储使用的是 cookie。但是 Web 存储需要更加的安全与快速. 这些数据不会被保存在服务器上, 但是这些数据只用于用户请求网站数据上.它也可以存储大量的数据, 而不影响网站的性能.

数据以 键/值 对存在, web 网页的数据只允许该网页访问使用。

浏览器支持



Internet Explorer 8+, Firefox, Opera, Chrome, 和 Safari 支持 Web 存储。

注意: Internet Explorer 7 及更早 IE 版本不支持 web 存储。

localStorage 和 sessionStorage

客户端存储数据的两个对象为:

- localStorage - 用于长久保存整个网站的数据, 保存的数据没有过期时间, 直到手动去除。
- sessionStorage - 用于临时保存同一窗口(或标签页)的数据, 在关闭窗口或标签页之后将会删除这些数据。

在使用 web 存储前,应检查浏览器是否支持 localStorage 和 sessionStorage:

```
if(typeof(Storage)!="undefined") { // 是的! 支持 localStorage sessionStorage 对象! // 一些代码..... } else { // 抱歉! 不支持 web 存储。 }
```

localStorage 对象

localStorage 对象存储的数据没有时间限制。第二天、第二周或下一年之后, 数据依然可用。

实例

```
// 存储 localStorage.setItem("sitename", "菜鸟教程"); // 查找 document.getElementById("result").innerHTML = "网站名: " + localStorage.getItem("sitename");
```

[尝试一下 »](#)

实例解析:

- 使用 **key="sitename"** 和 **value="菜鸟教程"** 创建一个 localStorage 键/值对。
- 检索键值为 "sitename" 的值然后将数据插入 **id="result"** 的元素中。

以上实例也可以这么写：

```
// 存储 localStorage.sitename = "菜鸟教程"; // 查找 document.getElementById("result").innerHTML =  
localStorage.sitename;
```

移除 localStorage 中的 "sitename"：

```
localStorage.removeItem("sitename");
```

不管是 localStorage，还是 sessionStorage，可使用的 API 都相同，常用的有如下几个（以 localStorage 为例）：

- 保存数据：localStorage.setItem(key,value);
- 读取数据：localStorage.getItem(key);
- 删除单个数据：localStorage.removeItem(key);
- 删除所有数据：localStorage.clear();
- 得到某个索引的 key：localStorage.key(index);

提示：键/值对通常以字符串存储，你可以按自己的需要转换该格式。

下面的实例展示了用户点击按钮的次数。

代码中的字符串值转换为数字类型：

实例

```
if (localStorage.clickcount) { localStorage.clickcount=Number(localStorage.clickcount)+1; } else {  
localStorage.clickcount=1; } document.getElementById("result").innerHTML=" 你已经点击了按钮 " +  
localStorage.clickcount + " 次 ";
```

[尝试一下 »](#)

sessionStorage 对象

sessionStorage 方法针对一个 session 进行数据存储。当用户关闭浏览器窗口后，数据会被删除。

如何创建并访问一个 sessionStorage：

实例

```
if (sessionStorage.clickcount) { sessionStorage.clickcount=Number(sessionStorage.clickcount)+1; } else {
sessionStorage.clickcount=1; } document.getElementById("result").innerHTML="在这个会话中你已经点击了该按钮 " +
sessionStorage.clickcount + " 次 ";
```

[尝试一下 »](#)

Web Storage 开发一个简单的网站列表程序

网站列表程序实现以下功能：

- 可以输入网站名，网址，以网站名作为 key 存入 localStorage；
- 根据网站名，查找网址；
- 列出当前已保存的所有网站；

以下代码用于保存与查找数据：

save() 与 find() 方法

```
//保存数据 function save(){ var siteurl = document.getElementById("siteurl").value; var sitename =
document.getElementById("sitename").value; localStorage.setItem(sitename, siteurl); alert("添加成功"); } //查找
数据 function find(){ var search_site = document.getElementById("search_site").value; var sitename =
```

```
localStorage.getItem(search_site); var find_result = document.getElementById("find_result");  
find_result.innerHTML = search_site + "的网址是: " + sitename; }
```

完整实例演示如下:

实例

```
<div style="border: 2px dashed #ccc;width:320px;text-align:center;"> <label for="sitename">网站名(key):  
</label> <input type="text" id="sitename" name="sitename" class="text"/> <br/> <label for="siteurl">网 址  
(value): </label> <input type="text" id="siteurl" name="siteurl"/> <br/> <input type="button" onclick="save()" value="新增记录"/> <hr/> <label for="search_site">输入网站名: </label> <input type="text" id="search_site" name="search_site"/> <input type="button" onclick="find()" value="查找网站"/> <p id="find_result"><br/></p>  
</div>
```

尝试一下 »

实现效果截图:

网站名(key): 菜鸟教程

网 址(value): www.runoob.com

新增记录

输入网站名: 菜鸟教程 查找网站

菜鸟教程的网址是: www.runoob.com

以上实例只是演示了简单的 localStorage 存储与查找, 更多情况下我们存储的数据会更复杂。

接下来我们将使用 `JSON.stringify` 来存储对象数据, `JSON.stringify` 可以将对象转换为字符串。

```
var site = new Object; ... var str = JSON.stringify(site); // 将对象转换为字符串
```

之后我们使用 `JSON.parse` 方法将字符串转换为 JSON 对象:

```
var site = JSON.parse(str);
```

JavaScript 实现代码:

save() 与 find() 方法

```
//保存数据 function save(){ var site = new Object; site.keyname = document.getElementById("keyname").value;
site.sitename = document.getElementById("sitename").value; site.siteurl =
document.getElementById("siteurl").value; var str = JSON.stringify(site); // 将对象转换为字符串
localStorage.setItem(site.keyname,str); alert("保存成功"); } //查找数据 function find(){ var search_site =
document.getElementById("search_site").value; var str = localStorage.getItem(search_site); var find_result =
document.getElementById("find_result"); var site = JSON.parse(str); find_result.innerHTML = search_site + "的
网站名是: " + site.sitename + ", 网址是: " + site.siteurl; }
```

完整实例如下:

实例

```
<div style="border: 2px dashed #ccc;width:320px;text-align:center;"> <label for="keyname">别名(key):</label>
<input type="text" id="keyname" name="keyname" class="text"/> <br/> <label for="sitename">网站名: </label>
<input type="text" id="sitename" name="sitename" class="text"/> <br/> <label for="siteurl">网 址: </label>
<input type="text" id="siteurl" name="siteurl"/> <br/> <input type="button" onclick="save()" value="新增记录"/>
<hr/> <label for="search_site">输入别名(key): </label> <input type="text" id="search_site" name="search_site"/>
<input type="button" onclick="find()" value="查找网站"/> <p id="find_result"><br/></p> </div>
```

尝试一下 »

实例中的 `loadAll` 输出了所有存储的数据, 你需要确保 `localStorage` 存储的数据都为 JSON 格式, 否则 `JSON.parse(str)` 将会报错。

输出结果演示:

别名(key):

网站名:

网 址:

输入别名(key):

runoob的网站名是: 菜鸟教程, 网址是:
www.runoob.com

[HTML5 语义元素](#)

[HTML5 Web SQL](#)

2 篇笔记 写笔记

1. 笑醉踏歌行
750***319@qq.com


```
//删除数据
function del(name) {
  localStorage.removeItem(name);
  alert("删除成功!");
  loadAll();
}
```

[尝试一下 »](#)

[笑醉踏歌行](#)

笑醉踏歌行

750***319@qq.com

6 年前 (2018-06-23)

2. Eg

240***9110@qq.com

[参考地址](#)

217

作用域



- o 这里的作用域指的是：如何隔离开不同页面之间的 `localStorage`（总不能在百度的页面上能读到腾讯的 `localStorage` 吧，哈哈）。
- o `localStorage` 只要在相同的协议、相同的主机名、相同的端口下，就能读取/修改到同一份 `localStorage` 数据。
- o `sessionStorage` 比 `localStorage` 更严苛一点，除了协议、主机名、端口外，还要求在同一窗口（也就是浏览器的标签页）下。

HTML5 Web SQL 数据库

Web SQL 数据库 API 并不是 HTML5 规范的一部分，但是它是一个独立的规范，引入了一组使用 SQL 操作客户端数据库的 APIs。

如果你是一个 Web 后端程序员，应该很容易理解 SQL 的操作。

你也可以参考我们的 [SQL 教程](#)，了解更多数据库操作知识。

Web SQL 数据库可以在最新版的 Safari, Chrome 和 Opera 浏览器中工作。

核心方法

以下是规范中定义的核心方法：

1. **openDatabase**：这个方法使用现有的数据库或者新建的数据库创建一个数据库对象。
 2. **transaction**：这个方法让我们能够控制一个事务，以及基于这种情况执行提交或者回滚。
 3. **executeSql**：这个方法用于执行实际的 SQL 查询。
-

打开数据库

我们可以使用 openDatabase() 方法来打开已存在的数据库，如果数据库不存在，则会创建一个新的数据库，使用代码如下：

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);
```

openDatabase() 方法对应的五个参数说明：

1. 数据库名称
2. 版本号
3. 描述文本
4. 数据库大小
5. 创建回调

第五个参数，创建回调会在创建数据库后被调用。

执行查询操作

执行操作使用 database.transaction() 函数:

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); db.transaction(function (tx) {  
tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); });
```

上面的语句执行后会在 'mydb' 数据库中创建一个名为 LOGS 的表。

插入数据

在执行上面的创建表语句后, 我们可以插入一些数据:

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); db.transaction(function (tx) {  
tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); tx.executeSql('INSERT INTO LOGS (id, log)  
VALUES (1, "菜鸟教程")'); tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "www.runoob.com")'); });
```

我们也可以使用动态值来插入数据:

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); db.transaction(function (tx) {  
tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); tx.executeSql('INSERT INTO LOGS (id,log)  
VALUES (?, ?)', [e_id, e_log]); });
```

实例中的 e_id 和 e_log 是外部变量, executeSql 会映射数组参数中的每个条目给 "?"。

读取数据

以下实例演示了如何读取数据库中已经存在的数据:

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); db.transaction(function (tx) {  
tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); tx.executeSql('INSERT INTO LOGS (id, log)  
VALUES (1, "菜鸟教程")'); tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "www.runoob.com")'); });  
db.transaction(function (tx) { tx.executeSql('SELECT * FROM LOGS', [], function (tx, results) { var len =
```

```
results.rows.length, i; msg = "<p>查询记录条数: " + len + "</p>"; document.querySelector('#status').innerHTML += msg; for (i = 0; i < len; i++){ alert(results.rows.item(i).log ); } }, null); });
```

完整实例

实例

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); var msg; db.transaction(function (tx) { tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); tx.executeSql('INSERT INTO LOGS (id, log) VALUES (1, "菜鸟教程")'); tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "www.runoob.com")'); msg = '<p>数据表已创建，且插入了两条数据。</p>'; document.querySelector('#status').innerHTML = msg; }); db.transaction(function (tx) { tx.executeSql('SELECT * FROM LOGS', [], function (tx, results) { var len = results.rows.length, i; msg = "<p>查询记录条数: " + len + "</p>"; document.querySelector('#status').innerHTML += msg; for (i = 0; i < len; i++){ msg = "<p><b>" + results.rows.item(i).log + "</b></p>"; document.querySelector('#status').innerHTML += msg; } }, null); });
```

[尝试一下 »](#)

以上实例运行结果如下图所示：

测试结果:

[runoob.co](http://runoob.com)

数据表已创建，且插入了两条数据。

查询记录条数: 2

菜鸟教程

www.runoob.com

删除记录

删除记录使用的格式如下：

```
db.transaction(function (tx) {  
    tx.executeSql('DELETE FROM LOGS WHERE id=1');  
});
```

删除指定的数据 id 也可以是动态的：

```
db.transaction(function(tx) {  
    tx.executeSql('DELETE FROM LOGS WHERE id=?', [id]);  
});
```

更新记录

更新记录使用的格式如下：

```
db.transaction(function (tx) {  
    tx.executeSql('UPDATE LOGS SET log=\'www.w3cschool.cc\' WHERE id=2');  
});
```

更新指定的数据 id 也可以是动态的：

```
db.transaction(function(tx) {  
    tx.executeSql('UPDATE LOGS SET log=\'www.w3cschool.cc\' WHERE id=?', [id]);  
});
```

完整实例

实例

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024); var msg; db.transaction(function (tx) {
tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)'); tx.executeSql('INSERT INTO LOGS (id, log)
VALUES (1, "菜鸟教程")'); tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "www.runoob.com")'); msg = '<p>
数据表已创建，且插入了两条数据。</p>'; document.querySelector('#status').innerHTML = msg; });
db.transaction(function (tx) { tx.executeSql('DELETE FROM LOGS WHERE id=1'); msg = '<p>删除 id 为 1 的记录。
</p>'; document.querySelector('#status').innerHTML = msg; }); db.transaction(function (tx) {
tx.executeSql('UPDATE LOGS SET log=\'www.w3cschool.cc\' WHERE id=2'); msg = '<p>更新 id 为 2 的记录。</p>';
document.querySelector('#status').innerHTML = msg; }); db.transaction(function (tx) { tx.executeSql('SELECT *
FROM LOGS', [], function (tx, results) { var len = results.rows.length, i; msg = "<p>查询记录条数： " + len +
"</p>"; document.querySelector('#status').innerHTML += msg; for (i = 0; i < len; i++){ msg = "<p><b>" +
results.rows.item(i).log + "</b></p>"; document.querySelector('#status').innerHTML += msg; } }, null); });
```

尝试一下 »

以上实例运行结果如下图所示：

运行结果：

更新 id 为 2 的记录。

查询记录条数: 1

runoob.com

[HTML5 Web 存储](#)

[HTML5 应用程序缓存](#)

点我分享笔记

HTML5 应用程序缓存

使用 HTML5，通过创建 cache manifest 文件，可以轻松地创建 web 应用的离线版本。

注意：manifest 的技术已被 web 标准废弃，不再推荐使用此功能。

什么是应用程序缓存（Application Cache）？

HTML5 引入了应用程序缓存，这意味着 web 应用可进行缓存，并可在没有因特网连接时进行访问。

应用程序缓存为应用带来三个优势：

1. 离线浏览 - 用户可在应用离线时使用它们
2. 速度 - 已缓存资源加载得更快
3. 减少服务器负载 - 浏览器将只从服务器下载更新过或更改过的资源。

浏览器支持



Internet Explorer 10, Firefox, Chrome, Safari 和 Opera 支持应用程序缓存。

HTML5 Cache Manifest 实例

下面的例子展示了带有 cache manifest 的 HTML 文档（供离线浏览）：

实例

```
<!DOCTYPE HTML>
<html manifest="demo.appcache">

<body>
文档内容.....
```

```
</body>
```

```
</html>
```

尝试一下 »

Cache Manifest 基础

如需启用应用程序缓存，请在文档的<html> 标签中包含 manifest 属性：

```
<!DOCTYPE HTML>
<html manifest="demo.appcache">
...
</html>
```

每个指定了 manifest 的页面在用户对其访问时都会被缓存。如果未指定 manifest 属性，则页面不会被缓存（除非在 manifest 文件中直接指定了该页面）。

manifest 文件的建议的文件扩展名是：".appcache"。

💡 请注意，manifest 文件需要配置正确的 MIME-type，即 "text/cache-manifest"。必须在 web 服务器上配置。

Manifest 文件

manifest 文件是简单的文本文件，它告知浏览器被缓存的内容（以及不缓存的内容）。

manifest 文件可分为三个部分：

- *CACHE MANIFEST* - 在此标题下列出的文件将在首次下载后进行缓存
- *NETWORK* - 在此标题下列出的文件需要与服务器的连接，且不会被缓存
- *FALLBACK* - 在此标题下列出的文件规定当页面无法访问时的回退页面（比如 404 页面）

CACHE MANIFEST

第一行, CACHE MANIFEST, 是必需的:

```
CACHE MANIFEST
/theme.css
/logo.gif
/main.js
```

上面的 manifest 文件列出了三个资源: 一个 CSS 文件, 一个 GIF 图像, 以及一个 JavaScript 文件。当 manifest 文件加载后, 浏览器会从网站的根目录下载这三个文件。然后, 无论用户何时与因特网断开连接, 这些资源依然是可用的。

NETWORK

下面的 NETWORK 小节规定文件 "login.php" 永远不会被缓存, 且离线时是不可用的:

```
NETWORK:
login.php
```

可以使用星号来指示所有其他资源/文件都需要因特网连接:

```
NETWORK:
*
```

FALLBACK

下面的 FALLBACK 小节规定如果无法建立因特网连接, 则用 "offline.html" 替代 /html5/ 目录中的所有文件:

```
FALLBACK:
/html/ /offline.html
```

注意: 第一个 URI 是资源, 第二个是替补。

更新缓存

一旦应用被缓存, 它就会保持缓存直到发生下列情况:

- 用户清空浏览器缓存
- manifest 文件被修改 (参阅下面的提示)


- 由程序来更新应用缓存

实例 - 完整的 Manifest 文件

```
CACHE MANIFEST
# 2012-02-21 v1.0.0
/theme.css
/logo.gif
/main.js
```

```
NETWORK:
login.php
```

```
FALLBACK:
/html/ /offline.html
```

 **提示:**以 "#" 开头的是注释行，但也可满足其他用途。应用的缓存会在其 manifest 文件更改时被更新。如果您编辑了一幅图片，或者修改了一个 JavaScript 函数，这些改变都不会被重新缓存。更新注释行中的日期和版本号是一种使浏览器重新缓存文件的办法。

关于应用程序缓存的说明

请留心缓存的内容。

一旦文件被缓存，则浏览器会继续展示已缓存的版本，即使您修改了服务器上的文件。为了确保浏览器更新缓存，您需要更新 manifest 文件。

注意: 浏览器对缓存数据的容量限制可能不太一样（某些浏览器设置的限制是每个站点 5MB）。

[HTML5 Web SQL](#)

[HTML5 Web Workers](#)

[点我分享笔记](#)

HTML5 Web Workers

web worker 是运行在后台的 JavaScript，不会影响页面的性能。

什么是 Web Worker?

当在 HTML 页面中执行脚本时，页面的状态是不可响应的，直到脚本已完成。

web worker 是运行在后台的 JavaScript，独立于其他脚本，不会影响页面的性能。您可以继续做任何愿意做的事情：点击、选取内容等等，而此时 web worker 在后台运行。

浏览器支持



Internet Explorer 10, Firefox, Chrome, Safari 和 Opera 都支持 Web workers.

HTML5 Web Workers 实例

下面的例子创建了一个简单的 web worker，在后台计数：

实例

计数：

开始 Worker 停止 Worker

尝试一下 »

demo_workers.js 文件代码

```
var i=0;

function timedCount()
{
    i=i+1;
    postMessage(i);
    setTimeout("timedCount()",500);
}

timedCount();
```

检测浏览器是否支持 Web Worker

在创建 web worker 之前，请检测用户的浏览器是否支持它：

```
if(typeof(Worker)!="undefined")
{
    // 是的！ Web worker 支持！
    // 一些代码.....
}
else
{
    //抱歉！ Web Worker 不支持
```

```
}
```

创建 web worker 文件

现在，让我们在一个外部 JavaScript 中创建我们的 web worker。
在这里，我们创建了计数脚本。该脚本存储于 "demo_workers.js" 文件中：

```
var i=0;

function timedCount()
{
    i=i+1;
    postMessage(i);
    setTimeout("timedCount()",500);
}

timedCount();
```

以上代码中重要的部分是 `postMessage()` 方法 - 它用于向 HTML 页面传回一段消息。

注意: web worker 通常不用于如此简单的脚本，而是用于更耗费 CPU 资源的任务。

创建 Web Worker 对象

我们已经有了 web worker 文件，现在我们需要从 HTML 页面调用它。

下面的代码检测是否存在 worker，如果不存在，- 它会创建一个新的 web worker 对象，然后运行 "demo_workers.js" 中的代码：

```
if(typeof(w)=="undefined")
{
    w=new Worker("demo_workers.js");
```



```
}
```

然后我们就可以从 web worker 发送和接收消息了。

向 web worker 添加一个 "onmessage" 事件监听器:

```
w.onmessage=function(event){  
    document.getElementById("result").innerHTML=event.data;  
};
```

```
<="" p="">
```

终止 Web Worker

当我们创建 web worker 对象后，它会继续监听消息（即使在外脚本完成之后）直到其被终止为止。

如需终止 web worker，并释放浏览器/计算机资源，请使用 `terminate()` 方法：

```
w.terminate();
```

完整的 Web Worker 实例代码

我们已经看到了 .js 文件中的 Worker 代码。下面是 HTML 页面的代码：

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> </head> <body> <p>  
计数: <output id="result"></output></p> <button onclick="startWorker()">开始工作</button> <button  
onclick="stopWorker()">停止工作</button> <p><strong>注意: </strong> Internet Explorer 9 及更早 IE 版本浏览器不支持  
Web Workers.</p> <script>  
var w; function startWorker() { if(typeof(Worker) !== "undefined") { if(typeof(w) == "undefined") { w = new  
Worker("demo_workers.js"); } w.onmessage = function(event) { document.getElementById("result").innerHTML =
```

```
event.data; }]; } else { document.getElementById("result").innerHTML = "抱歉, 你的浏览器不支持 Web Workers..."; } }  
function stopWorker() { w.terminate(); w = undefined; }  
</script> </body> </html>
```

尝试一下 »

Web Workers 和 DOM

由于 web worker 位于外部文件中, 它们无法访问下列 JavaScript 对象:

- window 对象
- document 对象
- parent 对象

[HTML5 应用程序缓存](#)

[HTML5 服务器发送事件\(Server-Sent Events\)](#)

[点我分享笔记](#)

HTML5 服务器发送事件(Server-Sent Events)

HTML5 服务器发送事件 (server-sent event) 允许网页获得来自服务器的更新。

Server-Sent 事件 - 单向消息传递

Server-Sent 事件指的是网页自动获取来自服务器的更新。

以前也可能做到这一点，前提是网页不得不询问是否有可用的更新。通过服务器发送事件，更新能够自动到达。

例子：Facebook/Twitter 更新、股价更新、新的博文、赛事结果等。

浏览器支持



所有主流浏览器均支持服务器发送事件，除了 Internet Explorer。

接收 Server-Sent 事件通知

EventSource 对象用于接收服务器发送事件通知：

实例

```
var source=new EventSource("demo_sse.php"); source.onmessage=function(event) {  
document.getElementById("result").innerHTML+=event.data + "<br>"; };
```

尝试一下 »

实例解析：

- 创建一个新的 EventSource 对象，然后规定发送更新的页面的 URL（本例中是 "demo_sse.php"）
- 每接收到一次更新，就会发生 onmessage 事件
- 当 onmessage 事件发生时，把已接收的数据推入 id 为 "result" 的元素中

检测 Server-Sent 事件支持

以下实例，我们编写了一段额外的代码来检测服务器发送事件的浏览器支持情况：

```
if(typeof(EventSource)!="undefined")
{
    // 浏览器支持 Server-Sent
    // 一些代码.....
}
else
{
    // 浏览器不支持 Server-Sent..
}
```

服务器端代码实例

为了让上面的例子可以运行，您还需要能够发送数据更新的服务器（比如 PHP 和 ASP）。服务器端事件流的语法是非常简单的。把 "Content-Type" 报头设置为 "text/event-stream"。现在，您可以开始发送事件流了。

实例

```
<?php
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');

$time = date('r');
echo "data: The server time is: {$time}\n\n";
```

```
flush();  
?>
```

ASP 代码 (VB) (demo_sse.asp):

```
<%  
Response.ContentType="text/event-stream"  
Response.Expires=-1  
Response.Write("data: " & now())  
Response.Flush()  
%>
```

代码解释:

- 把报头 "Content-Type" 设置为 "text/event-stream"
- 规定不对页面进行缓存
- 输出发送日期 (始终以 "data: " 开头)
- 向网页刷新输出数据

EventSource 对象

在上面的例子中, 我们使用 `onmessage` 事件来获取消息。不过还可以使用其他事件:

事件	描述
onopen	当通往服务器的连接被打开
onmessage	当接收到消息
onerror	当发生错误

[HTML5 Web Workers](#)

[HTML5 WebSocket](#)

1 篇笔记 写笔记

1. 启明星工作室
983***039@qq.com

75

在上面 asp 代码里，别忘记 `\n\n`。

如果你用 ASP.NET C#制作，则 demo_sse.aspx 代码如下：

```
<%@ Page Language="C#" %>

<script runat="server">
    protected void Page_Load(object sender, EventArgs e)
    {
        Response.ContentType = "text/event-stream";
        Response.Expires = -1;
        Response.Write("data:" + DateTime.Now + "\n\n");
        Response.Flush();
    }
</script>
```

需要以 `\n\n` 结尾，否则，会得不到数据。

[启明星工作室](#)

启明星工作室
983***039@qq.com

5 年前 (2019-02-26)

HTML5 WebSocket

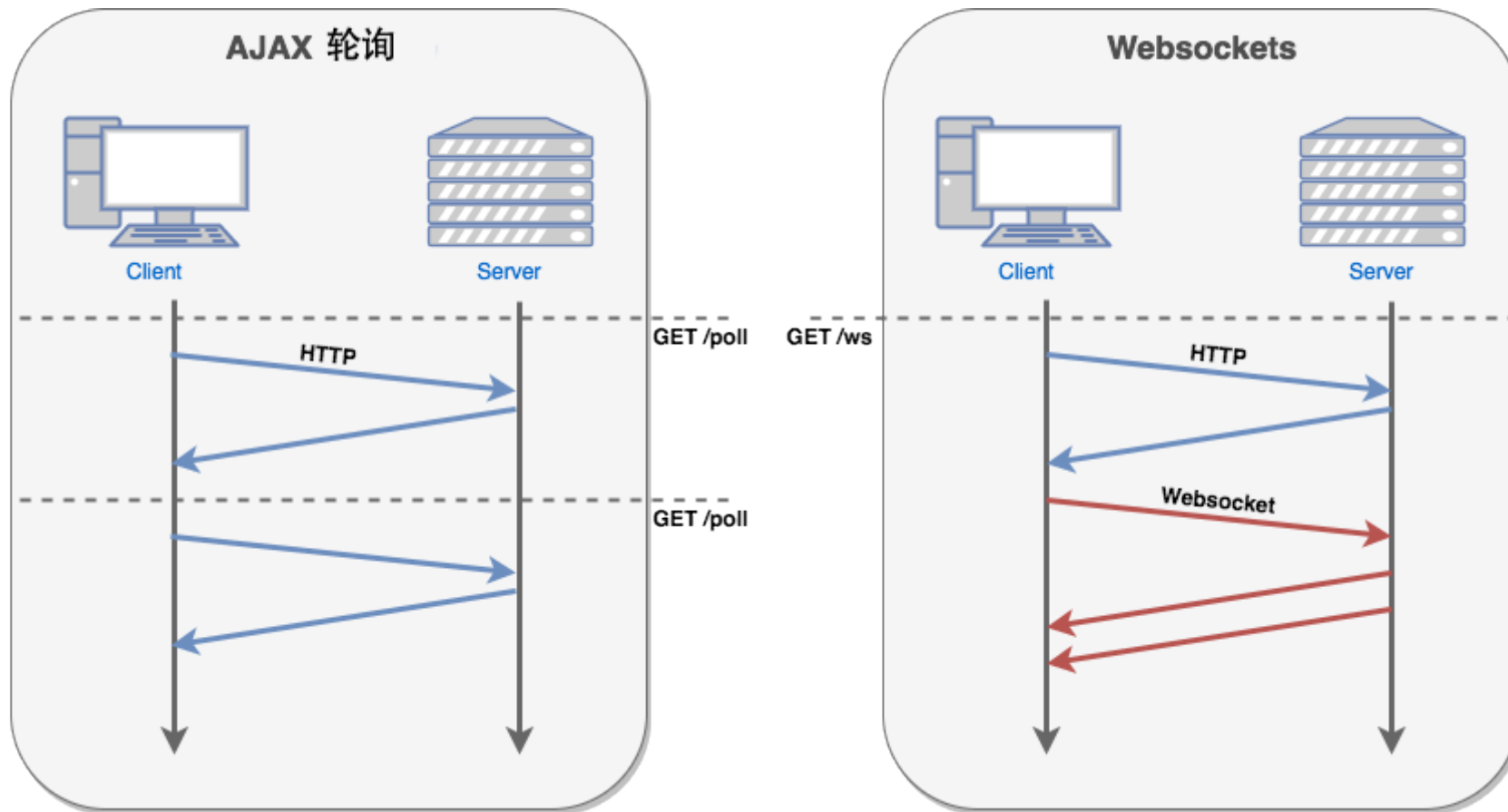
WebSocket 是 HTML5 开始提供的一种在单个 TCP 连接上进行全双工通讯的协议。

WebSocket 使得客户端和服务器之间的数据交换变得更加简单，允许服务端主动向客户端推送数据。在 WebSocket API 中，浏览器和服务器只需要完成一次握手，两者之间就直接可以创建持久性的连接，并进行双向数据传输。

在 WebSocket API 中，浏览器和服务器只需要做一个握手的动作，然后，浏览器和服务器之间就形成了一条快速通道。两者之间就直接可以数据互相传送。

现在，很多网站为了实现推送技术，所用的技术都是 Ajax 轮询。轮询是在特定的时间间隔（如每 1 秒），由浏览器对服务器发出 HTTP 请求，然后由服务器返回最新的数据给客户端的浏览器。这种传统的模式带来很明显的缺点，即浏览器需要不断的向服务器发出请求，然而 HTTP 请求可能包含较长的头部，其中真正有效的数据可能只是很小的一部分，显然这样会浪费很多的带宽等资源。

HTML5 定义的 WebSocket 协议，能更好的节省服务器资源和带宽，并且能够更实时地进行通讯。



浏览器通过 JavaScript 向服务器发出建立 WebSocket 连接的请求，连接建立以后，客户端和服务端就可以通过 TCP 连接直接交换数据。当你获取 Web Socket 连接后，你可以通过 **send()** 方法来向服务器发送数据，并通过 **onmessage** 事件来接收服务器返回的数据。以下 API 用于创建 WebSocket 对象。

```
var Socket = new WebSocket(url, [protocol] );
```

以上代码中的第一个参数 url, 指定连接的 URL。第二个参数 protocol 是可选的，指定了可接受的子协议。

WebSocket 属性

以下是 WebSocket 对象的属性。假定我们使用了以上代码创建了 Socket 对象：

属性	描述
Socket.readyState	只读属性 readyState 表示连接状态，可以是以下值： <ul style="list-style-type: none">• 0 - 表示连接尚未建立。• 1 - 表示连接已建立，可以进行通信。• 2 - 表示连接正在进行关闭。• 3 - 表示连接已经关闭或者连接不能打开。
Socket.bufferedAmount	只读属性 bufferedAmount 已被 send() 放入正在队列中等待传输，但是还没有发出的 UTF-8 文本字节数。

WebSocket 事件

以下是 WebSocket 对象的相关事件。假定我们使用了以上代码创建了 Socket 对象：

事件	事件处理程序	描述
open	Socket.onopen	连接建立时触发
message	Socket.onmessage	客户端接收服务端数据时触发
error	Socket.onerror	通信发生错误时触发
close	Socket.onclose	连接关闭时触发

WebSocket 方法

以下是 WebSocket 对象的相关方法。假定我们使用了以上代码创建了 Socket 对象：

方法	描述
Socket.send()	使用连接发送数据
Socket.close()	关闭连接

WebSocket 实例

WebSocket 协议本质上是一个基于 TCP 的协议。

为了建立一个 WebSocket 连接，客户端浏览器首先要向服务器发起一个 HTTP 请求，这个请求和通常的 HTTP 请求不同，包含了一些附加头信息，其中附加头信息"Upgrade: WebSocket"表明这是一个申请协议升级的 HTTP 请求，服务器端解析这些附加的头信息然后产生应答信息返回给客户端，客户端和服务器的 WebSocket 连接就建立起来了，双方就可以通过这个连接通道自由的传递信息，并且这个连接会持续存在直到客户端或者服务器的某一方主动的关闭连接。

客户端的 HTML 和 JavaScript

目前大部分浏览器支持 WebSocket() 接口，你可以在以下浏览器中尝试实例： Chrome, Mozilla, Opera 和 Safari。

runoob_websocket.html 文件内容

```
<!DOCTYPE HTML>
<html>
  <head>
    <meta charset="utf-8">
    <title>菜鸟教程 (runoob.com)</title>

    <script type="text/javascript">
```

```
function WebSocketTest()
{
    if ("WebSocket" in window)
    {
        alert("您的浏览器支持 WebSocket!");

        // 打开一个 web socket
        var ws = new WebSocket("ws://localhost:9998/echo");

        ws.onopen = function()
        {
            // Web Socket 已连接上, 使用 send() 方法发送数据
            ws.send("发送数据");
            alert("数据发送中...");
        };

        ws.onmessage = function (evt)
        {
            var received_msg = evt.data;
            alert("数据已接收...");
        };

        ws.onclose = function()
        {
            // 关闭 websocket
            alert("连接已关闭...");
        };
    }
}
```

```
    else
    {
        // 浏览器不支持 WebSocket
        alert("您的浏览器不支持 WebSocket!");
    }
}
</script>

</head>
<body>

    <div id="sse">
        <a href="javascript:WebSocketTest()">运行 WebSocket</a>
    </div>

</body>
</html>
```

安装 pywebsocket

在执行以上程序前，我们需要创建一个支持 WebSocket 的服务。从 [pywebsocket](https://github.com/googlearchive/pywebsocket) 下载 mod_pywebsocket ,或者使用 git 命令下载：

```
git clone https://github.com/googlearchive/pywebsocket
```

<p>

mod_pywebsocket 需要 python 环境支持</p>

mod_pywebsocket 是一个 Apache HTTP 的 Web Socket 扩展，安装步骤如下： </p>

```
<ul>
<li><p>解压下载的文件。</p></li>
<li><p>进入 <b>pywebsocket</b> 目录。</p></li>
<li><p>执行命令: </p>
<pre>$ python setup.py build
$ sudo python setup.py install
☒ 查看文档说明:
$ pydoc mod_pywebsocket
```

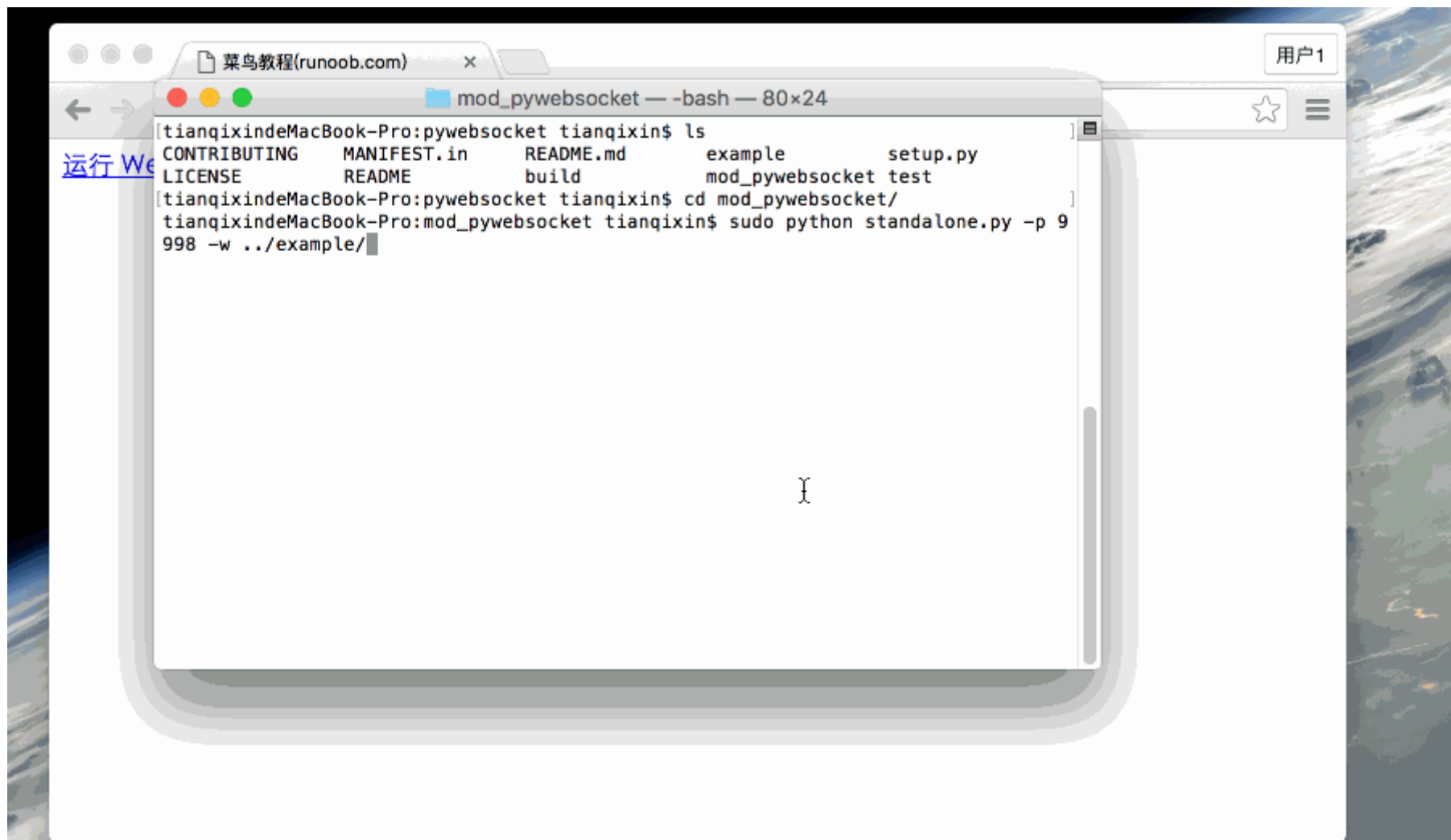
开启服务

在 **pywebsocket/mod_pywebsocket** 目录下执行以下命令:

```
$ sudo python standalone.py -p 9998 -w ../example/
```

以上命令会开启一个端口号为 9998 的服务, 使用 -w 来设置处理程序 echo_wsh.py 所在的目录。

现在我们可以 在 Chrome 浏览器打开前面创建的 runoob_websocket.html 文件。如果你的浏览器支持 WebSocket(), 点击"运行 WebSocket", 你就可以看到整个流程各个步骤弹出的窗口, 流程 Gif 演示:



在我们停止服务后，会弹出 "连接已关闭..."。

[HTML5 服务器发送事件\(Server-Sent Events\)](#)

[HTML5 测验](#)

2 篇笔记 写笔记

1. 0o3
234***242934@qq.com
参考地址

440

Websocket 使用 ws 或 wss 的统一资源标志符，类似于 HTTPS，其中 wss 表示在 TLS 之上的 Websocket。如：

```
ws://example.com/wsapi
wss://secure.example.com/
```

Websocket 使用和 HTTP 相同的 TCP 端口，可以绕过大多数防火墙的限制。默认情况下，Websocket 协议使用 80 端口；运行在 TLS 之上时，默认使用 443 端口。

一个典型的 **Websocket** 握手请求如下：

客户端请求

```
GET / HTTP/1.1
Upgrade: websocket
Connection: Upgrade
Host: example.com
Origin: http://example.com
Sec-WebSocket-Key: sN9cRrP/n9NdMgdcy2VJFQ==
Sec-WebSocket-Version: 13
```

服务器回应

```
HTTP/1.1 101 Switching Protocols
Upgrade: websocket
```

Connection: Upgrade

Sec-WebSocket-Accept: fFBooB7FAkLlXgRSz0BT3v4hq5s=

Sec-WebSocket-Location: ws://example.com/

- o Connection 必须设置 Upgrade，表示客户端希望连接升级。
- o Upgrade 字段必须设置 WebSocket，表示希望升级到 WebSocket 协议。
- o Sec-WebSocket-Key 是随机的字符串，服务器端会用这些数据来构造出一个 SHA-1 的信息摘要。把 “Sec-WebSocket-Key” 加上一个特殊字符串 “258EAF5E914-47DA-95CA-C5AB0DC85B11”，然后计算 SHA-1 摘要，之后进行 BASE-64 编码，将结果做为 “Sec-WebSocket-Accept” 头的值，返回给客户端。如此操作，可以尽量避免普通 HTTP 请求被误认为 WebSocket 协议。
- o Sec-WebSocket-Version 表示支持的 WebSocket 版本。RFC6455 要求使用的版本是 13，之前草案的版本均应当弃用。
- o Origin 字段是可选的，通常用来表示在浏览器中发起此 WebSocket 连接所在的页面，类似于 Referer。但是，与 Referer 不同的是，Origin 只包含了协议和主机名称。
- o 其他一些定义在 HTTP 协议中的字段，如 Cookie 等，也可以在 WebSocket 中使用。

在服务器方面，网上都有不同对 websocket 支持的服务器：

- o php - <http://code.google.com/p/phpwebsocket/>
- o jetty - <http://jetty.codehaus.org/jetty/>（版本 7 开始支持 websocket）
- o netty - <http://www.jboss.org/netty>
- o ruby - <http://github.com/gimite/web-socket-ruby>
- o Kaazing - <https://web.archive.org/web/20100923224709/http://www.kaazing.org/confluence/display/KAAZING/Home>
- o Tomcat - <http://tomcat.apache.org/>（7.0.27 支持 websocket，建议用 tomcat8，7.0.27 中的接口已经过时）
- o WebLogic - <http://www.oracle.com/us/products/middleware/cloud-app-foundation/weblogic/overview/index.html>（12.1.2 开始支持）
- o node.js - <https://github.com/Worlize/WebSocket-Node>
- o node.js - <http://socket.io>

- o nginx - <http://nginx.com/>
- o mojolicious - <http://mojolicio.us/>
- o python - <https://github.com/abourget/gevent-socketio>
- o Django - <https://github.com/stephenmcd/django-socketio>
- o erlang - <https://github.com/ninenines/cowboy.git>

0o3

0o3

234***242934@qq.com

参考地址

6 年前 (2018-09-19)

2. 讨人厌的团子蜀

mrf***uan@hotmail.com

678

说到 websocket 我觉得有必要说下跟 socket 的区别。

软件通信有七层结构，下三层结构偏向与数据通信，上三层更偏向于数据处理，中间的传输层则是连接上三层与下三层之间的桥梁，每一层都做不同的工作，上层协议依赖与下层协议。基于这个通信结构的概念。

Socket 其实并不是一个协议，是应用层与 TCP/IP 协议族通信的中间软件抽象层，它是一组接口。当两台主机通信时，让 Socket 去组织数据，以符合指定的协议。TCP 连接则更依靠于底层的 IP 协议，IP 协议的连接则依赖于链路层等更低层次。

WebSocket 则是一个典型的应用层协议。

总的来说：Socket 是传输控制层协议，WebSocket 是应用层协议。

讨人厌的团子蜀

讨人厌的团子蜀

mrf***uan@hotmail.com

5 年前 (2018-11-14)

HTML5 测验

- [HTML5 测验一](#)
- [HTML5 测验二](#)
- [HTML5 测验三](#)
- [HTML5 测验四](#)
- [HTML5 测验五](#)
- [HTML5 测验六](#)
- [HTML5 测验七](#)

HTML(5) 代码规范

HTML 代码约定

很多 Web 开发人员对 HTML 的代码规范知之甚少。

在 2000 年至 2010 年，许多 Web 开发人员从 HTML 转换到 XHTML。

使用 XHTML 开发人员逐渐养成了比较好的 HTML 编写规范。

而针对于 HTML5，我们应该形成比较好的代码规范，以下提供了几种规范的建议。

使用正确的文档类型

文档类型声明位于 HTML 文档的第一行：

```
<!DOCTYPE html>
```

如果你想跟其他标签一样使用小写，可以使用以下代码：

```
<!doctype html>
```

使用小写元素名

HTML5 元素名可以使用大写和小写字母。

推荐使用小写字母：

- 混合了大小写的风格是非常糟糕的。
- 开发人员通常使用小写 (类似 XHTML)。
- 小写风格看起来更加清爽。
- 小写字母容易编写。

不推荐：

```
<SECTION>  
  <p>这是一个段落。</p>  
</SECTION>
```

非常糟糕：

```
<Section>  
  <p>这是一个段落。</p>  
</SECTION>
```

推荐：

```
<section>
  <p>这是一个段落。</p>
</section>
```

关闭所有 HTML 元素

在 HTML5 中, 你不一定要关闭所有元素 (例如 <p> 元素), 但我们建议每个元素都要添加关闭标签。
不推荐:

```
<section>
  <p>这是一个段落。
  <p>这是一个段落。
</section>
```

推荐:

```
<section>
  <p>这是一个段落。</p>
  <p>这是一个段落。</p>
</section>
```

关闭空的 HTML 元素

在 HTML5 中, 空的 HTML 元素也不一定要关闭:
我们可以这么写:

```
<meta charset="utf-8">
```

也可以这么写:

```
<meta charset="utf-8" />
```

在 XHTML 和 XML 中斜线 (/) 是必须的。
如果你期望 XML 软件使用你的页面, 使用这种风格是非常好的。

使用小写属性名

HTML5 属性名允许使用大写和小写字母。

我们推荐使用小写字母属性名：

- 同时使用大小写是非常不好的习惯。
- 开发人员通常使用小写 (类似 XHTML)。
- 小写风格看起来更加清爽。
- 小写字母容易编写。

不推荐：

```
<div CLASS="menu">
```

推荐：

```
<div class="menu">
```

属性值

HTML5 属性值可以不用引号。

属性值我们推荐使用引号：

- 如果属性值含有空格需要使用引号。
- 混合风格不推荐的，建议统一风格。
- 属性值使用引号易于阅读。

以下实例属性值包含空格，没有使用引号，所以不能起作用：

```
<table class=table striped>
```

以下使用了双引号，是正确的：

```
<table class="table striped">
```

图片属性

图片通常使用 **alt** 属性。在图片不能显示时，它能替代图片显示。

```

```

定义好图片的尺寸，在加载时可以预留指定空间，减少闪烁。

```

```

空格和等号

等号前后可以使用空格。

```
<link rel = "stylesheet" href = "styles.css">
```

但我们推荐少用空格：

```
<link rel="stylesheet" href="styles.css">
```

避免一行代码过长

使用 HTML 编辑器，左右滚动代码是不方便的。

每行代码尽量少于 80 个字符。

空行和缩进

不要无缘无故添加空行。

为每个逻辑功能块添加空行，这样更易于阅读。

缩进使用两个空格，不建议使用 TAB。
比较短的代码间不要使用不必要的空行和缩进。

不必要的空行和缩进:

```
<body>

  <h1>菜鸟教程</h1>

  <h2>HTML</h2>

  <p>
    菜鸟教程，学的不仅是技术，更是梦想。
    菜鸟教程，学的不仅是技术，更是梦想。
    菜鸟教程，学的不仅是技术，更是梦想，
    菜鸟教程，学的不仅是技术，更是梦想。
  </p>

</body>
```

推荐:

```
<body>

<h1>菜鸟教程</h1>

<h2></h2>
<p>菜鸟教程，学的不仅是技术，更是梦想。
```

菜鸟教程，学的不仅是技术，更是梦想。
菜鸟教程，学的不仅是技术，更是梦想。
菜鸟教程，学的不仅是技术，更是梦想。 </p>

</body>

表格实例:

```
<table>
  <tr>
    <th>Name</th>
    <th>Description</th>
  </tr>
  <tr>
    <td>A</td>
    <td>Description of A</td>
  </tr>
  <tr>
    <td>B</td>
    <td>Description of B</td>
  </tr>
</table>
```

列表实例:

```
<ol>
  <li>London</li>
  <li>Paris</li>
```



```
<li>Tokyo</li>
</ol>
```

省略 <html> 和 <body>?

在标准 HTML5 中，<html> 和 <body> 标签是可以省略的。
以下 HTML5 文档是正确的：

实例：

```
<!DOCTYPE html>
<head>
  <title>页面标题</title>
</head>

<h1>这是一个标题</h1>
<p>这是一个段落。</p>
```

尝试一下 »

不推荐省略 <html> 和 <body> 标签。

<html> 元素是文档的根元素，用于描述页面的语言：

```
<!DOCTYPE html>
<html lang="zh">
```

声明语言是为了方便屏幕阅读器及搜索引擎。

省略 <html> 或 <body> 在 DOM 和 XML 软件中会崩溃。

省略 <body> 在旧版浏览器 (IE9) 会发生错误。

省略 <head>?

在标准 HTML5 中，<head> 标签是可以省略的。
默认情况下，浏览器会将 <body> 之前的内容添加到一个默认的 <head> 元素上。

实例

```
<!DOCTYPE html>
<html>
<title>页面标题</title>

<body>
  <h1>这是一个标题</h1>
  <p>这是一个段落。</p>
</body>

</html>
```

尝试一下 »



现在省略 head 标签还不推荐使用。

元数据

HTML5 中 <title> 元素是必须的，标题名描述了页面的主题：

```
<title>菜鸟教程</title>
```

标题和语言可以让搜索引擎很快了解你页面的主题:

```
<!DOCTYPE html>
<html lang="zh">
<head>
  <meta charset="UTF-8">
  <title>菜鸟教程</title>
</head>
```

HTML 注释

注释可以写在 `<!--` 和 `-->` 中:

```
<!-- 这是注释 -->
```

比较长的注释可以在 `<!--` 和 `-->` 中分行写:

```
<!--
  这是一个较长注释。 这是一个较长注释。这是一个较长注释。
  这是一个较长注释 这是一个较长注释。 这是一个较长注释。
-->
```

长注释第一个字符缩进两个空格，更易于阅读。

样式表

样式表使用简洁的语法格式 (`type` 属性不是必须的):

```
<link rel="stylesheet" href="styles.css">
```

短的规则可以写成一行:

```
p.into {font-family: Verdana; font-size: 16em;}
```

长的规则可以写成多行:

```
body {  
  background-color: lightgrey;  
  font-family: "Arial Black", Helvetica, sans-serif;  
  font-size: 16em;  
  color: black;  
}
```

- 将左花括号与选择器放在同一行。
- 左花括号与选择器间添加一个空格。
- 使用两个空格来缩进。
- 冒号与属性值之间添加一个空格。
- 逗号和符号之后使用一个空格。
- 每个属性与值结尾都要使用分号。
- 只有属性值包含空格时才使用引号。
- 右花括号放在新的一行。
- 每行最多 80 个字符。



在逗号和冒号后添加空格是常用的一个规则。

在 HTML 中载入 JavaScript

使用简洁的语法来载入外部的脚本文件 (type 属性不是必须的):

```
<script src="myscript.js">
```

使用 JavaScript 访问 HTML 元素

一个糟糕的 HTML 格式可能会导致 JavaScript 执行错误。

以下两个 JavaScript 语句会输出不同结果:

实例

```
var obj = getElementById("Demo")
```

```
var obj = getElementById("demo")
```

尝试一下 »

HTML 中 JavaScript 尽量使用相同的命名规则。

[访问 JavaScript 代码规范。](#)

使用小写文件名

大多 Web 服务器 (Apache, Unix) 对大小写敏感: london.jpg 不能通过 London.jpg 访问。

其他 Web 服务器 (Microsoft, IIS) 对大小写不敏感: london.jpg 可以通过 London.jpg 或 london.jpg 访问。

你必须保持统一的风格, 我们建议统一使用小写的文件名。

文件扩展名

HTML 文件后缀可以是 .html (或 .htm)。

CSS 文件后缀是 **.css**。

JavaScript 文件后缀是 **.js**。

.htm 和 .html 的区别

.htm 和 .html 的扩展名文件本质上是没有任何区别的。浏览器和 Web 服务器都会把它们当作 HTML 文件来处理。

区别在于：

.htm 应用在早期 DOS 系统，系统现在或者只能有三个字符。

在 Unix 系统中后缀没有特别限制，一般用 .html。

技术上区别

如果一个 URL 没有指定文件名 (如 <http://www.runoob.com/css/>), 服务器会返回默认的文件名。通常默认文件名为 index.html, index.htm, default.html, 和 default.htm。

如果服务器只配置了 "index.html" 作为默认文件，你必须将文件命名为 "index.html", 而不是 "index.htm"。

但是，通常服务器可以设置多个默认文件，你可以根据需要设置默认文件名。

不管怎样，HTML 完整的后缀是 ".html"。

[HTML5 测验](#)

[HTML 媒体\(Media\)](#)

点我分享笔记

HTML 多媒体

Web 上的多媒体指的是音效、音乐、视频和动画。

现代网络浏览器已支持很多多媒体格式。

什么是多媒体？

多媒体来自多种不同的格式。它可以是您听到或看到的任何内容，文字、图片、音乐、音效、录音、电影、动画等等。在因特网上，您会经常发现嵌入网页中的多媒体元素，现代浏览器已支持多种多媒体格式。在本教程中，您将了解到不同的多媒体格式，以及如何在您的网页中使用它们。

浏览器支持

第一款因特网浏览器只支持文本，而且即使是对文本的支持也仅限于单一字体和单一颜色。随后诞生了支持颜色、字体和文本样式的浏览器，图片支持也被加入。不同的浏览器以不同的方式处理对音效、动画和视频的支持。某些元素能够以内联的方式处理，而某些则需要额外的插件。您将在下面的章节学习更多有关插件的知识。

多媒体格式

格式 多媒体元素（比如视频和音频）存储于媒体文件中。确定媒体类型的最常用的方法是查看文件扩展名。当浏览器得到文件扩展名 .htm 或 .html 时，它会假定该文件是 HTML 页面。.xml 扩展名指示 XML 文件，而 .css 扩展名指示样式表。图片格式则通过 .gif 或 .jpg 来识别。多媒体元素也拥有带有不同扩展名的文件格式，比如 .swf、.wmv、.mp3 以及 .mp4。

视频格式



MP4 是互联网推出新的视频格式。

YouTube 推荐使用 MP4 。

Flash Players 支持 MP4

HTML5 支持 MP4。

格式	文件	描述
AVI	.avi	AVI (Audio Video Interleave) 格式是由微软开发的。所有运行 Windows 的计算机都支持 AVI 格式。它是因特网上很常见的格式，但非 Windows 计算机并不总是能够播放。
WMV	.wmv	Windows Media 格式是由微软开发的。Windows Media 在因特网上很常见，但是如果未安装额外的（免费）组件，就无法播放 Windows Media 电影。一些后期的 Windows Media 电影在所有非 Windows 计算机上都无法播放，因为没有合适的播放器。
MPEG	<ul style="list-style-type: none">• .mpg• .mpeg	MPEG (Moving Pictures Expert Group) 格式是因特网上最流行的格式。它是跨平台的，得到了所有最流行的浏览器的支持。
QuickTime	.mov	QuickTime 格式是由苹果公司开发的。QuickTime 是因特网上常见的格式，但是 QuickTime 电影不能在未安装额外的（免费）组件的 Windows 计算机上播放。
RealVideo	<ul style="list-style-type: none">• .rm	RealVideo 格式是由 Real Media 针对因特网开发的。该格式允许低带宽条件下（在线视频、网络电视）的视频流。由于是低带宽优先的，质量常会降低。

	• .ram	
Flash	<ul style="list-style-type: none"> • .swf • .flv 	Flash (Shockwave) 格式是由 Macromedia 开发的。Shockwave 格式需要额外的组件来播放。但是该组件会预装到 Firefox 或 IE 之类的浏览器上。
Mpeg-4	.mp4	Mpeg-4 (with H.264 video compression) 是一种针对因特网的新格式。事实上, YouTube 推荐使用 MP4。YouTube 接收多种格式, 然后全部转换为 .flv 或 .mp4 以供分发。越来越多的视频发布者转到 MP4, 将其作为 Flash 播放器和 HTML5 的因特网共享格式。



最新的 HTML5 标准只支持 MP4, WebM, 和 Ogg 视频格式。

声音格式

MP3 是一种音频压缩技术, 其全称是动态影像专家压缩标准音频层面 3 (Moving Picture Experts Group Audio Layer III), 简称为 MP3。它被设计用来大幅度地降低音频数据量。如果你的站点是音乐类型的, 你可以选择 mp3 格式。

格式	文件	描述
MIDI	<ul style="list-style-type: none"> • .mid • .midi 	<p>MIDI (Musical Instrument Digital Interface) 是一种针对电子音乐设备 (比如合成器和声卡) 的格式。MIDI 文件不含有声音, 但包含可被电子产品 (比如声卡) 播放的数字音乐指令。</p> <p>点击这里播放 The Beatles。</p> <p>因为 MIDI 格式仅包含指令, 所以 MIDI 文件极其小巧。上面的例子只有 23k 的大小, 但却能播放将近 5 分钟。MIDI 得到了广泛的平台上的大量软件的支持。大多数流行的网络浏览器都支持 MIDI。</p>
RealAudio	• .rm	RealAudio 格式是由 Real Media 针对因特网开发的。该格式也支持视频。该格式允许低带宽条件下

	• .ram	的音频流（在线音乐、网络音乐）。由于是低带宽优先的，质量常会降低。
Wave	.wav	Wave (waveform) 格式是由 IBM 和微软开发的。所有运行 Windows 的计算机和所有网络浏览器（除了 Google Chrome）都支持它。
WMA	.wma	WMA 格式 (Windows Media Audio)，质量优于 MP3，兼容大多数播放器，除了 iPod。WMA 文件可作为连续的数据流来传输，这使它对于网络电台或在线音乐很实用。
MP3	<ul style="list-style-type: none"> • .mp3 • .mpga 	MP3 文件实际上是 MPEG 文件的声音部分。MPEG 格式最初是由运动图像专家组开发的。MP3 是最受欢迎的针对音乐的声音格式。期待未来的软件系统都支持它。



HTML5 的最新标准支持 MP3, WAV, 和 Ogg 音频格式。

[HTML\(5\) 代码规范](#)

[HTML 插件](#)

点我分享笔记

分类导航

- **HTML / CSS**
- **JavaScript**

- 服务端
- 数据库
- 数据分析
- 移动端
- XML 教程
- ASP.NET
- Web Service
- 开发工具
- 网站建设

反馈/建议反馈/建议

HTML 插件

插件的功能是扩展 HTML 浏览器的功能。

HTML 助手（插件）

辅助应用程序（helper application）是可由浏览器启动的程序。辅助应用程序也称为插件。

辅助程序可用于播放音频和视频（以及其他）。辅助程序是使用 `<object>` 标签来加载的。

使用辅助程序播放视频和音频的一个优势是，您能够允许用户来控制部分或全部播放设置。

插件可以通过 `<object>` 标签或者 `<embed>` 标签添加在页面中。

大多数辅助应用程序允许对音量设置和播放功能（比如后退、暂停、停止和播放）的手工（或程序的）控制。

注意：

大多数浏览器不再支持 *Java* 小程序和插件。

大多数现代浏览器关闭了对 *Flash* 的支持。

以下 `swf` 文件的实例在现代浏览器中无法正常运行。



我们可以使用 `<video>` 和 `<audio>` 标签来显示视频和音频

`<object>` 元素

所有主流浏览器都支持 `<object>` 标签。

`<object>` 元素定义了 `HTML` 文档中嵌入的对象。

该标签用于插入对象 (例如在网页中嵌入 `Java` 小程序, `PDF` 阅读器, `Flash` 播放器)。

实例

```
<object width="400" height="50" data="bookmark.swf"></object>
```

尝试一下 »

`<object>` 元素同样可用于包含 `HTML` 文件:

实例

```
<object width="100%" height="500px" data="snippet.html"></object>
```

尝试一下 »

或者插入一张图片:

实例

```
<object data="audi.jpeg"></object>
```

尝试一下 »

<embed> 元素

所有主流浏览器都支持 <embed> 元素。

<embed> 元素表示一个 HTML Embed 对象。

<embed> 元素已经出现很长一段时间了，但是在 HTML5 前并未被详细说明，该元素在 HTML 5 页面上会被验证，在 HTML 4 上不会。

实例

```
<embed width="400" height="50" src="bookmark.swf">
```

尝试一下 »



注意 <embed> 元素没有关闭标签。不能使用替代文本。

<embed> 元素同样可用于包含 HTML 文件：

实例

```
<embed width="100%" height="500px" src="snippet.html">
```

尝试一下 »

或者插入一张图片：

实例

```
<embed src="audi.jpeg">
```

尝试一下 »

HTML 媒体(Media)

HTML 音频(Audio)

点我分享笔记

HTML 音频(Audio)

声音在 HTML 中可以以不同的方式播放。

问题以及解决方法

在 HTML 中播放音频并不容易！

您需要谙熟大量技巧，以确保您的音频文件在所有浏览器中（Internet Explorer, Chrome, Firefox, Safari, Opera）和所有硬件上（PC, Mac , iPad, iPhone）都能够播放。

在本章，菜鸟教程为您总结了问题和解决方法。

使用插件

浏览器插件是一种扩展浏览器标准功能的小型计算机程序。

插件可以使用 <object> 标签 或者 <embed> 标签添加在页面上。

这些标签定义资源（通常非 HTML 资源）的容器，根据类型，它们即会由浏览器显示，也会由外部插件显示。

使用 <embed> 元素

<embed>标签定义外部（非 HTML）内容的容器。（这是一个 HTML5 标签，在 HTML4 中是非法的，但是所有浏览器中都有效）。

下面的代码片段能够显示嵌入网页中的 MP3 文件：

实例

```
<embed height="50" width="100" src="horse.mp3">
```

尝试一下 »

问题：

- <embed> 标签在 HTML 4 中是无效的。页面无法通过 HTML 4 验证。
- 不同的浏览器对音频格式的支持也不同。
- 如果浏览器不支持该文件格式，没有插件的话就无法播放该音频。
- 如果用户的计算机未安装插件，无法播放音频。
- 如果把该文件转换为其他格式，仍然无法在所有浏览器中播放。

使用 <object> 元素

<object tag> 标签也可以定义外部（非 HTML）内容的容器。

下面的代码片段能够显示嵌入网页中的 MP3 文件：

实例

```
<object height="50" width="100" data="horse.mp3"></object>
```

尝试一下 »

问题:

- 不同的浏览器对音频格式的支持也不同。
 - 如果浏览器不支持该文件格式，没有插件的话就无法播放该音频。
 - 如果用户的计算机未安装插件，无法播放音频。
 - 如果把该文件转换为其他格式，仍然无法在所有浏览器中播放。
-

使用 HTML5 <audio> 元素

HTML5 <audio> 元素是一个 HTML5 元素，在 HTML 4 中是非法的，但在所有浏览器中都有效。
The <audio> element works in all modern browsers.

浏览器兼容

格中的数字表示支持该属性的第一个浏览器版本号。

元素					
<audio>	4.0	9.0	3.5	4.0	10.5

以下我们将使用 <audio> 标签来描述 MP3 文件(Internet Explorer、Chrome 以及 Safari 中是有效的), 同样添加了一个 OGG 类型文件(Firefox 和 Opera 浏览器中有效).如果失败，它会显示一个错误文本信息:

实例


```
<audio controls>
  <source src="horse.mp3" type="audio/mpeg">
  <source src="horse.ogg" type="audio/ogg">
  Your browser does not support this audio format.
</audio>
```

尝试一下 »

问题:

- <audio> 标签在 HTML 4 中是无效的。您的页面无法通过 HTML 4 验证。
- 您必须把音频文件转换为不同的格式。
- <audio> 元素在老式浏览器中不起作用。

最好的 HTML 解决方法

下面的例子使用了两个不同的音频格式。HTML5 <audio> 元素会尝试以 mp3 或 ogg 来播放音频。如果失败，代码将回退尝试 <embed> 元素。

实例

```
<audio controls height="100" width="100">
  <source src="horse.mp3" type="audio/mpeg">
  <source src="horse.ogg" type="audio/ogg">
  <embed height="50" width="100" src="horse.mp3">
</audio>
```

尝试一下 »

问题:

- 您必须把音频转换为不同的格式。
- `<embed>` 元素无法回退来显示错误消息。

使用超链接

如果网页包含指向媒体文件的超链接，大多数浏览器会使用"辅助应用程序"来播放文件。

以下代码片段显示指向 mp3 文件的链接。如果用户点击该链接，浏览器会启动"辅助应用程序"来播放该文件：

实例

```
<a href="horse.mp3">Play the sound</a>
```

尝试一下 »

内联的声音说明

当您在网页中包含声音，或者作为网页的组成部分时，它被称为内联声音。

如果您打算在 web 应用程序中使用内联声音，您需要意识到很多人都觉得内联声音令人恼火。同时请注意，用户可能已经关闭了浏览器中的内联声音选项。

我们最好的建议是只在用户希望听到内联声音的地方包含它们。一个正面的例子是，在用户需要听到录音并点击某个链接时，会打开页面然后播放录音。

HTML 多媒体标签

New : HTML5 新标签

标签	描述
<embed>	定义内嵌对象。HTML4 中不赞成，HTML5 中允许。
<object>	定义内嵌对象。
<param>	定义对象的参数。
<audio> New	定义了声音内容
<video> New	定义一个视频或者影片
<source> New	定义了 media 元素的多媒体资源(<video> 和 <audio>)
<track> New	规定 media 元素的字幕文件或其他包含文本的文件 (<video> 和<audio>)

[HTML 插件](#)

[HTML 视频（Video）播放](#)

点我分享笔记

HTML 视频（Video）

在 HTML 中播放视频的方法有很多种。

HTML 视频（Videos）播放

实例

```
<video width="320" height="240" controls> <source src="movie.mp4" type="video/mp4"> <source src="movie.ogg" type="video/ogg"> <source src="movie.webm" type="video/webm"> <object data="movie.mp4" width="320" height="240"> <embed src="movie.swf" width="320" height="240"> </object> </video>
```

尝试一下 »

问题以及解决方法

在 HTML 中播放视频并不容易！

您需要谙熟大量技巧，以确保您的视频文件在所有浏览器中（Internet Explorer, Chrome, Firefox, Safari, Opera）和所有硬件上（PC, Mac , iPad, iPhone）都能够播放。

在本章，菜鸟教程为您总结了问题和解决方法。

使用 <embed> 标签

<embed> 标签的作用是在 HTML 页面中嵌入多媒体元素。

下面的 HTML 代码显示嵌入网页的 Flash 视频：

实例

```
<embed src="intro.swf" height="200" width="200">
```

尝试一下 »

问题

- HTML4 无法识别 <embed> 标签。您的页面无法通过验证。
- 如果浏览器不支持 Flash，那么视频将无法播放
- iPad 和 iPhone 不能显示 Flash 视频。
- 如果您将视频转换为其他格式，那么它仍然不能在所有浏览器中播放。

使用 <object> 标签

<object> 标签的作用是在 HTML 页面中嵌入多媒体元素。

下面的 HTML 片段显示嵌入网页的一段 Flash 视频：

实例

```
<object data="intro.swf" height="200" width="200"></object>
```

尝试一下 »

问题：

- 如果浏览器不支持 Flash，将无法播放视频。

- iPad 和 iPhone 不能显示 Flash 视频。
 - 如果您将视频转换为其他格式，那么它仍然不能在所有浏览器中播放。
-

使用 HTML5 <video> 元素

HTML5 <video> 标签定义了一个视频或者影片。

<video> 元素在所有现代浏览器中都支持。

以下 HTML 片段会显示一段嵌入网页的 ogg、mp4 或 webm 格式的视频：

实例

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  <source src="movie.webm" type="video/webm">
```

您的浏览器不支持 video 标签。

</video>

尝试一下 »

问题：

- 您必须把视频转换为很多不同的格式。
 - <video> 元素在老式浏览器中无效。
-

最好的 HTML 解决方法

以下实例中使用了 4 种不同的视频格式。HTML 5 <video> 元素会尝试播放以 mp4、ogg 或 webm 格式中的一种来播放视频。如果均失败，则回退到 <embed> 元素。

HTML 5 + <object> + <embed>

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  <source src="movie.webm" type="video/webm">
  <object data="movie.mp4" width="320" height="240">
    <embed src="movie.swf" width="320" height="240">
  </object>
</video>
```

尝试一下 »

问题:

- 您必须把视频转换为很多不同的格式

使用超链接

如果网页包含指向媒体文件的超链接，大多数浏览器会使用"辅助应用程序"来播放文件。

以下代码片段显示指向 AVI 文件的链接。如果用户点击该链接，浏览器会启动"辅助应用程序"，比如 Windows Media Player 来播放这个 AVI 文件：

实例

```
<a href="intro.swf">Play a video file</a>
```

尝试一下 »

关于内联视频的说明

当视频被包含在网页中时，它被称为内联视频。

如果您打算在 web 应用程序中使用内联视频，您需要意识到很多人都觉得内联视频令人恼火。

同时请注意，用户可能已经关闭了浏览器中的内联视频选项。

我们最好的建议是只在用户希望看到内联视频的地方包含它们。一个正面的例子是，在用户需要看到视频并点击某个链接时，会打开页面然后播放视频。

HTML 多媒体标签

New : HTML5 新标签.

标签	描述
<embed>	定义内嵌对象。HTML4 中不赞成，HTML5 中允许。
<object>	定义内嵌对象。
<param>	定义对象的参数。
<audio> New	定义了声音内容
<video> New	定义一个视频或者影片

<code><source></code> New	定义了 media 元素的多媒体资源(<video> 和 <audio>)
<code><track></code> New	规定 media 元素的字幕文件或其他包含文本的文件 (<video> 和<audio>)

[HTML 音频\(Audio\)](#)

[HTML 实例](#)

[点我分享笔记](#)

HTML 实例

HTML 基础

[非常简单的 HTML 文档](#)

[HTML 标题](#)

[HTML 段落](#)

[HTML 链接](#)

[HTML 图片](#)

[实例解析](#)

HTML 标题

[HTML 标题](#)

[在 html 源码中插入注释](#)

[插入水平线](#)

[实例解析](#)

HTML 段落

[HTML 段落](#)

[更多段落](#)

[本例演示在 HTML 文档中折行的使用。](#)

[HTML 格式化的某些问题。](#)

[实例解析](#)

HTML 文本格式化

[文本格式化](#)

[此例演示如何使用 pre 标签对空行和空格进行控制。](#)

[此例演示不同的"计算机输出"标签的显示效果。](#)

[此例演示如何在 HTML 文件中写地址。](#)

[此例演示如何实现缩写或首字母缩写。](#)

[此例演示如何改变文字的方向。](#)

[此例演示如何实现长短不一的引用语。](#)

[文本下划线与删除线](#)

[实例解析](#)

HTML 样式

[HTML Style 元素](#)

[背景色样式](#)

[字体样式，颜色，大小](#)

[文本对齐样式](#)

[设置文本字体](#)

[设置文本字体大小](#)

[设置文本字体颜色](#)

[设置文本字体，字体大小，字体颜色](#)

[HTML 使用不同样式](#)

[没有下划线的链接](#)

[链接到一个外部样式表](#)

[实例解析](#)

HTML 链接

[创建超级链接](#)

[将图像作为链接](#)

[在新的浏览器窗口打开链接](#)

[链接到同一个页面的不同位置](#)

[跳出框架](#)

[创建电子邮件链接](#)

[创建电子邮件链接 2](#)

[实例解析](#)

HTML 图像

[插入图像](#)

[从不同的位置插入图片](#)

[排列图片](#)

[本例演示如何使图片浮动至段落的左边或右边。](#)

[制作图像链接](#)

[创建图像映射](#)

[实例解析](#)

HTML 表格

[简单的表格](#)

[没有边框的表格](#)

[表格中的表头](#)

[带有标题的表格](#)

[跨行或跨列的表格单元格](#)

[表格内的标签](#)

[单元格边距\(Cell padding\)](#)

[单元格间距\(Cell spacing\)](#)

[实例解析](#)

HTML 列表

[无序列表](#)

[有序列表](#)

[不同类型的有序列表](#)

[不同类型的无序列表](#)

[嵌套列表](#)

[嵌套列表 2](#)

[定义列表](#)

[实例解析](#)

HTML Forms 和 Input

[创建文本域\(Text fields\)](#)

[创建密码域](#)

[复选框](#)

[单选按钮](#)

[简单的下拉列表](#)

[预选下拉列表](#)

[本例演示如何创建一个文本域（多行文本输入控件）。](#)

[创建一个按钮](#)

[本例演示如何在数据周围绘制一个带标题的框。](#)

[带有文本域与输入域的表单](#)

[带有复选框与提交按钮的 form 表单](#)

[带有单选框与提交按钮的表单](#)

[发送邮件表单](#)

[实例解析](#)

HTML iframe

[内联框架 \(HTML 页面中插入框架\)](#)

[实例解析](#)

HTML 头部元素

[描述了文档标题](#)

[HTML 页面中默认的 URL 链接](#)

[提供文档元数据](#)

[实例解析](#)

HTML 脚本

[插入一个脚本](#)

[使用 <noscript> 标签](#)

[实例解析](#)

更多实例: <https://www.jyshare.com/examples/>

[HTML 视频 \(Video\) 播放](#)

[HTML 标签列表\(字母排序\)](#)

点我分享笔记

[HTML 标签列表 \(功能排序\)](#)

HTML 参考手册- (HTML5 标准)

按字母顺序排列

New : HTML5 新标签

标签	描述
<u><!--...--></u>	定义注释
<u><!DOCTYPE></u>	定义文档类型
<u><a></u>	定义超文本链接
<u><abbr></u>	定义缩写
<u><acronym></u>	定义只取首字母的缩写，不支持 HTML5
<u><address></u>	定义文档作者或拥有者的联系信息
<u><applet></u>	HTML5 中不赞成使用。定义嵌入的 applet。
<u><area></u>	定义图像映射内部的区域
<u><article></u> New	定义一个文章区域
<u><aside></u> New	定义页面的侧边栏内容

<u><audio></u> New	定义音频内容
<u></u>	定义文本粗体
<u><base></u>	定义页面中所有链接的默认地址或默认目标。
<u><basefont></u>	HTML5 不支持，不赞成使用。定义页面中文本的默认字体、颜色或尺寸。
<u><bdi></u> New	允许您设置一段文本，使其脱离其父元素的文本方向设置。
<u><bdo></u>	定义文字方向
<u><big></u>	定义大号文本，HTML5 不支持
<u><blockquote></u>	定义长的引用
<u><body></u>	定义文档的主体
<u>
</u>	定义换行
<u><button></u>	定义一个点击按钮
<u><canvas></u> New	定义图形，比如图表和其他图像,标签只是图形容器，您必须使用脚本来绘制图形
<u><caption></u>	定义表格标题
<u><center></u>	HTML5 不支持，不赞成使用。定义居中文本。

<u><cite></u>	定义引用(citation)
<u><code></u>	定义计算机代码文本
<u><col></u>	定义表格中一个或多个列的属性值
<u><colgroup></u>	定义表格中供格式化的列组
<u><command></u> New	定义命令按钮，比如单选按钮、复选框或按钮
<u><datalist></u> New	定义选项列表。请与 input 元素配合使用该元素，来定义 input 可能的值。
<u><dd></u>	定义定义列表中项目的描述
<u></u>	定义被删除文本
<u><details></u> New	用于描述文档或文档某个部分的细节
<u><dfn></u>	定义定义项目
<u><dialog></u> New	定义对话框，比如提示框
<u><dir></u>	HTML5 不支持，不赞成使用。定义目录列表。
<u><div></u>	定义文档中的节

<u><dl></u>	定义列表详情
<u><dt></u>	定义列表中的项目
<u></u>	定义强调文本
<u><embed></u> New	定义嵌入的内容，比如插件。
<u><fieldset></u>	定义围绕表单中元素的边框
<u><figcaption></u> New	定义<figure> 元素的标题
<u><figure></u> New	规定独立的流内容（图像、图表、照片、代码等等）。
<u></u>	HTML5 不支持，不赞成使用。定义文字的字体、尺寸和颜色。
<u><footer></u> New	定义 section 或 document 的页脚。
<u><form></u>	定义了 HTML 文档的表单
<u><frame></u>	定义框架集的窗口或框架
<u><frameset></u>	定义框架集
<u><h1> to <h6></u>	定义 HTML 标题

<u><head></u>	定义关于文档的信息
<u><header></u> New	定义了文档的头部区域
<u><hr></u>	定义水平线
<u><html></u>	定义 HTML 文档
<u><i></u>	定义斜体字
<u><iframe></u>	定义内联框架
<u></u>	定义图像
<u><input></u>	定义输入控件
<u><ins></u>	定义被插入文本
<u><kbd></u>	定义键盘文本
<u><keygen></u> New	规定用于表单的密钥对生成器字段。
<u><label></u>	定义 input 元素的标注
<u><legend></u>	定义 fieldset 元素的标题。
<u></u>	定义列表的项目
<u><link></u>	定义文档与外部资源的关系

<u><main></u>	定义文档的主体部分。
<u><map></u>	定义图像映射
<u><mark></u> New	定义带有记号的文本。请在需要突出显示文本时使用 标签。
<u><menu></u>	不赞成使用。定义菜单列表。
<u><meta></u>	定义关于 HTML 文档的元信息。
<u><meter></u> New	定义度量衡。仅用于已知最大和最小值的度量。
<u><nav></u> New	定义导航链接的部分
<u><noframes></u>	定义针对不支持框架的用户的替代内容。HTML5 不支持
<u><noscript></u>	定义针对不支持客户端脚本的用户的替代内容。
<u><object></u>	定义内嵌对象
<u></u>	定义有序列表。
<u><optgroup></u>	定义选择列表中相关选项的组合。
<u><option></u>	定义选择列表中的选项。

<u><output></u> New	定义不同类型的输出，比如脚本的输出。
<u><p></u>	定义段落。
<u><param></u>	定义对象的参数。
<u><pre></u>	定义预格式文本。
<u><progress></u> New	定义运行中的进度（进程）。
<u><q></u>	定义短的引用。
<u><rp></u> New	<rp> 标签在 ruby 注释中使用，以定义不支持 ruby 元素的浏览器所显示的内容。
<u><rt></u> New	<rt> 标签定义字符（中文注音或字符）的解释或发音。
<u><ruby></u> New	<ruby> 标签定义 ruby 注释（中文注音或字符）。
<u><s></u>	不赞成使用。定义加删除线的文本。
<u><samp></u>	定义计算机代码样本。
<u><script></u>	定义客户端脚本。

<u><section></u> New	<section> 标签定义文档中的节（section、区段）。比如章节、页眉、页脚或文档中的其他部分。
<u><select></u>	定义选择列表（下拉列表）。
<u><small></u>	定义小号文本。
<u><source></u> New	<source> 标签为媒介元素（比如 <video> 和 <audio>）定义媒介资源。
<u></u>	定义文档中的节。
<u><strike></u>	HTML5 不支持，不赞成使用。定义加删除线文本。
<u></u>	定义强调文本。
<u><style></u>	定义文档的样式信息。
<u><sub></u>	定义下标文本。
<u><summary></u> New	<summary> 标签包含 details 元素的标题，"details" 元素用于描述有关文档或文档片段的详细信息。
<u><sup></u>	定义上标文本。
<u><table></u>	定义表格。
<u><tbody></u>	定义表格中的主体内容。
<u><td></u>	定义表格中的单元。

<u><textarea></u>	定义多行的文本输入控件。
<u><tfoot></u>	定义表格中的表注内容（脚注）。
<u><th></u>	定义表格中的表头单元格。
<u><thead></u>	定义表格中的表头内容。
<u><time></u> New	定义日期或时间，或者两者。
<u><template></u> New	定义在页面加载时隐藏的一些内容。
<u><title></u>	定义文档的标题。
<u><tr></u>	定义表格中的行。
<u><track></u> New	<track> 标签为诸如 video 元素之类的媒介规定外部文本轨道。
<u><tt></u>	定义打字机文本。
<u><u></u>	不赞成使用。定义下划线文本。
<u></u>	定义无序列表。
<u><var></u>	定义文本的变量部分。

<u><video></u> <div>New</div>	<video> 标签定义视频，比如电影片段或其他视频流。
<u><wbr></u> <div>New</div>	规定在文本中的何处适合添加换行符。

[HTML 标签列表（功能排序）](#)

点我分享笔记

[HTML 标签列表\(字母排序\)](#)

[HTML 全局属性](#)

HTML 参考手册- (HTML5 标准)

功能排序

New

 : HTML5 新标签

标签	描述
基础	
<u><!DOCTYPE></u>	定义文档类型。
<u><html></u>	定义一个 HTML 文档

<u><title></u>	为文档定义一个标题
<u><body></u>	定义文档的主体
<u><h1> to <h6></u>	定义 HTML 标题
<u><p></u>	定义一个段落
<u>
</u>	定义简单的折行。
<u><hr></u>	定义水平线。
<u><!--...--></u>	定义一个注释
格式	
<u><acronym></u>	HTML5 不再支持。 定义只取首字母的缩写。
<u><abbr></u>	定义一个缩写。
<u><address></u>	定义文档作者或拥有者的联系信息。
<u></u>	定义粗体文本。
<u><bdi></u> 	允许您设置一段文本，使其脱离其父元素的文本方向设置。
<u><bdo></u>	定义文本的方向。
<u><big></u>	HTML5 不再支持。 定义大号文本。

<u><blockquote></u>	定义块引用。
<u><center></u>	HTML5 不再支持。 HTML 4.01 已废弃。 定义居中文本。
<u><cite></u>	定义引用(citation)。
<u><code></u>	定义计算机代码文本。
<u></u>	定义被删除文本。
<u><dfn></u>	定义定义项目。
<u></u>	定义强调文本。
<u></u>	HTML5 不再支持。 HTML 4.01 已废弃。 定义文本的字体、尺寸和颜色
<u><i></u>	定义斜体文本。
<u><ins></u>	定义被插入文本。
<u><kbd></u>	定义键盘文本。
<u><mark></u> New	定义带有记号的文本。
<u><meter></u> New	定义度量衡。仅用于已知最大和最小值的度量。
<u><pre></u>	定义预格式文本

<u><progress></u> New	定义运行中的任务进度（进程）。
<u><q></u>	定义短的引用。
<u><rp></u> New	定义不支持 ruby 元素的浏览器所显示的内容。
<u><rt></u> New	定义字符（中文注音或字符）的解释或发音。
<u><ruby></u> New	定义 ruby 注释（中文注音或字符）。
<u><s></u>	定义加删除线的文本。
<u><samp></u>	定义计算机代码样本。
<u><small></u>	定义小号文本。
<u><strike></u>	HTML5 不再支持。HTML 4.01 已废弃。定义加删除线的文本。
<u></u>	定义语气更为强烈的强调文本。
<u><sub></u>	定义下标文本。
<u><sup></u>	定义上标文本。
<u><time></u> New	定义一个日期/时间

<u><tt></u>	HTML5 不再支持。 定义打字机文本。
<u><u></u>	定义下划线文本。
<u><var></u>	定义文本的变量部分。
<u><wbr></u> New	规定在文本中的何处适合添加换行符。
表单	
<u><form></u>	定义一个 HTML 表单，用于用户输入。
<u><input></u>	定义一个输入控件
<u><textarea></u>	定义多行的文本输入控件。
<u><button></u>	定义按钮。
<u><select></u>	定义选择列表（下拉列表）。
<u><optgroup></u>	定义选择列表中相关选项的组合。
<u><option></u>	定义选择列表中的选项。
<u><label></u>	定义 input 元素的标注。
<u><fieldset></u>	定义围绕表单中元素的边框。
<u><legend></u>	定义 fieldset 元素的标题。

<u><datalist></u> New	规定了 input 元素可能的选项列表。
<u><keygen></u> New	规定用于表单的密钥对生成器字段。
<u><output></u> New	定义一个计算的结果
框架	
<u><frame></u>	HTML5 不再支持。 定义框架集的窗口或框架。
<u><frameset></u>	HTML5 不再支持。 定义框架集。
<u><noframes></u>	HTML5 不再支持。 定义针对不支持框架的用户的替代内容。
<u><iframe></u>	定义内联框架。
图像	
<u></u>	定义图像。
<u><map></u>	定义图像映射。
<u><area></u>	定义图像地图内部的区域。
<u><canvas></u> New	通过脚本（通常是 JavaScript）来绘制图形（比如图表和其他图像）。

<u><figcaption></u> New	定义一个 caption for a <figure> element
<u><figure></u> New	figure 标签用于对元素进行组合。
Audio/Video	
<u><audio></u> New	定义声音，比如音乐或其他音频流。
<u><source></u> New	定义 media 元素 (<video> 和 <audio>)的媒体资源。media
<u><track></u> New	为媒体(<video> 和 <audio>)元素定义外部文本轨道。
<u><video></u> New	定义一个音频或者视频
链接	
<u><a></u>	定义一个链接
<u><link></u>	定义文档与外部资源的关系。
<u><main></u>	定义文档的主体部分。
<u><nav></u> New	定义导航链接

列表	
<u></u>	定义一个无序列表
<u></u>	定义一个有序列表
<u></u>	定义一个列表项
<u><dir></u>	HTML5 不再支持。HTML 4.01 已废弃。定义目录列表。
<u><dl></u>	定义一个定义列表
<u><dt></u>	定义一个定义定义列表中的项目。
<u><dd></u>	定义定义列表中项目的描述。
<u><menu></u>	定义菜单列表。
<u><command></u> 	定义用户可能调用的命令（比如单选按钮、复选框或按钮）。
表格	
<u><table></u>	定义一个表格
<u><caption></u>	定义表格标题。
<u><th></u>	定义表格中的表头单元格。
<u><tr></u>	定义表格中的行。

<u><td></u>	定义表格中的单元。
<u><thead></u>	定义表格中的表头内容。
<u><tbody></u>	定义表格中的主体内容。
<u><tfoot></u>	定义表格中的表注内容（脚注）。
<u><col></u>	定义表格中一个或多个列的属性值。
<u><colgroup></u>	定义表格中供格式化的列组。
样式/节	
<u><style></u>	定义文档的样式信息。
<u><div></u>	定义文档中的节。
<u></u>	定义文档中的节。
<u><header></u> 	定义一个文档头部部分
<u><footer></u> 	定义一个文档底部
<u><section></u> 	定义了文档的某个区域

<u><article></u> New	定义一个文章内容
<u><aside></u> New	定义其所处内容之外的内容。
<u><details></u> New	定义了用户可见的或者隐藏的需求的补充细节。
<u><dialog></u> New	定义一个对话框或者窗口
<u><summary></u> New	定义一个可见的标题。 当用户点击标题时会显示出详细信息。
元信息	
<u><head></u>	定义关于文档的信息
<u><meta></u>	定义关于 HTML 文档的元信息。
<u><base></u>	定义页面中所有链接的默认地址或默认目标。
<u><basefont></u>	HTML5 不再支持。 HTML 4.01 已废弃。 定义页面中文本的默认字体、颜色或尺寸。
程序	
<u><script></u>	定义客户端脚本。
<u><noscript></u>	定义针对不支持客户端脚本的用户的替代内容。

<code><applet></code>	HTML5 不再支持。HTML 4.01 已废弃。定义嵌入的 applet。
<code><embed></code> New	定义了一个容器，用来嵌入外部应用或者互动程序（插件）。
<code><object></code>	定义嵌入的对象。
<code><param></code>	定义对象的参数。

[HTML 标签列表\(字母排序\)](#)

[HTML 全局属性](#)

点我分享笔记

HTML 全局属性

New : HTML5 新属性。

属性	描述
<code>accesskey</code>	设置访问元素的键盘快捷键。
<code>class</code>	规定元素的类名（classname）
<code>contenteditable</code> New	规定是否可编辑元素的内容。

contextmenuNew	指定一个元素的上下文菜单。当用户右击该元素，出现上下文菜单
data-*New	用于存储页面的自定义数据
dir	设置元素中内容的文本方向。
draggableNew	指定某个元素是否可以拖动
dropzoneNew	指定是否将数据复制，移动，或链接，或删除
hiddenNew	hidden 属性规定对元素进行隐藏。
id	规定元素的唯一 id
lang	设置元素中内容的语言代码。
spellcheckNew	检测元素是否拼写错误
style	规定元素的行内样式（inline style）
tabindex	设置元素的 Tab 键控制次序。
title	规定元素的额外信息（可在工具提示中显示）

translateNew	指定是否一个元素的值在页面载入时是否需要翻译
--------------	------------------------

HTML 标签列表（功能排序）

HTML 事件

点我分享笔记

HTML 事件属性

全局事件属性

HTML 4 的新特性之一是可以使 HTML 事件触发浏览器中的行为，比方说当用户点击某个 HTML 元素时启动一段 JavaScript。如果你想学习更多关于事件属性，请访问 [JavaScript 教程](#)

下面的表格提供了标准的事件属性，可以把它们插入 HTML/XHTML 元素中，以定义事件行为。

New : HTML5 新增属性事件。

窗口事件属性（Window Event Attributes）

由窗口触发该事件 (适用于 <body> 标签):

属性	值	描述
onafterprintNew	<i>script</i>	在打印文档之后运行脚本

onbeforeprintNew	script	在文档打印之前运行脚本
onbeforeunloadNew	script	在文档加载之前运行脚本
onblur	script	当窗口失去焦点时运行脚本
onerrorNew	script	当错误发生时运行脚本
onfocus	script	当窗口获得焦点时运行脚本
onhashchangeNew	script	当文档改变时运行脚本
onload	script	当文档加载时运行脚本
onmessageNew	script	当触发消息时运行脚本
onofflineNew	script	当文档离线时运行脚本
ononlineNew	script	当文档上线时运行脚本
onpagehideNew	script	当窗口隐藏时运行脚本

onpageshowNew	script	当窗口可见时运行脚本
onpopstateNew	script	当窗口历史记录改变时运行脚本
onredoNew	script	当文档执行再执行操作（redo）时运行脚本
onresizeNew	script	当调整窗口大小时运行脚本
onstorageNew	script	当 Web Storage 区域更新时（存储空间中的数据发生变化时）运行脚本
onundoNew	script	当文档执行撤销时运行脚本
onunloadNew	script	当用户离开文档时运行脚本

表单事件(Form Events)

表单事件在 HTML 表单中触发 (适用于所有 HTML 元素, 但该 HTML 元素需在 form 表单内):

属性	值	描述
onblur	script	当元素失去焦点时运行脚本

onchange	script	当元素改变时运行脚本
oncontextmenuNew	script	当触发上下文菜单时运行脚本
onfocus	script	当元素获得焦点时运行脚本
onformchangeNew	script	当表单改变时运行脚本
onforminputNew	script	当表单获得用户输入时运行脚本
oninputNew	script	当元素获得用户输入时运行脚本
oninvalidNew	script	当元素无效时运行脚本
onreset	script	当表单重置时运行脚本。 HTML 5 不支持。
onselect	script	当选取元素时运行脚本
onsubmit	script	当提交表单时运行脚本

键盘事件（Keyboard Events）

属性	值	描述
----	---	----

onkeydown	script	当按下按键时运行脚本
onkeypress	script	当按下并松开按键时运行脚本
onkeyup	script	当松开按键时运行脚本

鼠标事件（Mouse Events）

通过鼠标触发事件, 类似用户的行为:

属性	值	描述
onclick	script	当单击鼠标时运行脚本
ondblclick	script	当双击鼠标时运行脚本
ondragNew	script	当拖动元素时运行脚本
ondragendNew	script	当拖动操作结束时运行脚本
ondragenterNew	script	当元素被拖动至有效的拖放目标时运行脚本
ondragleaveNew	script	当元素离开有效拖放目标时运行脚本

ondragoverNew	<i>script</i>	当元素被拖动至有效拖放目标上方时运行脚本
ondragstartNew	<i>script</i>	当拖动操作开始时运行脚本
ondropNew	<i>script</i>	当被拖动元素正在被拖放时运行脚本
onmousedown	<i>script</i>	当按下鼠标按钮时运行脚本
onmousemove	<i>script</i>	当鼠标指针移动时运行脚本
onmouseout	<i>script</i>	当鼠标指针移出元素时运行脚本
onmouseover	<i>script</i>	当鼠标指针移至元素之上时运行脚本
onmouseup	<i>script</i>	当松开鼠标按钮时运行脚本
onmousewheelNew	<i>script</i>	当转动鼠标滚轮时运行脚本
onscrollNew	<i>script</i>	当滚动元素的滚动条时运行脚本

多媒体事件(Media Events)

通过视频（videos），图像（images）或者音频（audio）触发该事件，多应用于 HTML 媒体元素比如 <audio>, <embed>, , <object>, 和 <video>:

属性	值	描述
onabort	<i>script</i>	当发生中止事件时运行脚本
oncanplayNew	<i>script</i>	当媒介能够开始播放但可能因缓冲而需要停止时运行脚本
oncanplaythroughNew	<i>script</i>	当媒介能够无需因缓冲而停止即可播放至结尾时运行脚本
ondurationchangeNew	<i>script</i>	当媒介长度改变时运行脚本
onemptiedNew	<i>script</i>	当媒介资源元素突然为空时（网络错误、加载错误等）运行脚本
onendedNew	<i>script</i>	当媒介已抵达结尾时运行脚本
onerrorNew	<i>script</i>	当在元素加载期间发生错误时运行脚本
onloadeddataNew	<i>script</i>	当加载媒介数据时运行脚本
onloadedmetadataNew	<i>script</i>	当媒介元素的持续时间以及其他媒介数据已加载时运行脚本

onloadstartNew	<i>script</i>	当浏览器开始加载媒介数据时运行脚本
onpauseNew	<i>script</i>	当媒介数据暂停时运行脚本
onplayNew	<i>script</i>	当媒介数据将要开始播放时运行脚本
onplayingNew	<i>script</i>	当媒介数据已开始播放时运行脚本
onprogressNew	<i>script</i>	当浏览器正在取媒介数据时运行脚本
onratechangeNew	<i>script</i>	当媒介数据的播放速率改变时运行脚本
onreadystatechangeNew	<i>script</i>	当就绪状态（ready-state）改变时运行脚本
onseekedNew	<i>script</i>	当媒介元素的定位属性 [1] 不再为真且定位已结束运行时运行脚本
onseekingNew	<i>script</i>	当媒介元素的定位属性为真且定位已开始运行时运行脚本
onstalledNew	<i>script</i>	当取回媒介数据过程中（延迟）存在错误时运行脚本

onsuspendNew	script	当浏览器已在取媒介数据但在取回整个媒介文件之前停止时运行脚本
ontimeupdateNew	script	当媒介改变其播放位置时运行脚本
onvolumechangeNew	script	当媒介改变音量亦或当音量被设置为静音时运行脚本
onwaitingNew	script	当媒介已停止播放但打算继续播放时运行脚本

其他事件

属性	值	描述
onshowNew	script	当 <menu> 元素在上下文显示时触发
ontoggleNew	script	当用户打开或关闭 <details> 元素时触发

[HTML 全局属性](#)

[HTML 画布](#)

点我分享笔记

HTML5 <canvas> 参考手册

描述

HTML5 <canvas> 标签用于绘制图像（通过脚本，通常是 JavaScript）。

不过，<canvas> 元素本身并没有绘制能力（它仅仅是图形的容器） - 您必须使用脚本来完成实际的绘图任务。

getContext() 方法可返回一个对象，该对象提供了用于在画布上绘图的方法和属性。

本手册提供完整的 getContext("2d") 对象的属性和方法，可用于在画布上绘制文本、线条、矩形、圆形等等。

浏览器支持



Internet Explorer 9、Firefox、Opera、Chrome 和 Safari 支持 <canvas> 标签的属性及方法。

注意:Internet Explorer 8 及更早的 IE 版本不支持 <canvas> 元素。

颜色、样式和阴影

属性	描述
fillStyle	设置或返回用于填充绘画的颜色、渐变或模式。
strokeStyle	设置或返回用于笔触的颜色、渐变或模式。
shadowColor	设置或返回用于阴影的颜色。
shadowBlur	设置或返回用于阴影的模糊级别。
shadowOffsetX	设置或返回阴影与形状的水平距离。

shadowOffsetY	设置或返回阴影与形状的垂直距离。
---------------	------------------

方法	描述
createLinearGradient()	创建线性渐变（用在画布内容上）。
createPattern()	在指定的方向上重复指定的元素。
createRadialGradient()	创建放射状/环形的渐变（用在画布内容上）。
addColorStop()	规定渐变对象中的颜色和停止位置。

线条样式

属性	描述
lineCap	设置或返回线条的结束端点样式。
lineJoin	设置或返回两条线相交时，所创建的拐角类型。
lineWidth	设置或返回当前的线条宽度。
miterLimit	设置或返回最大斜接长度。

矩形

方法	描述
rect()	创建矩形。

<code>fillRect()</code>	绘制"被填充"的矩形。
<code>strokeRect()</code>	绘制矩形（无填充）。
<code>clearRect()</code>	在给定的矩形内清除指定的像素。

路径

方法	描述
<code>fill()</code>	填充当前绘图（路径）。
<code>stroke()</code>	绘制已定义的路径。
<code>beginPath()</code>	起始一条路径，或重置当前路径。
<code>moveTo()</code>	把路径移动到画布中的指定点，不创建线条。
<code>closePath()</code>	创建从当前点回到起始点的路径。
<code>lineTo()</code>	添加一个新点，然后在画布中创建从该点到最后指定点的线条。
<code>clip()</code>	从原始画布剪切任意形状和尺寸的区域。
<code>quadraticCurveTo()</code>	创建二次贝塞尔曲线。
<code>bezierCurveTo()</code>	创建三次贝塞尔曲线。
<code>arc()</code>	创建弧/曲线（用于创建圆形或部分圆）。
<code>arcTo()</code>	创建两切线之间的弧/曲线。

isPointInPath()	如果指定的点位于当前路径中，则返回 true，否则返回 false。
-----------------	------------------------------------

转换

方法	描述
scale()	缩放当前绘图至更大或更小。
rotate()	旋转当前绘图。
translate()	重新映射画布上的 (0,0) 位置。
transform()	替换绘图的当前转换矩阵。
setTransform()	将当前转换重置为单位矩阵。然后运行 transform()。

文本

属性	描述
font	设置或返回文本内容的当前字体属性。
textAlign	设置或返回文本内容的当前对齐方式。
textBaseline	设置或返回在绘制文本时使用的当前文本基线。

方法	描述
fillText()	在画布上绘制"被填充的"文本。

<code>strokeText()</code>	在画布上绘制文本（无填充）。
<code>measureText()</code>	返回包含指定文本宽度的对象。

图像绘制

方法	描述
<code>drawImage()</code>	向画布上绘制图像、画布或视频。

像素操作

属性	描述
<code>width</code>	返回 ImageData 对象的宽度。
<code>height</code>	返回 ImageData 对象的高度。
<code>data</code>	返回一个对象，其包含指定的 ImageData 对象的图像数据。

方法	描述
<code>createImageData()</code>	创建新的、空白的 ImageData 对象。
<code>getImageData()</code>	返回 ImageData 对象，该对象为画布上指定的矩形复制像素数据。
<code>putImageData()</code>	把图像数据（从指定的 ImageData 对象）放回画布上。

合成

属性	描述
globalAlpha	设置或返回绘图的当前 alpha 或透明值。
globalCompositeOperation	设置或返回新图像如何绘制到已有的图像上。

其他

方法	描述
save()	保存当前环境的状态。
restore()	返回之前保存过的路径状态和属性。
createEvent()	
getContext()	
toDataURL()	

[HTML 事件](#)

[HTML 音频/视频](#)

[点我分享笔记](#)

HTML 音频/视频 DOM 参考手册

HTML 音频/视频 DOM 参考手册

HTML5 DOM 为 <audio> 和 <video> 元素提供了方法、属性和事件。
这些方法、属性和事件允许您使用 JavaScript 来操作 <audio> 和 <video> 元素。

HTML 音频/视频 方法

方法	描述
<code>addTextTrack()</code>	向音频/视频添加新的文本轨道。
<code>canPlayType()</code>	检测浏览器是否能播放指定的音频/视频类型。
<code>load()</code>	重新加载音频/视频元素。
<code>play()</code>	开始播放音频/视频。
<code>pause()</code>	暂停当前播放的音频/视频。

HTML 音频/视频属性

属性	描述
<code>audioTracks</code>	返回表示可用音频轨道的 AudioTrackList 对象。
<code>autoplay</code>	设置或返回是否在加载完成后随即播放音频/视频。
<code>buffered</code>	返回表示音频/视频已缓冲部分的 TimeRanges 对象。
<code>controller</code>	返回表示音频/视频当前媒体控制器的 MediaController 对象。

controls	设置或返回音频/视频是否显示控件（比如播放/暂停等）。
crossOrigin	设置或返回音频/视频的 CORS 设置。
currentSrc	返回当前音频/视频的 URL。
currentTime	设置或返回音频/视频中的当前播放位置（以秒计）。
defaultMuted	设置或返回音频/视频默认是否静音。
defaultPlaybackRate	设置或返回音频/视频的默认播放速度。
duration	返回当前音频/视频的长度（以秒计）。
ended	返回音频/视频的播放是否已结束。
error	返回表示音频/视频错误状态的 MediaError 对象。
loop	设置或返回音频/视频是否应在结束时重新播放。
mediaGroup	设置或返回音频/视频所属的组合（用于连接多个音频/视频元素）。
muted	设置或返回音频/视频是否静音。
networkState	返回音频/视频的当前网络状态。
paused	设置或返回音频/视频是否暂停。
playbackRate	设置或返回音频/视频播放的速度。
played	返回表示音频/视频已播放部分的 TimeRanges 对象。

preload	设置或返回音频/视频是否应该在页面加载后进行加载。
readyState	返回音频/视频当前的就绪状态。
seekable	返回表示音频/视频可寻址部分的 TimeRanges 对象。
seeking	返回用户是否正在音频/视频中进行查找。
src	设置或返回音频/视频元素的当前来源。
startDate	返回表示当前时间偏移的 Date 对象。
textTracks	返回表示可用文本轨道的 TextTrackList 对象。
videoTracks	返回表示可用视频轨道的 VideoTrackList 对象。
volume	设置或返回音频/视频的音量。

HTML 音频/视频事件

事件	描述
abort	当音频/视频的加载已放弃时触发。
canplay	当浏览器可以开始播放音频/视频时触发。
canplaythrough	当浏览器可在不因缓冲而停顿的情况下进行播放时触发。
durationchange	当音频/视频的时长已更改时触发。
emptied	当目前的播放列表为空时触发。

ended	当目前的播放列表已结束时触发。
error	当在音频/视频加载期间发生错误时触发。
loadeddata	当浏览器已加载音频/视频的当前帧时触发。
loadedmetadata	当浏览器已加载音频/视频的元数据时触发。
loadstart	当浏览器开始查找音频/视频时触发。
pause	当音频/视频已暂停时触发。
play	当音频/视频已开始或不再暂停时触发。
playing	当音频/视频在因缓冲而暂停或停止后已就绪时触发。
progress	当浏览器正在下载音频/视频时触发。
ratechange	当音频/视频的播放速度已更改时触发。
seeked	当用户已移动/跳跃到音频/视频中的新位置时触发。
seeking	当用户开始移动/跳跃到音频/视频中的新位置时触发。
stalled	当浏览器尝试获取媒体数据，但数据不可用时触发。
suspend	当浏览器刻意不获取媒体数据时触发。
timeupdate	当目前的播放位置已更改时触发。
volumechange	当音量已更改时触发。

waiting	当视频由于需要缓冲下一帧而停止时触发。
---------	---------------------

[HTML 画布](#)

[HTML 有效 DOCTYPEs](#)

[点我分享笔记](#)

HTML 元素和有效 DOCTYPEs

HTML 元素 - 有效 DOCTYPEs

下面的表格列出了所有的 HTML5/HTML 4.01/XHTML 元素，以及它们会出现在什么文档类型 (!DOCTYPE) 中：

		HTML 4.01 / XHTML 1.0			
Tag	HTML5	Transitional	Strict	Frameset	XHTML 1.1
<u><a></u>	Yes	Yes	Yes	Yes	Yes
<u><abbr></u>	Yes	Yes	Yes	Yes	Yes
<u><acronym></u>	No	Yes	Yes	Yes	Yes
<u><address></u>	Yes	Yes	Yes	Yes	Yes
<u><applet></u>	No	Yes	No	Yes	No
<u><area></u>	Yes	Yes	Yes	Yes	No

<u><article></u>	Yes	No	No	No	No
<u><aside></u>	Yes	No	No	No	No
<u><audio></u>	Yes	No	No	No	No
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><base></u>	Yes	Yes	Yes	Yes	Yes
<u><basefont></u>	No	Yes	No	Yes	No
<u><bdi></u>	Yes	No	No	No	No
<u><bdo></u>	Yes	Yes	Yes	Yes	No
<u><big></u>	No	Yes	Yes	Yes	Yes
<u><blockquote></u>	Yes	Yes	Yes	Yes	Yes
<u><body></u>	Yes	Yes	Yes	Yes	Yes
<u>
</u>	Yes	Yes	Yes	Yes	Yes
<u><button></u>	Yes	Yes	Yes	Yes	Yes
<u><canvas></u>	Yes	No	No	No	No
<u><caption></u>	Yes	Yes	Yes	Yes	Yes
<u><center></u>	No	Yes	No	Yes	No

<u><cite></u>	Yes	Yes	Yes	Yes	Yes
<u><code></u>	Yes	Yes	Yes	Yes	Yes
<u><col></u>	Yes	Yes	Yes	Yes	No
<u><colgroup></u>	Yes	Yes	Yes	Yes	No
<u><command></u>	Yes	No	No	No	No
<u><datalist></u>	Yes	No	No	No	No
<u><dd></u>	Yes	Yes	Yes	Yes	Yes
<u></u>	Yes	Yes	Yes	Yes	No
<u><details></u>	Yes	No	No	No	No
<u><dfn></u>	Yes	Yes	Yes	Yes	Yes
<u><dir></u>	No	Yes	No	Yes	No
<u><div></u>	Yes	Yes	Yes	Yes	Yes
<u><dl></u>	Yes	Yes	Yes	Yes	Yes
<u><dt></u>	Yes	Yes	Yes	Yes	Yes
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><embed></u>	Yes	No	No	No	No

<u><fieldset></u>	Yes	Yes	Yes	Yes	Yes
<u><figcaption></u>	Yes	No	No	No	No
<u><figure></u>	Yes	No	No	No	No
<u></u>	No	Yes	No	Yes	No
<u><footer></u>	Yes	No	No	No	No
<u><form></u>	Yes	Yes	Yes	Yes	Yes
<u><frame></u>	No	No	No	Yes	No
<u><frameset></u>	No	No	No	Yes	No
<u><h1> to <h6></u>	Yes	Yes	Yes	Yes	Yes
<u><head></u>	Yes	Yes	Yes	Yes	Yes
<u><header></u>	Yes	No	No	No	No
<u><hgroup></u>	Yes	No	No	No	No
<u><hr></u>	Yes	Yes	Yes	Yes	Yes
<u><html></u>	Yes	Yes	Yes	Yes	Yes
<u><i></u>	Yes	Yes	Yes	Yes	Yes
<u><iframe></u>	Yes	Yes	No	Yes	No

<u></u>	Yes	Yes	Yes	Yes	Yes
<u><input></u>	Yes	Yes	Yes	Yes	Yes
<u><ins></u>	Yes	Yes	Yes	Yes	No
<u><kbd></u>	Yes	Yes	Yes	Yes	Yes
<u><keygen></u>	Yes	No	No	No	No
<u><label></u>	Yes	Yes	Yes	Yes	Yes
<u><legend></u>	Yes	Yes	Yes	Yes	Yes
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><link></u>	Yes	Yes	Yes	Yes	Yes
<u><map></u>	Yes	Yes	Yes	Yes	No
<u><mark></u>	Yes	No	No	No	No
<u><menu></u>	Yes	Yes	No	Yes	No
<u><meta></u>	Yes	Yes	Yes	Yes	Yes
<u><meter></u>	Yes	No	No	No	No
<u><nav></u>	Yes	No	No	No	No
<u><noframes></u>	No	Yes	No	Yes	No

<u><noscript></u>	Yes	Yes	Yes	Yes	Yes
<u><object></u>	Yes	Yes	Yes	Yes	Yes
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><optgroup></u>	Yes	Yes	Yes	Yes	Yes
<u><option></u>	Yes	Yes	Yes	Yes	Yes
<u><output></u>	Yes	No	No	No	No
<u><p></u>	Yes	Yes	Yes	Yes	Yes
<u><param></u>	Yes	Yes	Yes	Yes	Yes
<u><pre></u>	Yes	Yes	Yes	Yes	Yes
<u><progress></u>	Yes	No	No	No	No
<u><q></u>	Yes	Yes	Yes	Yes	Yes
<u><rp></u>	Yes	No	No	No	No
<u><rt></u>	Yes	No	No	No	No
<u><ruby></u>	Yes	No	No	No	No
<u><s></u>	Yes	Yes	No	Yes	No
<u><samp></u>	Yes	Yes	Yes	Yes	Yes

<u><script></u>	Yes	Yes	Yes	Yes	Yes
<u><section></u>	Yes	No	No	No	No
<u><select></u>	Yes	Yes	Yes	Yes	Yes
<u><small></u>	Yes	Yes	Yes	Yes	Yes
<u><source></u>	Yes	No	No	No	No
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><strike></u>	No	Yes	No	Yes	No
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><style></u>	Yes	Yes	Yes	Yes	Yes
<u><sub></u>	Yes	Yes	Yes	Yes	Yes
<u><summary></u>	Yes	No	No	No	No
<u><sup></u>	Yes	Yes	Yes	Yes	Yes
<u><table></u>	Yes	Yes	Yes	Yes	Yes
<u><tbody></u>	Yes	Yes	Yes	Yes	No
<u><td></u>	Yes	Yes	Yes	Yes	Yes
<u><textarea></u>	Yes	Yes	Yes	Yes	Yes

<u><tfoot></u>	Yes	Yes	Yes	Yes	No
<u><th></u>	Yes	Yes	Yes	Yes	Yes
<u><thead></u>	Yes	Yes	Yes	Yes	No
<u><time></u>	Yes	No	No	No	No
<u><title></u>	Yes	Yes	Yes	Yes	Yes
<u><tr></u>	Yes	Yes	Yes	Yes	Yes
<u><track></u>	Yes	No	No	No	No
<u><tt></u>	No	Yes	Yes	Yes	Yes
<u><u></u>	No	Yes	No	Yes	No
<u></u>	Yes	Yes	Yes	Yes	Yes
<u><var></u>	Yes	Yes	Yes	Yes	Yes
<u><video></u>	Yes	No	No	No	No
<u><wbr></u>	Yes	No	No	No	No

[HTML 音频/视频](#)

[HTML 颜色名](#)

HTML 颜色名

目前所有浏览器都支持以下颜色名。

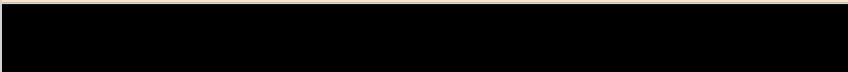
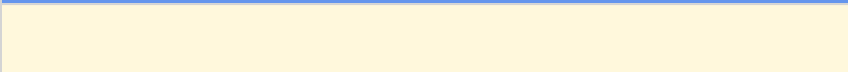
141 个颜色名称是在 HTML 和 CSS 颜色规范定义的（17 标准颜色，再加 124）。下表列出了所有颜色的值，包括十六进制值。
💡 提示: 17 标准颜色：黑色，蓝色，水，紫红色，灰色，绿色，石灰，栗色，海军，橄榄，橙，紫，红，白，银，蓝绿色，黄色。点击其中一个颜色名称（或一个十六进制值）就可以查看与不同文字颜色搭配的背景颜色。：



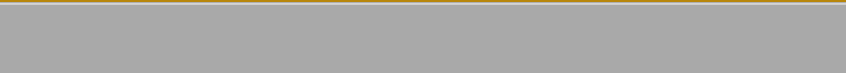
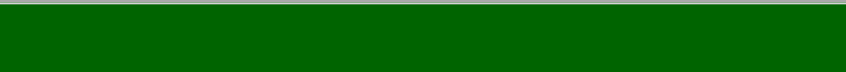

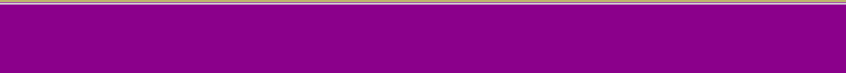
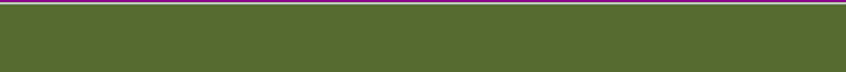






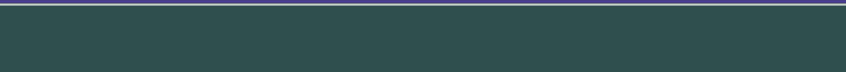


按颜色名排序

[按十六进制的值排序](#)

单击一个颜色名或者 16 进制值，就可以查看与不同文字颜色搭配的背景颜色。





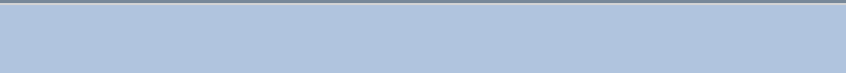



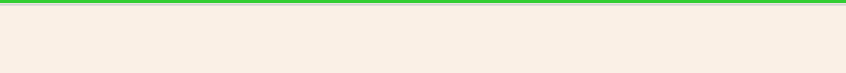


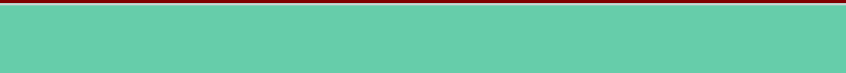
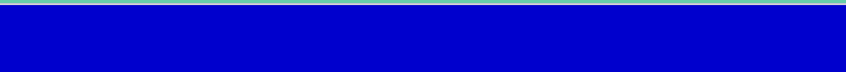



Color Name	HEX	Color
AliceBlue	#F0F8FF	
AntiqueWhite	#FAEBD7	
Aqua	#00FFFF	
Aquamarine	#7FFFD4	
Azure	#F0FFFF	
Beige	#F5F5DC	

<u>Bisque</u>	<u>#FFE4C4</u>	
<u>Black</u>	<u>#000000</u>	
<u>BlanchedAlmond</u>	<u>#FFEBCD</u>	
<u>Blue</u>	<u>#0000FF</u>	
<u>BlueViolet</u>	<u>#8A2BE2</u>	
<u>Brown</u>	<u>#A52A2A</u>	
<u>BurlyWood</u>	<u>#DEB887</u>	
<u>CadetBlue</u>	<u>#5F9EA0</u>	
<u>Chartreuse</u>	<u>#7FFF00</u>	
<u>Chocolate</u>	<u>#D2691E</u>	
<u>Coral</u>	<u>#FF7F50</u>	
<u>CornflowerBlue</u>	<u>#6495ED</u>	
<u>Cornsilk</u>	<u>#FFF8DC</u>	
<u>Crimson</u>	<u>#DC143C</u>	
<u>Cyan</u>	<u>#00FFFF</u>	
<u>DarkBlue</u>	<u>#00008B</u>	

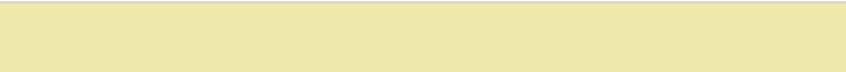
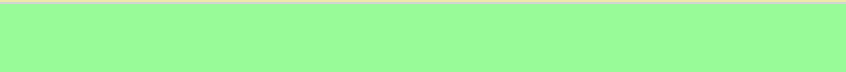
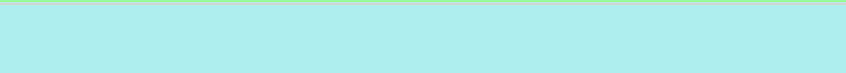

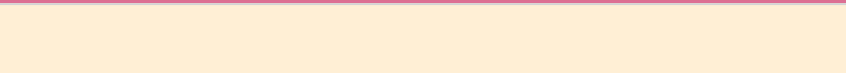








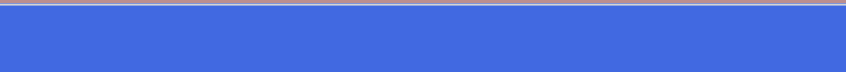
<u>DarkCyan</u>	<u>#008B8B</u>	
<u>DarkGoldenRod</u>	<u>#B8860B</u>	
<u>DarkGray</u>	<u>#A9A9A9</u>	
<u>DarkGreen</u>	<u>#006400</u>	
<u>DarkKhaki</u>	<u>#BDB76B</u>	
<u>DarkMagenta</u>	<u>#8B008B</u>	
<u>DarkOliveGreen</u>	<u>#556B2F</u>	
<u>DarkOrange</u>	<u>#FF8C00</u>	
<u>DarkOrchid</u>	<u>#9932CC</u>	
<u>DarkRed</u>	<u>#8B0000</u>	
<u>DarkSalmon</u>	<u>#E9967A</u>	
<u>DarkSeaGreen</u>	<u>#8FBC8F</u>	
<u>DarkSlateBlue</u>	<u>#483D8B</u>	
<u>DarkSlateGray</u>	<u>#2F4F4F</u>	
<u>DarkTurquoise</u>	<u>#00CED1</u>	
<u>DarkViolet</u>	<u>#9400D3</u>	



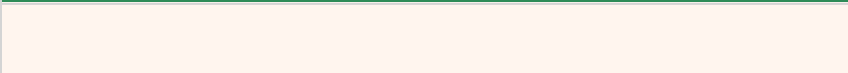

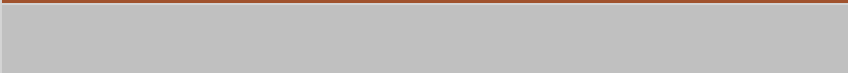



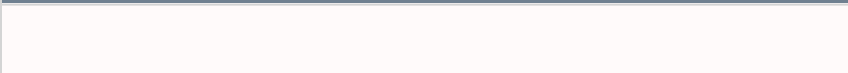



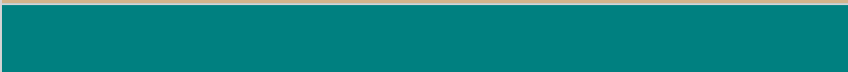



<u>DeepPink</u>	<u>#FF1493</u>	
<u>DeepSkyBlue</u>	<u>#00BFFF</u>	
<u>DimGray</u>	<u>#696969</u>	
<u>DodgerBlue</u>	<u>#1E90FF</u>	
<u>FireBrick</u>	<u>#B22222</u>	
<u>FloralWhite</u>	<u>#FFFAF0</u>	
<u>ForestGreen</u>	<u>#228B22</u>	
<u>Fuchsia</u>	<u>#FF00FF</u>	
<u>Gainsboro</u>	<u>#DCDCDC</u>	
<u>GhostWhite</u>	<u>#F8F8FF</u>	
<u>Gold</u>	<u>#FFD700</u>	
<u>GoldenRod</u>	<u>#DAA520</u>	
<u>Gray</u>	<u>#808080</u>	
<u>Green</u>	<u>#008000</u>	
<u>GreenYellow</u>	<u>#ADFF2F</u>	
<u>HoneyDew</u>	<u>#F0FFF0</u>	

<u>HotPink</u>	<u>#FF69B4</u>	
<u>IndianRed</u>	<u>#CD5C5C</u>	
<u>Indigo</u>	<u>#4B0082</u>	
<u>Ivory</u>	<u>#FFFFFF0</u>	
<u>Khaki</u>	<u>#F0E68C</u>	
<u>Lavender</u>	<u>#E6E6FA</u>	
<u>LavenderBlush</u>	<u>#FFF0F5</u>	
<u>LawnGreen</u>	<u>#7CFC00</u>	
<u>LemonChiffon</u>	<u>#FFFACD</u>	
<u>LightBlue</u>	<u>#ADD8E6</u>	
<u>LightCoral</u>	<u>#F08080</u>	
<u>LightCyan</u>	<u>#E0FFFF</u>	
<u>LightGoldenRodYellow</u>	<u>#FAFAD2</u>	
<u>LightGray</u>	<u>#D3D3D3</u>	
<u>LightGreen</u>	<u>#90EE90</u>	
<u>LightPink</u>	<u>#FFB6C1</u>	

<u>LightSalmon</u>	<u>#FFA07A</u>	
<u>LightSeaGreen</u>	<u>#20B2AA</u>	
<u>LightSkyBlue</u>	<u>#87CEFA</u>	
<u>LightSlateGray</u>	<u>#778899</u>	
<u>LightSteelBlue</u>	<u>#B0C4DE</u>	
<u>LightYellow</u>	<u>#FFFFE0</u>	
<u>Lime</u>	<u>#00FF00</u>	
<u>LimeGreen</u>	<u>#32CD32</u>	
<u>Linen</u>	<u>#FAF0E6</u>	
<u>Magenta</u>	<u>#FF00FF</u>	
<u>Maroon</u>	<u>#800000</u>	
<u>MediumAquaMarine</u>	<u>#66CDAA</u>	
<u>MediumBlue</u>	<u>#0000CD</u>	
<u>MediumOrchid</u>	<u>#BA55D3</u>	
<u>MediumPurple</u>	<u>#9370DB</u>	
<u>MediumSeaGreen</u>	<u>#3CB371</u>	

<u>MediumSlateBlue</u>	<u>#7B68EE</u>	
<u>MediumSpringGreen</u>	<u>#00FA9A</u>	
<u>MediumTurquoise</u>	<u>#48D1CC</u>	
<u>MediumVioletRed</u>	<u>#C71585</u>	
<u>MidnightBlue</u>	<u>#191970</u>	
<u>MintCream</u>	<u>#F5FFFA</u>	
<u>MistyRose</u>	<u>#FFE4E1</u>	
<u>Moccasin</u>	<u>#FFE4B5</u>	
<u>NavajoWhite</u>	<u>#FFDEAD</u>	
<u>Navy</u>	<u>#000080</u>	
<u>OldLace</u>	<u>#FDF5E6</u>	
<u>Olive</u>	<u>#808000</u>	
<u>OliveDrab</u>	<u>#6B8E23</u>	
<u>Orange</u>	<u>#FFA500</u>	
<u>OrangeRed</u>	<u>#FF4500</u>	
<u>Orchid</u>	<u>#DA70D6</u>	

<u>PaleGoldenRod</u>	<u>#EEE8AA</u>	
<u>PaleGreen</u>	<u>#98FB98</u>	
<u>PaleTurquoise</u>	<u>#AFEEEE</u>	
<u>PaleVioletRed</u>	<u>#DB7093</u>	
<u>PapayaWhip</u>	<u>#FFefd5</u>	
<u>PeachPuff</u>	<u>#FFDAB9</u>	
<u>Peru</u>	<u>#CD853F</u>	
<u>Pink</u>	<u>#FFC0CB</u>	
<u>Plum</u>	<u>#DDA0DD</u>	
<u>PowderBlue</u>	<u>#B0E0E6</u>	
<u>Purple</u>	<u>#800080</u>	
<u>Red</u>	<u>#FF0000</u>	
<u>RosyBrown</u>	<u>#BC8F8F</u>	
<u>RoyalBlue</u>	<u>#4169E1</u>	
<u>SaddleBrown</u>	<u>#8B4513</u>	
<u>Salmon</u>	<u>#FA8072</u>	

<u>SandyBrown</u>	<u>#F4A460</u>	
<u>SeaGreen</u>	<u>#2E8B57</u>	
<u>SeaShell</u>	<u>#FFF5EE</u>	
<u>Sienna</u>	<u>#A0522D</u>	
<u>Silver</u>	<u>#C0C0C0</u>	
<u>SkyBlue</u>	<u>#87CEEB</u>	
<u>SlateBlue</u>	<u>#6A5ACD</u>	
<u>SlateGray</u>	<u>#708090</u>	
<u>Snow</u>	<u>#FFFAFA</u>	
<u>SpringGreen</u>	<u>#00FF7F</u>	
<u>SteelBlue</u>	<u>#4682B4</u>	
<u>Tan</u>	<u>#D2B48C</u>	
<u>Teal</u>	<u>#008080</u>	
<u>Thistle</u>	<u>#D8BFD8</u>	
<u>Tomato</u>	<u>#FF6347</u>	
<u>Turquoise</u>	<u>#40E0D0</u>	

<u>Violet</u>	<u>#EE82EE</u>	
<u>Wheat</u>	<u>#F5DEB3</u>	
<u>White</u>	<u>#FFFFFF</u>	
<u>WhiteSmoke</u>	<u>#F5F5F5</u>	
<u>Yellow</u>	<u>#FFFF00</u>	
<u>YellowGreen</u>	<u>#9ACD32</u>	

[HTML 有效 DOCTYPEs](#)

[HTML 取色器/拾色器](#)

点我分享笔记

HTML 取色器/拾色器

选取颜色:	选择的颜色:	淡 / 暗:		
	<div>黑色文本</div> <div>阴影</div>	100%		#ffffff
		95%		#ffe5e5
		90%		#ffcccc



或输入颜色值:

OK

或使用 HTML5:

白色文本

阴影

red




#ff0000

rgb(255, 0, 0)

hsl(0, 100%, 50%)


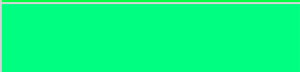


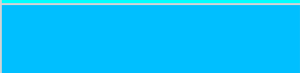
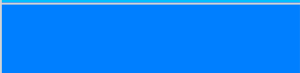










hsv(0, 100%, 100%)

85%		#ffb3b3
80%		#ff9999
75%		#ff8080
70%		#ff6666
65%		#ff4d4d
60%		#ff3333
55%		#ff1a1a
50%		#ff0000
45%		#e60000
40%		#cc0000
35%		#b30000
30%		#990000
25%		#800000
20%		#660000
15%		#4d0000

10%		#330000
5%		#1a0000
0%		#000000

Hue

	Hue	Hex	Rgb	Hsl	Hsv
	0	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)
	15	#ff4000	rgb(255, 64, 0)	hsl(15, 100%, 50%)	hsv(15, 100%, 100%)
	30	#ff8000	rgb(255, 128, 0)	hsl(30, 100%, 50%)	hsv(30, 100%, 100%)
	45	#ffbf00	rgb(255, 191, 0)	hsl(45, 100%, 50%)	hsv(45, 100%, 100%)
	60	#ffff00	rgb(255, 255, 0)	hsl(60, 100%, 50%)	hsv(60, 100%, 100%)
	75	#bfff00	rgb(191, 255, 0)	hsl(75, 100%, 50%)	hsv(75, 100%, 100%)
	90	#80ff00	rgb(128, 255, 0)	hsl(90, 100%, 50%)	hsv(90, 100%, 100%)
	105	#40ff00	rgb(64, 255, 0)	hsl(105, 100%, 50%)	hsv(105, 100%, 100%)
	120	#00ff00	rgb(0, 255, 0)	hsl(120, 100%, 50%)	hsv(120, 100%, 100%)

	135	#00ff40	rgb(0, 255, 64)	hsl(135, 100%, 50%)	hsv(135, 100%, 100%)
	150	#00ff80	rgb(0, 255, 128)	hsl(150, 100%, 50%)	hsv(150, 100%, 100%)
	165	#00ffbf	rgb(0, 255, 191)	hsl(165, 100%, 50%)	hsv(165, 100%, 100%)
	180	#00ffff	rgb(0, 255, 255)	hsl(180, 100%, 50%)	hsv(180, 100%, 100%)
	195	#00bfff	rgb(0, 191, 255)	hsl(195, 100%, 50%)	hsv(195, 100%, 100%)
	210	#007fff	rgb(0, 127, 255)	hsl(210, 100%, 50%)	hsv(210, 100%, 100%)
	225	#0040ff	rgb(0, 64, 255)	hsl(225, 100%, 50%)	hsv(225, 100%, 100%)
	240	#0000ff	rgb(0, 0, 255)	hsl(240, 100%, 50%)	hsv(240, 100%, 100%)
	255	#4000ff	rgb(64, 0, 255)	hsl(255, 100%, 50%)	hsv(255, 100%, 100%)
	270	#7f00ff	rgb(127, 0, 255)	hsl(270, 100%, 50%)	hsv(270, 100%, 100%)
	285	#bf00ff	rgb(191, 0, 255)	hsl(285, 100%, 50%)	hsv(285, 100%, 100%)
	300	#ff00ff	rgb(255, 0, 255)	hsl(300, 100%, 50%)	hsv(300, 100%, 100%)
	315	#ff00bf	rgb(255, 0, 191)	hsl(315, 100%, 50%)	hsv(315, 100%, 100%)
	330	#ff0080	rgb(255, 0, 128)	hsl(330, 100%, 50%)	hsv(330, 100%, 100%)
	345	#ff0040	rgb(255, 0, 64)	hsl(345, 100%, 50%)	hsv(345, 100%, 100%)
	360	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)

HSL Saturation

	Sat	Hex	Rgb	Hsl	Hsv
	100%	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)
	95%	#f90606	rgb(249, 6, 6)	hsl(0, 95%, 50%)	hsv(0, 97%, 98%)
	90%	#f20d0d	rgb(242, 13, 13)	hsl(0, 90%, 50%)	hsv(0, 95%, 95%)
	85%	#ec1313	rgb(236, 19, 19)	hsl(0, 85%, 50%)	hsv(0, 92%, 93%)
	80%	#e61919	rgb(230, 25, 25)	hsl(0, 80%, 50%)	hsv(0, 89%, 90%)
	75%	#df2020	rgb(223, 32, 32)	hsl(0, 75%, 50%)	hsv(0, 86%, 88%)
	70%	#d92626	rgb(217, 38, 38)	hsl(0, 70%, 50%)	hsv(0, 82%, 85%)
	65%	#d22d2d	rgb(210, 45, 45)	hsl(0, 65%, 50%)	hsv(0, 79%, 83%)
	60%	#cc3333	rgb(204, 51, 51)	hsl(0, 60%, 50%)	hsv(0, 75%, 80%)
	55%	#c63939	rgb(198, 57, 57)	hsl(0, 55%, 50%)	hsv(0, 71%, 78%)
	50%	#bf4040	rgb(191, 64, 64)	hsl(0, 50%, 50%)	hsv(0, 67%, 75%)
	45%	#b94646	rgb(185, 70, 70)	hsl(0, 45%, 50%)	hsv(0, 62%, 73%)
	40%	#b34d4d	rgb(179, 77, 77)	hsl(0, 40%, 50%)	hsv(0, 57%, 70%)

	35%	#ac5353	rgb(172, 83, 83)	hsl(0, 35%, 50%)	hsv(0, 52%, 68%)
	30%	#a65959	rgb(166, 89, 89)	hsl(0, 30%, 50%)	hsv(0, 46%, 65%)
	25%	#9f6060	rgb(159, 96, 96)	hsl(0, 25%, 50%)	hsv(0, 40%, 63%)
	20%	#996666	rgb(153, 102, 102)	hsl(0, 20%, 50%)	hsv(0, 33%, 60%)
	15%	#936c6c	rgb(147, 108, 108)	hsl(0, 15%, 50%)	hsv(0, 26%, 58%)
	10%	#8c7373	rgb(140, 115, 115)	hsl(0, 10%, 50%)	hsv(0, 18%, 55%)
	5%	#867979	rgb(134, 121, 121)	hsl(0, 5%, 50%)	hsv(0, 10%, 53%)
	0%	#808080	rgb(128, 128, 128)	hsl(0, 0%, 50%)	hsv(0, 0%, 50%)

HSL 淡 / 暗

	Lum	Hex	Rgb	Hsl	Hsv
	100%	#ffffff	rgb(255, 255, 255)	hsl(0, 0%, 100%)	hsv(0, 0%, 100%)
	95%	#ffe5e5	rgb(255, 229, 229)	hsl(0, 100%, 95%)	hsv(0, 10%, 100%)
	90%	#ffcccc	rgb(255, 204, 204)	hsl(0, 100%, 90%)	hsv(0, 20%, 100%)
	85%	#ffb3b3	rgb(255, 179, 179)	hsl(0, 100%, 85%)	hsv(0, 30%, 100%)
	80%	#ff9999	rgb(255, 153, 153)	hsl(0, 100%, 80%)	hsv(0, 40%, 100%)

	75%	#ff8080	rgb(255, 128, 128)	hsl(0, 100%, 75%)	hsv(0, 50%, 100%)
	70%	#ff6666	rgb(255, 102, 102)	hsl(0, 100%, 70%)	hsv(0, 60%, 100%)
	65%	#ff4d4d	rgb(255, 77, 77)	hsl(0, 100%, 65%)	hsv(0, 70%, 100%)
	60%	#ff3333	rgb(255, 51, 51)	hsl(0, 100%, 60%)	hsv(0, 80%, 100%)
	55%	#ff1a1a	rgb(255, 26, 26)	hsl(0, 100%, 55%)	hsv(0, 90%, 100%)
	50%	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)
	45%	#e60000	rgb(230, 0, 0)	hsl(0, 100%, 45%)	hsv(0, 100%, 90%)
	40%	#cc0000	rgb(204, 0, 0)	hsl(0, 100%, 40%)	hsv(0, 100%, 80%)
	35%	#b30000	rgb(179, 0, 0)	hsl(0, 100%, 35%)	hsv(0, 100%, 70%)
	30%	#990000	rgb(153, 0, 0)	hsl(0, 100%, 30%)	hsv(0, 100%, 60%)
	25%	#800000	rgb(128, 0, 0)	hsl(0, 100%, 25%)	hsv(0, 100%, 50%)
	20%	#660000	rgb(102, 0, 0)	hsl(0, 100%, 20%)	hsv(0, 100%, 40%)
	15%	#4d0000	rgb(77, 0, 0)	hsl(0, 100%, 15%)	hsv(0, 100%, 30%)
	10%	#330000	rgb(51, 0, 0)	hsl(0, 100%, 10%)	hsv(0, 100%, 20%)
	5%	#1a0000	rgb(26, 0, 0)	hsl(0, 100%, 5%)	hsv(0, 100%, 10%)
	0%	#000000	rgb(0, 0, 0)	hsl(0, 0%, 0%)	hsv(0, 0%, 0%)

HSV Saturation

	Sat	Hex	Rgb	Hsl	Hsv
	100%	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)
	95%	#ff0d0d	rgb(255, 13, 13)	hsl(0, 100%, 53%)	hsv(0, 95%, 100%)
	90%	#ff1919	rgb(255, 25, 25)	hsl(0, 100%, 55%)	hsv(0, 90%, 100%)
	85%	#ff2626	rgb(255, 38, 38)	hsl(0, 100%, 57%)	hsv(0, 85%, 100%)
	80%	#ff3333	rgb(255, 51, 51)	hsl(0, 100%, 60%)	hsv(0, 80%, 100%)
	75%	#ff4040	rgb(255, 64, 64)	hsl(0, 100%, 63%)	hsv(0, 75%, 100%)
	70%	#ff4d4d	rgb(255, 77, 77)	hsl(0, 100%, 65%)	hsv(0, 70%, 100%)
	65%	#ff5959	rgb(255, 89, 89)	hsl(0, 100%, 68%)	hsv(0, 65%, 100%)
	60%	#ff6666	rgb(255, 102, 102)	hsl(0, 100%, 70%)	hsv(0, 60%, 100%)
	55%	#ff7373	rgb(255, 115, 115)	hsl(0, 100%, 73%)	hsv(0, 55%, 100%)
	50%	#ff8080	rgb(255, 128, 128)	hsl(0, 100%, 75%)	hsv(0, 50%, 100%)
	45%	#ff8c8c	rgb(255, 140, 140)	hsl(0, 100%, 78%)	hsv(0, 45%, 100%)
	40%	#ff9999	rgb(255, 153, 153)	hsl(0, 100%, 80%)	hsv(0, 40%, 100%)

	35%	#ffa6a6	rgb(255, 166, 166)	hsl(0, 100%, 83%)	hsv(0, 35%, 100%)
	30%	#ffb3b3	rgb(255, 179, 179)	hsl(0, 100%, 85%)	hsv(0, 30%, 100%)
	25%	#ffbfbf	rgb(255, 191, 191)	hsl(0, 100%, 88%)	hsv(0, 25%, 100%)
	20%	#ffcccc	rgb(255, 204, 204)	hsl(0, 100%, 90%)	hsv(0, 20%, 100%)
	15%	#ffd9d9	rgb(255, 217, 217)	hsl(0, 100%, 93%)	hsv(0, 15%, 100%)
	10%	#ffe6e6	rgb(255, 230, 230)	hsl(0, 100%, 95%)	hsv(0, 10%, 100%)
	5%	#fff2f2	rgb(255, 242, 242)	hsl(0, 100%, 98%)	hsv(0, 5%, 100%)
	0%	#ffffff	rgb(255, 255, 255)	hsl(0, 0%, 100%)	hsv(0, 0%, 100%)

HSV 亮 / 暗

	Value	Hex	Rgb	Hsl	Hsv
	100%	#ff0000	rgb(255, 0, 0)	hsl(0, 100%, 50%)	hsv(0, 100%, 100%)
	95%	#f20000	rgb(242, 0, 0)	hsl(0, 100%, 48%)	hsv(0, 100%, 95%)
	90%	#e60000	rgb(230, 0, 0)	hsl(0, 100%, 45%)	hsv(0, 100%, 90%)
	85%	#d90000	rgb(217, 0, 0)	hsl(0, 100%, 43%)	hsv(0, 100%, 85%)
	80%	#cc0000	rgb(204, 0, 0)	hsl(0, 100%, 40%)	hsv(0, 100%, 80%)

	75%	#bf0000	rgb(191, 0, 0)	hsl(0, 100%, 38%)	hsv(0, 100%, 75%)
	70%	#b30000	rgb(179, 0, 0)	hsl(0, 100%, 35%)	hsv(0, 100%, 70%)
	65%	#a60000	rgb(166, 0, 0)	hsl(0, 100%, 33%)	hsv(0, 100%, 65%)
	60%	#990000	rgb(153, 0, 0)	hsl(0, 100%, 30%)	hsv(0, 100%, 60%)
	55%	#8c0000	rgb(140, 0, 0)	hsl(0, 100%, 28%)	hsv(0, 100%, 55%)
	50%	#800000	rgb(128, 0, 0)	hsl(0, 100%, 25%)	hsv(0, 100%, 50%)
	45%	#730000	rgb(115, 0, 0)	hsl(0, 100%, 23%)	hsv(0, 100%, 45%)
	40%	#660000	rgb(102, 0, 0)	hsl(0, 100%, 20%)	hsv(0, 100%, 40%)
	35%	#590000	rgb(89, 0, 0)	hsl(0, 100%, 18%)	hsv(0, 100%, 35%)
	30%	#4d0000	rgb(77, 0, 0)	hsl(0, 100%, 15%)	hsv(0, 100%, 30%)
	25%	#400000	rgb(64, 0, 0)	hsl(0, 100%, 13%)	hsv(0, 100%, 25%)
	20%	#330000	rgb(51, 0, 0)	hsl(0, 100%, 10%)	hsv(0, 100%, 20%)
	15%	#260000	rgb(38, 0, 0)	hsl(0, 100%, 8%)	hsv(0, 100%, 15%)
	10%	#1a0000	rgb(26, 0, 0)	hsl(0, 100%, 5%)	hsv(0, 100%, 10%)
	5%	#0d0000	rgb(13, 0, 0)	hsl(0, 100%, 3%)	hsv(0, 100%, 5%)
	0%	#000000	rgb(0, 0, 0)	hsl(0, 0%, 0%)	hsv(0, 0%, 0%)

RGB (Red, Green, Blue)

Red	Green	Blue
255	0	0
<div></div>	<div></div>	<div></div>

rgb(255, 0, 0) #ff0000

HTML 颜色名

HTML 颜色混搭

点我分享笔记

HTML 颜色混搭

混搭两种颜色，并查看效果：

选择颜色：

顶部颜色：

#FF0000

#FF9999



#FF0000

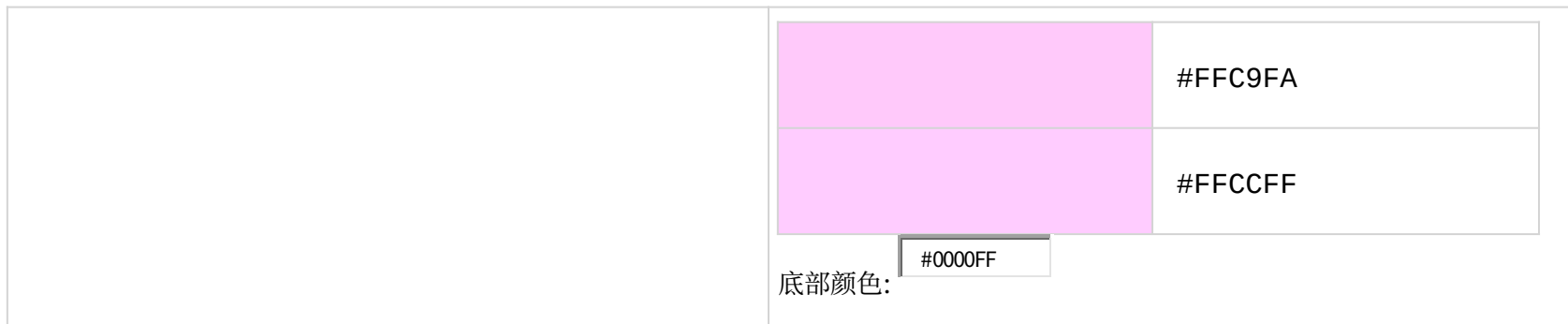


#0000FF



	#FF9C9E
	#FF9EA3
	#FFA1A8
	#FFA3AD
	#FFA6B2
	#FFA8B8
	#FFABBD
	#FFADC2
	#FFB0C7

		#FFB2CC
		#FFB5D1
		#FFB8D6
		#FFBADB
		#FFBDE0
		#FFBFE6
		#FFC2EB
		#FFC4F0
		#FFC7F5



[HTML 取色器/拾色器](#)

[HTML Emoji](#)

[点我分享笔记](#)

HTML Emoji

Emoji 是来自 UTF-8 字符集的字符。

表情符号（英语：emoji，日语：絵文字／えもじ emoji），是使用在网页和聊天中的形意符号，最初是日本在无线通信中所使用的视觉情感符号（图画文字）。表情意指面部表情，图标则是图形标志的意思，可用来代表多种表情，如笑脸表示笑、蛋糕表示食物等。Emoji 看起来像一张图片或图标，其实不是。

Emoji 实际上是 UTF-8 (Unicode) 字符集上的字符。

UTF-8 几乎涵盖了世界上所有的字符和符号。

HTML charset 属性

想要正常显示一个 HTML 页面，浏览器就需要知道网页使用的字符集。网页中的字符集使用 `<meta>` 标签来指定：

```
<meta charset="UTF-8">
```

注：如果我们没有刻意指定 `meta` 属性，默认的字符集编码也是 UTF-8。

更多 UTF-8 编码可以参考：[HTML Unicode \(UTF-8\) 参考手册](#)

UTF-8 字符

很多 UTF-8 字符无法在键盘上输入，但我们可以使用数字（称为实体编号）来表示：

- A 为 65
- B 为 66
- C 为 67

实例

```
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> </head> <body> <p>显示结果： A B C</p> <p>显示结果： &#65;
&#66; &#67;</p> </body> </html>
```

尝试一下 »

实例解析：

`<meta charset="UTF-8">` 定义来字符集。

A, B, 和 C 也可以使用 65, 66, 和 67 来表示。

实体编号需要以 `&#` 开头并以分号 `;` 结尾，这样才能正确显示一个字符。

同样 Emoji 也是字符，可以在 HTML 页面中跟其他字符一样使用它：

实例

```
<!DOCTYPE html> <html> <head> <meta charset="UTF-8"> </head> <body> <h1>Emoji 标签符号</h1> <p>可以通过 font-size
属性，像设置字体大小一样，设置表情的大小。</p> <p style="font-size:48px"> &#128512; &#128516; &#128525; &#128151;
</p> </body> </html>
```

尝试一下 »

Emoji 表情符号

下表列出来一些 Emoji 表情符号：

Emoji	值
	🗻
	🗼
	🗽
	🗾
	🗿
	😀
	😁
	😂
	😃
	😄
	😅

更多 Emoji 内容参考：[Emoji 参考手册](#)

[HTML 颜色混搭](#)

HTML 字符集

HTML 字符集

如需正确地显示 HTML 页面，浏览器必须知道使用何种字符集。

万维网早期使用的字符集是 ASCII。ASCII 支持 0-9 的数字，大写和小写英文字母表，以及一些特殊字符。[完整的 ASCII 参考手册](#)。

由于很多国家使用的字符并不属于 ASCII，现代浏览器的默认字符集是 ISO-8859-1。[完整的 ISO-8859-1 参考手册](#)。

如果网页使用不同于 ISO-8859-1 的字符集，就应该在 <meta> 标签进行指定。[尝试一下](#)

ISO 字符集

ISO 字符集是国际标准组织 (ISO) 针对不同的字母表/语言定义的标准字符集。

下面列出了世界各地使用的不同字符集：

字符集	描述	使用范围
ISO-8859-1	Latin alphabet part 1	北美、西欧、拉丁美洲、加勒比海、加拿大、非洲
ISO-8859-2	Latin alphabet part 2	东欧
ISO-8859-3	Latin alphabet part 3	SE Europe、世界语、其他杂项

ISO-8859-4	Latin alphabet part 4	斯堪的纳维亚/波罗的海（以及其他没有包括在 ISO-8859-1 中的部分）
ISO-8859-5	Latin/Cyrillic part 5	使用古代斯拉夫语字母表的语言，比如保加利亚语、白俄罗斯文、俄罗斯语、马其顿语
ISO-8859-6	Latin/Arabic part 6	使用阿拉伯字母的语言
ISO-8859-7	Latin/Greek part 7	现代希腊语，以及由希腊语衍生的数学符号
ISO-8859-8	Latin/Hebrew part 8	使用希伯来语的语言
ISO-8859-9	Latin 5 part 9	土耳其语。除了土耳其字符取代了冰岛文字，其它与 ISO-8859-1 相同。
ISO-8859-10	Latin 6	拉普兰语、日耳曼语、爱斯基摩北欧语
ISO-8859-15	Latin 9 (aka Latin 0)	与 ISO 8859-1 类似，欧元符号和其他一些字符取代了一些较少使用的符号
ISO-2022-JP	Latin/Japanese part 1	日语
ISO-2022-JP-2	Latin/Japanese part 2	日语
ISO-2022-KR	Latin/Korean part 1	韩语

Unicode 标准

由于上面列出的字符集都有容量限制，而且不兼容多语言环境，Unicode 联盟开发了 Unicode 标准。Unicode 标准涵盖了世界上的所有字符、标点和符号。不论是何种平台、程序或语言，Unicode 都能够进行文本数据的处理、存储和交换。

Unicode 联盟

Unicode 联盟开发了 Unicode 标准。他们的目标是用标准的 Unicode 转换格式 (UTF) 来取代现有的字符集。Unicode 标准已经获得了成功，在 XML、Java、ECMAScript (JavaScript)、LDAP、CORBA 3.0、WML 中，Unicode 已经得到了实现。在许多操作系统以及所有的现代浏览器中，Unicode 同样得到了支持。Unicode 联盟与领导性的标准发展组织进行合作，比如 ISO、W3C 以及 ECMA。Unicode 可以被不同的字符集兼容。最常用的编码方式是 UTF-8 和 UTF-16：

字符集	描述
UTF-8	UTF8 中的字符可以是 1-4 个字节长。UTF-8 可以表示 Unicode 标准中的任意字符。UTF-8 向后兼容 ASCII。UTF-8 是网页和电子邮件的首选编码。
UTF-16	16 比特的 Unicode 转换格式是一种 Unicode 可变字符编码，能够对全部 Unicode 指令表进行编码。UTF-16 主要被用于操作系统和环境中，比如微软的 Windows 2000/XP/2003/Vista/CE 以及 Java 和 .NET 字节代码环境。

提示： 最前面的 256 个 Unicode 字符集字符对应于 256 个 ISO-8859-1 字符。
提示： 所有 HTML 4 浏览器都已支持 UTF-8，而所有 XHTML 和 XML 处理器支持 UTF-8 和 UTF-16！

[HTML Emoji](#)

[HTML ASCII 参考手册](#)

点我分享笔记

HTML ASCII 参考手册

ASCII 字符集被用于因特网上不同计算机间传输信息。

ASCII 字符集

ASCII，它的全称是"美国信息交换标准代码"。它设计于 60 年代早期，是计算机和诸如打印机、磁带驱动器之类的硬件设备的标准字符集。ASCII 是 7 比特字符集，包含了 128 个不同的字符值。ASCII 支持 0-9 的数字，A-Z 大写和小写英文字母，以及一些特殊字符。被广泛使用于现代计算机、HTML 和因特网上的字符集都是基于 ASCII。以下表格列举了 128 个 ASCII 字符以及对应的 HTML 实体编码。

ASCII 可打印的字符

字符	编号	描述
	32	空格 (space)
!	33	感叹号 (exclamation mark)
"	34	引号 (quotation mark)
#	35	数字符号 (number sign)
\$	36	美元符号 (dollar sign)
%	37	百分比符号 (percent sign)

&	38	& 符号 (ampersand)
'	39	撇号 (apostrophe)
(40	左括号 (left parenthesis)
)	41	右括号 (right parenthesis)
*	42	星号 (asterisk)
+	43	加号 (plus sign)
,	44	逗号 (comma)
-	45	连字符 (hyphen)
.	46	句号 (period)
/	47	斜线 (slash)
0	48	数字 0
1	49	数字 1
2	50	数字 2
3	51	数字 3
4	52	数字 4
5	53	数字 5

6	54	数字 6
7	55	数字 7
8	56	数字 8
9	57	数字 9
:	58	冒号 (colon)
;	59	分号 (semicolon)
<	60	小于号 (less-than)
=	61	等于号 (equals-to)
>	62	大于号 (greater-than)
?	63	问号 (question mark)
@	64	@ 符号 (at sign)
A	65	大写字母 A
B	66	大写字母 B
C	67	大写字母 C
D	68	大写字母 D
E	69	大写字母 E

F	70	大写字母 F
G	71	大写字母 G
H	72	大写字母 H
I	73	大写字母 I
J	74	大写字母 J
K	75	大写字母 K
L	76	大写字母 L
M	77	大写字母 M
N	78	大写字母 N
O	79	大写字母 O
P	80	大写字母 P
Q	81	大写字母 Q
R	82	大写字母 R
S	83	大写字母 S
T	84	大写字母 T
U	85	大写字母 U

V	86	大写字母 V
W	87	大写字母 W
X	88	大写字母 X
Y	89	大写字母 Y
Z	90	大写字母 Z
[91	左方括号 (left square bracket)
\	92	反斜线 (backslash)
]	93	右方括号 (right square bracket)
^	94	插入符号 (caret)
_	95	下划线 (underscore)
`	96	重音符 (grave accent)
a	97	小写字母 a
b	98	小写字母 b
c	99	小写字母 c
d	100	小写字母 d
e	101	小写字母 e

f	102	小写字母 f
g	103	小写字母 g
h	104	小写字母 h
i	105	小写字母 i
j	106	小写字母 j
k	107	小写字母 k
l	108	小写字母 l
m	109	小写字母 m
n	110	小写字母 n
o	111	小写字母 o
p	112	小写字母 p
q	113	小写字母 q
r	114	小写字母 r
s	115	小写字母 s
t	116	小写字母 t
u	117	小写字母 u

v	118	小写字母 v
w	119	小写字母 w
x	120	小写字母 x
y	121	小写字母 y
z	122	小写字母 z
{	123	左花括号 (left curly brace)
	124	竖线 (vertical bar)
}	125	右花括号 (right curly brace)
~	126	波浪线 (tilde)

ASCII 设备控制字符

ASCII 控制字符（00-31，加上 127）最初被设计用来控制诸如打印机和磁带驱动器之类的硬件设备。控制字符（除了水平制表符、换行、回车之外）在 HTML 文档中不起任何作用。

字符	编号	描述
NUL	00	空字符 (null character)
SOH	01	标题开始 (start of header)

STX	02	正文开始 (start of text)
ETX	03	正文结束 (end of text)
EOT	04	传输结束 (end of transmission)
ENQ	05	请求 (enquiry)
ACK	06	收到通知/响应 (acknowledge)
BEL	07	响铃 (bell)
BS	08	退格 (backspace)
HT	09	水平制表符 (horizontal tab)
LF	10	换行 (line feed)
VT	11	垂直制表符 (vertical tab)
FF	12	换页 (form feed)
CR	13	回车 (carriage return)
SO	14	不用切换 (shift out)
SI	15	启用切换 (shift in)
DLE	16	数据链路转义 (data link escape)
DC1	17	设备控制 1 (device control 1)

DC2	18	设备控制 2 (device control 2)
DC3	19	设备控制 3 (device control 3)
DC4	20	设备控制 4 (device control 4)
NAK	21	拒绝接收/无响应 (negative acknowledge)
SYN	22	同步空闲 (synchronize)
ETB	23	传输块结束 (end transmission block)
CAN	24	取消 (cancel)
EM	25	已到介质末端/介质存储已满 (end of medium)
SUB	26	替补/替换 (substitute)
ESC	27	溢出/逃离/取消 (escape)
FS	28	文件分隔符 (file separator)
GS	29	组分隔符 (group separator)
RS	30	记录分隔符 (record separator)
US	31	单元分隔符 (unit separator)
DEL	127	删除 (delete)

ASCII 与 16 进制转换对照表

十六进制代码	MCS 字符或缩写	DEC 多国字符名
ASCII 控制字符 1		
00	NUL	空字符
01	SOH	标题起始 (Ctrl/A)
02	STX	文本起始 (Ctrl/B)
03	ETX	文本结束 (Ctrl/C)
04	EOT	传输结束 (Ctrl/D)
05	ENQ	询问 (Ctrl/E)
06	ACK	认可 (Ctrl/F)
07	BEL	铃 (Ctrl/G)
08	BS	退格 (Ctrl/H)
09	HT	水平制表栏 (Ctrl/I)
0A	LF	换行 (Ctrl/J)
0B	VT	垂直制表栏 (Ctrl/K)

0C	FF	换页 (Ctrl/L)
0D	CR	回车 (Ctrl/M)
0E	SO	移出 (Ctrl/N)
0F	SI	移入 (Ctrl/O)
10	DLE	数据链接丢失 (Ctrl/P)
11	DC1	设备控制 1 (Ctrl/Q)
12	DC2	设备控制 2 (Ctrl/R)
13	DC3	设备控制 3 (Ctrl/S)
14	DC4	设备控制 4 (Ctrl/T)
15	NAK	否定接受 (Ctrl/U)
16	SYN	同步闲置符 (Ctrl/V)
17	ETB	传输块结束 (Ctrl/W)
18	CAN	取消 (Ctrl/X)
19	EM	媒体结束 (Ctrl/Y)
1A	SUB	替换 (Ctrl/Z)
1B	ESC	换码符

1C	FS	文件分隔符
1D	GS	组分分隔符
1E	RS	记录分隔符
1F	US	单位分隔符
ASCII 特殊和数字字符		
20	SP	空格
21	!	感叹号
22	"	引号 (双引号)
23	#	数字符号
24	\$	美元符
25	%	百分号
26	&	和号
27	"	省略号 (单引号)
28	(左圆括号
29)	右圆括号
2A	*	星号

2B		加号
2C	,	逗号
2D	--	连字号或减号
2E	.	句点或小数点
2F	/	斜杠
30	0	零
31	1	1
32	2	2
33	3	3
34	4	4
35	5	5
36	6	6
37	7	7
38	8	8
39	9	9
3A	:	冒号

3B	;	分号
3C	<	小于
3D	=	等于
3E	>	大于
3F	?	问号
ASCII 字母字符		
40	@	商业 at 符号
41	A	大写字母 A
42	B	大写字母 B
43	C	大写字母 C
44	D	大写字母 D
45	E	大写字母 E
46	F	大写字母 F
47	G	大写字母 G
48	H	大写字母 H
49	I	大写字母 I

4A	J	大写字母 J
4B	K	大写字母 K
4C	L	大写字母 L
4D	M	大写字母 M
4E	N	大写字母 N
4F	O	大写字母 O
50	P	大写字母 P
51	Q	大写字母 Q
52	R	大写字母 R
53	S	大写字母 S
54	T	大写字母 T
55	U	大写字母 U
56	V	大写字母 V
57	W	大写字母 W
58	X	大写字母 X
59	Y	大写字母 Y

5A	Z	大写字母 Z
5B	[左中括号
5C	\	反斜杠
5D]	右中括号
5E	^	音调符号
5F	_	下划线
60	`	重音符
61	a	小写字母 a
62	b	小写字母 b
63	c	小写字母 c
64	d	小写字母 d
65	e	小写字母 e
66	f	小写字母 f
67	g	小写字母 g
68	h	小写字母 h
69	i	小写字母 i

6A	j	小写字母 j
6B	k	小写字母 k
6C	l	小写字母 l
6D	m	小写字母 m
6E	n	小写字母 n
6F	o	小写字母 o
70	p	小写字母 p
71	q	小写字母 q
72	r	小写字母 r
73	s	小写字母 s
74	t	小写字母 t
75	u	小写字母 u
76	v	小写字母 v
77	w	小写字母 w
78	x	小写字母 x
79	y	小写字母 y

7A	z	小写字母 z
7B	{	左大括号
7C		垂直线
7D	}	右大括号 (ALTMODE)
7E	~	代字号 (ALTMODE)
7F	DEL	擦掉 (DELETE)
控制字符		
80		[保留]
81		[保留]
82		[保留]
83		[保留]
84	IND	索引
85	NEL	下一行
86	SSA	被选区域起始
87	ESA	被选区域结束
88	HTS	水平制表符集

89	HTJ	对齐的水平制表符集
8A	VTs	垂直制表符集
8B	PLD	部分行向下
8C	PLU	部分行向上
8D	RI	反向索引
8E	SS2	单移 2
8F	SS3	单移 3
90	DCS	设备控制字符串
91	PU1	专用 1
92	PU2	专用 2
93	STS	设置传输状态
94	CCH	取消字符
95	MW	消息等待
96	SPA	保护区起始
97	EPA	保护区结束
98		[保留]

99		[保留]
9A		[保留]
9B	CSI	控制序列引导符
9C	ST	字符串终止符
9D	OSC	操作系统命令
9E	PM	秘密消息
9F	APC	应用程序
其他字符		
A0		[保留] 2
A1	?	反向感叹号
A2	?	分币符
A3	?	英磅符
A4		[保留] 2
A5	?	人民币符
A6		[保留] 2
A7	§	章节符

A8	¤	通用货币符号 2
A9	?	版权符号
AA	?	阴性顺序指示符
AB	?	左角引号
AC		[保留] 2
AD		[保留] 2
AE		[保留] 2
AF		[保留] 2
B0	°	温度符
B1	±	加/减号
B2	?	上标 2
B3	?	上标 3
B4		[保留] 2
B5	?	微符
B6	?	段落符, pilcrow
B7	·	中点

B8		[保留] 2
B9	?	上标 1
BA	?	阳性顺序指示符
BB	?	右角引号
BC	?	分数四分之一
BD	?	分数二分之一
BE		[保留] 2
BF	?	反向问号
C0	?	带重音符的大写字母 A
C1	?	带尖锐重音的大写字母 A
C2	?	带音调符号的大写字母 A
C3	?	带代字号的大写字母 A
C4	?	带元音变音 (分音符号) 的大写字母 A
C5	?	带铃声的大写字母 A
C6	?	大写字母 AE 双重元音
C7	?	带变音符号的大写字母 C

C8	?	带重音符的大写字母 E
C9	?	带尖锐重音的大写字母 E
CA	?	带音调符号的大写字母 E
CB	?	带元音变音 (分音符号) 的大写字母 E
CC	?	带重音符的大写字母 I
CD	?	带尖锐重音的大写字母 I
CE	?	带音调符号的大写字母 I
CF	?	带元音变音 (分音符号) 的大写字母 I
D0		[保留] 2
D1	?	带代字号的大写字母 N
D2	?	带重音符的大写字母 O
D3	?	带尖锐重音的大写字母 O
D4	?	带音调符号的大写字母 O
D5	?	带代字号的大写字母 O
D6	?	带元音变音 (分音符号) 的大写字母 O
D7	OE	大写字母 OE 连字 2

D8	?	帶斜杠的大写字母 O
D9	?	帶重音符的大写字母 U
DA	?	帶尖锐重音的大写字母 U
DB	?	帶音调符号的大写字母 U
DC	?	帶元音变音 (分音符号) 的大写字母 U
DD	Y	帶元音变音 (分音符号) 的大写字母 Y
DE		[保留] 2
DF	?	德语高调小写字母 s
E0	à	帶重音符的小写字母 a
E1	á	帶尖锐重音的小写字母 a
E2	?	帶音调符号的小写字母 a
E3	?	帶代字号的小写字母 a
E4	?	帶元音变音 (分音符号) 的小写字母 a
E5	?	帶铃声的小写字母 a
E6	?	小写字母 ae 双重元音
E7	?	帶变音符号的小写字母 c

E8	è	带重音符的小写字母 e
E9	é	带尖锐重音的小写字母 e
EA	ê	带音调符号的小写字母 e
EB	?	带元音变音 (分音符号) 的小写字母 e
EC	ì	带重音符的小写字母 i
ED	í	带尖锐重音的小写字母 i
EE	?	带音调符号的小写字母 i
EF	?	带元音变音 (分音符号) 的小写字母 i
F0		[保留] 2
F1	?	带代字号的小写字母 n
F2	ò	带重音符的小写字母 o
F3	ó	带尖锐重音的小写字母 o
F4	?	带音调符号的小写字母 o
F5	?	带代字号的小写字母 o
F6	?	带元音变音 (分音符号) 的小写字母 o
F7	oe	小写字母 oe 连字 2

F8	?	带斜杠的小写字母 o
F9	ù	带重音符的小写字母 u
FA	ú	带尖锐重音的小写字母 u
FB	?	带音调符号的小写字母 u
FC	ü	带元音变音 (分音符号) 的小写字母 u
FD	?	带元音变音 (分音符号) 的小写字母 y 2
FE		[保留] 2
FF		[保留] 2

[HTML 字符集](#)

[HTML ISO-8859-1 参考手册](#)

点我分享笔记

HTML ISO-8859-1 参考手册

现代的浏览器支持的字符集：

- [ASCII 字符集](#)
- [标准 ISO 字符集](#)
- [数学符号、希腊字母、其他符号](#)

ISO-8859-1

ISO-8859-1 是大多数浏览器默认的字符集。

ISO-8859-1 的较低部分（从 1 到 127 之间的代码）是最初的 ASCII 字符集（0-9 的数字，大写和小写英文字母表，以及一些特殊字符）。

ISO-8859-1 的较高部分（从 160 到 255 之间的代码）包含了一些西欧国家使用的字符和一些被广泛使用的特殊字符，它们全都有实体名称。这些符号中的大多数都可以在不进行实体引用的情况下使用，但是实体名称或实体编号为那些不容易通过键盘键入的符号提供了表达的方法。

HTML 预留字符

HTML and XHTML 预留了一些字符。比如，您不能使用包含这些字符的文本，因为浏览器可能会误以为是 HTML 标签。

HTML and XHTML 中央处理器必须识别以下表格所列举的五种特殊字符：

字符	实体编号	实体名称	描述
	 	 	非间断空格（non-breaking space）
¡	¡	¡	倒置感叹号（inverted exclamation mark）
¢	¢	¢	美分符号（cent）
£	£	£	英镑符号（pound）
¤	¤	¤	货币符号（currency）
¥	¥	¥	人民币/日元符号（yen）
	¦	¦	间断的竖杠（broken vertical bar）
§	§	§	小节号（section）

¨	¨	¨	分音符号 (spacing diaeresis)
©	©	©	版权所有 (copyright)
^a	ª	ª	阴性序数记号 (feminine ordinal indicator)
«	«	«	左双角引号 (angle quotation mark (left))
¬	¬	¬	否定符号 (negation)
	­	­	软连字符 (soft hyphen)
®	®	®	注册商标 (registered trademark)
-	¯	¯	长音符号 (spacing macron)
°	°	°	度符号 (degree)
±	±	±	加减号/正负号 (plus-or-minus)
²	²	²	上标 2 (superscript 2)
³	³	³	上标 3 (superscript 3)
´	´	´	尖音符号 (spacing acute)
μ	µ	µ	微米符号 (micro)
¶	¶	¶	段落符号 (paragraph)
·	·	·	中间点 (middle dot)

¸	¸	¸	变音符号 (spacing cedilla)
¹	¹	¹	上标 1 (superscript 1)
º	º	º	阳性序数记号 (masculine ordinal indicator)
»	»	»	右双角引号 (angle quotation mark (right))
¼	¼	¼	1/4 分数 (fraction 1/4)
½	½	½	1/2 分数 (fraction 1/2)
¾	¾	¾	3/4 分数 (fraction 3/4)
¿	¿	¿	倒置问号 (inverted question mark)

ISO 8859-1 字符实体

字符	实体编号	实体名称	描述
À	À	À	大写字母 A, 重音 (grave accent)
Á	Á	Á	大写字母 A, 尖音 (acute accent)
Â	Â	Â	大写字母 A, 抑扬音 (circumflex accent)
Ã	Ã	Ã	大写字母 A, 腭化 (tilde)
Ä	Ä	Ä	大写字母 A, 带有变音符号标记 (umlaut mark)
Å	Å	Å	大写字母 A, 带有上圆圈 (ring)

Æ	Æ	Æ	大写字母 AE
Ç	Ç	Ç	大写字母 C, 变音 (cedilla)
È	È	È	大写字母 E, 重音 (grave accent)
É	É	É	大写字母 E, 尖音 (acute accent)
Ê	Ê	Ê	大写字母 E, 抑扬音 (circumflex accent)
Ë	Ë	Ë	大写字母 E, 带有变音符号标记 (umlaut mark)
Ì	Ì	Ì	大写字母 I, 重音 (grave accent)
Í	Í	Í	大写字母 I, 尖音 (acute accent)
Î	Î	Î	大写字母 I, 抑扬音 (circumflex accent)
Ï	Ï	Ï	大写字母 I, 带有变音符号标记 (umlaut mark)
Ð	Ð	Ð	冰岛语大写字母 eth
Ñ	Ñ	Ñ	大写字母 N, 腭化 (tilde)
Ò	Ò	Ò	大写字母 O, 重音 (grave accent)
Ó	Ó	Ó	大写字母 O, 尖音 (acute accent)
Ô	Ô	Ô	大写字母 O, 抑扬音 (circumflex accent)
Õ	Õ	Õ	大写字母 O, 腭化 (tilde)

Ö	Ö	Ö	大写字母 O, 带有变音符号标记 (umlaut mark)
×	×	×	乘号 (multiplication)
Ø	Ø	Ø	大写字母 O, 带有斜线 (slash)
Û	Ù	Ù	大写字母 U, 重音 (grave accent)
Ú	Ú	Ú	大写字母 U, 尖音 (acute accent)
Û	Û	Û	大写字母 U, 抑扬音 (circumflex accent)
Ü	Ü	Ü	大写字母 U, 带有变音符号标记 (umlaut mark)
Ý	Ý	Ý	大写字母 Y, 尖音 (acute accent)
Þ	Þ	Þ	冰岛语大写字母 THORN
ß	ß	ß	德语小写字母 sharp s
à	à	à	小写字母 a, 重音 (grave accent)
á	á	á	小写字母 a, 尖音 (acute accent)
â	â	â	小写字母 a, 抑扬音 (circumflex accent)
ã	ã	ã	小写字母 a, 腭化 (tilde)
ä	ä	ä	小写字母 a, 带有变音符号标记 (umlaut mark)
å	å	å	小写字母 a, 带有上圆圈 (ring)

æ	æ	æ	小写字母 ae
ç	ç	ç	小写字母 c, 变音 (cedilla)
è	è	è	小写字母 e, 重音 (grave accent)
é	é	é	小写字母 e, 尖音 (acute accent)
ê	ê	ê	小写字母 e, 抑扬音 (circumflex accent)
ë	ë	ë	小写字母 e, 带有变音符号标记 (umlaut mark)
ì	ì	ì	小写字母 i, 重音 (grave accent)
í	í	í	小写字母 i, 尖音 (acute accent)
î	î	î	小写字母 i, 抑扬音 (circumflex accent)
ï	ï	ï	小写字母 i, 带有变音符号标记 (umlaut mark)
ð	ð	ð	冰岛语小写字母 eth
ñ	ñ	ñ	小写字母 n, 腭化 (tilde)
ò	ò	ò	小写字母 o, 重音 (grave accent)
ó	ó	ó	小写字母 o, 尖音 (acute accent)
ô	ô	ô	小写字母 o, 抑扬音 (circumflex accent)
õ	õ	õ	小写字母 o, 腭化 (tilde)

ö	ö	ö	小写字母 o, 带有变音符号标记 (umlaut mark)
÷	÷	÷	除号 (division)
ø	ø	ø	小写字母 o, 带有斜线 (slash)
ù	ù	ù	小写字母 u, 重音 (grave accent)
ú	ú	ú	小写字母 u, 尖音 (acute accent)
û	û	û	小写字母 u, 抑扬音 (circumflex accent)
ü	ü	ü	小写字母 u, 带有变音符号标记 (umlaut mark)
ý	ý	ý	小写字母 y, 尖音 (acute accent)
þ	þ	þ	冰岛语小写字母 thorn
ÿ	ÿ	ÿ	小写字母 y, 带有变音符号标记 (umlaut mark)

[HTML ASCII 参考手册](#)

[HTML 符号实体参考手册](#)

点我分享笔记

HTML 符号实体参考手册

HTML 符号实体

本字符实体参考手册包括了数学符号、希腊字符、各种箭头记号、科技符号以及形状。

注释： 实体名称对大小写敏感。

HTML 支持的数学符号

字符	实体编号	实体名称	描述
∀	∀	∀	for all
∂	∂	∂	part
∃	∃	∃	exists
∅	∅	∅	empty
∇	∇	∇	nabla
∈	∈	∈	isin
∉	∉	∉	notin
⊃	∋	∋	ni
∏	∏	∏	prod
∑	∑	∑	sum
−	−	−	minus

*	∗	∗	lowast
√	√	√	square root
∞	∝	∝	proportional to
∞	∞	∞	infinity
∠	∠	∠	angle
∧	∧	∧	and
∨	∨	∨	or
∩	∩	∩	cap
∪	∪	∪	cup
∫	∫	∫	integral
∴	∴	∴	therefore
∼	∼	∼	similar to
≅	≅	≅	congruent to
≈	≈	≈	almost equal
≠	≠	≠	not equal
≡	≡	≡	equivalent

\leq	≤	≤	less or equal
\geq	≥	≥	greater or equal
\subset	⊂	⊂	subset of
\supset	⊃	⊃	superset of
$\not\subset$	⊄	⊅	not subset of
\subseteq	⊆	⊆	subset or equal
\supseteq	⊇	⊇	superset or equal
\oplus	⊕	⊕	circled plus
\otimes	⊗	⊗	circled times
\perp	⊥	⊥	perpendicular
\cdot	⋅	⋅	dot operator

HTML 支持的希腊字母

字符	实体编号	实体名称	描述
A	Α	Α	Alpha
B	Β	Β	Beta
Γ	Γ	Γ	Gamma

Δ	Δ	Δ	Delta
E	Ε	Ε	Epsilon
Z	Ζ	Ζ	Zeta
H	Η	Η	Eta
Θ	Θ	Θ	Theta
I	Ι	Ι	Iota
K	Κ	Κ	Kappa
Λ	Λ	Λ	Lambda
M	Μ	Μ	Mu
N	Ν	Ν	Nu
Ξ	Ξ	Ξ	Xi
O	Ο	Ο	Omicron
Π	Π	Π	Pi
P	Ρ	Ρ	Rho
	undefined		Sigmaf
Σ	Σ	Σ	Sigma

T	Τ	Τ	Tau
Y	Υ	Υ	Upsilon
Φ	Φ	Φ	Phi
X	Χ	Χ	Chi
Ψ	Ψ	Ψ	Psi
Ω	Ω	Ω	Omega
α	α	α	alpha
β	β	β	beta
γ	γ	γ	gamma
δ	δ	δ	delta
ε	ε	ε	epsilon
ζ	ζ	ζ	zeta
η	η	η	eta
θ	θ	θ	theta
ι	ι	ι	iota

κ	κ	κ	kappa
λ	λ	λ	lambda
μ	μ	μ	mu
ν	ν	ν	nu
ξ	ξ	ξ	xi
ο	ο	ο	omicron
π	π	π	pi
ρ	ρ	ρ	rho
ς	ς	ς	sigmaf
σ	σ	σ	sigma
τ	τ	τ	tau
υ	υ	υ	upsilon
φ	φ	φ	phi
χ	χ	χ	chi
ψ	ψ	ψ	psi
ω	ω	ω	omega

θ	ϑ	ϑ	theta symbol
Υ	ϒ	ϒ	upsilon symbol
π	ϖ	ϖ	pi symbol

HTML 支持的其他实体

字符	实体编号	实体名称	描述
Œ	Œ	Œ	capital ligature OE
œ	œ	œ	small ligature oe
Š	Š	Š	capital S with caron
š	š	š	small S with caron
Ÿ	Ÿ	Ÿ	capital Y with diaeres
ƒ	ƒ	ƒ	f with hook
ˆ	ˆ	ˆ	modifier letter circumflex accent
˜	˜	˜	small tilde
	 	 	en space
	 	 	em space

	 	 	thin space
	‌	‌	zero width non-joiner
	‍	‍	zero width joiner
	‎	‎	left-to-right mark
	‏	‏	right-to-left mark
–	–	–	en dash
—	—	—	em dash
'	‘	‘	left single quotation mark
'	’	’	right single quotation mark
,	‚	‚	single low-9 quotation mark
"	“	“	left double quotation mark
"	”	”	right double quotation mark
„	„	„	double low-9 quotation mark
†	†	†	dagger
‡	‡	‡	double dagger
·	•	•	bullet

...	…	…	horizontal ellipsis
‰	‰	‰	per mille
'	′	′	minutes
"	″	″	seconds
<	‹	&lquo;	single left angle quotation
>	›	&rquo;	single right angle quotation
—	‾	‾	overline
€	€	€	euro
™	™ or ™	™	trademark
←	←	←	left arrow
↑	↑	↑	up arrow
→	→	→	right arrow
↓	↓	↓	down arrow
↔	↔	↔	left right arrow
↵	↵	↵	carriage return arrow
⌈	⌈	⌈	left ceiling

l	⌉	⌉	right ceiling
l	⌊	⌊	left floor
l	⌋	⌋	right floor
◊	◊	◊	lozenge
♠	♠	♠	spade
♣	♣	♣	club
♥	♥	♥	heart
♦	♦	♦	diamond

[HTML ISO-8859-1 参考手册](#)

[HTML URL 编码参考手册](#)

[点我分享笔记](#)

HTML URL 编码 参考手册

URL 编码会将字符转换为可通过因特网传输的格式。

URL - 统一资源定位器

Web 浏览器通过 URL 从 web 服务器请求页面。

URL 是网页的地址，比如：**https://www.runoob.com**。

URL 编码

URL 只能使用 [ASCII 字符集](#) 来通过因特网进行发送。

由于 URL 常常会包含 ASCII 集合之外的字符，URL 必须转换为有效的 ASCII 格式。

URL 编码使用 "%" 其后跟随两位的十六进制数来替换非 ASCII 字符。

URL 不能包含空格。URL 编码通常使用 + 来替换空格。

尝试一下

如果您点击下面的"提交"按钮，浏览器会在发送输入之前对其进行 URL 编码。服务器上的页面会显示出接收到的输入。

Hello Gintar	提交
--------------	----

试着输入一些其他字符，然后再次点击提交按钮。

URL 编码函数

JavaScript、PHP、ASP 都提供了对字符串进行 URL 编码的函数。

JavaScript 中使用 `encodeURIComponent()` 函数，PHP 中使用 `rawurlencode()` 函数，ASP 中使用 `Server.URLEncode()` 函数。

点击"URL 编码"按钮，看看 JavaScript 函数是怎么对文本进行编码的。

Hello Gintar

注释：JavaScript 函数将空格编码成 %20 。

URL 编码参考手册

ASCII 字符	URL-编码
space	%20
!	%21
"	%22
#	%23
\$	%24
%	%25
&	%26
'	%27
(%28
)	%29
*	%2A
+	%2B
,	%2C
-	%2D

.	%2E
/	%2F
0	%30
1	%31
2	%32
3	%33
4	%34
5	%35
6	%36
7	%37
8	%38
9	%39
:	%3A
;	%3B
<	%3C
=	%3D

>	%3E
?	%3F
@	%40
A	%41
B	%42
C	%43
D	%44
E	%45
F	%46
G	%47
H	%48
I	%49
J	%4A
K	%4B
L	%4C
M	%4D

N	%4E
O	%4F
P	%50
Q	%51
R	%52
S	%53
T	%54
U	%55
V	%56
W	%57
X	%58
Y	%59
Z	%5A
[%5B
\	%5C
]	%5D

^	%5E
_	%5F
`	%60
a	%61
b	%62
c	%63
d	%64
e	%65
f	%66
g	%67
h	%68
i	%69
j	%6A
k	%6B
l	%6C
m	%6D

n	%6E
o	%6F
p	%70
q	%71
r	%72
s	%73
t	%74
u	%75
v	%76
w	%77
x	%78
y	%79
z	%7A
{	%7B
	%7C
}	%7D

~	%7E
	%7F
`	%80
☒	%81
,	%82
<i>f</i>	%83
”	%84
...	%85
†	%86
‡	%87
^	%88
%0	%89
Š	%8A
<	%8B
Œ	%8C
☒	%8D

ž	%8E
☒	%8F
☒	%90
'	%91
'	%92
"	%93
"	%94
.	%95
—	%96
—	%97
~	%98
™	%99
š	%9A
>	%9B
œ	%9C
☒	%9D

ž	%9E
Ÿ	%9F
	%A0
ı	%A1
¢	%A2
£	%A3
¤	%A4
¥	%A5
 	%A6
§	%A7
..	%A8
©	%A9
a	%AA
<<	%AB
¬	%AC
	%AD

®	%AE
-	%AF
°	%B0
±	%B1
²	%B2
³	%B3
´	%B4
µ	%B5
¶	%B6
·	%B7
¸	%B8
¹	%B9
º	%BA
»	%BB
¼	%BC
½	%BD

¾	%BE
¿	%BF
À	%C0
Á	%C1
Â	%C2
Ã	%C3
Ä	%C4
Å	%C5
Æ	%C6
Ç	%C7
È	%C8
É	%C9
Ê	%CA
Ë	%CB
Ì	%CC
Í	%CD

Î	%CE
İ	%CF
Đ	%D0
Ñ	%D1
Ò	%D2
Ó	%D3
Ô	%D4
Õ	%D5
Ö	%D6
×	%D7
Ø	%D8
Ù	%D9
Ú	%DA
Û	%DB
Ü	%DC
Ý	%DD

Ð	%DE
ß	%DF
à	%E0
á	%E1
â	%E2
ã	%E3
ä	%E4
å	%E5
æ	%E6
ç	%E7
è	%E8
é	%E9
ê	%EA
ë	%EB
ì	%EC
í	%ED

î	%EE
ï	%EF
ð	%F0
ñ	%F1
ò	%F2
ó	%F3
ô	%F4
õ	%F5
ö	%F6
÷	%F7
ø	%F8
ù	%F9
ú	%FA
û	%FB
ü	%FC
ý	%FD

þ	%FE
ÿ	%FF

URL 编码参考手册

ASCII 设备控制字符最初被设计为用来控制诸如打印机和磁带驱动器之类的硬件设备。在 URL 中这些字符不会起任何作用。

ASCII 字符	描述	URL-编码
NUL	null character	%00
SOH	start of header	%01
STX	start of text	%02
ETX	end of text	%03
EOT	end of transmission	%04
ENQ	enquiry	%05
ACK	acknowledge	%06
BEL	bell (ring)	%07
BS	backspace	%08
HT	horizontal tab	%09

LF	line feed	%0A
VT	vertical tab	%0B
FF	form feed	%0C
CR	carriage return	%0D
SO	shift out	%0E
SI	shift in	%0F
DLE	data link escape	%10
DC1	device control 1	%11
DC2	device control 2	%12
DC3	device control 3	%13
DC4	device control 4	%14
NAK	negative acknowledge	%15
SYN	synchronize	%16
ETB	end transmission block	%17
CAN	cancel	%18
EM	end of medium	%19

SUB	substitute	%1A
ESC	escape	%1B
FS	file separator	%1C
GS	group separator	%1D
RS	record separator	%1E
US	unit separator	%1F

[HTML 符号实体参考手册](#)

[HTML 语言代码参考手册](#)

点我分享笔记

HTML 语言代码 参考手册

ISO 语言代码

HTML 的 `lang` 属性可用于声明网页或部分网页的语言，这对搜索引擎和浏览器是有帮助的。
根据 W3C 推荐标准，您应该通过 `<html>` 标签中的 `lang` 属性对每张页面中的主要语言进行声明：
比如声明原文版语言：

```
<html lang="en"> ... </html>
```

在 XHTML 中，采用如下方式在 `<html>` 标签中对语言进行声明：

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en"> ... </html>
```

ISO 639-1 语言代码

ISO 639-1 为各种语言定义了缩略词。您可以在 HTML 和 XHTML 中的 lang 和 xml:lang 属性中使用它们。

语言	ISO 代码
Abkhazian	ab
Afar	aa
Afrikaans	af
Albanian	sq
Amharic	am
Arabic	ar
Aragonese	an
Armenian	hy
Assamese	as
Aymara	ay
Azerbaijani	az
Bashkir	ba

Basque	eu
Bengali (Bangla)	bn
Bhutani	dz
Bihari	bh
Bislama	bi
Breton	br
Bulgarian	bg
Burmese	my
Byelorussian (Belarusian)	be
Cambodian	km
Catalan	ca
Cherokee	
Chewa	
Chinese (简体)	zh
Chinese (繁体)	zh
Corsican	co

Croatian	hr
Czech	cs
Danish	da
Divehi	
Dutch	nl
Edo	
English	en
Esperanto	eo
Estonian	et
Faeroese	fo
Farsi	fa
Fiji	fj
Finnish	fi
Flemish	
French	fr
Frisian	fy

Fulfulde	
Galician	gl
Gaelic (Scottish)	gd
Gaelic (Manx)	gv
Georgian	ka
German	de
Greek	el
Greenlandic	kl
Guarani	gn
Gujarati	gu
Haitian Creole	ht
Hausa	ha
Hawaiian	
Hebrew	he, iw
Hindi	hi
Hungarian	hu

Ibibio	
Icelandic	is
Ido	io
Igbo	
Indonesian	id, in
Interlingua	ia
Interlingue	ie
Inuktitut	iu
Inupiak	ik
Irish	ga
Italian	it
Japanese	ja
Javanese	jv
Kannada	kn
Kanuri	
Kashmiri	ks

Kazakh	kk
Kinyarwanda (Ruanda)	rw
Kirghiz	ky
Kirundi (Rundi)	rn
Konkani	
Korean	ko
Kurdish	ku
Laothian	lo
Latin	la
Latvian (Lettish)	lv
Limburgish (Limburger)	li
Lingala	ln
Lithuanian	lt
Macedonian	mk
Malagasy	mg
Malay	ms

Malayalam	ml
Maltese	mt
Maori	mi
Marathi	mr
Moldavian	mo
Mongolian	mn
Nauru	na
Nepali	ne
Norwegian	no
Occitan	oc
Oriya	or
Oromo (Afaan Oromo)	om
Papiamentu	
Pashto (Pushto)	ps
Polish	pl

Portuguese	pt
Punjabi	pa
Quechua	qu
Rhaeto-Romance	rm
Romanian	ro
Russian	ru
Sami (Lappish)	
Samoan	sm
Sangro	sg
Sanskrit	sa
Serbian	sr
Serbo-Croatian	sh
Sesotho	st
Setswana	tn
Shona	sn
Sichuan Yi	ii

Sindhi	sd
Sinhalese	si
Siswati	ss
Slovak	sk
Slovenian	sl
Somali	so
Spanish	es
Sundanese	su
Swahili (Kiswahili)	sw
Swedish	sv
Syriac	
Tagalog	tl
Tajik	tg
Tamazight	
Tamil	ta
Tatar	tt

Telugu	te
Thai	th
Tibetan	bo
Tigrinya	ti
Tonga	to
Tsonga	ts
Turkish	tr
Turkmen	tk
Twi	tw
Uighur	ug
Ukrainian	uk
Urdu	ur
Uzbek	uz
Venda	
Vietnamese	vi
Volapük	vo

Wallon	wa
Welsh	cy
Wolof	wo
Xhosa	xh
Yi	
Yiddish	yi, ji
Yoruba	yo
Zulu	zu

[HTML URL 编码参考手册](#)

[HTML 国家/地区代码](#)

[点我分享笔记](#)

HTML 国家/地区 参考手册

ISO 国家和地区代码

在 HTML 中，国家/地区代码可作为 **lang** 属性中语言代码的补充。
语言代码的前两个字符定义网页的语言（[请参阅语言代码参考](#)）。

最后两个字符定义国家/地区。

以下实例将中文指定为语言，将中国指定为国家：

```
<html lang="zh-CN"> ... </html>
```

以下实例将英语指定为语言，将美国指定为国家：

```
<html lang="en-US"> ... </html>
```

ISO 639-1 国家/地区代码

国家/地区	ISO 代码
AFGHANISTAN	AF
ALBANIA	AL
ALGERIA	DZ
AMERICAN SAMOA	AS
ANDORRA	AD
ANGOLA	AO
ANTARCTICA	AQ
ANTIGUA AND BARBUDA	AG
ARGENTINA	AR
ARMENIA	AM

ARUBA	AW
AUSTRALIA	AU
AUSTRIA	AT
AZERBAIJAN	AZ
BAHAMAS	BS
BAHRAIN	BH
BANGLADESH	BD
BARBADOS	BB
BELARUS	BY
BELGIUM	BE
BELIZE	BZ
BENIN	BJ
BERMUDA	BM
BHUTAN	BT
BOLIVIA	BO
BOSNIA AND HERZEGOVINA	BA

BOTSWANA	BW
BOUVET ISLAND	BV
BRAZIL	BR
BRITISH INDIAN OCEAN TERRITORY	IO
BRUNEI DARUSSALAM	BN
BULGARIA	BG
BURKINA FASO	BF
BURUNDI	BI
CAMBODIA	KH
CAMEROON	CM
CANADA	CA
CAPE VERDE	CV
CAYMAN ISLANDS	KY
CENTRAL AFRICAN REPUBLIC	CF
CHAD	TD
CHILE	CL

CHINA	CN
CHRISTMAS ISLAND	CX
COCOS (KEELING) ISLANDS	CC
COLOMBIA	CO
COMOROS	KM
CONGO	CG
CONGO, THE DEMOCRATIC REPUBLIC OF THE	CD
COOK ISLANDS	CK
COSTA RICA	CR
CÔTE D'IVOIRE	CI
CROATIA	HR
CUBA	CU
CYPRUS	CY
CZECH REPUBLIC	CZ
DENMARK	DK
DJIBOUTI	DJ

DOMINICA	DM
DOMINICAN REPUBLIC	DO
ECUADOR	EC
EGYPT	EG
EL SALVADOR	SV
EQUATORIAL GUINEA	GQ
ERITREA	ER
ESTONIA	EE
ETHIOPIA	ET
FALKLAND ISLANDS (MALVINAS)	FK
FAROE ISLANDS	FO
FIJI	FJ
FINLAND	FI
FRANCE	FR
FRENCH GUIANA	GF
FRENCH POLYNESIA	PF

FRENCH SOUTHERN TERRITORIES	TF
GABON	GA
GAMBIA	GM
GEORGIA	GE
GERMANY	DE
GHANA	GH
GIBRALTAR	GI
GREECE	GR
GREENLAND	GL
GRENADA	GD
GUADELOUPE	GP
GUAM	GU
GUATEMALA	GT
GUINEA	GN
GUINEA-BISSAU	GW
GUYANA	GY

HAITI	HT
HEARD ISLAND AND MCDONALD ISLANDS	HM
HONDURAS	HN
HONG KONG	HK
HUNGARY	HU
ICELAND	IS
INDIA	IN
INDONESIA	ID
IRAN, ISLAMIC REPUBLIC OF	IR
IRAQ	IQ
IRELAND	IE
ISRAEL	IL
ITALY	IT
JAMAICA	JM
JAPAN	JP
JORDAN	JO

KAZAKHSTAN	KZ
KENYA	KE
KIRIBATI	KI
KOREA, DEMOCRATIC PEOPLE'S REPUBLIC OF	KP
KOREA, REPUBLIC OF	KR
KUWAIT	KW
KYRGYZSTAN	KG
LAO PEOPLE'S DEMOCRATIC REPUBLIC (LAOS)	LA
LATVIA	LV
LEBANON	LB
LESOTHO	LS
LIBERIA	LR
LIBYA, STATE OF	LY
LIECHTENSTEIN	LI
LITHUANIA	LT
LUXEMBOURG	LU

MACAO	MO
MACEDONIA, THE FORMER YUGOSLAV REPUBLIC OF	MK
MADAGASCAR	MG
MALAWI	MW
MALAYSIA	MY
MALDIVES	MV
MALI	ML
MALTA	MT
MARSHALL ISLANDS	MH
MARTINIQUE	MQ
MAURITANIA	MR
MAURITIUS	MU
MAYOTTE	YT
MEXICO	MX
MICRONESIA, FEDERATED STATES OF	FM
MOLDOVA, REPUBLIC OF	MD

MONACO	MC
MONGOLIA	MN
MONTENEGRO	ME
MONTSERRAT	MS
MOROCCO	MA
MOZAMBIQUE	MZ
MYANMAR	MM
NAMIBIA	NA
NAURU	NR
NEPAL, FEDERAL DEMOCRATIC REPUBLIC OF	NP
NETHERLANDS	NL
NETHERLANDS ANTILLES	AN
NEW CALEDONIA	NC
NEW ZEALAND	NZ
NICARAGUA	NI
NIGER	NE

NIGERIA	NG
NIUE	NU
NORFOLK ISLAND	NF
NORTHERN MARIANA ISLANDS	MP
NORWAY	NO
OMAN	OM
PAKISTAN	PK
PALAU	PW
PALESTINE, STATE OF	PS
PANAMA	PA
PAPUA NEW GUINEA	PG
PARAGUAY	PY
PERU	PE
PHILIPPINES	PH
PITCAIRN	PN
POLAND	PL

PORTUGAL	PT
PUERTO RICO	PR
QATAR	QA
RÉUNION	RE
ROMANIA	RO
RUSSIAN FEDERATION	RU
RWANDA	RW
SAINT HELENA	SH
SAINT KITTS AND NEVIS	KN
SAINT LUCIA	LC
SAINT PIERRE AND MIQUELON	PM
SAINT VINCENT AND THE GRENADINES	VC
SAMOA	WS
SAN MARINO	SM
SAO TOME AND PRINCIPE	ST
SAUDI ARABIA	SA

SENEGAL	SN
SERBIA	RS
SEYCHELLES	SC
SIERRA LEONE	SL
SINGAPORE	SG
SLOVAKIA	SK
SLOVENIA	SI
SOLOMON ISLANDS	SB
SOMALIA	SO
SOUTH AFRICA	ZA
SOUTH GEORGIA AND THE SOUTH SANDWICH ISLANDS	GS
SOUTH SUDAN	SS
SPAIN	ES
SRI LANKA	LK
SUDAN	SD
SURINAME	SR

SVALBARD AND JAN MAYEN	SJ
SWAZILAND	SZ
SWEDEN	SE
SWITZERLAND	CH
SYRIAN ARAB REPUBLIC	SY
TAIWAN, CHINA	TW
TAJIKISTAN	TJ
TANZANIA, UNITED REPUBLIC OF	TZ
THAILAND	TH
TIMOR-LESTE	TL
TOGO	TG
TOKELAU	TK
TONGA	TO
TRINIDAD AND TOBAGO	TT
TUNISIA	TN
TURKEY	TR

TURKMENISTAN	TM
TURKS AND CAICOS ISLANDS	TC
TUVALU	TV
UGANDA	UG
UKRAINE	UA
UNITED ARAB EMIRATES	AE
UNITED KINGDOM	GB
UNITED STATES	US
UNITED STATES MINOR OUTLYING ISLANDS	UM
URUGUAY	UY
UZBEKISTAN	UZ
VANUATU	VU
VENEZUELA	VE
VIET NAM	VN
VIRGIN ISLANDS, BRITISH	VG
VIRGIN ISLANDS, U.S.	VI

WALLIS AND FUTUNA	WF
WESTERN SAHARA	EH
YEMEN	YE
ZAMBIA	ZM
ZIMBABWE	ZW

[HTML 语言代码参考手册](#)

[HTTP 状态消息](#)

[点我分享笔记](#)

HTTP 状态消息

当浏览器从 **web** 服务器请求服务时，可能会发生错误。
以下列举了有可能会返回的一系列 HTTP 状态消息：

1xx: 信息

消息：	描述：
100 Continue	服务器仅接收到部分请求，如果服务器没有拒绝该请求，客户端应该继续发送其余的请求。

101 Switching Protocols	服务器转换协议：服务器将遵从客户的请求转换到另外一种协议。
103 Checkpoint	用于 PUT 或者 POST 请求恢复失败时的恢复请求建议。

2xx: 成功

消息：	描述：
200 OK	请求成功（这是对 HTTP 请求成功的标准应答。）
201 Created	请求被创建完成，同时新的资源被创建。
202 Accepted	供处理的请求已被接受，但是处理未完成。
203 Non-Authoritative Information	请求已经被成功处理，但是一些应答头可能不正确，因为使用的是其他文档的拷贝。
204 No Content	请求已经被成功处理，但是没有返回新文档。浏览器应该继续显示原来的文档。如果用户定期地刷新页面，而 Servlet 可以确定用户文档足够新，这个状态代码是很有用的。
205 Reset Content	请求已经被成功处理，但是没有返回新文档。但浏览器应该重置它所显示的内容。用来强制浏览器清除表单输入内容。
206 Partial Content	客户发送了一个带有 Range 头的 GET 请求，服务器完成了它。

3xx: 重定向

消息：	描述：
-----	-----

300 Multiple Choices	多重选择。链接列表。用户可以选择某链接到达目的地。最多允许五个地址。
301 Moved Permanently	所请求的页面已经转移至新的 URL 。
302 Found	所请求的页面已经临时转移至新的 URL 。
303 See Other	所请求的页面可在别的 URL 下被找到。
304 Not Modified	未按预期修改文档。客户端有缓冲的文档并发出了一个条件性的请求（一般是提供 If-Modified-Since 头表示客户只想比指定日期更新的文档）。服务器告诉客户，原来缓冲的文档还可以继续使用。
305 Use Proxy	客户请求的文档应该通过 Location 头所指明的代理服务器提取。
306 Switch Proxy	<i>目前已不再使用，但是代码依然被保留。</i>
307 Temporary Redirect	被请求的页面已经临时移至新的 URL 。
308 Resume Incomplete	用于 PUT 或者 POST 请求恢复失败时的恢复请求建议。

4xx: 客户端错误

消息：	描述：
400 Bad Request	因为语法错误，服务器未能理解请求。
401 Unauthorized	合法请求，但对被请求页面的访问被禁止。因为被请求的页面需要身份验证，客户端没有提供或者身份验证失败。
402 Payment Required	<i>此代码尚无法使用。</i>

403 Forbidden	合法请求，但对被请求页面的访问被禁止。
404 Not Found	服务器无法找到被请求的页面。
405 Method Not Allowed	请求中指定的方法不被允许。
406 Not Acceptable	服务器生成的响应无法被客户端所接受。
407 Proxy Authentication Required	用户必须首先使用代理服务器进行验证，这样请求才会被处理。
408 Request Timeout	请求超出了服务器的等待时间。
409 Conflict	由于冲突，请求无法被完成。
410 Gone	被请求的页面不可用。
411 Length Required	"Content-Length" 未被定义。如果无此内容，服务器不会接受请求。
412 Precondition Failed	请求中的前提条件被服务器评估为失败。
413 Request Entity Too Large	由于所请求的实体太大，服务器不会接受请求。
414 Request-URI Too Long	由于 URL 太长，服务器不会接受请求。当 POST 请求被转换为带有很长的查询信息的 GET 请求时，就会发生这种情况。
415 Unsupported Media Type	由于媒介类型不被支持，服务器不会接受请求。
416 Requested Range Not Satisfiable	客户端请求部分文档，但是服务器不能提供被请求的部分。
417 Expectation Failed	服务器不能满足客户在请求中指定的请求头。

5xx: 服务器错误

消息:	描述:
500 Internal Server Error	请求未完成。服务器遇到不可预知的情况。
501 Not Implemented	请求未完成。服务器不支持所请求的功能，或者服务器无法完成请求。
502 Bad Gateway	请求未完成。服务器充当网关或者代理的角色时，从上游服务器收到一个无效的响应。
503 Service Unavailable	服务器当前不可用（过载或者当机）。
504 Gateway Timeout	网关超时。服务器充当网关或者代理的角色时，未能从上游服务器收到一个及时的响应。
505 HTTP Version Not Supported	服务器不支持请求中指明的 HTTP 协议版本。
511 Network Authentication Required	用户需要提供身份验证来获取网络访问入口。

[HTML 国家/地区代码](#)

[HTTP 方法: GET 对比 POST](#)

点我分享笔记

分类导航

- [HTML / CSS](#)
- [JavaScript](#)

- 服务端
- 数据库
- 数据分析
- 移动端
- XML 教程
- ASP.NET
- Web Service
- 开发工具
- 网站建设

反馈/建议反馈/建议

HTTP 方法：GET 对比 POST

两种最常用的 HTTP 方法是：GET 和 POST。

什么是 HTTP ？

超文本传输协议（HTTP）的设计目的是保证客户端与服务器之间的通信。

HTTP 的工作方式是客户端与服务器之间的请求-应答协议。

web 浏览器可能是客户端，而计算机上的网络应用程序也可能作为服务器端。

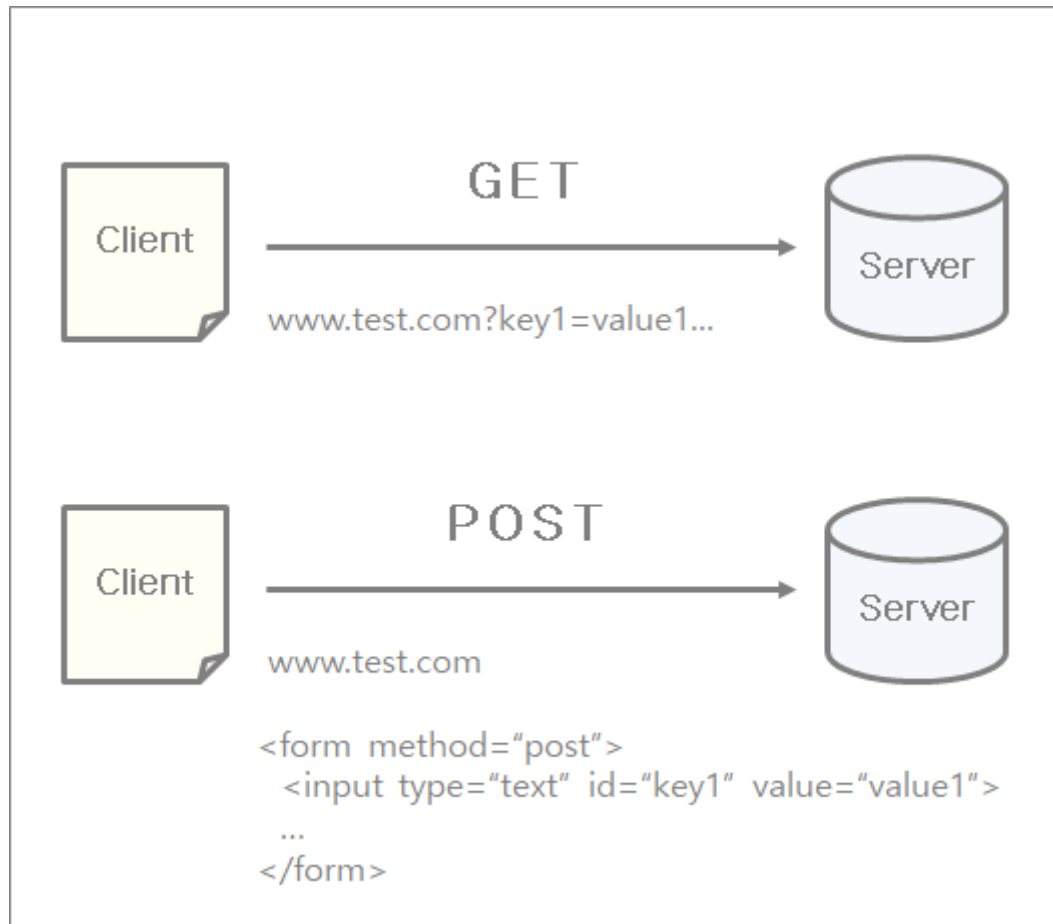
举例：客户端（浏览器）向服务器提交 HTTP 请求；服务器向客户端返回响应。响应包含关于请求的状态信息以及可能被请求的内容。

两种 HTTP 请求方法：GET 和 POST

在客户机和服务器之间进行请求-响应时，两种最常被用到的方法是：GET 和 POST。

- **GET** - 从指定的资源请求数据。
- **POST** - 向指定的资源提交要被处理的数据。

GET 提交参数一般显示在 URL 上，POST 通过表单提交不会显示在 URL 上，POST 更具隐蔽性：



GET 方法

请注意，查询字符串（名称/值对）是在 **GET** 请求的 **URL** 中发送的：

```
/test/demo_form.php?name1=value1&name2=value2
```

有关 **GET** 请求的其他一些注释：

- GET 请求可被缓存
- GET 请求保留在浏览器历史记录中
- GET 请求可被收藏为书签
- GET 请求不应在处理敏感数据时使用
- GET 请求有长度限制
- GET 请求只应当用于取回数据

POST 方法

请注意，查询字符串（名称/值对）是在 **POST** 请求的 **HTTP** 消息主体中发送的：

```
POST /test/demo_form.php HTTP/1.1
```

```
Host: runoob.com
```

```
name1=value1&name2=value2
```

有关 **POST** 请求的其他一些注释：

- POST 请求不会被缓存
- POST 请求不会保留在浏览器历史记录中

- POST 不能被收藏为书签
- POST 请求对数据长度没有要求

比较 GET 与 POST

下面的表格比较了两种 HTTP 方法：GET 和 POST。

	GET	POST
后退按钮/刷新	无害	数据会被重新提交（浏览器应该告知用户数据会被重新提交）。
书签	可收藏为书签	不可收藏为书签
缓存	能被缓存	不能缓存
编码类型	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data。为二进制数据使用多重编码。
历史	参数保留在浏览器历史中。	参数不会保存在浏览器历史中。
对数据长度的限制	是的。当发送数据时，GET 方法向 URL 添加数据；URL 的长度是受限制的（URL 的最大长度是 2048 个字符）。	无限制。
对数据类型的限制	只允许 ASCII 字符。	没有限制。也允许二进制数据。
安全性	与 POST 相比，GET 的安全性较差，因为所	POST 比 GET 更安全，因为参数不会被保存

	发送的数据是 URL 的一部分。 在发送密码或其他敏感信息时绝不要使用 GET ！	在浏览器历史或 web 服务器日志中。
可见性	数据在 URL 中对所有人都是可见的。	数据不会显示在 URL 中。

其他 HTTP 请求方法

下面的表格列出了其他一些 HTTP 请求方法：

方法	描述
HEAD	与 GET 相同，但只返回 HTTP 报头，不返回文档主体。
PUT	上传指定的 URI 表示。
DELETE	删除指定资源。
OPTIONS	返回服务器支持的 HTTP 方法。
CONNECT	把请求连接转换到透明的 TCP/IP 通道。

[HTTP 状态消息](#)

[Px/Em 换算工具](#)

1 篇笔记 写笔记

Form 中的 get 和 post 方法，在数据传输过程中分别对应了 HTTP 协议中的 GET 和 POST 方法。二者主要区别如下：

- o 1、Get 是用来从服务器上获得数据，而 Post 是用来向服务器上传递数据。
- o 2、Get 将表单中数据的按照 variable=value 的形式，添加到 action 所指向的 URL 后面，并且两者使用 “?”连接，而各个变量之间使用 “&”连接；Post 是将表单中的数据放在 form 的数据体中，按照变量和值相对应的方式，传递到 action 所指向 URL。
- o 3、Get 是不安全的，因为在传输过程，数据被放在请求的 URL 中，而如今现有的很多服务器、代理服务器或者用户代理都会将请求 URL 记录到日志文件中，然后放在某个地方，这样就可能会有一些隐私的信息被第三方看到。另外，用户也可以在浏览器上直接看到提交的数据，一些系统内部消息将会一同显示在用户面前。Post 的所有操作对用户来说都是不可见的。
- o 4、Get 传输的数据量小，这主要是因为受 URL 长度限制；而 Post 可以传输大量的数据，所以在上传文件只能使用 Post（当然还有一个原因，将在后面的提到）。
- o 5、Get 限制 Form 表单的数据集的值必须为 ASCII 字符；而 Post 支持整个 ISO10646 字符集。
- o 6、Get 是 Form 的默认方法。

使用 Post 传输的数据，可以通过设置编码的方式正确转化中文；而 Get 传输的数据却没有变化。在以后的程序中，我们一定要注意这一点。

风哥

风哥
liu***7080@qq.com

5 年前 (2019-03-07)

Px、Em 换算工具

以下工具提供了 em 和 px 的换算工具。

- 第一个输入框：设置了网页默认的字体像素 (通常 16px)
- 第二个输入框：输入像素值，可以将 px 换算为 em。
- 第三个输入框：输入 em（相对长度单位）值，可以将 em 换算为 px。

设置默认的像素大小:

px

PX 换算为 EM:

px

EM 换算为 PX:

em

转换

换算结果:

文本字体大小

下表列出了在网页字体默认值为 16px 时，px 和 em 及网页字体百分比的换算数据。

px	em	百分比
5px	0.3125em	31.25%
6px	0.3750em	37.50%
7px	0.4375em	43.75%
8px	0.5000em	50.00%
9px	0.5625em	56.25%
10px	0.6250em	62.50%
11px	0.6875em	68.75%
12px	0.7500em	75.00%
13px	0.8125em	81.25%
14px	0.8750em	87.50%
15px	0.9375em	93.75%
16px	1.0000em	100.00%
17px	1.0625em	106.25%
18px	1.1250em	112.50%
19px	1.1875em	118.75%

20px	1.2500em	125.00%
21px	1.3125em	131.25%
22px	1.3750em	137.50%
23px	1.4375em	143.75%
24px	1.5000em	150.00%
25px	1.5625em	156.25%

相关文章

- [px,pt,em 换算表](#)
- [px、em、rem 区别介绍](#)
-

[HTTP 方法: GET 对比 POST](#)

[键盘快捷键](#)

点我分享笔记

键盘快捷键

通过使用键盘快捷键可以节省时间。

Windows 和 Mac 的键盘快捷键

在现代操作系统中和计算机软件程序中，键盘快捷键经常被使用。
使用键盘快捷键能帮您节省很多时间。

基本的快捷键

描述	Windows	Mac OS
编辑菜单	Alt + E	Ctrl + F2 + F
文件菜单	Alt + F	Ctrl + F2 + E
视图菜单	Alt + V	Ctrl + F2 + V
全选文本	Ctrl + A	Cmd + A
复制文本	Ctrl + C	Cmd + C
查找文本	Ctrl + F	Cmd + F
查找替换文本	Ctrl + H	Cmd + F
新建文档	Ctrl + N	Cmd + N
打开文件	Ctrl + O	Cmd + O
打印选项	Ctrl + P	Cmd + P

保存文件	Ctrl + S	Cmd + S
粘贴文本	Ctrl + V	Cmd + V
剪切文本	Ctrl + X	Cmd + X
重做文本	Ctrl + Y	Shift + Cmd + Z
撤销文本	Ctrl + Z	Cmd + Z

文本编辑

描述	Windows	Mac OS
光标移动		
将文本插入光标向右移动或者移动到下一行行首	Right Arrow	Right Arrow
将文本插入光标向左移动或者移动到上一行行尾	Left Arrow	Left Arrow
将文本插入光标向上移动一行	Up Arrow	Up Arrow
将文本插入光标向下移动一行	Down Arrow	Down Arrow
将文本插入光标移动到当前行的行首	Home	Cmd + Left Arrow

将文本插入光标移动到当前行的行尾	End	Cmd + Right Arrow
将文本插入光标移动到文档的开头	Ctrl + Home	Cmd + Up Arrow
将文本插入光标移动到文档的结尾	Ctrl + End	Cmd + Down Arrow
将文本插入光标移动到上一个文本框	Page Up	Fn + Up Arrow
将文本插入光标移动到下一个文本框	Page Down	Fn + Down Arrow
将文本插入光标向左移动到前一个词的开头	Ctrl + Left Arrow	Option + Left Arrow
将文本插入光标向右移动到后一个词的开头	Ctrl + Right Arrow	Option + Right Arrow
将文本插入光标移动到行首	Ctrl + Up Arrow	Cmd + Left Arrow
将文本插入光标移动到行尾	Ctrl + Down Arrow	Cmd + Right Arrow
文本选择		
选择文本插入光标左边的字符	Shift + Left Arrow	Shift + Left Arrow
选择文本插入光标右边的字符	Shift + Right Arrow	Shift + Right Arrow
向上选择一行文本	Shift + Up Arrow	Shift + Up Arrow
向下选择一行文本	Shift + Down Arrow	Shift + Down Arrow
选择文本插入光标左边的字	Shift + Ctrl + Left	Shift + Opt + Left

选择文本插入光标右边的字	Shift + Ctrl + Right	Shift + Opt + Right
向左选择一段文本	Shift + Ctrl + Up	Shift + Opt + Up
向右选择一段文本	Shift + Ctrl + Down	Shift + Opt + Down
选择文本插入光标与当前行行首之间的文本	Shift + Home	Cmd + Shift + Left Arrow
选择文本插入光标与当前行行尾之间的文本	Shift + End	Cmd + Shift + Right Arrow
选择文本插入光标与文档开头之间的文本	Shift + Ctrl + Home	Cmd + Shift + Up Arrow or Cmd + Shift + Fn + Left Arrow
选择文本插入光标与文档结尾之间的文本	Shift + Ctrl + End	Cmd + Shift + Down Arrow or Cmd + Shift + Fn + Right Arrow
向上选择一个文本框	Shift + Page Up	Shift + Fn + Up Arrow
向下选择一个文本框	Shift + Page Down	Shift + Fn + Down Arrow
全选文本	Ctrl + A	Cmd + A
查找文本	Ctrl + F	Cmd + F
文本排版		

将所选文本设置为粗体	Ctrl + B	Cmd + B
将所选文本设置为斜体	Ctrl + I	Cmd + I
将所选文本加下划线	Ctrl + U	Cmd + U
将所选文本设置为上标	Ctrl + Shift + =	Cmd + Shift + =
将所选文本设置为下标	Ctrl + =	Cmd + =
文本编辑		
删除文本插入光标左边的字符	Backspace	Backspace
删除文本插入光标右边的字符	Delete	Fn + Backspace
删除文本插入光标右边的字	Ctrl + Del	Cmd + Backspace
删除文本插入光标左边的字	Ctrl + Backspace	Cmd + Fn + Backspace
增加缩进量	Tab	Tab
减少缩进量	Shift + Tab	Shift + Tab
复制文本	Ctrl + C	Cmd + C
查找替换文本	Ctrl + H	Cmd + F
粘贴文本	Ctrl + V	Cmd + V

剪切文本	Ctrl + X	Cmd + X
重做文本	Ctrl + Y	Shift + Cmd + Z
撤销文本	Ctrl + Z	Cmd + Z

Web 浏览器

描述	Windows	Mac OS
Navigation		
向下滚动框	Space or Page Down	Space or Fn + Down Arrow
向上滚动框	Shift + Space or Page Up	Shift + Space or Fn + Up Arrow
定位到页面底部	End	Cmd + Down Arrow
定位到页面头部	Home	Cmd + Up Arrow
回退	Alt + Left Arrow or Backspace	Cmd + Left Arrow
前进	Alt + Right Arrow or Shift + Backspace	Cmd + Right Arrow

刷新网页	F5	Cmd + R
刷新网页（无缓存）	Ctrl + F5	Cmd + Shift + R
停止	Esc	Esc
全屏切换	F11	Cmd + Shift + F
放大	Ctrl + +	Cmd + +
缩小	Ctrl + -	Cmd + -
重置 100%（默认）	Ctrl + 0	Cmd + 0
打开主页	Alt + Home	Option + Home or Option + Fn + Left Arrow
查找文本	Ctrl + F	Cmd + F
标签/窗口管理		
打开一个新的标签页	Ctrl + T	Cmd + T
关闭当前标签页	Ctrl + W	Cmd + W
关闭所有标签页	Ctrl + Shift + W	Cmd + Q
关闭除当前标签页以外的其它标签页	Ctrl + Alt + F4	Cmd + Opt + W

定位到下一个标签页	Ctrl + Tab	Control + Tab or Cmd + Shift + Right Arrow
定位到上一个标签页	Ctrl + Shift + Tab	Shift + Control + Tab or Cmd + Shift + Left Arrow
定位到指定编号的标签页	Ctrl + 1-8	Cmd + 1-8
定位到最后一个标签页	Ctrl + 9	Cmd + 9
重新打开最后一个关闭的标签页	Ctrl + Shift + T	Cmd + Shift + T
打开一个新窗口	Ctrl + N	Cmd + N
关闭当前窗口	Alt + F4	Cmd + W
定位到下一个窗口	Alt + Tab	Cmd + Tab
定位到上一个窗口	Alt + Shift + Tab	Cmd + Shift + Tab
重新打开最后一个关闭的窗口	Ctrl + Shift + N	
在背景的标签页中打开链接	Ctrl + Click	Cmd + Click
在前景的标签页中打开链接	Ctrl + Shift + Click	Cmd + Shift + Click
打印当前网页	Ctrl + P	Cmd + P
保存当前网页	Ctrl + S	Cmd + S

地址栏		
在工具栏、搜索栏和页面元素间进行循环定位	Tab	Tab
定位到浏览器的地址栏	Ctrl + L or Alt + D	Cmd + L
聚焦并选中浏览器的搜索栏	Ctrl + E	Cmd + E / Cmd + K
在新的标签页打开地址栏位置	Alt + Enter	Cmd + Enter
显示一系列之前访问过的地址	F4	
在地址栏输入文本的开头增加"www."，结尾增加".com"。（比如，地址栏输入"baidu"，然后按下 Ctrl + Enter 键，即打开"www.baidu.com"。）	Ctrl + Enter	Cmd + Enter or Control + Enter
书签		
打开书签菜单	Ctrl + B	Cmd + B
将当前页加为书签	Ctrl + D	Cmd + Opt + B or Cmd + Shift + B
打开浏览历史记录	Ctrl + H	Cmd + Shift + H or Cmd + Y
打开下载历史记录	Ctrl + J	Cmd + J or Cmd + Shift + J

屏幕截图

描述	Windows	Mac OS
将整个屏幕的截图保存为文件		Cmd + Shift + 3
复制整个屏幕的截图到剪贴板	PrtScr (Print Screen) or Ctrl + PrtScr	Cmd + Ctrl + Shift + 3
将窗口截图保存为文件		Cmd + Shift + 4, then Space
复制窗口截图到剪贴板	Alt + PrtScr	Cmd + Ctrl + Shift + 4, then Space
复制选定区域截图到剪贴板		Cmd + Ctrl + Shift + 4
将选定区域截图保存为文件		Cmd + Shift + 4

注释： 由于不同的键盘设置，一些快捷键可能并不适用于所有用户。

[Px/Em 换算工具](#)

[HTML <!--...--> 注释标签](#)

点我分享笔记

HTML 标签

HTML `<!--...-->` 注释标签

实例

HTML 注释：

```
<!-- 这是一个注释，注释在浏览器中不会显示 --> <p>这是一个段落</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<!--...-->` 注释标签。

标签定义及使用说明

`<!--...-->` 注释标签用来在源文档中插入注释。注释不会在浏览器中显示。

您可使用注释对您的代码进行解释，这样做有助于您在以后的时间对代码的编辑。特别是代码量很大的情况下很有用。

您也可以在注释内容存储针对程序所定制的信息。在这种情况下，这些信息对用户是不可见的，但是对程序来说是可用的。一个好的习惯是把注释或样式元素放入注释文本中，这样就可避免不支持脚本或样式的老浏览器把它们显示为纯文本。

```
<script type="text/javascript"> <!-- function displayMsg() { alert("Hello World!") } //--> </script>
```

注释：命令行最后的两个正斜杠（//）是 JavaScript 注释符号。这确保了 JavaScript 不会执行 `-->` 标签。

除了在源文档中有非常明显的作用外，许多 Web 服务器也利用注释来实现文档服务端软件特有的特性。这些服务器可以扫描文档，从传统的 HTML/XHTML 注释中找到特定的字符序列，然后再根据嵌在注释中的命令采取相应的动作。这些动作可能是简单的包括其他文件中的文本（即所谓的服务器端包含，server-inside include），也可能是复杂地执行其他命令去动态生成文档的内容。

HTML 4.01 与 HTML5 之间的差异

无。

标准属性

<!--...--> 注释标签不支持任何标准属性。

如需更多有关 HTML 标准属性的信息，请访问 [标准属性](#)。

事件属性

<!--...--> 注释标签不支持任何事件属性。

如需更多有关 HTML 事件属性的信息，请访问 [事件属性](#)。

[HTML <!--...--> 注释标签](#)

[HTML <a> 标签](#)

HTML <!DOCTYPE> 声明

<!DOCTYPE> 声明的目的是让浏览器能够正确地渲染页面，声明的形式如下：

```
<!DOCTYPE html>
```

以上代码是 HTML5 的文档类型声明，它告诉浏览器当前页面是使用 HTML5 规范编写的，HTML5 是最新的 HTML 版本，拥有更多的功能和优化，因此推荐在新的 Web 页面中使用它。

在 HTML 文档中，`<!DOCTYPE>` 声明通常是文档的第一行，位于 `<html>` 标签之前。

实例

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容..... </body> </html>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<!DOCTYPE>` 声明。

标签定义及使用说明

在 HTML 中，`<!DOCTYPE>` 声明是文档类型声明（Document Type Declaration）的缩写，用于指示浏览器当前页面使用的 HTML 版本。

`<!DOCTYPE>` 声明位于文档中的最前面的位置，处于 `<html>` 标签之前。

`<!DOCTYPE>` 声明不是一个 HTML 标签，因此不需要关闭标签。。

在 HTML 4.01 及之前版本中，`<!DOCTYPE>` 声明可能会指向 DTD（文档类型定义）文件，它定义了文档的结构和元素规则。因为 HTML 4.01 是基于 SGML（Standard Generalized Markup Language 标准通用标记语言）。DTD 指定了标记语言的规则，确保了浏览器能够正确的渲染内容。

HTML5 不是基于 SGML，因此不再依赖 DTD 文件，而是使用更简单的声明。

提示：总是给您的 HTML 文档添加 `<!DOCTYPE>` 声明，确保浏览器能够预先知道文档类型。

HTML 4.01 与 HTML5 之间的差异

HTML 4.01 规定了三种不同的 <!DOCTYPE> 声明，分别是：Strict、Transitional 和 Frameset。

HTML5 中仅规定了一种：

```
<!DOCTYPE html>
```

HTML 元素和 Doctypes

参阅 [HTML 元素与合法的 Doctype](#)，看看每一个 HTML 元素都出现在哪一种 Doctype 中。

提示和注释

注释：<!DOCTYPE> 标签没有结束标签。

提示：<!DOCTYPE> 声明不区分大小写。

提示：使用 [W3C 的验证](#) 检查您是否编写了一个带有正确 DTD 的合法的 HTML / XHTML 文档！

常见的 DOCTYPE 声明

HTML 5

```
<!DOCTYPE html>
```

HTML 4.01 Strict

这个 DTD 包含所有 HTML 元素和属性，但不包括表象或过时的元素（如 font）。框架集是不允许的。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

HTML 4.01 Transitional

这个 DTD 包含所有 HTML 元素和属性，包括表象或过时的元素（如 font ）。框架集是不允许的。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

HTML 4.01 Frameset

这个 DTD 与 HTML 4.01 Transitional 相同，但是允许使用框架集内容。

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
```

XHTML 1.0 Strict

这个 DTD 包含所有 HTML 元素和属性，但不包括表象或过时的元素（如 font ）。框架集是不允许的。结构必须按标准格式的 XML 进行书写。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

XHTML 1.0 Transitional

这个 DTD 包含所有 HTML 元素和属性，包括表象或过时的元素（如 font ）。框架集是不允许的。结构必须按标准格式的 XML 进行书写。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

XHTML 1.0 Frameset

这个 DTD 与 XHTML 1.0 Transitional 相同，但是允许使用框架集内容。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

XHTML 1.1

这个 DTD 与 XHTML 1.0 Strict 相同，但是允许您添加模块（例如为东亚语言提供 ruby 支持）。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

[HTML <!--...--> 注释标签](#)

[HTML <a> 标签](#)

1 篇笔记 写笔记

1. 度ei它
h28***8@163.com
参考地址

388

<!DOCTYPE html> 是 HTML5 中唯一的 doctype, 也被视作将网页 "升级" 到 HTML5 的第一步。

很多国外网站的 <!DOCTYPE html> 和 <HEAD> 之间都会有一段注释,如:

```
<!--[if IE 6 ]><html class="ie ielt9 ielt8 ielt7 ie6" lang="en-US"><![endif]-->
<!--[if IE 7 ]><html class="ie ielt9 ielt8 ie7" lang="en-US"><![endif]-->
<!--[if IE 8 ]><html class="ie ielt9 ie8" lang="en-US"><![endif]-->
<!--[if IE 9 ]><html class="ie ie9" lang="en-US"><![endif]-->
<!--[if (gt IE 9)|!(IE)]><!--><html lang="en-US"><!--<![endif]-->
```

该代码作用于 css, 来写一些针对 IE 各版本的样式差异。

先判断用户用的哪个 IE 版本, 然后在标签上加上该版本的 class, 这样可以方便 hack。

css 文件是这样写的:

```
.ie6 xxx {};  
.ie7 xxx {};
```

这是目前最好的 hack 方式之一。

度ei它

度ei它
h28***8@163.com
参考地址

6 年前 (2018-05-07)

[HTML <!DOCTYPE> 声明](#)

[HTML <abbr> 标签](#)

HTML <a> 标签

实例

链接到菜鸟教程：

```
<a href="https://www.runoob.com">访问菜鸟教程!</a>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 <a> 标签。

标签定义及使用说明

<a> 标签定义超链接，用于从一个页面链接到另一个页面。

<a> 元素最重要的属性是 href 属性，它指定链接的目标。

在所有浏览器中，链接的默认外观如下：

- 未被访问的链接带有下划线而且是蓝色的

- 已被访问的链接带有下划线而且是紫色的
- 活动链接带有下划线而且是红色的

提示和注释

提示：如果没有使用 href 属性，则不能使用 hreflang、media、rel、target 以及 type 属性。

提示：通常在当前浏览器窗口中显示被链接页面，除非规定了其他 target。

提示：请使用 CSS 来改变链接的样式。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，<a> 标签既可以是超链接，也可以是锚。在 HTML5 中，<a> 标签是超链接，但是假如没有 href 属性，它仅仅是超链接的一个占位符。

HTML5 有一些新的属性，同时不再支持一些 HTML 4.01 的属性。

属性

New：HTML5 中的新属性。

属性	值	描述
<u>charset</u>	<i>char_encoding</i>	HTML5 不支持。规定目标 URL 的字符编码。
<u>coords</u>	<i>coordinates</i>	HTML5 不支持。规定链接的坐标。

<u>download</u> New	<i>filename</i>	指定下载链接
<u>href</u>	<i>URL</i>	规定链接的目标 URL。
<u>hreflang</u>	<i>language_code</i>	规定目标 URL 的基准语言。仅在 href 属性存在时使用。
<u>media</u> New	<i>media_query</i>	规定目标 URL 的媒介类型。默认值：all。仅在 href 属性存在时使用。
<u>name</u>	<i>section_name</i>	HTML5 不支持。规定锚的名称。
<u>rel</u>	alternate author bookmark help license next nofollow noreferrer prefetch prev search tag	规定当前文档与目标 URL 之间的关系。仅在 href 属性存在时使用。
<u>rev</u>	<i>text</i>	HTML5 不支持。规定目标 URL 与当前文档之间的关系。
<u>shape</u>	default rect	HTML5 不支持。规定链接的形状。

	circle poly	
target	_blank _parent _self _top <i>framename</i>	规定在何处打开目标 URL。仅在 href 属性存在时使用。 <ul style="list-style-type: none"> • _blank: 新窗口打开。 • _parent: 在父窗口中打开链接。 • _self: 默认, 当前页面跳转。 • _top: 在当前窗体打开链接, 并替换当前的整个窗体(框架页)。
type 	<i>MIME_type</i>	规定目标 URL 的 MIME 类型。仅在 href 属性存在时使用。 注: MIME = Multipurpose Internet Mail Extensions。

全局属性

<a> 标签支持 [HTML 的全局属性](#)。

事件属性

<a> 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[创建超级链接](#)

本例演示如何在 HTML 文档中创建链接。

[将图像作为链接](#)

本例演示如何使用图像作为链接。

[在新的浏览器窗口打开链接](#)

本例演示如何在新窗口打开一个页面，这样的话访问者就无需离开您的站点了。

[创建电子邮件链接](#)

本例演示如何链接到一个邮件。（本例在安装邮件客户端程序后才能工作。）

[创建电子邮件链接 2](#)

本例演示更加复杂的邮件链接。

[使用锚跳转到同一个页面的不同位置](#)

本例演示如何使用锚的 id 属性跳转到页面的不同位置（HTML5 不支持 name 属性）。

相关文章

HTML 教程: [HTML 链接](#)

HTML DOM 参考手册: [Anchor 对象](#)

[HTML <!DOCTYPE> 声明](#)

[HTML <abbr> 标签](#)

1 篇笔记 写笔记

1. 猫猫单
221***6201@qq.com

设置 target 属性时，top 与 parent 的打开方式十分类似，需仔细区分。

比如网 A 中镶嵌了 iframe 网页 B，网页 B 又镶嵌了 iframe 网页 C。

- o 如果网页 C 中连接设置 **target=_parent**，则跳转将网页 B 去掉直接在 A 中嵌入网页 C 中链接页面。
- o 如果网页 C 中 **target=_top**，则直接跳出所有 iframe 框架，直接转向 C 中链接页面。

猫猫单

猫猫单

221***6201@qq.com

5 年前 (2019-09-21)

[HTML <a> 标签](#)

[HTML <acronym> 标签](#)

HTML <abbr> 标签

实例

被标记的缩写词如下：

The<abbr title="World Health Organization">WHO</abbr> was founded in 1948.

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<abbr>` 标签。

注释：IE 6 或更早版本的 IE 浏览器不支持 `<abbr>` 标签。

标签定义及使用说明

`<abbr>` 标签用来表示一个缩写词或者首字母缩略词，如"WWW"或者"NATO"。

通过对缩写词语进行标记，您就能够为浏览器、拼写检查程序、翻译系统以及搜索引擎分度器提供有用的信息。

提示和注释

提示：在某些浏览器中，当您把鼠标移至带有 `<abbr>` 标签的缩写词/首字母缩略词上时，`<abbr>` 标签的 `title` 属性可被用来展示缩写词/首字母缩略词的完整版本。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<abbr>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<abbr>` 标签支持 [HTML 的事件属性](#)。

[HTML `<a>` 标签](#)

[HTML <acronym> 标签](#)

[HTML <abbr> 标签](#)

[HTML <address> 标签](#)

HTML <acronym> 标签 - HTML5 不支持

实例

被标记的首字母缩略词如下：

```
Can I get this <acronym title="as soon as possible">ASAP</acronym>?
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <acronym> 。

注释：IE 5.5 或更早版本的 IE 浏览器不支持 <acronym> 标签。

标签定义及使用说明

HTML5 不支持 <acronym> 标签。请使用 <abbr> 标签代替它。

<acronym> 标签定义首字母缩略词。

如果首字母缩略词是一个单词，则可以被读出来，例如 NATO, NASA, ASAP, GUI。

通过对只取首字母缩略词进行标记，您就能够为浏览器、拼写检查程序、翻译系统以及搜索引擎分度器提供有用的信息。

提示和注释

提示：在某些浏览器中，当您把鼠标移至带有 <acronym> 标签的首字母缩略词上时，<acronym> 标签的 title 属性可被用来展示首字母缩略词的完整版本。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <acronym> 标签，HTML 4.01 支持 <acronym> 标签。

标准属性

在 HTML 4.01 中，<acronym> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式

title	<i>text</i>	规定元素的额外信息
xml:lang	<i>language_code</i>	规定 XHTML 文档中元素内容的语言代码

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<acronym> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML <abbr> 标签](#)

[HTML <address> 标签](#)

点我分享笔记

[HTML <acronym> 标签](#)

[HTML <applet> 标签](#)

HTML <address> 标签

实例

Example.com 的联系信息：

```
<address> Written by <a href="mailto:webmaster@example.com">Jon Doe</a>.<br> Visit us at:<br> Example.com<br> Box 564, Disneyland<br> USA </address>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <address> 标签。

标签定义及使用说明

`<address>` 标签定义文档作者/所有者的联系信息。

如果 `<address>` 元素位于 `<body>` 元素内部，则它表示该文档作者/所有者的联系信息。

如果 `<address>` 元素位于 `<article>` 元素内部，则它表示该文章作者/所有者的联系信息。

`<address>` 元素的文本通常呈现为斜体。大多数浏览器会在该元素的前后添加换行。

提示和注释

提示： 不应该使用 `<address>` 标签来描述邮政地址，除非这些信息是联系信息的组成部分。

提示： `<address>` 元素通常被包含在 [<footer>](#) 元素的其他信息中。

HTML 4.01 与 HTML5 之间的差异

HTML 4.01 不支持 `<article>` 标签，因此在 HTML 4.01 中 `<address>` 标签总是定义文档的作者/所有者的联系信息。

全局属性

`<address>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<address>` 标签支持 [HTML 的事件属性](#)。

[HTML `<acronym>` 标签](#)

[HTML <applet> 标签](#)

点我分享笔记

[HTML <address> 标签](#)

[HTML <area> 标签](#)

HTML <applet> 标签 - HTML5 不支持

实例

一个嵌入的 Java applet:

```
<applet code="Bubbles.class" width="350" height="350"> Java applet that draws animated bubbles. </applet>
```

浏览器支持



注释：某些浏览器中依然存在对 <applet> 但是需要额外的插件和安装过程才能起作用。

标签定义及使用说明

HTML5 不支持 <applet> 标签。请使用 [<object>](#) 标签代替它。

在 HTML 4.01 中，<applet> 元素 [已废弃](#)。

<applet> 标签定义嵌入的 applet。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <applet> 标签，HTML 4.01 已废弃 <applet> 标签。

必需的属性

属性	值	描述
code	<i>URL</i>	规定 Java applet 的文件名。
object	<i>name</i>	规定了包含该 applet 的一系列版本的资源名称。

可选的属性

属性	值	描述
align	left right top bottom middle baseline	规定 applet 相对于周围元素的对齐方式。
alt	<i>text</i>	规定 applet 的替换文本。
archive	<i>URL</i>	规定档案文件的位置。

codebase	<i>URL</i>	规定 code 属性中指定的 applet 的基准 URL。
height	<i>pixels</i>	规定 applet 的高度。
hspace	<i>pixels</i>	定义围绕 applet 的水平间隔。
name	<i>name</i>	定义 applet 的名称（用在脚本中的）。
vspace	<i>pixels</i>	定义围绕 applet 的垂直间隔。
width	<i>pixels</i>	规定 applet 的宽度。

标准属性

在 HTML 4.01 中，<applet> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
id	<i>id</i>	规定元素的唯一 id
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<applet> 标签不支持任何事件属性。
如需完整的描述，请访问[事件属性](#)。

[HTML <address> 标签](#)

[HTML <area> 标签](#)

点我分享笔记

[HTML <applet> 标签](#)

[HTML <article> 标签](#)

HTML <area> 标签

实例

带有可点击区域的图像映射：

```
 <map name="planetmap">
<area shape="rect" coords="0,0,82,126" alt="Sun" href="sun.htm"> <area shape="circle" coords="90,58,3"
alt="Mercury" href="mercur.htm"> <area shape="circle" coords="124,58,8" alt="Venus" href="venus.htm"> </map>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<area>` 标签。

标签定义及使用说明

`<area>` 标签定义图像映射内部的区域（图像映射指的是带有可点击区域的图像）。

`<area>` 元素始终嵌套在 `<map>` 标签内部。

注释： `` 标签中的 `usemap` 属性与 `<map>` 元素中的 `name` 相关联，以创建图像与映射之间的关系。

HTML 4.01 与 HTML5 之间的差异

HTML5 提供了一些新属性，同时不再支持 HTML 4.01 中的某些属性。

HTML 与 XHTML 之间的差异

在 HTML 中，`<area>` 标签没有结束标签。

在 XHTML 中，`<area>` 标签必须正确地关闭。

属性

New：HTML5 中的新属性。

属性	值	描述
<code>alt</code>	<i>text</i>	规定区域的替代文本。如果使用 <code>href</code> 属性，则该属性是必需的。

<u>coords</u>	<i>coordinates</i>	规定区域的坐标。
<u>href</u>	<i>URL</i>	规定区域的目标 URL。
<u>hreflang</u> New	<i>language_code</i>	规定目标 URL 的语言。
<u>media</u> New	<i>media query</i>	规定目标 URL 是为何种媒介/设备优化的。默认：all。
<u>nohref</u>	<i>value</i>	HTML5 不支持。规定没有相关链接的区域。
<u>rel</u> New	alternate author bookmark help license next nofollow noreferrer prefetch prev search tag	规定当前文档与目标 URL 之间的关系。
<u>shape</u>	default rect circle	规定区域的形状。

	poly	
target	_blank _parent _self _top <i>framename</i>	规定在何处打开目标 URL。
type New	<i>MIME_type</i>	规定目标 URL 的 MIME 类型。 注：MIME = Multipurpose Internet Mail Extensions。

全局属性

<area> 标签支持 [HTML 的全局属性](#)。

事件属性

<area> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML DOM 参考手册：[Area 对象](#)

[HTML <applet> 标签](#)

[HTML <article> 标签](#)

点我分享笔记

[HTML <area> 标签](#)

[HTML <aside> 标签](#)

HTML <article> 标签

实例

```
<article> <h1>Internet Explorer 9</h1> <p> Windows Internet Explorer 9(缩写为 IE9 )在2011年3月14日 21:00 发布。</p> </article>
```

[尝试一下 »](#)

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari 都支持 <article> 标签。

注释： IE 8 或更早版本的 IE 浏览器不支持 <article> 标签。

标签定义及使用说明

<article> 标签定义独立的内容。

<article> 标签定义的内容本身必须是有意义的且必须是独立于文档的其余部分。

<article> 的潜在来源：

- 论坛帖子

- 博客文章
- 新闻故事
- 评论

HTML 4.01 与 HTML5 之间的差异

<article> 标签是 HTML5 的新标签。

全局属性

<article> 标签支持 [HTML 的全局属性](#)。

事件属性

<article> 标签支持 [HTML 的事件属性](#)。

[HTML <area> 标签](#)

[HTML <aside> 标签](#)

点我分享笔记

[HTML <article> 标签](#)

[HTML <audio> 标签](#)

HTML `<aside>` 标签

实例

```
<p>My family and I visited The Epcot center this summer.</p> <aside> <h4>Epcot Center</h4> <p>The Epcot Center is a theme park in Disney World, Florida.</p> </aside>
```

[尝试一下 »](#)

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari 都支持 `<aside>` 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 `<aside>` 标签。

标签定义及使用说明

`<aside>` 标签定义 `<article>` 标签外的内容。

`aside` 的内容应该与附近的内容相关。

HTML 4.01 与 HTML5 之间的差异

`<aside>` 标签是 HTML5 的新标签。

提示和注释

提示：<aside> 的内容可用作文章的侧栏。

全局属性

<aside> 标签支持 [HTML 的全局属性](#)。

事件属性

<aside> 标签支持 [HTML 的事件属性](#)。

[HTML <article> 标签](#)

[HTML <audio> 标签](#)

点我分享笔记

[HTML <aside> 标签](#)

[HTML 标签](#)

HTML <audio> 标签

实例

播放声音：

```
<audio controls> <source src="horse.ogg" type="audio/ogg"> <source src="horse.mp3" type="audio/mpeg"> 您的浏览器不支持 audio 元素。</audio>
```

尝试一下 »

浏览器支持

表格中的数字表示支持该标签的第一个浏览器版本号。

元素					
<audio>	4.0	9.0	3.5	4.0	10.5

标签定义及使用说明

<audio> 标签定义声音，比如音乐或其他音频流。
目前，<audio> 元素支持的 3 种文件格式：MP3、Wav、Ogg。

浏览器	MP3	Wav	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

HTML 4.01 与 HTML5 之间的差异

<audio> 标签是 HTML5 的新标签。

提示和注释

提示：可以在 <audio> 和 </audio> 之间放置文本内容，这些文本信息将会被显示在那些不支持 <audio> 标签的浏览器中。

属性

New：HTML5 中的新属性。

属性	值	描述
<u>autoplay</u> New	autoplay	如果出现该属性，则音频在就绪后马上播放。
<u>controls</u> New	controls	如果出现该属性，则向用户显示音频控件（比如播放/暂停按钮）。
<u>loop</u> New	loop	如果出现该属性，则每当音频结束时重新开始播放。
<u>muted</u> New	muted	如果出现该属性，则音频输出为静音。

preload New	auto metadata none	规定当网页加载时，音频是否默认被加载以及如何被加载。
src New	<i>URL</i>	规定音频文件的 URL。

全局属性

<audio> 标签支持 [HTML 的全局属性](#)。

事件属性

<audio> 标签支持 [HTML 的事件属性](#)。

[HTML <aside> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <audio> 标签](#)

[HTML <base> 标签](#)

HTML 标签

实例

<p>这是一个普通的文本- 这是一个加粗文本。</p>

尝试一下 »

浏览器支持



所有主流浏览器都支持 标签。

标签定义及使用说明

 标签定义粗体的文本。

HTML 4.01 与 HTML5 之间的差异

无。

提示和注释

注释：根据 HTML 5 的规范， 标签应该做为最后的选择，只有在没有其他标记比较合适时才使用它。HTML 5 规范声明：标题应该用 <h1> - <h6> 标签表示，被强调的文本应该用 标签表示，重要的文本应该用 标签表示，被标记的或者高亮显示的文本应该用 <mark> 标签表示。

提示：您也可以使用 CSS 的 "font-weight" 属性设置粗体文本。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。

[HTML <audio> 标签](#)

[HTML <base> 标签](#)

1 篇笔记 写笔记

1. 彬少
177***4238@qq.com
参考地址

112

 的效果是加粗，(强调)的效果也是加粗，有什么区别吗？

查了一下资料发现：strong 是 web 标准中 xhtml 的标签，strong 的意思是 "强调"；b 是 html 的，b 的意思是 bold（粗体）。

为什么用 strong 代替 b？其实这个问题不妨改问：xhtml 和 html 有什么不同，为什么要用 xhtml 代替 html？

简单地说：web 标准主张 xhtml 不涉及具体的表现形式，"强调"可以用加粗来强调，也可以用其它方式来强调，比如下划线，比如字体加大，比如红色，等等，可以通过 css 来改变 strong 的具体表现，这就是为什么 b 不能替代 strong。

[彬少](#)

彬少
177***4238@qq.com

[HTML 标签](#)

[HTML <basefont> 标签](#)

HTML <base> 标签

实例

规定页面上所有链接的默认 URL 和默认目标:

```
<head> <base href="http://www.runoob.com/images/" target="_blank"> </head> <body>  <a href="http://www.runoob.com">runoob.com</a> </body>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <base> 标签。

标签定义及使用说明

<base> 标签为页面上的所有的相对链接规定默认 URL 或默认目标。

在一个文档中，最多能使用一个 <base> 元素。<base> 标签必须位于 <head> 元素内部。

提示和注释

提示：请把 <base> 标签排在 <head> 元素中第一个元素的位置，这样 head 区域中其他元素就可以使用 <base> 元素中的信息了。

注释：如果使用了 <base> 标签，则必须具备 href 属性或者 target 属性或者两个属性都具备。

HTML 4.01 与 HTML5 之间的差异

无。

HTML 与 XHTML 之间的差异

在 HTML 中，<base> 标签没有结束标签。

在 XHTML 中，<base> 标签必须被正确地关闭。

属性

属性	值	描述
href	<i>URL</i>	规定页面中所有相对链接的基准 URL。
target	<i>_blank</i> <i>_parent</i> <i>_self</i> <i>_top</i> <i>framename</i>	规定页面中所有的超链接和表单在何处打开。该属性会被每个链接中的 target 属性覆盖。

全局属性

<base> 标签支持 [HTML 的全局属性](#)。

事件属性

<base> 标签不支持任何的事件属性。

相关文章

HTML DOM 参考手册: [Base 对象](#)

[HTML 标签](#)

[HTML <basefont> 标签](#)

点我分享笔记

[HTML <base> 标签](#)

[HTML <bdi> 标签](#)

HTML <basefont> 标签 - HTML5 不支持

实例

规定页面上文本的默认颜色和大小：

```
<head>
<basefont color="red" size="5" />
</head>

<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
</body>
```

[尝试一下 »](#)

浏览器支持



只有 IE 9 和更早版本的 IE 浏览器支持 `<basefont>` 标签。应该避免使用该标签。

标签定义及使用说明

HTML5 不支持 `<basefont>` 标签。请用 CSS 代替。

在 HTML 4.01 中，`<basefont>` 元素 已废弃。

`<basefont>` 标签定义文档中所有文本的默认颜色、大小和字体。

提示和注释

提示：使用 CSS 为文档中的文本规定默认颜色、大小和字体。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <basefont> 标签，HTML 4.01 已废弃 <basefont> 标签。

可选的属性

属性	值	描述
color	<i>color</i>	HTML5 不支持。HTML 4.01 已废弃。规定文档中文本的默认颜色。
face	<i>font_family</i>	HTML5 不支持。HTML 4.01 已废弃。规定文档中文本的默认字体。
size	<i>number</i>	HTML5 不支持。HTML 4.01 已废弃。规定文档中文本的默认大小。

标准属性

在 HTML 4.01 中，<basefont> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id

lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<basefont> 标签不支持任何事件属性。

如需完整的描述，请访问[事件属性](#)。

[HTML <base> 标签](#)

[HTML <bdi> 标签](#)

点我分享笔记

[HTML <basefont> 标签](#)

[HTML <bdo> 标签](#)

HTML <bdi> 标签

实例

将用户名从周围的文本方向设置中隔离出来：


```
<ul> <li>用户 <bdi>hrefs</bdi>: 60 分</li> <li>用户 <bdi>jdoe</bdi>: 80 分</li> <li>用户 <bdi>إيان</bdi>: 90 分</li> </ul>
```

[尝试一下 »](#)

浏览器支持

元素					
<bdi>	Yes	不支持	Yes	Yes	Yes

标签定义及使用说明

bdi 指的是 bidi 隔离（Bi-directional Isolation）。

<bdi> 标签允许您设置一段文本，使其脱离其父元素的文本方向设置。

在发布用户评论或其他您无法完全控制的内容时，该标签很有用。

HTML 4.01 与 HTML5 之间的差异

<bdi> 标签是 HTML5 的新标签。

全局属性

<bdi> 标签支持 [HTML 的全局属性](#)。

事件属性

<bdi> 标签支持 [HTML 的事件属性](#)。

[HTML <basefont> 标签](#)

[HTML <bdo> 标签](#)

1 篇笔记 写笔记

1. 没有水的木
257***1053@qq.com
参考地址

40

HTML <bdi> 元素 (双向隔离元素) 会隔离可能以不同方向进行格式化的外部文本。

当不知道是从什么方向嵌入文本，如来自于数据库的文本（有起数据库的文本方向）的时候，该元素是十分有用的。

注意：尽管同样的显示效果可以通过使用 CSS 规则 [unicode-bidi](#)：隔离或者其他文本格式化元素，但语义信息只能

通过<bdi>元素传递。特别是，当浏览器允许忽略 CSS 样式时，在这种情况下，使用<bdi>仍然可以保证文本正确显示，而使用 CSS 样式来传递语义时就显得毫无用处。

示例：

```
<p dir="ltr">This arabic word <bdi>ARABIC_PLACEHOLDER</bdi> is automatically displayed right-to-left.</p>
```

结果：

This arabic word REDLOHECALP_CIBARA is automatically displayed right-to-left.

没有水的木

没有水的木

257***1053@qq.com

参考地址

7 年前 (2017-09-08)

[HTML <bdi> 标签](#)

[HTML <big> 标签](#)

HTML <bdo> 标签

实例

指定文本方向：

<p>该段落文字从左到右显示。</p> <p><bdo dir="rtl">该段落文字从右到左显示。</bdo></p>

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <bdo> 标签。

标签定义及使用说明

bdo 指的是 bidi 覆盖（Bi-Directional Override）。
<bdo> 标签用来覆盖默认的文本方向。

HTML 4.01 与 HTML5 之间的差异

无。

属性

属性	值	描述
dir	ltr rtl	必需。规定 <bdo> 元素内的文本方向。

全局属性

<bdo> 标签支持 [HTML 的全局属性](#)。

事件属性

<bdo> 标签支持 [HTML 的事件属性](#)。

[HTML <bdi> 标签](#)

[HTML <big> 标签](#)

2 篇笔记 写笔记

1. lodedi
598***855@qq.com

229

ltr 是英文 left to right 的首字母缩写，即从左到右。

同理

rtl 是英文 right to left 即从右到左。

lodedi

lodedi
598***855@qq.com

7 年前 (2017-06-28)

2. WH-KING
104***5930@qq.com

50

bdo 元素一般用于把一段文本的方向规定为与周围文本的自然方向相反的方向。方向由**必需属性 dir** 指定。**bdo** 元素很少使用，只用于某些**多语言文档**。在这种文档中，可能有某一段文本使用的语言的阅读方式与文档中其他部分使用的语言的阅读方式不同。

WH-KING

WH-KING
104***5930@qq.com

7 年前 (2017-09-08)

[HTML <bdo> 标签](#)

[HTML <blockquote> 标签](#)

HTML <big> 标签 - HTML5 不支持

实例

让文本比常规的字体大一号：

```
<p><big>这个文本比较大。</big></p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <big> 标签。

标签定义及使用说明

HTML5 不支持 <big> 标签。请用 CSS 代替。

<big> 标签用来制作更大的文本。

提示和注释

提示：在文档中使用 CSS 来规定[文本大小](#)。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <big> 标签，HTML 4.01 支持 <big> 标签。

标准属性

在 HTML 4.01 中，<big> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息
xml:lang	<i>language_code</i>	规定 XHTML 文档中元素内容的语言代码

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<big> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML <bdo> 标签](#)

[HTML <blockquote> 标签](#)

1 篇笔记 写笔记

1. WH-KING

- o 浏览器显示包含在 `big` 元素中的文字时，其字体将比周围的文字大 **1 号**，直到上限 **7 号** 字体。也就是说，如果文字已经是最大号字体，那么 `<big>` 标签将不起作用。
- o 可以嵌套 `<big>` 标签使用，每一对 `<big>` 标签都可以使字体放大 1 号，直到上限 7 号字体。
- o 对于那些不支持 `<big>` 标签的浏览器来说，它经常被解释成粗体。

[HTML <big> 标签](#)

[HTML <body> 标签](#)

HTML `<blockquote>` 标签

实例

定义一个摘自另一个源的块引用：

```
<blockquote cite="http://www.worldwildlife.org/who/index.html">
```

For 50 years, WWF has been protecting the future of nature. The world's leading conservation organization, WWF works in 100 countries and is supported by 1.2 million members in the United States and close to 5 million globally.

```
</blockquote>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<blockquote>` 标签。

标签定义及使用说明

`<blockquote>` 标签定义摘自另一个源的块引用。

浏览器通常会对 `<blockquote>` 元素进行缩进。

提示和注释

提示：如果标记是不需要段落分隔的短引用，请使用 `<q>`。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<blockquote>` 标签定义一段长引用。

在 HTML5 中，`<blockquote>` 标签定义摘自另一个源的块引用。

HTML 与 XHTML 之间的差异

注释：如需把页面作为 XHTML 进行验证，那么 `<blockquote>` 元素必须包含块级元素，比如：

```
<blockquote>
<p>这是一个长引用，这是一个长引用。</p>
</blockquote>
```

属性

属性	值	描述
cite	<i>URL</i>	规定引用的来源。

全局属性

<blockquote> 标签支持 [HTML 的全局属性](#)。

事件属性

<blockquote> 标签支持 [HTML 的事件属性](#)。

[HTML <big> 标签](#)

[HTML <body> 标签](#)

1 篇笔记 写笔记

1. WH-KING
104***5930@qq.com

- o blockquote 元素中一般嵌套 p 元素，用于标记被引用的文本，这些引用文本并不是直接位于 blockquote 元素中。

- o 如果引文来自在线资源（包括自己网站中的其他地方），那么可以在<blockquote>标签的 cite（引用）属性中指明原始来源的 URL。
- o 通常浏览器会把 blockquote 元素呈现为一段左右两侧缩进(40px)的文本。

2. [HTML <blockquote> 标签](#)

3. [HTML
 标签](#)

4. HTML <body> 标签

5.

6. 实例

7. 一个简单的 HTML 文档，包含尽可能少的必需的标签：

```
8. <!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容.....</body> </html>
```

9.

[尝试一下 »](#)

10.

11. 浏览器支持



13. 所有主流浏览器都支持 <body> 标签。

14.

15. 标签定义及使用说明

- 16. <body> 标签定义文档的主体。
- 17. <body> 元素包含文档的所有内容（比如文本、超链接、图像、表格和列表等等）。
- 18.

19. HTML 4.01 与 HTML5 之间的差异

- 20. 在 HTML 5 中，删除了所有 body 元素的"呈现属性"。
- 21. 在 HTML 4.01 中，所有 body 元素的"呈现属性" 已废弃。
- 22.

23. 属性

属性	值	描述
alink	color	HTML5 不支持。 HTML 4.01 已废弃。 规定文档中活动链接的颜色。
background	URL	HTML5 不支持。 HTML 4.01 已废弃。 规定文档的背景图像。
bgcolor	color	HTML5 不支持。 HTML 4.01 已废弃。 规定文档的背景颜色。
link	color	HTML5 不支持。 HTML 4.01 已废弃。 规定文档中未访问链接的颜色。
text	color	HTML5 不支持。 HTML 4.01 已废弃。 规定文档中所有文本的颜色。
vlink	color	HTML5 不支持。 HTML 4.01 已废弃。 规定文档中已被访问链接的颜色。

- 24.
- 25.

26. 全局属性

27. <body> 标签支持 [HTML 的全局属性](#)。

28.

29. 事件属性

30. <body> 标签支持 [HTML 的事件属性](#)。

31.

32. 相关文章

33. HTML 教程: [HTML 元素](#)

34. HTML DOM 参考手册: [Body 对象](#)

35. [HTML <blockquote> 标签](#)

36. [HTML
 标签](#)

37. [点我分享笔记](#)

38. [HTML <body> 标签](#)

39. [HTML <button> 标签](#)

40. HTML
 标签

41.

42. 实例

43. 以下代码标记一个换行：

44. `<p>` 使用 `br` 元素`
`在文本中`
`换行。 `</p>`

45.

尝试一下 »

46.

47. 浏览器支持

48. 

49. 所有主流浏览器都支持 `
` 标签。

50.

51. 标签定义及使用说明

52. `
` 标签插入一个简单的换行符。

53. `
` 标签是一个空标签，意味着它没有结束标签。

54.

55. 提示和注释

56. **提示：** 在写地址信息或者写诗词时 `
` 标签非常有用。

57. **注释：** 请使用 `
` 标签来输入空行，而不是分割段落。

58.

59.HTML 4.01 与 HTML5 之间的差异

60. 无。

61.

62.HTML 与 XHTML 之间的差异

63. 在 HTML 中，
 标签没有结束标签。

64. 在 XHTML 中，
 标签必须被正确地关闭，比如这样：
。

65.

66.全局属性

67.
 标签支持 [HTML 的全局属性](#)。

68.

69.事件属性

70.
 标签支持 [HTML 的事件属性](#)。

71.

72.相关文章

73. HTML 教程： [HTML 段落](#)

74. [HTML <body> 标签](#)

75. [HTML <button> 标签](#)

76. 点我分享笔记

77. [HTML
 标签](#)

78. [HTML <canvas> 标签](#)

79. HTML <button> 标签

80.

81. 实例

82. 以下代码标记一个按钮:

83. `<button type="button">点我!</button>`

84.

尝试一下 »

85.

86. 浏览器支持

87. 

88. 所有主流浏览器都支持 <button> 标签。

89.

90. 标签定义及使用说明

91. <button> 标签定义一个按钮。
92. 在 <button> 元素内部，您可以放置内容，比如文本或图像。这是该元素与使用 <input> 元素创建的按钮之间的不同之处。
93. 提示：请始终为 <button> 元素规定 type 属性。不同的浏览器对 <button> 元素的 type 属性使用不同的默认值。
- 94.

95. 提示和注释

96. 注释：如果在 HTML 表单中使用 <button> 元素，不同的浏览器可能会提交不同的按钮值。请使用 <input> 在 HTML 表单中创建按钮。
- 97.

98. HTML 4.01 与 HTML5 之间的差异

99. HTML5 中的新属性：autofocus、form、formaction、formenctype、formmethod、formnovalidate 以及 formtarget。
- 100.

101. 属性

102. **New**：HTML5 中的新属性。

属性	值	描述
<u>autofocus</u> New	autofocus	规定当页面加载时按钮应当自动地获得焦点。
<u>disabled</u>	disabled	规定应该禁用该按钮。

<u>form</u> New	<i>form_id</i>	规定按钮属于一个或多个表单。
<u>formaction</u> New	<i>URL</i>	规定当提交表单时向何处发送表单数据。覆盖 form 元素的 action 属性。该属性与 type="submit" 配合使用。
<u>formenctype</u> New	application/x-www-form-urlencoded multipart/form-data text/plain	规定在向服务器发送表单数据之前如何对其进行编码。覆盖 form 元素的 enctype 属性。该属性与 type="submit" 配合使用。
<u>formmethod</u> New	get post	规定用于发送表单数据的 HTTP 方法。覆盖 form 元素的 method 属性。该属性与 type="submit" 配合使用。
<u>formnovalidate</u> New	formnovalidate	如果使用该属性，则提交表单时不进行验证。覆盖 form 元素的 novalidate 属性。该属性与 type="submit" 配合使用。
<u>formtarget</u> New	_blank _self _parent _top <i>framename</i>	规定在何处打开 action URL。覆盖 form 元素的 target 属性。该属性与 type="submit" 配合使用。
<u>name</u>	<i>name</i>	规定按钮的名称。
<u>type</u>	button reset	规定按钮的类型。

	submit	
value	<i>text</i>	规定按钮的初始值。可由脚本进行修改。

103.

104.

105. 全局属性

106. <button> 标签支持 [HTML 的全局属性](#)。

107.

108. 事件属性

109. <button> 标签支持 [HTML 的事件属性](#)。

110.

111. 相关文章

112. HTML DOM 参考手册: [Button 对象](#)

113. [HTML
 标签](#)

114. [HTML <canvas> 标签](#)

115. 点我分享笔记

116. [HTML <button> 标签](#)

117. [HTML <caption> 标签](#)

118. HTML <canvas> 标签

119.

120. 实例

121. 通过 <canvas> 元素来显示一个红色的矩形:

```
122.   <canvas           id="myCanvas"></canvas>           <script           type="text/javascript">           var  
      canvas=document.getElementById('myCanvas');   var   ctx=canvas.getContext('2d');   ctx.fillStyle='#FF0000';  
      ctx.fillRect(0,0,80,100); </script>
```

123.

[尝试一下 »](#)

124.

125. 浏览器支持



126.

127. IE 9、Firefox、Opera、Chrome 和 Safari 支持 <canvas> 标签。

128. 注释: IE 8 或更早版本的 IE 浏览器不支持 <canvas> 标签。

129.

130. 标签定义及使用说明

131. <canvas> 标签通过脚本（通常是 JavaScript）来绘制图形（比如图表和其他图像）。

132. <canvas> 标签只是图形容器，您必须使用脚本来绘制图形。

133.

134. HTML 4.01 与 HTML5 之间的差异

135. <canvas> 标签是 HTML5 中的新标签。

136.

137. 提示和注释

138. 注释：<canvas> 元素中的任何文本将会被显示在不支持 <canvas> 的浏览器中。

139. 提示：如想了解 canvas 对象的所有属性和方法，请参阅 [HTML 画布参考手册](#)。

140.

141. 属性

142. **New**：HTML5 中的新属性。

属性	值	描述
height New	<i>pixels</i>	规定画布的高度。
width New	<i>pixels</i>	规定画布的宽度。

143. 全局属性

144. <canvas> 标签支持 [HTML 的全局属性](#)。

145.

146. 事件属性

147. <canvas> 标签支持 [HTML 的事件属性](#)。

148. [HTML <button> 标签](#)

149. [HTML <caption> 标签](#)

150. [点我分享笔记](#)

151. [HTML <canvas> 标签](#)

152. [HTML <center> 标签](#)

153. HTML <caption> 标签

154.

155. 实例

156. 带有标题的表格:

```
157. <table border="1"> <caption>Monthly savings</caption> <tr> <th>Month</th> <th>Savings</th> </tr>
    <tr> <td>January</td> <td>$100</td> </tr> </table>
```

158.

[尝试一下 »](#)

159.

160. 浏览器支持



161.

162. 所有主流浏览器都支持 <caption> 标签。

163.

164. 标签定义及使用说明

165. <caption> 标签定义表格的标题。

166. <caption> 标签必须直接放置到 <table> 标签之后。

167. 您只能对每个表格定义一个标题。

168. 提示：通常这个标题会被居中于表格之上。然而，CSS 属性 "text-align" 和 "caption-side" 能用来设置标题的对齐方式和显示位置。

169.

170. HTML 4.01 与 HTML5 之间的差异

171. HTML5 不支持 align 属性。

172. HTML 4.01 已废弃 align 属性。

173.

174. 属性

属性	值	描述
<u>align</u>	left right	HTML5 不支持。HTML 4.01 已废弃。 定义标题的对齐方式。

	top bottom	
--	---------------	--

175.

176.

177. 全局属性

178. <caption> 标签支持 [HTML 的全局属性](#)。

179.

180. 事件属性

181. <caption> 标签支持 [HTML 的事件属性](#)。

182. [HTML <canvas> 标签](#)

183. [HTML <center> 标签](#)

184. [点我分享笔记](#)

185. [HTML <caption> 标签](#)

186. [HTML <cite> 标签](#)

187. HTML <center> 标签 - HTML5 不支持

188.

189. 实例

190. 将 HTML 网页中的文本进行水平居中处理：

191. `<center>`这个文本居中对齐。`</center>`

192.

[尝试一下 »](#)

193.

194. 浏览器支持

195. 

196. 所有主流浏览器都支持 `<center>` 标签。

197.

198. 标签定义及使用说明

199. **HTML5 不支持 `<center>` 标签。请用 CSS 代替。**

200. 在 HTML 4.01 中，`<center>` 元素 已废弃。

201. `<center>` 对其所包括的文本进行水平居中。

202.

203. 提示和注释

204. **提示：**请使用 CSS 样式来居中文本！在 CSS 教程中您能了解到更多关于居中文本的细节。

205.

206. HTML 4.01 与 HTML5 之间的差异

207. HTML5 不支持 <center> 标签，HTML 4.01 已废弃 <center> 标签。
208.

209. 标准属性

210. 在 HTML 4.01 中，<center> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

211. 如需完整的描述，请访问[标准属性](#)。
212.

213. 事件属性

214. 在 HTML 4.01 中，<center> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

215. 如需完整的描述，请访问[事件属性](#)。

216. [HTML <caption> 标签](#)

217. [HTML <cite> 标签](#)

218. 点我分享笔记

[HTML <center> 标签](#)

[HTML <code> 标签](#)

HTML <cite> 标签

实例

使用 <cite> 标签来定义作品的标题：

```
<p><cite>The Scream</cite> by Edward Munch. Painted in 1893.</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <cite> 标签。

标签定义及使用说明

<cite> 标签定义作品（比如书籍、歌曲、电影、电视节目、绘画、雕塑等等）的标题。

注释：人名不属于作品的标题。

HTML 4.01 与 HTML5 之间的差异

在 HTML5 中，<cite> 标签定义作品的标题。
在 HTML 4.01 中，<cite> 标签定义一个引用。

全局属性

<cite> 标签支持 [HTML 的全局属性](#)。

事件属性

<cite> 标签支持 [HTML 的事件属性](#)。

[HTML <center> 标签](#)

[HTML <code> 标签](#)

1 篇笔记 写笔记

1. WH-KING
104***5930@qq.com

- o 按照惯例，引用的文本将以斜体显示。
- o 用<cite>标签把指向其他文档的引用分离出来，尤其是分离那些传统媒体中的文档，如书籍、杂志、期刊等。如果引用的这些文档有联机版本，最好是把该引用包含于一个 a 元素中，从而把一个超链接指向该联机版本。
- o <cite>标签还有一个隐藏功能：它可以使你或者其他从文档中自动摘录参考书目。可以很容易想象一个浏览器，它能够自动整理引用表格，并把它们作为脚注或者独立的文档来显示。

WH-KING

WH-KING

104***5930@qq.com

7 年前 (2017-09-08)

[HTML <cite> 标签](#)

[HTML <col> 标签](#)

HTML `<code>` 标签

实例

对文档中的文本进行格式化：

```
<code>一段电脑代码 print("Hello World")</code>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<code>` 标签。

标签定义及使用说明

`<code>` 标签是一个短语标签，用来定义计算机代码文本。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 CSS，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
<code></code>	呈现为被强调的文本。
<code></code>	定义重要的文本。
<code><dfn></code>	定义一个定义项目。
<code><code></code>	定义计算机代码文本。
<code><samp></code>	定义样本文本。
<code><kbd></code>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<code><var></code>	定义变量。您可以将此标签与 <code><pre></code> 及 <code><code></code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<code>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<code>` 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML <cite> 标签](#)

[HTML <col> 标签](#)

点我分享笔记

[HTML <code> 标签](#)

[HTML <colgroup> 标签](#)

HTML `<col>` 标签

实例

`<colgroup>` 和 `<col>` 标签为表格中的三个列设置了背景色:

```
<table border="1">
  <colgroup>
    <col span="2" style="background-color:red">
    <col style="background-color:yellow">
```

```
</colgroup>
<tr>
  <th>ISBN</th>
  <th>Title</th>
  <th>Price</th>
</tr>
<tr>
  <td>3476896</td>
  <td>My first HTML</td>
  <td>$53</td>
</tr>
</table>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<col>` 标签。

标签定义及使用说明

`<col>` 标签规定了 `<colgroup>` 元素内部的每一列的列属性。

通过使用 `<col>` 标签，可以向整个列应用样式，而不需要重复为每个单元格或每一行设置样式。

HTML 4.01 与 HTML5 之间的差异

HTML5 中不再支持 HTML 4.01 中的大部分属性。

HTML 与 XHTML 之间的差异

在 HTML 中，<col> 标签没有结束标签。
在 XHTML 中，<col> 标签必须被正确的关闭。

属性

属性	值	描述
align	left right center justify char	HTML5 不支持。规定与 <col> 元素相关的内容的水平对齐方式。
char	<i>character</i>	HTML5 不支持。规定根据哪个字符来对齐与 <col> 元素相关的内容。
charoff	<i>number</i>	HTML5 不支持。规定第一个对齐字符的偏移量。
span	<i>number</i>	规定 <col> 元素应该横跨的列数。
valign	top middle bottom	HTML5 不支持。规定与 <col> 元素相关的内容的垂直对齐方式。

	baseline	
width	<i>%</i> <i>pixels</i> <i>relative_length</i>	HTML5 不支持。Specifies the width of a <col> element

全局属性

<col> 标签支持 [HTML 的全局属性](#)。

事件属性

<col> 标签支持 [HTML 的事件属性](#)。

[HTML <code> 标签](#)

[HTML <colgroup> 标签](#)

点我分享笔记

[HTML <col> 标签](#)

[HTML <command> 标签](#)

HTML <colgroup> 标签

实例

`<colgroup>` 和 `<col>` 标签为表格中的三个列设置了背景色：

```
<table border="1">
  <colgroup>
    <col span="2" style="background-color:red">
    <col style="background-color:yellow">
  </colgroup>
  <tr>
    <th>ISBN</th>
    <th>Title</th>
    <th>Price</th>
  </tr>
  <tr>
    <td>3476896</td>
    <td>My first HTML</td>
    <td>$53</td>
  </tr>
</table>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<colgroup>` 标签。

标签定义及使用说明

`<colgroup>` 标签用于对表格中的列进行组合，以便对其进行格式化。

通过使用 `<colgroup>` 标签，可以向整个列应用样式，而不需要重复为每个单元格或每一行设置样式。

注释：只能在 `<table>` 元素之内，在任何一个 `<caption>` 元素之后，在任何一个 `<thead>`、`<tbody>`、`<tfoot>`、`<tr>` 元素之前使用 `<colgroup>` 标签。

提示：如果想对 `<colgroup>` 中的某列定义不同的属性，请在 `<colgroup>` 标签内使用 `<col>` 标签。

HTML 4.01 与 HTML5 之间的差异

HTML5 中不再支持 HTML 4.01 中的大部分属性。

属性

属性	值	描述
align	left right center justify char	HTML5 不支持。规定在列组合中内容的水平对齐方式。
char	<i>character</i>	HTML5 不支持。规定根据哪个字符来对齐列组中的内容。
charoff	<i>number</i>	HTML5 不支持。规定第一个对齐字符的偏移量。
span	<i>number</i>	规定列组应该横跨的列数。

valign	top middle bottom baseline	HTML5 不支持。定义在列组合中内容的垂直对齐方式。
width	<i>pixels</i> % <i>relative_length</i>	HTML5 不支持。规定列组合的宽度。

全局属性

<colgroup> 标签支持 [HTML 的全局属性](#)。

事件属性

<colgroup> 标签支持 [HTML 的事件属性](#)。

[HTML <col> 标签](#)

[HTML <command> 标签](#)

点我分享笔记

[HTML <colgroup> 标签](#)

[HTML <datalist> 标签](#)

HTML `<command>` 标签

实例

`<command>` 可以进行如下标记：

```
<menu>
```

```
<command type="command" label="Save" onclick="save()">Save</command>
```

```
</menu>
```

[尝试一下 »](#)

浏览器支持



目前，主流浏览器都不支持 `<command>` 标签。

注释：只有 IE 9 支持 `<command>` 标签，其他之前版本或者之后版本的 IE 浏览器不支持 `<command>` 标签。

标签定义及使用说明

`<command>` 标签可以定义用户可能调用的命令（比如单选按钮、复选框或按钮）。

当使用 `<menu>` 元素时，`command` 元素将作为菜单或者工具栏的一部分显示出来。但是，用 `command` 规定键盘快捷键时，`command` 元素能被放置在页面的任何位置，但元素不可见。

HTML 4.01 与 HTML5 之间的差异

<command> 标签是 HTML 5 中的新标签。

属性

New：HTML5 中的新属性。

属性	值	描述
checked New	checked	规定当页面加载时，command 是否被选中。仅用于 radio 或 checkbox 类型。
disabled New	disabled	规定 command 是否可用。
icon New	<i>URL</i>	规定作为 command 来显示的图像的 URL。
label New	<i>text</i>	必需。规定 command 的名字，对用户可见。
radiogroup New	<i>groupname</i>	规定可进行切换且将被切换的 command 所属的组名。仅在类型为 radio 时使用。
type New	checkbox command radio	定义 command 的类型。默认是 "command"。

全局属性

<command> 标签支持 [HTML 的全局属性](#)。

事件属性

<command> 标签支持 [HTML 的事件属性](#)。

[HTML <colgroup> 标签](#)

[HTML <datalist> 标签](#)

点我分享笔记

[HTML <command> 标签](#)

[HTML <dd> 标签](#)

HTML <datalist> 标签

实例

下面是一个 <input> 元素，<datalist> 中描述了其可能的值：

```
<input list="browsers"> <datalist id="browsers"> <option value="Internet Explorer"> <option value="Firefox">  
<option value="Chrome"> <option value="Opera"> <option value="Safari"> </datalist>
```

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

元素					
<datalist>	20.0	10.0	4.0	12.1	9.0

标签定义及使用说明

<datalist> 标签规定了 <input> 元素可能的选项列表。

<datalist> 标签被用来在为 <input> 元素提供"自动完成"的特性。用户能看到一个下拉列表，里边的选项是预先定义好的，将作为用户的输入数据。

请使用 <input> 元素的 list 属性来绑定 <datalist> 元素。

HTML 4.01 与 HTML5 之间的差异

<datalist> 标签是 HTML5 中的新标签。

全局属性

<datalist> 标签支持 [HTML 的全局属性](#)。

事件属性

<datalist> 标签支持 [HTML 的事件属性](#)。

[HTML <command> 标签](#)

[HTML <dd> 标签](#)

点我分享笔记

[HTML <datalist> 标签](#)

[HTML 标签](#)

HTML <dd> 标签

实例

带有项目和描述的描述列表：

```
<dl>
  <dt>Coffee</dt>
  <dd>Black hot drink</dd>
  <dt>Milk</dt>
  <dd>White cold drink</dd>
</dl>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<dd>` 标签。

标签定义及使用说明

`<dd>` 标签被用来对一个描述列表中的项目/名字进行描述。

`<dd>` 标签与 [`<dl>`](#)（定义一个描述列表）和 [`<dt>`](#)（定义项目/名字）一起使用。

在 `<dd>` 标签内，您能放置段落、换行、图片、链接、列表等等。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<dd>` 标签被用来描述一个定义列表中的条目。

在 HTML5 中，`<dd>` 标签被用来描述一个描述列表的项目/名字。

全局属性

`<dd>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<dd>` 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程：[HTML 列表](#)

[HTML <datalist> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <dd> 标签](#)

[HTML <details> 标签](#)

HTML 标签

实例

一段带有已删除部分和新插入部分的文本：

```
<p>My favorite color is <del>blue</del> <ins>red</ins>!</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 标签。

标签定义及使用说明

 标签定义文档中已删除的文本。

提示和注释

提示：您也可以看看 [<ins>](#) 标签如何标记先插入的文本。
提示： 和 <ins> 一起使用，描述文档中的更新和修正。浏览器通常会在已删除文本上添加一条删除线，在新插入文本下添加一条下划线。

HTML 4.01 与 HTML5 之间的差异

无。

属性

属性	值	描述
cite	<i>URL</i>	规定一个解释了文本被删除的原因的文档的 URL。
datetime	<i>YYYY-MM-DDThh:mm:ssTZD</i>	规定文本被删除的日期和时间。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。

[HTML <dd> 标签](#)

[HTML <details> 标签](#)

点我分享笔记

[HTML 标签](#)

[HTML <dfn> 标签](#)

HTML <details> 标签

实例

使用 <details> 元素：

```
<details> <summary>Copyright 1999-2011.</summary> <p> - by Refsnes Data. All Rights Reserved.</p> <p>All content and graphics on this web site are the property of the company Refsnes Data.</p> </details>
```

[尝试一下 »](#)

浏览器支持



目前，只有 Chrome 和 Safari 6 支持 `<details>` 标签。

标签定义及使用说明

`<details>` 标签规定了用户可见的或者隐藏的需求的补充细节。

`<details>` 标签用来供用户开启关闭的交互式控件。任何形式的内容都能被放在 `<details>` 标签里边。

`<details>` 元素的内容对用户是不可见的，除非设置了 `open` 属性。

HTML 4.01 与 HTML5 之间的差异

`<details>` 标签是 HTML5 中的新标签。

提示和注释

提示：与 `<summary>` 标签配合使用可以为 `details` 定义标题。标题是可见的，用户点击标题时，会显示出 `details`。

属性

New：HTML5 中的新属性。

属性	值	描述
<code>open</code> New	<code>open</code>	规定 <code>details</code> 是否可见。

全局属性

<details> 标签支持 [HTML 的全局属性](#)。

事件属性

<details> 标签支持 [HTML 的事件属性](#)。

[HTML 标签](#)

[HTML <dfn> 标签](#)

点我分享笔记

[HTML <details> 标签](#)

[HTML <dialog> 标签](#)

HTML <dfn> 标签

实例

对文档中的文本进行格式化：

```
<dfn>定义项目</dfn>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<dfn>` 标签。

标签定义及使用说明

`<dfn>` 标签是一个短语标签，用来定义一个定义项目。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 **CSS**，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
<code></code>	呈现为被强调的文本。
<code></code>	定义重要的文本。
<code><dfn></code>	定义一个定义项目。
<code><code></code>	定义计算机代码文本。
<code><samp></code>	定义样本文本。
<code><kbd></code>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<code><var></code>	定义变量。您可以将此标签与 <code><pre></code> 及 <code><code></code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

<dfn> 标签支持 [HTML 的全局属性](#)。

事件属性

<dfn> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML <details> 标签](#)

[HTML <dialog> 标签](#)

点我分享笔记

[HTML <dfn> 标签](#)

[HTML <dir> 标签](#)

HTML [<dialog>](#) 标签

实例

使用 <dialog> 元素:

<table border="1"> <tr> <th>January <dialog open>This is an open dialog window</dialog></th> <th>February</th> <th>March</th> </tr> <tr> <td>31</td> <td>28</td> <td>31</td> </tr> </table>

尝试一下 »

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

元素					
<dialog>	37.0	不支持	59.0	6.0	24.0

标签定义及使用说明

<dialog> 标签定义一个对话框、确认框或窗口。

HTML 4.01 与 HTML5 之间的差异

<dialog> 标签是 HTML5 中的新标签。

属性

New：HTML5 中的新属性。

属性	值	描述
open New	open	规定 dialog 元素是有效的，用户可以与它进行交互。

全局属性

<dialog> 标签支持 [HTML 的全局属性](#)。

事件属性

<dialog> 标签支持 [HTML 的事件属性](#)。

[HTML <dfn> 标签](#)

[HTML <dir> 标签](#)

点我分享笔记

[HTML <dialog> 标签](#)

[HTML <div> 标签](#)

HTML <dir> 标签 - HTML5 不支持

实例

目录列表：

```
<dir> <li>html</li> <li>xhtml</li> <li>css</li> </dir>
```

[尝试一下»](#)

浏览器支持



所有主流浏览器都支持 `<dir>` 标签。

标签定义及使用说明

HTML5 不支持 `<dir>` 标签。请用 CSS 代替。

在 HTML 4.01 中，`<dir>` 元素 [已废弃](#)。

`<dir>` 标签被用来定义目录列表。

提示和注释

提示： 请使用 CSS 来为列表添加样式！在我们的 CSS 教程中，您可以找到更多有关[为列表添加样式](#)的细节。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 `<dir>` 标签，HTML 4.01 已废弃 `<dir>` 标签。

可选的属性

属性	值	描述
compact	compact	HTML5 不支持。HTML 4.01 已废弃。规定列表必须比常规状态小一号呈现。

标准属性

在 HTML 4.01 中，<dir> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<dir> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML <dialog> 标签](#)

[HTML <div> 标签](#)

点我分享笔记

[HTML <div> 标签](#)

[HTML <dl> 标签](#)

HTML <div> 标签

实例

文档中的一个区域将显示为蓝色：

```
<div style="color:#0000FF"> <h3>这是一个在 div 元素中的标题。</h3> <p>这是一个在 div 元素中的文本。</p> </div>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <div> 标签。

标签定义及使用说明

<div> 标签定义 HTML 文档中的一个分隔区块或者一个区域部分。

<div> 标签常用于组合块级元素，以便通过 CSS 来对这些元素进行格式化。

提示和注释

提示： <div> 元素经常与 CSS 一起使用，用来布局网页。

注释： 默认情况下，浏览器通常会在 <div> 元素前后放置一个换行符。然而，您可以通过使用 CSS 改变这种情况。

HTML 4.01 与 HTML5 之间的差异

HTML5 中不支持 align 属性。

在 HTML 4.01 中，align 属性 [已废弃](#)。

属性

属性	值	描述
align	left right center justify	HTML5 不支持。HTML 4.01 已废弃。 规定 <div> 元素中的内容的对齐方式。

全局属性

<div> 标签支持 [HTML 的全局属性](#)。

事件属性

<div> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 布局](#)

[HTML <dir> 标签](#)

[HTML <dl> 标签](#)

点我分享笔记

[HTML <div> 标签](#)

[HTML <dt> 标签](#)

HTML **<dl>** 标签

实例

带有项目和描述的描述列表:

```
<dl> <dt>Coffee</dt> <dd>Black hot drink</dd> <dt>Milk</dt> <dd>White cold drink</dd> </dl>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<dl>` 标签。

标签定义及使用说明

`<dl>` 标签定义一个描述列表。

`<dl>` 标签与 [<dt>](#)（定义项目/名字）和 [<dd>](#)（描述每一个项目/名字）一起使用。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<dl>` 标签定义一个定义列表。

在 HTML5 中，`<dl>` 标签定义一个描述列表。

全局属性

`<dl>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<dl>` 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程： [HTML 列表](#)

[HTML <div> 标签](#)

[HTML <dt> 标签](#)

点我分享笔记

[HTML <dl> 标签](#)

[HTML 标签](#)

HTML <dt> 标签

实例

带有项目和描述的描述列表：

```
<dl> <dt>Coffee</dt> <dd>Black hot drink</dd> <dt>Milk</dt> <dd>White cold drink</dd> </dl>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <dt> 标签。

标签定义及使用说明

<dt> 标签定义一个描述列表的项目/名字。

`<dt>` 标签与 [<dl>](#)（定义一个描述列表）和 [<dd>](#)（描述每一个项目/名字）一起使用。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<dt>` 标签定义一个定义列表的条目。
在 HTML5 中，`<dt>` 标签定义一个描述列表的项目/名字。

全局属性

`<dt>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<dt>` 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程： [HTML 列表](#)

[HTML <dl> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <dt> 标签](#)

[HTML <embed> 标签](#)

HTML 标签

实例

对文档中的文本进行格式化：

强调文本

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 标签。

标签定义及使用说明

 标签是一个短语标签，用来呈现为被强调的文本。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 CSS，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
	呈现为被强调的文本。

	定义重要的文本。
<dfn>	定义一个定义项目。
<code>	定义计算机代码文本。
<samp>	定义样本文本。
<kbd>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<var>	定义变量。您可以将此标签与 <pre> 及 <code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML <dt> 标签](#)

[HTML <embed> 标签](#)

点我分享笔记

[HTML 标签](#)

[HTML <fieldset> 标签](#)

HTML <embed> 标签

实例

嵌入图片:

```
<embed type="image/jpg" src="https://static.jyshare.com/images/runoob-logo.png" width="258" height="39">
```

[尝试一下 »](#)

实例

嵌入 HTML 页面:

```
<embed type="text/html" src="snippet.html" width="500" height="200">
```

[尝试一下 »](#)

实例

嵌入视频：

```
<embed type="video/webm" src="video.mp4" width="400" height="300">
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<embed>` 标签。

标签定义及使用说明

`<embed>` 标签定义了一个容器，用来嵌入外部应用或者互动程序（插件）。

注意：现在已经不建议使用 `<embed>` 标签了，可以使用 ``、`<iframe>`、`<video>`、`<audio>` 等标签代替。

HTML 4.01 与 HTML5 之间的差异

`<embed>` 标签是 HTML 5 中的新标签。

属性

New：HTML5 中的新属性。

属性	值	描述
height ^{New}	<i>pixels</i>	规定嵌入内容的高度。
src ^{New}	<i>URL</i>	规定被嵌入内容的 URL。
type ^{New}	<i>MIME_type</i>	规定嵌入内容的 MIME 类型。 注：MIME = Multipurpose Internet Mail Extensions。
width ^{New}	<i>pixels</i>	规定嵌入内容的宽度。

全局属性

<embed> 标签支持 [HTML 的全局属性](#)。

事件属性

<embed> 标签支持 [HTML 的事件属性](#)。

[HTML 标签](#)

[HTML <fieldset> 标签](#)

点我分享笔记

[HTML <embed> 标签](#)

[HTML <figcaption> 标签](#)

HTML <fieldset> 标签

实例

对表单中的相关元素进行分组：

```
<form> <fieldset> <legend>Personalia:</legend> Name: <input type="text"><br> Email: <input type="text"><br>
Date of birth: <input type="text"> </fieldset> </form>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <fieldset> 标签。

标签定义及使用说明

<fieldset> 标签可以将表单内的相关元素分组。

<fieldset> 标签会在相关表单元素周围绘制边框。

提示和注释

提示：<legend> 标签为 <fieldset> 元素定义标题。

HTML 4.01 与 HTML5 之间的差异

HTML5 中新增了一些 <fieldset> 的新属性：disabled、form、name，HTML 4.01 中不支持这些属性。

属性

New：HTML5 中的新属性。

属性	值	描述
disabled New	disabled	规定该组中的相关表单元素应该被禁用。
form New	<i>form_id</i>	规定 fieldset 所属的一个或多个表单。
name New	<i>text</i>	规定 fieldset 的名称。

全局属性

<fieldset> 标签支持 [HTML 的全局属性](#)。

事件属性

<fieldset> 标签支持 [HTML 的事件属性](#)。

[HTML <embed> 标签](#)

[HTML <figcaption> 标签](#)

点我分享笔记

[HTML <fieldset> 标签](#)

[HTML <figure> 标签](#)

HTML <figcaption> 标签

实例

使用 <figure> 元素标记文档中的一个图像。<figure> 元素带有一个标题：

```
<figure>  <figcaption>Fig1. - A view  
of the pulpit rock in Norway.</figcaption> </figure>
```

[尝试一下 »](#)

浏览器支持



IE 9、Firefox、Opera、Chrome 和 Safari 支持 <figcaption> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 `<figcaption>` 标签。

标签定义及使用说明

`<figcaption>` 标签为 [<figure>](#) 元素定义标题。

`<figcaption>` 元素应该被置于 `<figure>` 元素的第一个或最后一个子元素的位置。

HTML 4.01 与 HTML5 之间的差异

`<figcaption>` 标签是 HTML5 中的新标签。

全局属性

`<figcaption>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<figcaption>` 标签支持 [HTML 的事件属性](#)。

[HTML <fieldset> 标签](#)

[HTML <figure> 标签](#)

点我分享笔记

[HTML <figcaption> 标签](#)

[HTML 标签](#)

HTML <figure> 标签

实例

使用 <figure> 元素标记文档中的一个图像：

```
<figure>  </figure>
```

[尝试一下 »](#)

浏览器支持



IE 9、Firefox、Opera、Chrome 和 Safari 支持 <figure> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <figure> 标签。

标签定义及使用说明

<figure> 标签规定独立的流内容（图像、图表、照片、代码等等）。

<figure> 元素的内容应该与主内容相关，同时元素的位置相对于主内容是独立的。如果被删除，则不应影响文档流。

HTML 4.01 与 HTML5 之间的差异

<figure> 标签是 HTML 5 中的新标签。

提示和注释

提示: [<figcaption>](#) 元素被用来为 <figure> 元素定义标题。

全局属性

<figure> 标签支持 [HTML 的全局属性](#)。

事件属性

<figure> 标签支持 [HTML 的事件属性](#)。

[HTML <figcaption> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <figure> 标签](#)

[HTML <footer> 标签](#)

HTML 标签 - HTML5 不支持

实例

规定文本的尺寸、字体和颜色：

```
<font size="3" color="red">这是一些文本!</font> <font size="2" color="blue">这是一些文本!</font> <font face="verdana" color="green">这是一些文本!</font>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `` 标签。

标签定义及使用说明

HTML5 不支持 `` 标签。请用 CSS 代替。

在 HTML 4.01 中，`` 元素 **已废弃**。

`` 标签规定文本的字体、字体尺寸、字体颜色。

提示和注释

提示：请使用 CSS 来定义文本的**字体、尺寸、颜色**。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 `` 标签，HTML 4.01 已废弃 `<center>` 标签。

可选的属性

属性	值	描述
color	<i>rgb(x,x,x)</i> <i>#xxxxxxx</i> <i>colorname</i>	HTML5 不支持。HTML 4.01 已废弃。 规定文本的颜色。
face	<i>font_family</i>	HTML5 不支持。HTML 4.01 已废弃。 规定文本的字体。
size	<i>number</i>	HTML5 不支持。HTML 4.01 已废弃。 规定文本的尺寸。

标准属性

在 HTML 4.01 中， 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式

title	text	规定元素的额外信息
-------	------	-----------

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中， 标签不支持任何的事件属性。

如需完整的描述，请访问[事件属性](#)。

[HTML <figure> 标签](#)

[HTML <footer> 标签](#)

1 篇笔记 写笔记

1. WH-KING
104***5930@qq.com

65

类似于 **font** 元素这种只具有表现性的标记代码被废弃的原因：

在 Web 早期，HTML 文档编辑人员缺乏改变页面排版样式的途径，无法使用与当时 Web 浏览器中内置字形、字号和颜色不同的设置，随后 HTML 中引入了 font 元素，文档编辑人员可以通过表现性标签改变字体相关属性。这使得文档编辑人员在文本的表现性控制方面获得了更多的主动权。但是，随之而来的问题比较突出，就是在文档中插入了大量的、类似于 font 元素的这种表现性标记代码，使 HTML 文档变得臃肿不堪，倘若网站需要重新设计，势必需要找出每个类似于 font 元素的这种表现性标记代码并加以修改，并且一个文本片段往往对应大量的表现性标记代码。

CSS 的出现给了 HTML 文档编辑人员不需要额外的标记代码就能控制排版样式的途径，使得修改整个网站的设计都可以通过一个单独的文件来完成。因此，类似于 font 元素这种只具有表现性的标记代码被正式废弃。

[WH-KING](#)

WH-KING

104***5930@qq.com

7 年前 (2017-09-12)

[HTML 标签](#)

[HTML <form> 标签](#)

HTML <footer> 标签

实例

文档的页脚：

```
<footer> <p>Posted by: Hege Refsnes</p> <p><time pubdate datetime="2012-03-01"></time></p> </footer>
```

[尝试一下 »](#)

浏览器支持



IE 9、Firefox、Opera、Chrome 和 Safari 支持 <footer> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <footer> 标签。

标签定义及使用说明

<footer> 标签定义文档或者文档的一部分区域的页脚。

<footer> 元素应该包含它所包含的元素的信息。

在典型情况下，该元素会包含文档创作者的姓名、文档的版权信息、使用条款的链接、联系信息等等。

在一个文档中，您可以定义多个 <footer> 元素。

HTML 4.01 与 HTML5 之间的差异

<footer> 标签是 HTML 5 中的新标签。

提示和注释

提示：假如您使用 <footer> 元素来插入联系信息，应该在 <footer> 元素内使用 [<address>](#) 标签。

全局属性

<footer> 标签支持 [HTML 的全局属性](#)。

事件属性

<footer> 标签支持 [HTML 的事件属性](#)。

[HTML 标签](#)

[HTML <form> 标签](#)

点我分享笔记

[HTML <footer> 标签](#)

[HTML <frame> 标签](#)

HTML <form> 标签

实例

带有两个输入字段和一个提交按钮的 HTML 表单：

```
<form action="demo_form.php" method="get"> First name: <input type="text" name="fname"><br> Last name: <input type="text" name="lname"><br> <input type="submit" value="提交"> </form>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 <form> 标签。

标签定义及使用说明

<form> 标签用于创建供用户输入的 HTML 表单。

<form> 元素包含一个或多个如下的表单元素：

- [<input>](#)
- [<textarea>](#)
- [<button>](#)

- [<select>](#)
 - [<option>](#)
 - [<optgroup>](#)
 - [<fieldset>](#)
 - [<label>](#)
-

HTML 4.01 与 HTML5 之间的差异

HTML5 新增了两个新的属性：autocomplete 和 novalidate，同时不再支持 HTML 4.01 中的某些属性。

HTML 与 XHTML 之间的差异

在 XHTML 中，name 属性已被废弃。使用全局 id 属性代替。

属性

New：HTML5 中的新属性。

属性	值	描述
accept	<i>MIME_type</i>	HTML5 不支持。 规定服务器接收到的文件的类型。（文件是通过文件上传提交的）
accept-charset	<i>character_set</i>	规定服务器可处理的表单数据字符集。
action	<i>URL</i>	规定当提交表单时向何处发送表单数据。

autocomplete New	on off	规定是否启用表单的自动完成功能。
enctype	application/x-www-form-urlencoded multipart/form-data text/plain	规定在向服务器发送表单数据之前如何对其进行编码。 (适用于 method="post" 的情况)
method	get post	规定用于发送表单数据的 HTTP 方法。
name	<i>text</i>	规定表单的名称。
novalidate New	novalidate	如果使用该属性，则提交表单时不进行验证。
target	_blank _self _parent _top	规定在何处打开 action URL。

全局属性

<form> 标签支持 [HTML 的全局属性](#)。

事件属性

<form> 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[带有复选框的表单](#)

此表单包含两个复选框和一个提交按钮。

[带有单选按钮的表单](#)

此表单包含两个单选框和一个提交按钮。

相关文章

HTML 教程: [HTML 表单和输入](#)

HTML DOM 参考手册: [Form 对象](#)

[HTML <footer> 标签](#)

[HTML <frame> 标签](#)

1 篇笔记 写笔记

1. 温水青蛙
135***4809@qq.com

form 标签是表单标签

action 属性设置提交的服务器地址

method 属性设置提交的方式 GET(默认值)或 POST

表单提交的时候，数据没有发送给服务器的三种情况：

- 1、表单项没有 name 属性值
- 2、单选、复选（下拉列表中的 option 标签）都需要添加 value 属性，以便发送给服务器
- 3、表单项不在提交的 form 标签中

GET 请求的特点是：

- 1、浏览器地址栏中的地址是：action 属性[+?+请求参数]

请求参数的格式是：name=value&name=value

- 2、不安全
- 3、它有数据长度的限制

POST 请求的特点是：

- 1、浏览器地址栏中只有 action 属性值
- 2、相对于 GET 请求要安全
- 3、理论上没有数据长度的限制

[温水青蛙](#)

温水青蛙

135***4809@qq.com

3 年前 (2021-04-18)

[HTML <form> 标签](#)

[HTML <frameset> 标签](#)

HTML <frame> 标签 - HTML5 不支持

实例

简单的三框架页面：

```
<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">
</frameset>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 `<frame>` 标签。

标签定义及使用说明

HTML5 不支持 `<frame>` 标签。

`<frame>` 标签定义 `<frameset>` 中的子窗口（框架）。

`<frameset>` 中的每个 `<frame>` 都可以设置不同的属性，比如 `border`、`scrolling`、`noresize` 等等。

注释：如果您希望验证包含框架的页面，请确保 `<!DOCTYPE>` 被设置为 "HTML Frameset DTD" 或者 "XHTML Frameset DTD"。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <frame> 标签，HTML 4.01 支持 <frame> 标签。

HTML 与 XHTML 之间的差异

在 HTML 中，<frame> 标签没有结束标签。在 XHTML 中，<frame> 标签必须被正确地关闭。

可选的属性

属性	值	描述
frameborder	0 1	HTML5 不支持。规定是否显示框架周围的边框。
longdesc	URL	HTML5 不支持。规定一个包含有关框架内容的长描述的页面。
marginheight	pixels	HTML5 不支持。规定框架的上方和下方的边距。
marginwidth	pixels	HTML5 不支持。规定框架的左侧和右侧的边距。
name	name	HTML5 不支持。规定框架的名称。
noresize	noresize	HTML5 不支持。规定无法调整框架的大小。
scrolling	yes no auto	HTML5 不支持。规定是否在框架中显示滚动条。

src	<i>URL</i>	HTML5 不支持。规定在框架中显示的文档的 URL。
---------------------	------------	-----------------------------

标准属性

在 HTML 4.01 中，<frame> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
id	<i>id</i>	规定元素的唯一 id
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

根据 W3C 的标准，在 HTML 4.01 中，<frame> 标签不支持任何的事件属性。

但是，所有的浏览器都支持 onload 事件。

如需完整的描述，请访问[事件属性](#)。



尝试一下 - 实例

[水平框架](#)

本例演示：如何使用三份不同的文档制作一个水平框架。

[混合结构框架](#)

本例演示如何制作含有三份文档的框架结构，同时将他们混合置于行和列之中。

[含有 noresize="noresize" 属性的框架结构](#)

本例演示 noresize 属性。在本例中，框架是不可调整尺寸的。在框架间的边框上拖动鼠标，您会发现边框是无法移动的。

[HTML <form> 标签](#)

[HTML <frameset> 标签](#)

点我分享笔记

[HTML <frame> 标签](#)

[HTML <head> 标签](#)

HTML <frameset> 标签 - HTML5 不支持

实例

简单的三框架页面：

```
<frameset cols="25%,*,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
```



```
<frame src="frame_c.htm">
</frameset>
```

尝试一下 »

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 `<frameset>` 标签。

标签定义及使用说明

HTML5 不支持 `<frameset>` 标签。

`<frameset>` 标签定义一个框架集。

`<frameset>` 元素被用来组织一个或者多个 `<frame>` 元素。每个 `<frame>` 有各自独立的文档。

`<frameset>` 元素规定在框架集中存在多少列或多少行，以及每行每列占用的百分比/像素。

注释：如果您希望验证包含框架的页面，请确保 `<!DOCTYPE>` 被设置为 "HTML Frameset DTD" 或者 "XHTML Frameset DTD" 。

HTML 与 XHTML 之间的差异

无。

可选的属性

属性	值	描述
cols	<i>pixels</i> <i>%</i> <i>*</i>	HTML5 不支持。规定框架集中列的数目和尺寸。
rows	<i>pixels</i> <i>%</i> <i>*</i>	HTML5 不支持。规定框架集中行的数目和尺寸。

标准属性

在 HTML 4.01 中，<frameset> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
id	<i>id</i>	规定元素的唯一 id
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<frameset> 标签支持如下事件属性：

属性	值	描述
onload	<i>script</i>	当文档被载入时执行脚本
onunload	<i>script</i>	当文档被卸下时执行脚本

如需完整的描述，请访问[事件属性](#)。



尝试一下 - 实例

[水平框架](#)

本例演示：如何使用三份不同的文档制作一个水平框架。

[混合结构框架](#)

本例演示如何制作含有三份文档的框架结构，同时将他们混合置于行和列之中。

[含有 noresize="noresize" 属性的框架结构](#)

本例演示 noresize 属性。在本例中，框架是不可调整尺寸的。在框架间的边框上拖动鼠标，您会发现边框是无法移动的。

相关文章

HTML DOM 参考手册：[Frameset 对象](#)

[HTML <frame> 标签](#)

[HTML <head> 标签](#)

点我分享笔记

[HTML <frameset> 标签](#)

[HTML <header> 标签](#)

HTML <head> 标签

实例

一份在头部带有 <title> 标签的 HTML 文档：

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容..... </body> </html>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 <head> 标签。

标签定义及使用说明

<head> 元素是所有头部元素的容器。

<head> 元素必须包含文档的标题（title），可以包含脚本、样式、meta 信息 以及其他更多的信息。

以下列出的元素能被用在 <head> 元素内部：

- [<title>](#) （在头部中，这个元素是必需的）
 - [<style>](#)
 - [<base>](#)
 - [<link>](#)
 - [<meta>](#)
 - [<script>](#)
 - [<noscript>](#)
-

HTML 4.01 与 HTML5 之间的差异

HTML5 不再支持 profile 属性。

属性

属性	值	描述
profile	<i>URL</i>	HTML5 不支持。 规定文档 URL 的一系列规则。这些规则能被浏览器识别并且准确读取 <meta> 标签的内容属性中的信息。

全局属性

<head> 标签支持 [HTML 的全局属性](#)。



尝试一下 - 实例

[在 <head> 中使用 <base> 标签](#)

本例演示如何使用 <base> 标签规定页面中所有链接的默认 URL 和默认 target。

[在 <head> 中使用 <style> 标签](#)

本例演示如何在 <head> 部分添加样式信息。

[在 <head> 中使用 <link> 标签](#)

本例演示如何使用 <link> 标签链接到一个外部样式表。

相关文章

HTML 教程: [HTML 头部](#)

[HTML <frameset> 标签](#)

[HTML <header> 标签](#)

点我分享笔记

[HTML <head> 标签](#)

[HTML <hgroup> 标签](#)

HTML <header> 标签

实例

<article> 的页眉:

```
<article>
  <header>
    <h1>Internet Explorer 9</h1>
    <p><time pubdate datetime="2011-03-15"></time></p>
  </header>
  <p> Windows Internet Explorer 9(缩写为 IE9 )是在 2011 年 3 月 14 日 21:00 发布的</p>
</article>
```

[尝试一下 »](#)

浏览器支持



IE 9、Firefox、Opera、Chrome 和 Safari 支持 <header> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <header> 标签。

标签定义及使用说明

<header> 标签定义文档或者文档的一部分区域的页眉。

<header> 元素应该作为介绍内容或者导航链接栏的容器。

在一个文档中，您可以定义多个 <header> 元素。

注释：<header> 标签不能被放在 <footer>、<address> 或者另一个 <header> 元素内部。

HTML 4.01 与 HTML5 之间的差异

<header> 标签是 HTML 5 中的新标签。

全局属性

<header> 标签支持 [HTML 的全局属性](#)。

事件属性

<header> 标签支持 [HTML 的事件属性](#)。

[HTML <head> 标签](#)

[HTML <hgroup> 标签](#)

点我分享笔记

[HTML <header> 标签](#)

[HTML <h1> – <h6> 标签](#)

HTML <hgroup> 标签

实例

使用 <hgroup> 对标题进行组合：

```
<hgroup>
<h1>Welcome to my WWF</h1>
<h2>For a living planet</h2>
</hgroup>

<p>The rest of the content...</p>
```

[尝试一下 »](#)

浏览器支持



IE 9、Firefox、Opera、Chrome 和 Safari 支持 <hgroup> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <hgroup> 标签。

标签定义及使用说明

<hgroup> 标签被用来对标题元素进行分组。

当标题有多个层级（副标题）时，<hgroup> 元素被用来对一系列 <h1> - <h6> 元素进行分组。

HTML 4.01 与 HTML5 之间的差异

<hgroup> 标签是 HTML 5 中的新标签。

全局属性

<hgroup> 标签支持 [HTML 的全局属性](#)。

事件属性

<hgroup> 标签支持 [HTML 的事件属性](#)。

[HTML <header> 标签](#)

[HTML <h1> – <h6> 标签](#)

点我分享笔记

[HTML <hgroup> 标签](#)

[HTML <hr> 标签](#)

HTML <h1> - <h6> 标签

实例

六个不同的 HTML 标题：

<h1>这是标题 1</h1> <h2>这是标题 2</h2> <h3>这是标题 3</h3> <h4>这是标题 4</h4> <h5>这是标题 5</h5> <h6>这是标题 6</h6>

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<h1>` - `<h6>` 标签。

标签定义及使用说明

`<h1>` - `<h6>` 标签被用来定义 HTML 标题。

`<h1>` 定义重要等级最高的标题。`<h6>` 定义重要等级最低的标题。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<h1>` - `<h6>` 的 "align" 属性已被废弃。在 HTML 5 中，`<h1>` - `<h6>` 元素的 "align" 属性不被支持。请使用 CSS 来排列元素。

属性

属性	值	描述
align	left center right justify	HTML5 不支持。HTML 4.01 已废弃。 规定标题中文本的排列。

全局属性

<h1> - <h6> 标签支持 [HTML 的全局属性](#)。

事件属性

<h1> - <h6> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 标题](#)

[HTML <hgroup> 标签](#)

[HTML <hr> 标签](#)

点我分享笔记

[HTML <h1> – <h6> 标签](#)

[HTML <i> 标签](#)

HTML **<hr>** 标签

实例

当内容的主题发生变化时，使用 <hr> 标签进行分隔：

```
<h1>HTML</h1>
```

```
<p>HTML 是用于描述 web 页面的一种语言。</p>
```

```
<hr>
```

```
<h1>CSS</h1>
```

```
<p>CSS 定义如何显示 HTML 元素。</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<hr>` 标签。

标签定义及使用说明

`<hr>` 标签定义 HTML 页面中的主题变化（比如话题的转移），并显示为一条水平线。

`<hr>` 元素被用来分隔 HTML 页面中的内容（或者定义一个变化）。

HTML 4.01 与 HTML5 之间的差异

在 HTML5 中，`<hr>` 定义内容中的主题变化，并显示为一条水平线。

在 HTML 4.01 中，`<hr>` 标签仅仅显示为一条水平线。

在 HTML 4.01 中，所有的布局属性都 已废弃。在 HTML5 中不再支持这些属性。请使用 CSS 来为 `<hr>` 元素定义样式。

HTML 与 XHTML 之间的差异

在 HTML 中，<hr> 标签没有结束标签。
在 XHTML 中，<hr> 标签必须被正确地关闭，比如 <hr />。

属性

属性	值	描述
align	left center right	HTML5 不支持。HTML 4.01 已废弃。 规定 <hr> 元素的对齐方式
noshade	noshade	HTML5 不支持。HTML 4.01 已废弃。 规定 <hr> 元素的颜色呈现为纯色。
size	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。 规定 <hr> 元素的高度。
width	<i>pixels</i> %	HTML5 不支持。HTML 4.01 已废弃。 规定 <hr> 元素的宽度。

全局属性

<hr> 标签支持 [HTML 的全局属性](#)。

事件属性

<hr> 标签支持 [HTML 的事件属性](#)。

[HTML <h1> – <h6> 标签](#)

[HTML <i> 标签](#)

点我分享笔记

[HTML <hr> 标签](#)

[HTML <iframe> 标签](#)

HTML <i> 标签

实例

```
<p>He named his car <i>The lightning</i>, because it was very fast.</p>
```

尝试一下 »

浏览器支持



所有主流浏览器都支持 <i> 标签。

标签定义及使用说明

<i> 定义与文本中其余部分不同的部分，并把这部分文本呈现为斜体文本。

<i> 标签被用来表示科技术语、其他语种的成语俗语、想法、宇宙飞船的名字等等。

在没有其他适当语义的元素可以使用时，请使用 `<i>` 元素。其他语义的元素如下：

- [``](#)（被强调的文本）
- [``](#)（重要的文本）
- [`<mark>`](#)（被标记的/高亮显示的文本）
- [`<cite>`](#)（作品的标题）
- [`<dfn>`](#)（一个定义项目）

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，`<i>` 标签呈现斜体的文本。然而，在 HTML5 中没有必要这么做，可以使用样式表来格式化 `<i>` 元素中的文本。

全局属性

`<i>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<i>` 标签支持 [HTML 的事件属性](#)。

[HTML `<hr>` 标签](#)

[HTML `<iframe>` 标签](#)

点我分享笔记

[HTML `<i>` 标签](#)

[HTML 标签](#)

HTML <iframe> 标签

实例

标记一个内联框架：

```
<iframe src="https://www.runoob.com"></iframe>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <iframe> 标签。

标签定义及使用说明

<iframe> 标签规定一个内联框架。

一个内联框架被用来在当前 HTML 文档中嵌入另一个文档。

提示和注释

提示：您可以把需要的文本放置在 <iframe> 和 </iframe> 之间，这样就可以应对不支持 <iframe> 的浏览器。

提示：使用 CSS 为 <iframe> （包括滚动条）定义样式。

HTML 4.01 与 HTML5 之间的差异

HTML5 新增了一些新的属性，同时去掉了 HTML 4.01 中的一些属性。

HTML 与 XHTML 之间的差异

在 XHTML 中，name 属性已被废弃，并将被去掉。请使用 id 属性代替。

属性

New：HTML5 中的新属性。

属性	值	描述
align	left right top middle bottom	HTML5 不支持。HTML 4.01 已废弃。规定如何根据周围的元素来对齐 <iframe>。
frameborder	1 0	HTML5 不支持。规定是否显示 <iframe> 周围的边框。
height	<i>pixels</i>	规定 <iframe> 的高度。
longdesc	<i>URL</i>	HTML5 不支持。规定一个页面，该页面包含了有关 <iframe> 的较长描述。

<u>marginheight</u>	<i>pixels</i>	HTML5 不支持。规定 <iframe> 的顶部和底部的边距。
<u>marginwidth</u>	<i>pixels</i>	HTML5 不支持。规定 <iframe> 的左侧和右侧的边距。
<u>name</u>	<i>name</i>	规定 <iframe> 的名称。
<u>sandbox</u> ^{New}	"" allow-forms allow-same-origin allow-scripts allow-top-navigation	对 <iframe> 的内容定义一系列额外的限制。
<u>scrolling</u>	yes no auto	HTML5 不支持。规定是否在 <iframe> 中显示滚动条。
<u>seamless</u> ^{New}	seamless	规定 <iframe> 看起来像是父文档中的一部分。
<u>src</u>	<i>URL</i>	规定在 <iframe> 中显示的文档的 URL。
<u>srcdoc</u> ^{New}	<i>HTML_code</i>	规定页面中的 HTML 内容显示在 <iframe> 中。
<u>width</u>	<i>pixels</i>	规定 <iframe> 的宽度。

全局属性

<iframe> 标签支持 [HTML 的全局属性](#)。

事件属性

<iframe> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 框架](#)

HTML DOM 参考手册: [IFrame 对象](#)

[HTML <i> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <iframe> 标签](#)

[HTML input 标签](#)

HTML 标签

实例

如何插入图像:

```

```

尝试一下 »

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 `` 标签。

标签定义及使用说明

`` 标签定义 HTML 页面中的图像。

`` 标签有两个必需的属性：`src` 和 `alt`。

注释：从技术上讲，图像并不会插入 HTML 页面中，而是链接到 HTML 页面上。`` 标签的作用是为被引用的图像创建占位符。

提示：通过在 `<a>` 标签中嵌套 `` 标签，给图像添加到另一个文档的链接。

HTML 4.01 与 HTML5 之间的差异

HTML5 中不支持以下属性：`align`、`border`、`hspace`、`longdesc`、`vspace`。

在 HTML 4.01 中，以下属性：`align`、`border`、`hspace`、`vspace` [已废弃](#)。

HTML 与 XHTML 之间的差异

在 HTML 中，`` 标签没有结束标签。

在 XHTML 中，`` 标签必须被正确地关闭。

属性

New：HTML5 中的新属性。

属性	值	描述
align	top bottom middle left right	HTML5 不支持。HTML 4.01 已废弃。 规定如何根据周围的文本来排列图像。
loading	eager：立即加载 lazy：延迟加载	指定浏览器是应立即加载图像还是延迟加载图像。
alt	<i>text</i>	规定图像的替代文本。
border	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。 规定图像周围的边框。
crossorigin New	anonymous use-credentials	设置图像的跨域属性
height	<i>pixels</i>	规定图像的高度。
hspace	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。 规定图像左侧和右侧的空白。
ismap	ismap	将图像规定为服务器端图像映射。
longdesc	<i>URL</i>	HTML5 不支持。HTML 4.01 已废弃。 指向包含长的图像描述文档的 URL。

src	<i>URL</i>	规定显示图像的 URL。
usemap	<i>#mapname</i>	将图像定义为客户器端图像映射。
vspace	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。 规定图像顶部和底部的空白。
width	<i>pixels</i>	规定图像的宽度。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[从不同的位置插入图片](#)

本例演示如何将其他文件夹或服务器的图片显示到网页中。

[制作图像链接](#)

本例演示如何将图像作为一个链接使用。

创建图像地图

本例演示如何创建带有可供点击区域的图像地图。其中的每个区域都是一个超级链接。

相关文章

HTML 教程: [HTML 图像](#)

HTML DOM 参考手册: [Image 对象](#)

[HTML <iframe> 标签](#)

[HTML input 标签](#)

2 篇笔记 写笔记

1. SFD
554***f2123@qq.com

121

alt 与 title 属性区别:

图片中的 **alt** 属性是在图片不能正常显示时出现的文本提示。

图片中的 **title** 属性是在鼠标在移动到元素上的文本提示。

[SFD](#)

SFD
554***f2123@qq.com

6 年前 (2018-06-28)

2. 技术大神养成中
935***287@qq.com

alt 属性和 title 属性的区别：

1、alt 属性的特点

A、alt 属性（注意是“属性”而不是“标签”）包括替换说明，对于图像和图像热点是必须的。它只能用在 img、area 和 input 元素中（包括 applet 元素）。对于 input 元素，alt 属性意在用来替换提交按钮的图片。比如：

```
<input type="image" src="image.gif" alt="Submit" />
```

B、alt 属性保证那些文字确实为那些看不到图像的人提供了说明信息，并且在上下文中有意义。对于那些装饰性的图片可以使用空的值（alt=""，引号中间没有空格），而不是使用不相关的替换文字比如“blue bullet”。

C、Alt 属性值得长度必须少于 100 个英文字符或者用户必须保证替换文字尽可能的短。

2、title 属性的特点

A、title 属性为设置该属性的元素提供建议性的信息，即提供非本质的额外信息，大部分的可视化浏览器在鼠标悬浮在特定元素上时显示 title 文字为提示信息（tool tip），然而这又由制造商来决定如何渲染 title 文字。一些浏览器会将 title 文字显示在状态栏里。比如早期版本的 Safari 浏览器。

B、title 属性可以用在除了 base, basefont, head, html, meta, param, script 和 title 之外的所有标签。但是并不是必须的。

C、title 属性有一个很好的用途，即为链接添加描述性文字，特别是当连接本身并不是十分清楚的表达了链接的目的。这样就使得访问者知道那些链接将会带他们到什么地方，他们就不会加载一个可能完全不感兴趣的页面。另外一个潜在的应用就是为图像提供额外的说明信息，比如日期或者其他非本质的信息。

D、title 属性值可以比 alt 属性值设置的更长。不过要注意的是，有些浏览器会截断过长的文字（比如工具提示或其他）。比如 Mozilla 核心的浏览器只能显示最先的 60 个字符。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>测试 alt 属性和 title 属性</title>
```

```
</head>
<body>
<p>一个图像: <br/>
  一个图像: </p>
</body>
</html>
```

以下是测试结果:

图一是 alt 的显示结果:



图二是 title 的显示结果:



[HTML 标签](#)

[HTML <ins> 标签](#)

HTML <input> 标签

实例

一个简单的 HTML 表单，包含两个文本输入框和一个提交按钮：

<form action="demo_form.php"> First name: <input type="text" name="fname">
 Last name: <input type="text" name="lname">
 <input type="submit" value="提交"> </form>

尝试一下 »

(本页底部可以查看更多实例)

浏览器支持

元素					
<input>	Yes	Yes	Yes	Yes	Yes

标签定义及使用说明

<input> 标签规定了用户可以在其中输入数据的输入字段。

<input> 元素在 <form> 元素中使用，用来声明允许用户输入数据的 input 控件。

输入字段可通过多种方式改变，取决于 type 属性。

提示和注释

注意: <input> 元素是空的,它只包含标签属性。

提示: 你可以使用 <label> 元素来定义 <input> 元素的标注。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中, "align" 数据已经不再使用。HTML5 中不支持该属性。 可以使用 CSS 来定义 <input> 元素的对齐方式。
在 HTML5 中, <input> 添加了几个属性, 并且添加了对应的值。

HTML 与 XHTML 之间的差异

在 HTML 中, <input> 标签没有结束标签。
在 XHTML 中, <input> 标签必须被正确地关闭。

属性

New : HTML5 新标签。

属性	值	描述
accept	audio/* video/* image/* <i>MIME_type</i>	规定通过文件上传来提交的文件的类型。(只针对 type="file")
align	left right top middle bottom	HTML5 已废弃, 不赞成使用。规定图像输入的对齐方式。(只针对 type="image")
alt	<i>text</i>	定义图像输入的替代文本。(只针对 type="image")
autocomplete New	on off	autocomplete 属性规定 <input> 元素输入字段是否应该启用自动完成功能。

<u>autofocus</u> New	autofocus	属性规定当页面加载时 <input> 元素应该自动获得焦点。
<u>checked</u>	checked	checked 属性规定在页面加载时应该被预先选定的 <input> 元素。(只针对 type="checkbox" 或者 type="radio")
<u>disabled</u>	disabled	disabled 属性规定应该禁用的 <input> 元素。
<u>form</u> New	<i>form_id</i>	form 属性规定 <input> 元素所属的一个或多个表单。
<u>formaction</u> New	<i>URL</i>	属性规定当表单提交时处理输入控件的文件的 URL。(只针对 type="submit" 和 type="image")
<u>formenctype</u> New	application/x-www-form-urlencoded multipart/form-data text/plain	属性规定当表单数据提交到服务器时如何编码(只适合 type="submit" 和 type="image")。
<u>formmethod</u> New	get post	定义发送表单数据到 action URL 的 HTTP 方法。(只适合 type="submit" 和 type="image")
<u>formnovalidate</u> New	formnovalidate	formnovalidate 属性覆盖 <form> 元素的 novalidate 属性。
<u>formtarget</u> New	_blank _self _parent _top <i>framename</i>	规定表示提交表单后在哪里显示接收到响应的名称或关键词。(只适合 type="submit" 和 type="image")
<u>height</u> New	<i>pixels</i>	规定 <input>元素的高度。(只针对 type="image")

<u>list</u> New	<i>datalist_id</i>	属性引用 <datalist> 元素，其中包含 <input> 元素的预定义选项。
<u>max</u> New	<i>number date</i>	属性规定 <input> 元素的最大值。
<u>maxlength</u>	<i>number</i>	属性规定 <input> 元素中允许的最大字符数。
<u>min</u> New	<i>number date</i>	属性规定 <input>元素的最小值。
<u>multiple</u> New	<i>multiple</i>	属性规定允许用户输入到 <input> 元素的多个值。
<u>name</u>	<i>text</i>	name 属性规定 <input> 元素的名称。
<u>pattern</u> New	<i>regexp</i>	pattern 属性规定用于验证 <input> 元素的值的正则表达式。
<u>placeholder</u> New	<i>text</i>	placeholder 属性规定可描述输入 <input> 字段预期值的简短的提示信息。
<u>readonly</u>	<i>readonly</i>	readonly 属性规定输入字段是只读的。
<u>required</u> New	<i>required</i>	属性规定必需在提交表单之前填写输入字段。
<u>size</u>	<i>number</i>	size 属性规定以字符数计的 <input> 元素的可见宽度。

src	<i>URL</i>	src 属性规定显示为提交按钮的图像的 URL。（只针对 type="image"）
step <div>New</div>	<i>number</i>	step 属性规定 <input> 元素的合法数字间隔。
type	button checkbox color date datetime datetime-local email file hidden image month number password radio range reset search submit tel text time	type 属性规定要显示的 <input> 元素的类型。

	url week	
value	<i>text</i>	指定 <input> 元素 value 的值。
width 	<i>pixels</i>	width 属性规定 <input> 元素的宽度。(只针对 type="image")

全局属性

<input> 标签支持全局属性， 查看完整属性表 [HTML 全局属性](#)。

事件属性

<input> 标签支持所有 [HTML 事件属性](#)。

[HTML 标签](#)

[HTML <ins> 标签](#)

点我分享笔记

[HTML input 标签](#)

[HTML <kbd> 标签](#)

HTML `<ins>` 标签

实例

一段带有已删除部分和新插入部分的文本：

```
<p>My favorite color is <del>blue</del> <ins>red</ins>!</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<ins>` 标签。

标签定义及使用说明

`<ins>` 标签定义已经被插入文档中的文本。

提示和注释

提示：您也可以看看标记已删除文本的 `` 标签。

提示：`` 和 `<ins>` 一起使用，描述文档中的更新和修正。浏览器通常会在已删除文本上添加一条删除线，在新插入文本下添加一条下划线。

HTML 4.01 与 HTML5 之间的差异

无。

属性

属性	值	描述
cite	<i>URL</i>	规定一个文档的 URL，该文档解释了文本被插入的原因。
datetime	<i>YYYY-MM-DDThh:mm:ssTZD</i>	规定文本被插入的日期和时间。

全局属性

<ins> 标签支持 [HTML 的全局属性](#)。

事件属性

<ins> 标签支持 [HTML 的事件属性](#)。

[HTML input 标签](#)

[HTML <kbd> 标签](#)

点我分享笔记

[HTML <ins> 标签](#)

[HTML5 <keygen> 标签](#)

HTML <kbd> 标签

实例

在文档中格式化文本:

<kbd>键盘输入</kbd>

尝试一下 »

浏览器支持

元素					
<kbd>	Yes	Yes	Yes	Yes	Yes

标签定义及使用说明

<kbd> 标签定义键盘文本样式。

HTML 键盘输入元素 (<kbd>) 用于表示用户输入，它将产生一个行内元素，以浏览器的默认 monospace 字体显示。

提示: 不推荐使用 <kbd> 标签，更推荐使用 CSS 实现丰富的效果。

所有标签短语:

标签	描述
----	----

	呈现为被强调的文本。
	定义重要的文本。
<dfn>	定义一个定义项目。
<code>	定义计算机代码文本。
<samp>	定义样本文本。
<kbd>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<var>	定义变量。您可以将此标签与 <pre> 及 <code> 标签配合使用。

更多实例

实例

在文档中格式化文本：

<p>在对话框中输入：<kbd>cmd</kbd>
然后点击 OK 按钮。</p> <p>保存文件请使用快捷键 <kbd>Ctrl</kbd> + <kbd>S</kbd></p>

尝试一下 »

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<kbd>` 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

`<kbd>` 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML `<ins>` 标签](#)

[HTML5 `<keygen>` 标签](#)

点我分享笔记

[HTML `<kbd>` 标签](#)

[HTML `<label>` 标签](#)

HTML `<keygen>` 标签

该标签在新的 Web 标准中已废弃。

实例

带有 keygen 字段的表单：

```
<form action="demo_keygen.asp" method="get">  
  用户名: <input type="text" name="usr_name">  
  加密: <keygen name="security">  
  <input type="submit">  
</form>
```

[尝试一下 »](#)

浏览器支持



Firefox、Opera、Chrome 和 Safari 6 都支持 <keygen> 标签。

标签定义及使用说明

<keygen> 标签规定用于表单的密钥对生成器字段。

当提交表单时，私钥存储在本地，公钥发送到服务器。

HTML 4.01 与 HTML5 之间的差异

<keygen> 元素是 HTML5 新标签。

属性

New: HTML5 新属性。

属性	值	描述
autofocus New	autofocus	使 <keygen> 字段在页面加载时获得焦点。
challenge New	challenge	如果使用，则将 keygen 的值设置为在提交时询问。
disabled New	disabled	禁用 <keygen> 元素字段。
form New	<i>form_id</i>	定义该 <keygen> 字段所属的一个或多个表单。
keytype New	rsa dsa ec	定义密钥的安全算法。
name New	<i>name</i>	定义 <keygen> 元素的唯一名称。 name 属性用于在提交表单时搜集字段的值。

全局属性

<keygen> 标签支持全局属性， 查看完整属性表 [HTML 全局属性](#)。

事件属性

<keygen> 标签支持所有 [HTML 事件属性](#).

[HTML <kbd> 标签](#)

[HTML <label> 标签](#)

点我分享笔记

[HTML5 <keygen> 标签](#)

[HTML <legend> 标签](#)

HTML <label> 标签

实例

带有两个输入字段和相关标记的简单 HTML 表单：

```
<form action="demo_form.php"> <label for="male">Male</label> <input type="radio" name="sex" id="male"
value="male"><br> <label for="female">Female</label> <input type="radio" name="sex" id="female"
value="female"><br><br> <input type="submit" value="提交"> </form>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<label>` 标签。

标签定义及使用说明

`<label>` 标签为 `input` 元素定义标注（标记）。

`label` 元素不会向用户呈现任何特殊效果。不过，它为鼠标用户改进了可用性。如果您在 `label` 元素内点击文本，就会触发此控件。就是说，当用户选择该标签时，浏览器就会自动将焦点转到和标签相关的表单控件上。

`<label>` 标签的 `for` 属性应当与相关元素的 `id` 属性相同。

提示和注释

提示：“`for`” 属性可把 `label` 绑定到另外一个元素。请把 “`for`” 属性的值设置为相关元素的 `id` 属性的值。

HTML 4.01 与 HTML5 之间的差异

“`form`” 属性是 HTML5 的新属性。

属性

New: HTML5 新属性。

属性	值	描述
<code>for</code>	<code>element_id</code>	规定 <code>label</code> 与哪个表单元素绑定。

<code>form</code> New	<code>form_id</code>	规定 label 字段所属的一个或多个表单。
------------------------------	----------------------	------------------------

全局属性

<label> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<label> 标签支持所有 [HTML 事件属性](#)。

[HTML5 <keygen> 标签](#)

[HTML <legend> 标签](#)

点我分享笔记

[HTML <label> 标签](#)

[HTML 标签](#)

HTML <legend> 标签

实例

组合表单中的相关元素:

```
<form>
  <fieldset>
    <legend>Personalia:</legend>
    Name: <input type="text" size="30"><br>
    Email: <input type="text" size="30"><br>
    Date of birth: <input type="text" size="10">
  </fieldset>
</form>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<legend>` 标签。

标签定义及使用说明

The `<legend>` 元素为 `<fieldset>` 元素定义标题。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中 "align" 属性已被废弃, HTML5 不支持该属性。不建议使用。请使用 CSS 来设置 `<legend>` 元素的对齐方式。

属性

属性	值	描述
align	top bottom left right	HTML5 不支持。HTML 4.01 已废弃。不建议使用。请使用样式代替。为 fieldset 中的标题定义对齐方式。

全局属性

<legend> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<legend> 标签支持所有 [HTML 事件属性](#)。

[HTML <label> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <legend> 标签](#)

[HTML <link> 标签](#)

HTML 标签

实例

HTML 两个列表实例： 一个有序列表 () 和一个无序列表 ()：

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>

<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

[尝试一下 »](#)

(本页底部查看更多实例)

浏览器支持



目前多数主流浏览器支持 标签。

标签定义及使用说明

 标签定义列表项目。
 标签可用在有序列表（）、无序列表（）和菜单列表（<menu>）中。

HTML 4.01 与 HTML5 之间的差异

"type" 属性 在 HTML 4.01 已被废弃。HTML5 不支持该属性。
"value" 属性 在 HTML 4.01 已被废弃。HTML5 不支持该属性。

提示和注释

提示: 请使用 CSS 来定义列表和列表项目的类型。

属性

属性	值	描述
type	1 A a I i disc square circle	HTML5 不支持该属性。HTML 4.01 已废弃该属性。 不赞成使用。请使用样式取代它。 规定使用哪种项目符号。

value	<i>number</i>	不赞成使用。请使用样式取代它。规定列表项目的数字。
-----------------------	---------------	---------------------------

全局属性

 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

 标签支持所有 [HTML 事件属性](#)。



在线实例

[一个嵌套列表](#)

列表内的列表。

[另外一个嵌套列表](#)

更复杂的嵌套列表。

相关文章

HTML 教程: [HTML 列表](#)

[HTML <legend> 标签](#)

[HTML <link> 标签](#)

1 篇笔记 写笔记

1. IdanSuce
275***6743@qq.com

20

方便记忆的方法：

- o ol 的全称可能是 **ordered list**。
- o ul 的全称可能是 **unordered list**。
- o li 的全称可能是 **list**。

[IdanSuce](#)

IdanSuce
275***6743@qq.com

1 年前 (2022-11-11)

[HTML 标签](#)

[HTML <main> 标签](#)

HTML <link> 标签

实例

链接到外部样式文件：


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

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<link>` 标签。

标签定义及使用说明

`<link>` 标签定义文档与外部资源的关系。

`<link>` 标签最常见的用途是链接样式表。

注意： `link` 元素是空元素，它仅包含属性。

注意： 此元素只能存在于 `head` 部分，不过它可出现任何次数。

HTML 4.01 与 HTML5 之间的差异

一些 HTML 4.01 属性在 HTML5 中不支持。

HTML5 新增了 `"sizes"` 属性。

HTML 与 XHTML 之间的差异

在 HTML 中，`<link>` 标签没有结束标签。

在 XHTML 中，`<link>` 标签必须被正确地关闭。

属性

New: HTML5 新属性。

属性	值	描述
charset	<i>char_encoding</i>	HTML5 不支持该属性。 定义被链接文档的字符编码方式。
href	<i>URL</i>	定义被链接文档的位置。
hreflang	<i>language_code</i>	定义被链接文档中文本的语言。
media	<i>media_query</i>	规定被链接文档将显示在什么设备上。
rel	alternate archives author bookmark external first help icon last license next nofollow norereferrer	必需。定义当前文档与被链接文档之间的关系。rel 是 relationship 的英文缩写。

	pingback prefetch prev search sidebar stylesheet tag up	
rev	<i>reversed relationship</i>	HTML5 不支持该属性。定义被链接文档与当前文档之间的关系。
sizes 	<i>HeightxWidth</i> any	定义了链接属性大小，只对属性 rel="icon" 起作用。
target	_blank _self _top _parent <i>frame_name</i>	HTML5 不支持该属性。定义在何处加载被链接文档。
type	<i>MIME_type</i>	规定被链接文档的 MIME 类型。

全局属性

<link> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<link> 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程: [HTML 样式](#)

HTML DOM 参考手册: [Link 对象](#)

[HTML 标签](#)

[HTML <main> 标签](#)

点我分享笔记

[HTML <link> 标签](#)

[HTML <map> 标签](#)

HTML main 标签

实例

使用 main 标签来展示文档的主体部分:

```
<main> <h1>Web 浏览器</h1> <p>Google Chrome、Firefox 以及 Internet Explorer 是目前最流行的浏览器。</p> <article>
<h1>Google Chrome 浏览器</h1> <p>Google Chrome 浏览器是由 Google 开发的一款免费的开源 web 浏览器，于 2008 年发布。</p>
</article> <article> <h1>Internet Explorer 浏览器</h1> <p>Internet Explorer 浏览器由微软开发的一款免费的 web 浏览器，
```

发布于 1995 年。

```
</p> </article> <article> <h1>Mozilla Firefox 浏览器</h1> <p>Firefox 浏览器是一款来自 Mozilla 的免费开源 web 浏览器，发布于 2004 年。</p> </article> </main>
```

[尝试一下 »](#)

定义和用法

`<main>` 标签用于指定文档的主体内容。

`<main>` 标签中的内容在文档中是唯一的。它不应包含在文档中重复出现的内容，比如侧栏、导航栏、版权信息、站点标志或搜索表单。注意在一个文档中，`<main>` 元素是唯一的，所以不能出现一个以上的 `<main>` 元素。`<main>` 元素不能是以下元素的后代：`<article>`、`<aside>`、`<footer>`、`<header>` 或 `<nav>`。

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

元素					
<main>	6.0	12.0	4.0	5.0	11.1

HTML 4.01 与 HTML5 之间的差异

`<main>` 标签是 HTML5 中新增加的。

[HTML <link>标签](#)

[HTML <map> 标签](#)

点我分享笔记

[HTML <main> 标签](#)

[HTML5 <mark> 标签](#)

HTML <map> 标签

实例

带有可点击区域的图像映射：

```

```

```
<map name="planetmap">
```

```
  <area shape="rect" coords="0,0,82,126" href="sun.htm" alt="Sun">
```

```
  <area shape="circle" coords="90,58,3" href="mercur.htm" alt="Mercury">
```

```
  <area shape="circle" coords="124,58,8" href="venus.htm" alt="Venus">
```

```
</map>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 <map> 标签。

标签定义及使用说明

<map> 标签用于客户端图像映射。图像映射指带有可点击区域的一幅图像。

中的 usemap 属性可引用 <map> 中的 id 或 name 属性（取决于浏览器），所以我们应同时向 <map> 添加 id 和 name 属性。

area 元素永远嵌套在 map 元素内部。area 元素可定义图像映射中的区域。

HTML 4.01 与 HTML5 之间的差异

注意: 在 HTML5 中, 如果 id 属性在<map> 标签中指定, 则你必须同样指定 name 属性。

HTML 与 XHTML 之间的差异

在 XHTML 中, name 属性已经废弃, 使用 id 属性替换它。

属性

属性	值	描述
name	<i>mapname</i>	必需。为 image-map 规定的名称。

全局属性

<map> 标签支持全局属性, 查看完整属性表 [HTML 全局属性](#)。

事件属性

<map> 标签支持所有 [HTML 事件属性](#)。

[HTML <main> 标签](#)

[HTML5 <mark> 标签](#)

点我分享笔记

[HTML <map> 标签](#)

[HTML <menu> 标签](#)

HTML5 <mark> 标签

实例

部分文本高亮显示：

```
<p>Do not forget to buy <mark>milk</mark> today.</p>
```

[尝试一下 »](#)

浏览器支持



Internet Explorer 9+、Firefox、Opera、Chrome 和 Safari 支持 <mark> 标签。

注意：Internet Explorer 8 及更早版本不支持 <mark> 标签。

标签定义及使用说明

<mark> 标签定义带有记号的文本。

请在需要突出显示文本时使用 <mark> 标签。

HTML 4.01 与 HTML5 之间的差异

<mark> 是 HTML5 新标签。

全局属性

<mark> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<mark> 标签支持所有 [HTML 事件属性](#)。

[HTML <map> 标签](#)

[HTML <menu> 标签](#)

点我分享笔记

[HTML5 <mark> 标签](#)

[HTML <meta> 标签](#)

HTML <menu> 标签

实例

两个菜单按钮系列选项实例 ("File" 和 "Edit") :

```
<menu type="toolbar"> <li> <menu label="File"> <button type="button" onclick="file_new()">New...</button>
<button type="button" onclick="file_open()">Open...</button> <button type="button"
onclick="file_save()">Save</button> </menu> </li> <li> <menu label="Edit"> <button type="button"
onclick="edit_cut()">Cut</button> <button type="button" onclick="edit_copy()">Copy</button> <button
type="button" onclick="edit_paste()">Paste</button> </menu> </li> </menu>
```

[尝试一下 »](#)

浏览器支持



目前主流浏览器并不支持 <menu> 标签。

标签定义及使用说明

<menu> 标签定义了一个命令列表或菜单。

<menu> 标签通常用于文本菜单，工具条和命令列表选项。

提示和注释

提示: 使用 CSS 来定义菜单列表样式。

HTML 4.01 与 HTML5 之间的差异

HTML 4.01 的 <menu> 元素已废弃。
HTML5 中 <menu> 元素已被重新定义。

属性

New: HTML5 新属性。

属性	值	描述
label New	<i>text</i>	描述菜单项的标记。
type New	context toolbar list	描述显示菜单类型. 默认为 "list"。

全局属性

<menu> 标签支持全局属性, 查看完整属性表 [HTML 全局属性](#)。

事件属性

<menu> 标签支持所有 [HTML 事件属性](#)。

[HTML5 <mark> 标签](#)

[HTML <meta> 标签](#)

点我分享笔记

[HTML <menu> 标签](#)

[HTML <meter> 标签](#)

HTML <meta> 标签

实例

描述 HTML 文档的元数据：

```
<head> <meta name="description" content="免费在线教程"> <meta name="keywords" content="HTML,CSS,XML,JavaScript">
<meta name="author" content="runoob"> <meta charset="UTF-8"> </head>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <meta> 标签。

标签定义及使用说明

元数据（Metadata）是数据的数据信息。

`<meta>` 标签提供了 HTML 文档的元数据。元数据不会显示在客户端，但是会被浏览器解析。

META 元素通常用于指定网页的描述，关键词，文件的最后修改时间，作者及其他元数据。

元数据可以被使用浏览器（如何显示内容或重新加载页面），搜索引擎（关键词），或其他 Web 服务调用。

提示和注释

注意：`<meta>` 标签通常位于 `<head>` 区域内。

注意：元数据通常以 名称/值 对出现。

注意：如果没有提供 `name` 属性，那么名称/值对中的名称会采用 `http-equiv` 属性的值。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 `scheme` 属性。

在 HTML5 中，有一个新的 `charset` 属性，它使字符集的定义更加容易：

- HTML 4.01: `<meta http-equiv="content-type" content="text/html; charset=UTF-8">`
 - HTML5: `<meta charset="UTF-8">`
-

HTML 与 XHTML 之间的差异

在 HTML 中 `<meta>` 标签没有结束标签。

在 XHTML 中 `<meta>` 标签必须包含结束标签。

实例

实例 1 - 定义文档关键词，用于搜索引擎：

```
<meta name="keywords" content="HTML, CSS, XML, XHTML, JavaScript">
```

实例 2 - 定义 web 页面描述：

```
<meta name="description" content="Free Web tutorials on HTML and CSS">
```

实例 3 - 定义页面作者：

```
<meta name="author" content="Hege Refsnes">
```

实例 4 - 每 30 秒刷新页面：

```
<meta http-equiv="refresh" content="30">
```

属性

New：HTML5 新属性。

属性	值	描述
charset New	<i>character_set</i>	定义文档的字符编码。
content	<i>text</i>	定义与 http-equiv 或 name 属性相关的元信息。
http-equiv	content-type default-style refresh	把 content 属性关联到 HTTP 头部。

name	application-name author description generator keywords	把 content 属性关联到一个名称。
scheme	<i>format/URI</i>	HTML5 不支持。 定义用于翻译 content 属性值的格式。

事件属性

<meta> 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程: [HTML 头部](#)

[HTML <menu> 标签](#)

[HTML <meter> 标签](#)

2 篇笔记 写笔记

1. 柒沐
743***609@qq.com

```
<meta name="HandheldFriendly" content="true">
<!-- 微软的老式浏览器 -->
<meta name="MobileOptimized" content="320">
<!-- uc 强制竖屏 -->
<meta name="screen-orientation" content="portrait">
<!-- QQ 强制竖屏 -->
<meta name="x5-orientation" content="portrait">
<!-- UC 强制全屏 -->
<meta name="full-screen" content="yes">
<!-- QQ 强制全屏 -->
<meta name="x5-fullscreen" content="true">
<!-- UC 应用模式 -->
<meta name="browsermode" content="application">
<!-- QQ 应用模式 -->
<meta name="x5-page-mode" content="app">
<!-- windows phone 点击无高光 -->
```

柒沐

柒沐

743***609@qq.com

7 年前 (2017-06-26)

2. brysj22952

huz***gqiang2010@hotmail.com

148

还有一个 **viewport** 属性，这里面没有提到，它用于移动端显示优化的。

可以参考这篇文章：[viewport 深入理解](https://www.runoob.com/w3cnote/viewport-deep-understanding.html) (<https://www.runoob.com/w3cnote/viewport-deep-understanding.html>)

里面提到 6 个属性：

width	设置 <i>layout viewport</i> 的宽度，为一个正整数，或字符串"width-device"
initial-scale	设置页面的初始缩放值，为一个数字，可以带小数
minimum-scale	允许用户的最小缩放值，为一个数字，可以带小数
maximum-scale	允许用户的最大缩放值，为一个数字，可以带小数
height	设置 <i>layout viewport</i> 的高度，这个属性对我们并不重要，很少使用
user-scalable	是否允许用户进行缩放，值为 "no" 或 "yes", no 代表不允许，yes 代表允许

还有一个属性：minimal-ui，参考：

iOS 7.1 的 Safari 为 meta 标签新增 minimal-ui 属性，在网页加载时隐藏地址栏与导航栏（<http://36kr.com/p/210516.html>），这个在 iOS8 中已经废弃了。

[brysj22952](#)

brysj22952

huz***gqiang2010@hotmail.com

6 年前 (2018-07-22)

[HTML <meta> 标签](#)

[HTML <nav> 标签](#)

HTML <meter> 标签

实例

使用 meter 元素展示给定的数据范围：

<meter value="2" min="0" max="10">2 out of 10</meter>
 <meter value="0.6">60%</meter>

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该元素的第一个浏览器的版本号。

元素					
<meter>	8.0	13.0	6.0	6.0	11.0

标签定义及使用说明

<meter> 标签定义度量衡。仅用于已知最大和最小值的度量。
比如：磁盘使用情况，查询结果的相关性等。
注意： <meter> 不能作为一个进度条来使用， 进度条 [<progress>](#) 标签。

HTML 4.01 与 HTML5 之间的差异

<meter> 是 HTML5 的新标签。

属性

New: HTML5 新属性。

属性	值	描述
<u>form</u> New	<i>form_id</i>	规定 <meter> 元素所属的一个或多个表单。
<u>high</u> New	<i>number</i>	规定被界定为高的值的范围。
<u>low</u> New	<i>number</i>	规定被界定为低的值的范围。
<u>max</u> New	<i>number</i>	规定范围的最大值。
<u>min</u> New	<i>number</i>	规定范围的最小值。
<u>optimum</u> New	<i>number</i>	规定度量的最优值。
<u>value</u> New	<i>number</i>	必需。规定度量的当前值。

全局属性

<meter> 标签支持全局属性， 查看完整属性表 [HTML 全局属性](#)。

事件属性

<meter> 标签支持所有 [HTML 事件属性](#)。

[HTML <meta> 标签](#)

[HTML <nav> 标签](#)

点我分享笔记

[HTML <meter> 标签](#)

[HTML <noframes> 标签](#)

HTML <nav> 标签

实例

一个导航链接实例：

```
<nav> <a href="/html/">HTML</a> | <a href="/css/">CSS</a> | <a href="/js/">JavaScript</a> | <a href="/jquery/">jQuery</a> </nav>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<nav>` 标签。

标签定义及使用说明

`<nav>` 标签定义导航链接的部分。

并不是所有的 HTML 文档都要使用到 `<nav>` 元素。`<nav>` 元素只是作为标注一个导航链接的区域。在不同设备上（手机或者 PC）可以制定导航链接是否显示，以适应不同屏幕的需求。

HTML 4.01 与 HTML5 中的差异

`<nav>` 是 HTML5 的新标签。

全局属性

`<nav>` 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

`<nav>` 标签支持所有 [HTML 事件属性](#)。

[HTML <meter> 标签](#)

[HTML <noframes> 标签](#)

点我分享笔记

[HTML <nav> 标签](#)

[HTML <noscript> 标签](#)

HTML <noframes> 标签 HTML5 不支持该标签

实例

显示三个 frame 框架，如果不支持 frame 输出 <noframes> 标签的文本：

```
<html>

<frameset cols="25%,50%,25%">
  <frame src="frame_a.htm">
  <frame src="frame_b.htm">
  <frame src="frame_c.htm">
  <noframes>Sorry, your browser does not handle frames!</noframes>
</frameset>

</html>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 <noframes> 标签。

标签定义及使用说明

HTML5 不支持 <noframes> 标签。

<noframes> 元素可为那些不支持框架的浏览器显示文本。noframes 元素位于 frameset 元素内部。

<noframes> 元素插入在 <frameset> 元素中使用。

注意： 如果您希望验证包含框架的页面，请确保 DTD 被设置为 "Frameset DTD"。

HTML 4.01 与 HTML5 的差异

HTML5 不支持 <noframes> 标签，HTML 4.01 支持该标签。

HTML 与 XHTML 的差异

重要： 在 XHTML Frameset DTD，位于 <noframes> 元素中的文本信息必须有关闭标签。

标准属性

在 HTML 4.01 中，<noframes> 支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向

id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息
xml:lang	<i>language_code</i>	规定 XHTML 文档中元素内容的语言代码

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<noframes> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本

onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML <nav> 标签](#)

[HTML <noscript> 标签](#)

点我分享笔记

[HTML <noframes> 标签](#)

[HTML <object> 标签](#)

HTML <noscript> 标签

实例

<noscript> 标签的使用：

```
<script> document.write("Hello World!") </script> <noscript>抱歉，你的浏览器不支持 JavaScript!</noscript>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<noscript>` 标签。

标签定义及使用说明

`noscript` 元素用来定义在脚本未被执行时的替代内容（文本）。

此标签可被用于可识别 `<noscript>` 标签但无法支持其中的脚本的浏览器。

提示和注释

提示：如果浏览器支持脚本，那么它不会显示出 `noscript` 元素中的文本。

注释：无法识别 `<script>` 标签的浏览器会把标签的内容显示到页面上。为了避免浏览器这样做，您应当在注释标签中隐藏脚本。老式的（无法识别 `<script>` 标签的）浏览器会忽略注释，这样就不会把标签的内容写到页面上，而新式的浏览器则懂得执行这些脚本，即使它们被包围在注释标签中！

```
<script>
<!--
function displayMsg()
{
    alert("Hello World!")
}
//-->
</script>
```

在 HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，<noscript> 标签只允许插入到 <body> 元素中。

在 HTML5 中，<noscript> 标签可以插入到 <head> 和 <body> 区域中。

在 HTML 与 XHTML 之间的差异

XHTML 不支持 <noscript> 标签。

全局属性

<noscript> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

相关文章

HTML 教程: [HTML 脚本](#)

[HTML <noframes> 标签](#)

[HTML <object> 标签](#)

1 篇笔记 写笔记

noscript 标签可以包含不仅仅是文本的内容，并且其中内容在脚本可用时全无意义，如：

1. 一个段落，或其它可视的内容，例如 frame。
2. 一个提示框，比如类似 Facebook 的提示页：
3. 一些动态执行的内容，如 style, link 甚至 script（但最后这个当然往往无用……）。

灵活运用这个标签可以实现很多功能。比如：

```
<noscript><style>script{ display:none }</style></noscript>
```

这理论上能够防止浏览器把 script 内的内容当作文本显示。因为在浏览器无法使用脚本时，style 标签生效，将可能被认为是文本的 script 标签隐藏（实际上它们本来就应当是隐藏的）。

但一般来说，既然浏览器支持 noscript，那么应该也不至于将脚本的内容直接输出（仅仅是不运行而已），因此这通常是不必要的。若要防止脚本内容被输出，应当在脚本内加上注释标签。

但在实际项目中，即使果真无法使用脚本，noscript 也能防止出现糟糕的用户体验，请看下例：

```
<noscript>  
<p>警告：您的浏览器不支持或您禁止了脚本。</p>  
<style>  
#content { display:none }
```

```
</style>
</noscript>
<div id="content">
  <p>我是页面本来的内容! </p>
  <script>document.write("我是页面本来的脚本! ")</script>
</div>
```

如果脚本失效，页面原先的依赖于脚本的内容可能出现各种奇怪的问题。但在 `noscript` 内的任何元素，都只会在禁止脚本的时候发挥作用，而平时它们大致相当于注释。

因此使用上述方式，即可在脚本不可用时提示错误信息，并隐藏原有内容以防止错误发生。

用此方法也可构建一个提示框之类的区域，并可以指示用户如何打开脚本。但要注意基于脚本的框架不一定能在没有脚本的情况下运作，不过仅使用 `CSS` 也能做出比较美观的布局。



[HTML <noscript> 标签](#)

[HTML 标签](#)

HTML <object> 标签

实例

使用<object> 元素在 HTML 加入 Flash 文件：

```
<object width="400" height="400" data="helloworld.swf"></object>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<object>` 标签。

标签定义及使用说明

定义一个嵌入的对象。请使用此元素向您的 XHTML 页面添加多媒体。此元素允许您规定插入 HTML 文档中的对象的数据和参数，以及可用来显示和操作数据的代码。

`<object>` 标签用于包含对象，比如图像、音频、视频、Java applets、ActiveX、PDF 以及 Flash。

`object` 的初衷是取代 `img` 和 `applet` 元素。不过由于漏洞以及缺乏浏览器支持，这一点并未实现。

浏览器的对象支持有赖于对象类型。不幸的是，主流浏览器都使用不同的代码来加载相同的对象类型。

而幸运的是，`object` 对象提供了解决方案。如果未显示 `object` 元素，就会执行位于 `<object>` 和 `</object>` 之间的代码。通过这种方式，我们能够嵌套多个 `object` 元素（每个对应一个浏览器）。

HTML 4.01 与 HTML5 中的差异

一些 HTML 4.01 属性在 HTML5 中不被支持。

"form" 是 HTML5 定义的新属性。

在 HTML5 中，`objects` 可以在 `form` 表单中提交。

在 HTML5 中，`objects` 不再出现在 `<head>` 元素区域内。

属性

New: HTML5 新属性。

属性	值	描述
<u>align</u>	top bottom middle left right	HTML5 不支持。HTML 4.01 已废弃。规定 <object> 元素相对于周围元素的对齐方式。
archive	URL	HTML5 不支持。由空格分隔的指向档案文件的 URL 列表。这些档案文件包含了与对象相关的资源。
<u>border</u>	pixels	HTML5 不支持。HTML 4.01 已废弃。规定 <object> 周围的边框宽度。
classid	class_ID	HTML5 不支持。定义嵌入 Windows Registry 中或某个 URL 中的类的 ID 值，此属性可用来指定浏览器中包含的对象的位置，通常是一个 Java 类。
codebase	URL	HTML5 不支持。定义在何处可找到对象所需的代码，提供一个基准 URL。
codetype	MIME_type	HTML5 不支持。通过 classid 属性所引用的代码的 MIME 类型。
<u>data</u>	URL	规定对象使用的资源的 URL。
declare	declare	HTML5 不支持。定义该对象仅可被声明，但不能被创建或例示，直到该对象得到应用为止。
<u>form</u> New	form_id	规定对象所属的一个或多个表单。
<u>height</u>	pixels	规定对象的高度。

hspace	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。规定对象左侧和右侧的空白。
name	<i>name</i>	为对象规定名称。
standby	<i>text</i>	HTML5 不支持。定义当对象正在加载时所显示的文本。
type	<i>MIME_type</i>	规定 data 属性中规定的数据的 MIME 类型。
usemap	<i>#mapname</i>	规定与对象一同使用的客户端图像映射的名称。
vspace	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。规定对象的顶部和底部的空白。
width	<i>pixels</i>	规定对象的宽度。

全局属性

<object> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<object> 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程： [HTML Object 元素](#)

[HTML <noscript> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <object> 标签](#)

[HTML <optgroup> 标签](#)

HTML 标签

实例

2 个不同的有序列表实例：

```
<ol> <li>Coffee</li> <li>Tea</li> <li>Milk</li> </ol> <ol start="50"> <li>Coffee</li> <li>Tea</li>
<li>Milk</li> </ol>
```

[尝试一下 »](#)

浏览器支持

元素					
	Yes	Yes	Yes	Yes	Yes

标签定义及使用说明

 标签定义了一个有序列表. 列表排序以数字来显示。
使用 标签来定义列表选项。

提示和注释

提示: 可以使用 CSS 来渲染, 详细查看 [CSS 列表](#)。
提示:无序列表, 可以使用 标签。

HTML 4.01 与 HTML5 中的差异

"reversed" 属性是 HTML5 中的新属性。
在 HTML 4.01 中"compact" 属性已经废弃,在 HTML5 中不支持该属性。

属性

New: HTML5 新属性。

属性	值	描述
compact	compact	HTML5 中不支持, 不赞成使用。请使用样式取代它。 规定列表呈现的效果比正常情况更小巧。
reversed New	reversed	指定列表倒序(9,8,7...)

<u>start</u>	<i>number</i>	一个整数值属性，指定了列表编号的起始值。 这个属性在 HTML4 中弃用，但是在 HTML5 中被重新引入。
<u>type</u>	<ul style="list-style-type: none">• a 表示小写英文字母编号• A 表示大写英文字母编号• i 表示小写罗马数字编号• I 表示大写罗马数字编号• 1 表示数字编号（默认）	规定列表的类型。不赞成使用。请使用样式代替。

更多实例

实例

设置不同的列表样式(使用 CSS):

```
<ol style="list-style-type:upper-roman"> <li>Coffee</li> <li>Tea</li> <li>Milk</li> </ol> <ol style="list-style-type:lower-alpha"> <li>Coffee</li> <li>Tea</li> <li>Milk</li> </ol>
```

尝试一下 »

实例

使用 CSS 显示不同的列表样式:

```
ol.a {list-style-type: armenian;} ol.b {list-style-type: cjk-ideographic;} ol.c {list-style-type: decimal;}  
ol.d {list-style-type: decimal-leading-zero;} ol.e {list-style-type: georgian;} ol.f {list-style-type:  
hebrew;} ol.g {list-style-type: hiragana;} ol.h {list-style-type: hiragana-iroha;} ol.i {list-style-type:  
katakana;} ol.j {list-style-type: katakana-iroha;} ol.k {list-style-type: lower-alpha;} ol.l {list-style-type:  
lower-greek;} ol.m {list-style-type: lower-latin;} ol.n {list-style-type: lower-roman;} ol.o {list-style-type:  
upper-alpha;} ol.p {list-style-type: upper-latin;} ol.q {list-style-type: upper-roman;} ol.r {list-style-type:  
none;} ol.s {list-style-type: inherit;}
```

尝试一下 »

实例

列表嵌套:

```
<ol> <li>Coffee</li> <li>Tea <ul> <li>Black tea</li> <li>Green tea</li> </ul> </li> <li>Milk</li> </ol>
```

尝试一下 »

全局属性

 标签支持全局属性, 查看完整属性表 [HTML 全局属性](#)。

事件属性

 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程: [HTML 列表](#)

[HTML <object> 标签](#)

[HTML <optgroup> 标签](#)

点我分享笔记

[HTML 标签](#)

[HTML <option> 标签](#)

HTML <optgroup> 标签

实例

通过 <optgroup> 标签把相关的选项组合在一起:

```
<select>
  <optgroup label="Swedish Cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
```

```
</optgroup>
<optgroup label="German Cars">
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</optgroup>
</select>
```

[尝试一下 »](#)

浏览器支持



大多数主流浏览器支持 `<optgroup>` 标签。

标签定义及使用说明

`<optgroup>` 标签经常用于把相关的选项组合在一起。

如果你有很多选项组合, 你可以使用 `<optgroup>` 标签能够很简单的将相关选项组合在一起。

HTML 4.01 与 HTML5 中的差异

无。

属性

属性	值	描述
disabled	disabled	规定禁用该选项组。
label	<i>text</i>	为选项组规定描述。

全局属性

<optgroup> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<optgroup> 标签支持所有 [HTML 事件属性](#)。

[HTML 标签](#)

[HTML <option> 标签](#)

点我分享笔记

[HTML <optgroup> 标签](#)

[HTML <output> 标签](#)

HTML <option> 标签

实例

创建带有 4 个选项的选择列表：

```
<select>
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="opel">Opel</option>
  <option value="audi">Audi</option>
</select>
```

[尝试一下 »](#)

浏览器支持



大多数主流浏览器支持 `<option>` 标签。

标签定义及使用说明

The `<option>` 标签定义下拉列表中的一个选项（一个条目）。

`<option>` 标签中的内容作为 `<select>` 或者 `<datalist>` 一个元素使用。

提示和注释

注释： `<option>` 标签可以在不带有任何属性的情况下使用，但是您通常需要使用 `value` 属性，此属性会指示出被送往服务器的内容。

注释： 请与 `select` 元素配合使用此标签，否则这个标签是没有意义的。

提示：如果列表选项很多，可以使用 <optgroup> 标签对相关选项进行组合。

属性

属性	值	描述
disabled	disabled	规定此选项应在首次加载时被禁用。
label	<i>text</i>	定义当使用 <optgroup> 时所使用的标注。
selected	selected	规定选项（在首次显示在列表中时）表现为选中状态。
value	<i>text</i>	定义送往服务器的选项值。

全局属性

<option> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<option> 标签支持所有 [HTML 事件属性](#)。

[HTML <optgroup> 标签](#)

[HTML <output> 标签](#)

点我分享笔记

[HTML <option> 标签](#)

[HTML <p> 标签](#)

HTML <output> 标签

实例

将计算结果显示在 <output> 元素中:

<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">0 <input type="range" id="a" value="50">100
+<input type="number" id="b" value="50"> =<output name="x" for="a b"></output>

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该属性的第一个浏览器版本号。

元素					
<output>	10.0	13.0	4.0	5.1	11.0

标签定义及使用说明

<output> 标签作为计算结果输出显示(比如执行脚本的输出)。

在 HTML 4.01 与 HTML5 中的差异

<output> 标签是 HTML 5 中的新标签。

属性

New: HTML5 新属性。

属性	值	描述
<u>for</u> New	<i>element_id</i>	描述计算中使用的元素与计算结果之间的关系。
<u>form</u> New	<i>form_id</i>	定义输入字段所属的一个或多个表单。
<u>name</u> New w	<i>name</i>	定义对象的唯一名称（表单提交时使用）。

全局属性

<output> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<output> 标签支持所有 [HTML 事件属性](#)。

[HTML <option> 标签](#)

[HTML <p> 标签](#)

点我分享笔记

[HTML <output> 标签](#)

[HTML <param> 标签](#)

HTML <p> 标签

实例

以下代码标记了一个段落：

```
<p>这是一个段落。</p>
```

[尝试一下 »](#)

(在页面下部，您可以找到更多实例)

浏览器支持



目前大多数浏览器支持 <p> 标签。

标签定义及使用说明

<p> 标签定义段落。
<p>元素会自动在其前后创建一些空白。浏览器会自动添加这些空间，您也可以在样式表中规定。

在 HTML 4.01 与 HTML5 中的差异

HTML 4.01 中标签的 align 属性已经废弃，HTML5 不支持该属性。

属性

属性	值	描述
align	left right center justify	HTML5 不支持。HTML 4.01 已废弃。 不赞成使用。请使用样式取代它。规定段落中文本的对齐方式。

全局属性

<p> 标签支持全局属性， 查看完整属性表 [HTML 全局属性](#)。

事件属性

<p> 标签支持所有 [HTML 事件属性](#)。

相关文章

HTML 教程: [HTML 段落](#)

[HTML <output> 标签](#)

[HTML <param> 标签](#)

点我分享笔记

[HTML <p> 标签](#)

[HTML <pre> 标签](#)

HTML <param> 标签

实例

设置参数 "autoplay" 为 "true", 音频载入后会自动播放:

```
<object data="horse.wav">  
  <param name="autoplay" value="true">  
</object>
```

[尝试一下 »](#)

浏览器支持



<param> 标签支持大部分主流浏览器。但是 <object> 定义的文件格式不是所有的浏览器都支持。

标签定义及使用说明

<param>元素允许您为插入 XHTML 文档的对象规定 run-time 设置，也就是说，此标签可为包含它的 [<object>](#) 或者 [<applet>](#) 标签提供参数。

在 HTML 4.01 与 HTML5 之间的差异

HTML 4.01 属性: "type" 和 "valuetype", 在 HTML5 中不支持。

Differences Between HTML and XHTML

在 HTML 中，<param> 标签没有结束标签。

在 XHTML 中，<param> 标签必须被正确地关闭。

属性

属性	值	描述
name	<i>name</i>	定义参数的名称（用在脚本中）。
type	<i>MIME_type</i>	HTML5 不支持。 定义 MIME 类型参数。
value	<i>value</i>	描述参数值。

valuetype	data ref object	HTML5 不支持。描述值的类型。
-----------	-----------------------	-------------------

全局属性

<param> 标签支持全局属性，查看完整属性表 [HTML 全局属性](#)。

事件属性

<param> 标签支持所有 [HTML 事件属性](#)。

[HTML <p> 标签](#)

[HTML <pre> 标签](#)

点我分享笔记

[HTML <param> 标签](#)

[HTML <html> 标签](#)

HTML <pre> 标签

实例

预格式化的文本：

```
<pre>
```

此例演示如何使用 `pre` 标签

对空行和 空格

进行控制

```
</pre>
```

[尝试一下 »](#)

浏览器支持



目前大多数浏览器支持 `<pre>` 标签。

标签定义及使用说明

`<pre>` 标签可定义预格式化的文本。

被包围在 `<pre>` 标签 元素中的文本通常会保留空格和换行符。而文本也会呈现为等宽字体。

提示和注释

提示：`<pre>` 标签的一个常见应用就是用来表示计算机的源代码。

在 HTML 4.01 与 HTML5 中的差异

在 HTML 4.01 中, "width" 属性已废弃, 不可使用。HTML5 不支持"width"属性。

属性

属性	值	描述
width	<i>number</i>	HTML5 不支持该属性。HTML 4.01 已废弃该属性。定义每行的最大字符数（通常是 40、80 或 132）。

全局属性

<pre> 标签支持全局属性, 查看完整属性表 [HTML 全局属性](#)。

事件属性

<pre> 标签支持所有 [HTML 事件属性](#)。

[HTML <param> 标签](#)

[HTML <html> 标签](#)

2 篇笔记 写笔记

1. WH-KING
104***5930@qq.com

- o `pre` 元素是块级元素，但是只能包含文本或行内元素。也就是说，任何块级元素（常见为可以导致段落断开的标签）都不能位于 `pre` 元素中。
- o `pre` 元素中允许的文本可以包含物理样式和基于内容的样式变化，还有链接、图像和水平分隔线。当把其他标签，比如 `<a>` 标签放到 `<pre>` 块中时，就像放在 HTML/XHTML 文档的其他部分中一样即可。
- o 制表符 `tab` 在 `<pre>` 标签定义的块当中可以起到应有的作用，每个制表符占据 8 个字符的位置，但并不推荐使用 `tab`，因为在不同的浏览器中，`tab` 的表现形式各不相同。在用 `<pre>` 标签格式化的文档段中使用空格，可以确保文本正确的水平位置。
- o 如果希望使用 `<pre>` 标签来定义计算机源代码，比如 HTML 源代码，请使用符号实体来表示特殊字符。一般将 `<pre>` 标签与 `<code>` 标签结合起来使用，以获得更加精确的语义，用于标记页面中需要呈现的源代码。
- o 如果想要把某一段规定好的文本格式放在 HTML 中，那么就要利用 `pre` 元素的特性。

WH-KING

WH-KING

104***5930@qq.com

参考地址

7 年前 (2017-09-12)

2. 技术大神养成中

935***287@qq.com

- 1、该标签可定义预格式化的文本，被包围在 `pre` 元素中的文本通常会保留空格和换行符，并且文本会呈现为等宽字体。该标签的一个常见应用就是用来表示计算机的源代码。
- 2、该元素中允许的文本可以包括物理样式和基于内容的样式变化，还有链接、图像和水平分隔线。当把其他标签（比如 `<a>` 标签）放到 `<pre>` 块中时，就像放在 HTML/XHTML 文档的其他部分中一样即可。
- 3、标签中的特殊符号被转换为符号实体，比如 `"<"` 代表 `"<"`，`">"` 代表 `">"`。
- 4、可以导致段落断开的标签（比如 `<h1>`、`<p>` 和 `<address>` 标签）尽量不要包含在 `<pre>` 所定义的块里，我试过现在的浏览器（Google、IE 和火狐），虽然可以把段落结束标签解释为简单地换行，但是在代码编辑器里会报错 `"Invalid location of tag (h1)."`

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
</head>
<body>
  <pre>你好 我是 Natalie

  我即将成为<h1>一名技术</h1>大神。
```

以下是火狐浏览器运行的结果

你好 我是Natalie

我即将成为

一名技术

大神。

[HTML <pre> 标签](#)

[HTML5 <picture> 标签](#)

HTML **<html>** 标签

实例

简单的 HTML5 文档:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>文档标题</title>
</head>

<body>
```

文档内容.....

</body>

</html>

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <html> 标签。

标签定义及使用说明

<html> 标签告知浏览器这是一个 HTML 文档。

<html> 标签是 HTML 文档中最外层的元素。

<html> 标签是所有其他 HTML 元素（除了 [<!DOCTYPE>](#) 标签）的容器。

HTML 4.01 与 HTML5 之间的差异

HTML5 中，增加了一个新属性：manifest。

HTML 与 XHTML 之间的差异

xmlns 属性在 XHTML 中是必需的，但在 HTML 中不是。

然而，即使 XHTML 文档中的 <html> 没有使用 xmlns 属性，W3C 上的 HTML 验证器也不会报错。这是因为 "xmlns=http://www.w3.org/1999/xhtml" 是一个固定值，即使您没有包含它，此值也会被添加到 <html> 标签中。

属性

New：HTML5 中的新属性。

| 属性 | 值 | 描述 |
|-------------------------------------|------------------------------|---|
| manifest New | <i>URL</i> | 定义一个 URL，在这个 URL 上描述了文档的缓存信息。 |
| xmlns | http://www.w3.org/1999/xhtml | HTML 不支持。只有 XHTML 支持。 规定 XML 的 namespace 属性（如果您需要您的内容符合 XHTML，则使用这个属性。）。 |

全局属性

<html> 标签支持 **HTML 的全局属性**。

[HTML <pre> 标签](#)

[HTML5 <picture> 标签](#)

点我分享笔记

[HTML <html> 标签](#)

[HTML5 progress 标签](#)

HTML <picture> 元素

实例

根据屏幕匹配的不同尺寸显示不同图片，如果没有匹配到或浏览器不支持 picture 属性则使用 img 元素：

```
<picture> <source media="(min-width: 650px)" srcset="demo1.jpg"> <source media="(min-width: 465px)"
srcset="demo2.jpg">  </picture>
```

[尝试一下 »](#)

定义

picture 元素允许我们在不同的设备上显示不同的图片，一般用于响应式。

HTML5 引入了 <picture> 元素，该元素可以让图片资源的调整更加灵活。

<picture> 元素零或多个 <source> 元素和一个 元素，每个 <source> 元素匹配不同的设备并引用不同的图像源，如果没有匹配的，就选择 元素的 src 属性中的 url。

注意: 元素是放在最后一个 <picture> 元素之后，如果浏览器不支持该属性则显示 元素的图片。

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

| | | | | | |
|-----------|------|------|------|-----|------|
| 元素 | | | | | |
| <picture> | 38.0 | 13.0 | 38.0 | 9.1 | 25.0 |

HTML 4.01 与 HTML5 之间的差异

<picture> 属性是 HTML5 新定义的。

全局属性

<picture> 标签支持 [HTML 的全局属性](#)。

事件属性

<picture> 标签支持 [HTML 的事件属性](#)。

[HTML <html> 标签](#)

[HTML5 progress 标签](#)

点我分享笔记

[HTML5 <picture> 标签](#)

[HTML <q> 标签](#)

HTML5 <progress> 标签

实例

标记"下载进度":

```
<progress value="22" max="100"></progress>
```

[尝试一下 »](#)

浏览器支持



IE 10、Firefox、Opera、Chrome 和 Safari 6 支持 <progress> 标签。

注释：IE 9 或者更早版本的 IE 浏览器不支持 <progress> 标签。

标签定义及使用说明

<progress> 标签定义运行中的任务进度（进程）。

HTML 4.01 与 HTML5 之间的差异

<progress> 标签是 HTML5 中的新标签。

提示和注释

提示： 请将 <progress> 标签与 JavaScript 一起使用来显示任务的进度。

注释： <progress> 标签不适合用来表示度量衡（例如，磁盘空间使用情况或相关的查询结果）。表示度量衡，请使用 [<meter>](#) 标签代替。

属性

New：HTML5 中的新属性。

| 属性 | 值 | 描述 |
|----------------------------------|---------------|-----------|
| max New | <i>number</i> | 规定需要完成的值。 |
| value New | <i>number</i> | 规定进程的当前值。 |

全局属性

<progress> 标签支持 [HTML 的全局属性](#)。

事件属性

<progress> 标签支持 [HTML 的事件属性](#)。

[HTML5 <picture> 标签](#)

[HTML <q> 标签](#)

1 篇笔记 写笔记

1. Evan
lig***iyi@qq.com

53

下载进度:

```
<progress id='progress1' value="0" max="100">
</progress>
<button onclick="start_run(100)">下载</button>
<script>
function start_run(n)
{
    if(n==0){alert("下载完成")}
    var progress1=document.getElementById("progress1")
    n=n-1
    cur_task=100-n
    progress1.value=cur_task
    setTimeout("start_run('+n+')",100)
}
</script>
```

[尝试一下 »](#)

[Evan](#)

Evan
lig***iyi@qq.com

4 年前 (2019-11-27)

[HTML5 progress 标签](#)

[HTML <rp> 标签](#)

HTML <q> 标签

实例

标记一个短的引用：

```
<p>WWF's goal is to:
```

```
<q>Build a future where people live in harmony with nature.</q>
```

```
We hope they succeed.</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <q> 标签。

标签定义及使用说明

<q> 标签定义一个短的引用。

浏览器经常会在这种引用的周围插入引号。

提示和注释

提示：请使用 [<blockquote>](#) 来标记摘自另一个源的块引用。

HTML 4.01 与 HTML5 之间的差异

无。

属性

属性	值	描述
cite	<i>URL</i>	规定引用的源 URL。

全局属性

<q> 标签支持 [HTML 的全局属性](#)。

事件属性

<q> 标签支持 [HTML 的事件属性](#)。

[HTML5 progress 标签](#)

[HTML <rp> 标签](#)

点我分享笔记

[HTML <q> 标签](#)

[HTML <rt> 标签](#)

HTML <rp> 标签

实例

一个 ruby 注释：

```
<ruby> 漢 <rp>(</rp><rt>han</rt><rp>)</rp> 字 <rp>(</rp><rt>zi</rt><rp>)</rp> </ruby>
```

[尝试一下 »](#)

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari 支持 <rp> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <rp> 标签。

标签定义及使用说明

<rp> 标签在 ruby 注释中使用，以定义不支持 ruby 元素的浏览器所显示的内容。

ruby 注释是中文注音或字符。在东亚使用，显示的是东亚字符的发音。

将 `<rp>` 标签与 `<ruby>` 和 `<rt>` 标签一起使用：

`<ruby>` 元素由一个或多个需要解释/发音的字符和一个提供该信息的 `<rt>` 元素组成，还包括可选的 `<rp>` 元素，定义当浏览器不支持 "ruby" 元素时显示的内容。

HTML 4.01 与 HTML5 之间的差异

`<rp>` 标签是 HTML5 中的新标签。

全局属性

`<rp>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<rp>` 标签支持 [HTML 的事件属性](#)。

[HTML <q> 标签](#)

[HTML <rt> 标签](#)

点我分享笔记

[HTML <rp> 标签](#)

[HTML <ruby> 标签](#)

HTML `<rt>` 标签

实例

一个 ruby 注释：

```
<ruby> 汉 <rp>(</rp><rt>Han</rt><rp>)</rp> 字 <rp>(</rp><rt>zi</rt><rp>)</rp> </ruby>
```

尝试一下 »

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari 支持 <rt> 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 <rt> 标签。

标签定义及使用说明

<rt> 标签定义字符（中文注音或字符）的解释或发音。

将 <rt> 标签与 [<ruby>](#) 和 [<rp>](#) 标签一起使用：

<ruby> 元素由一个或多个需要解释/发音的字符和一个提供该信息的 <rt> 元素组成，还包括可选的 <rp> 元素，定义当浏览器不支持 "ruby" 元素时显示的内容。

HTML 4.01 与 HTML5 之间的差异

<rt> 标签是 HTML5 中的新标签。

全局属性

<rt> 标签支持 [HTML 的全局属性](#)。

事件属性

<rt> 标签支持 [HTML 的事件属性](#)。

[HTML <rp> 标签](#)

[HTML <ruby> 标签](#)

[HTML <rt> 标签](#)

[HTML <s> 标签](#)

HTML <ruby> 标签

实例

一个 ruby 注释：

```
<ruby> 汉 <rp>(</rp><rt>Han</rt><rp>)</rp> 字 <rp>(</rp><rt>zi</rt><rp>)</rp> </ruby>
```

[尝试一下 »](#)

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari 支持 `<ruby>` 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 `<ruby>` 标签。

标签定义及使用说明

`<ruby>` 标签定义 ruby 注释（中文注音或字符）。

在东亚使用，显示的是东亚字符的发音。

将 `<ruby>` 标签与 `<rt>` 和 `<rp>` 标签一起使用：

`<ruby>` 元素由一个或多个需要解释/发音的字符和一个提供该信息的 `<rt>` 元素组成，还包括可选的 `<rp>` 元素，定义当浏览器不支持 "ruby" 元素时显示的内容。

HTML 4.01 与 HTML5 之间的差异

`<ruby>` 标签是 HTML5 中的新标签。

全局属性

`<ruby>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<ruby>` 标签支持 [HTML 的事件属性](#)。

[HTML `<rt>` 标签](#)

[HTML <s> 标签](#)

点我分享笔记

[HTML <ruby> 标签](#)

[HTML <samp> 标签](#)

HTML <s> 标签

实例

标记不再正确的文本：

```
<p><s>My car is blue.</s></p>
<p>My new car is silver.</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <s> 标签。

标签定义及使用说明

<s> 标签对那些不正确、不准确或者没有用的文本进行标识。

<s> 标签不应该用来定义替换的或者删除的文本。如果要定义替换的或者删除的文本，请使用 [](#) 标签。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，<s> 元素 [已废弃](#)，用来给文本加删除线。

HTML5 重定义了 <s> 元素，现在是被用来定义那些不正确的文本。

全局属性

<s> 标签支持 [HTML 的全局属性](#)。

事件属性

<s> 标签支持 [HTML 的事件属性](#)。

[HTML <ruby> 标签](#)

[HTML <samp> 标签](#)

点我分享笔记

[HTML <s> 标签](#)

[HTML <script> 标签](#)

HTML [<samp>](#) 标签

实例

对文档中的文本进行格式化：

`<samp>计算机样本</samp>`

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<samp>` 标签。

标签定义及使用说明

`<samp>` 标签是一个短语标签，用来定义计算机程序的样本文本。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 **CSS**，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
<code></code>	呈现为被强调的文本。
<code></code>	定义重要的文本。
<code><dfn></code>	定义一个定义项目。

<code>	定义计算机代码文本。
<samp>	定义样本文本。
<kbd>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<var>	定义变量。您可以将此标签与 <pre> 及 <code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

<samp> 标签支持 [HTML 的全局属性](#)。

事件属性

<samp> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML <s> 标签](#)

[HTML <script> 标签](#)

[HTML <samp> 标签](#)

[HTML <section> 标签](#)

HTML <script> 标签

实例

通过 JavaScript 输出 "Hello world":

```
<script> document.write("Hello World!") </script>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <script> 标签。

标签定义及使用说明

<script> 标签用于定义客户端脚本，比如 JavaScript。

<script> 元素既可包含脚本语句，也可以通过 "src" 属性指向外部脚本文件。

JavaScript 通常用于图像操作、表单验证以及动态内容更改。

提示和注释

注释：如果使用 "src" 属性，则 <script> 元素必须是空的。

提示：请参阅 [<noscript>](#) 元素，对于那些在浏览器中禁用脚本或者其浏览器不支持客户端脚本的用户来说，该元素非常有用。

注释：有多种执行外部脚本的方法：

- 如果 async="async"：脚本相对于页面的其余部分异步地执行（当页面继续进行解析时，脚本将被执行）
- 如果不使用 async 且 defer="defer"：脚本将在页面完成解析时执行
- 如果既不使用 async 也不使用 defer：在浏览器继续解析页面之前，立即读取并执行脚本

HTML 4.01 与 HTML5 之间的差异

在 HTML 4 中，"type" 属性是必需的，但在 HTML5 中是可选的。

"async" 属性是 HTML5 中的新属性。

HTML5 中不再支持 HTML 4.01 中的某些属性："xml:space"。

HTML 与 XHTML 之间的差异

在 XHTML 中，脚本中的内容类型声明为 #PCDATA（代替 CDATA），就是说会对实体进行解析。

这意味着，在 XHTML 中，应该编码所有特殊的字符，或者把所有内容嵌套在 CDATA 部分中：

```
<script type="text/javascript">
//<![CDATA[
var i=10;
if (i<5)
{
```

```
// 代码内容
}  
//]]>  
</script>
```

属性

New：HTML5 中的新属性。

| 属性 | 值 | 描述 |
|-------------------------|------------------|---------------------------------|
| <u>async</u> New | async | 规定异步执行脚本（仅适用于外部脚本）。 |
| <u>charset</u> | <i>charset</i> | 规定在脚本中使用的字符编码（仅适用于外部脚本）。 |
| <u>defer</u> | defer | 规定当页面已完成解析后，执行脚本（仅适用于外部脚本）。 |
| <u>src</u> | <i>URL</i> | 规定外部脚本的 URL。 |
| <u>type</u> | <i>MIME-type</i> | 规定脚本的 MIME 类型。 |
| xml:space | preserve | HTML5 不支持。 规定是否保留代码中的空白。 |

全局属性

<script> 标签支持 [HTML 的全局属性](#)。

相关文章

HTML 教程: [HTML 脚本](#)

[HTML <samp> 标签](#)

[HTML <section> 标签](#)

点我分享笔记

[HTML <script> 标签](#)

[HTML <select> 标签](#)

HTML <section> 标签

实例

文档的某个区域，解释了什么是 WWF：

```
<section>
<h1>WWF</h1>
<p>The World Wide Fund for Nature (WWF) is....</p>
</section>
```

[尝试一下 »](#)

浏览器支持



IE 9+、Firefox、Opera、Chrome 和 Safari `<section>` 标签。

注释：IE 8 或更早版本的 IE 浏览器不支持 `<section>` 标签。

标签定义及使用说明

`<section>` 标签定义了文档的某个区域。比如章节、头部、底部或者文档的其他区域。

HTML 4.01 与 HTML5 之间的差异

`<section>` 标签是 HTML5 中的新标签。

全局属性

`<section>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<section>` 标签支持 [HTML 的事件属性](#)。

[HTML `<script>` 标签](#)

[HTML `<select>` 标签](#)

[HTML <section> 标签](#)

[HTML <small> 标签](#)

HTML <select> 标签

实例

创建带有 4 个选项的选择列表：

```
<select> <option value="volvo">Volvo</option> <option value="saab">Saab</option> <option value="mercedes">Mercedes</option> <option value="audi">Audi</option> </select>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <select> 标签。

标签定义及使用说明

<select> 元素用来创建下拉列表。

<select> 元素中的 [<option>](#) 标签定义了列表中的可用选项。

提示和注释

提示：<select> 元素是一种表单控件，可用于在表单中接受用户输入。

HTML 4.01 与 HTML5 之间的差异

HTML5 增加了一些新的属性。

属性

New: HTML5 中的新属性。

属性	值	描述
<u>autofocus</u> New	autofocus	规定在页面加载时下拉列表自动获得焦点。
<u>disabled</u>	disabled	当该属性为 true 时，会禁用下拉列表。
<u>form</u> New	<i>form_id</i>	定义 select 字段所属的一个或多个表单。
<u>multiple</u>	multiple	当该属性为 true 时，可选择多个选项。
<u>name</u>	<i>text</i>	定义下拉列表的名称。
<u>required</u> New	required	规定用户在提交表单前必须选择一个下拉列表中的选项。

size	<i>number</i>	规定下拉列表中可见选项的数目。
----------------------	---------------	-----------------

全局属性

<select> 标签支持 [HTML 的全局属性](#)。

事件属性

<select> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML DOM 参考手册: [Select 对象](#)

[HTML <section> 标签](#)

[HTML <small> 标签](#)

点我分享笔记

[HTML <select> 标签](#)

[HTML <source> 标签](#)

HTML <small> 标签

实例

定义小型文本：

```
<p> runoob.com - the world's largest web development site.</p>
```

```
<p><small> Copyright 1999-2050 by Refsnes Data.</small></p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<small>` 标签。

标签定义及使用说明

`<small>` 标签定义小型文本（和旁注）。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<small>` 标签支持 [HTML 的全局属性](#)。

事件属性

<small> 标签支持 **HTML 的事件属性**。

[HTML <select> 标签](#)

[HTML <source> 标签](#)

点我分享笔记

[HTML <small> 标签](#)

[HTML 标签](#)

HTML <source> 标签

实例

带有两个源文件的音频播放器。浏览器需要选择它所支持的源文件（如果都支持则任选一个）：

```
<audio controls> <source src="horse.ogg" type="audio/ogg"> <source src="horse.mp3" type="audio/mpeg"> 您的浏览器不支持 audio 元素。 </audio>
```

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该标签的第一个浏览器版本号。

标签					
<source>	4.0	9.0	3.5	4.0	10.5

标签定义及使用说明

<source> 标签为媒体元素（比如 <video> 和 <audio>）定义媒体资源。
<source> 标签允许您规定两个视频/音频文件供浏览器根据它对媒体类型或者编解码器的支持进行选择。

HTML 4.01 与 HTML5 之间的差异

<source> 标签是 HTML5 中的新标签。

属性

New: HTML5 中的新属性。

属性	值	描述
<u>media</u> New	<i>media_query</i>	规定媒体资源的类型，供浏览器决定是否下载。
<u>src</u> New	<i>URL</i>	规定媒体文件的 URL。
<u>type</u>	<i>MIME_type</i>	规定媒体资源的 MIME 类型。

sizes		不同页面布局设置不同图片大小。
srcset	URL	<source> 应用于 <picture> 标签时需要使用到。指定在不同情况下使用的图像 URL。

更多实例

实例

在 <video> 标签中使用 <source> 来设置视频：

```
<video width="320" height="240" controls> <source src="movie.mp4" type="video/mp4"> <source src="movie.ogg" type="video/ogg"> Your browser does not support the video tag. </video>
```

[尝试一下 »](#)

实例

在 <picture> 标签中使用 <source> 来设置不同屏幕显示的图片：

```
<picture> <source media="(min-width:650px)" srcset="https://static.jyshare.com/images/runoob-logo.png"> <source media="(min-width:465px)" srcset="https://static.jyshare.com/images/code-icon-script.png">  </picture>
```

[尝试一下 »](#)

全局属性

<source> 标签支持 [HTML 的全局属性](#)。

事件属性

<source> 标签支持 **HTML 的事件属性**。

[HTML <small> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <source> 标签](#)

[HTML <strike> 标签](#)

HTML 标签

实例

使用 元素对文本中的一部分进行着色：

```
<p>我的母亲有 <span style="color:blue">蓝色</span> 的眼睛。</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `` 标签。

标签定义及使用说明

`` 用于对文档中的行内元素进行组合。

`` 标签没有固定的格式表现。当对它应用样式时，它才会产生视觉上的变化。如果不对 `` 应用样式，那么 `` 元素中的文本与其他文本不会有任何视觉上的差异。

`` 标签提供了一种将文本的一部分或者文档的一部分独立出来的方式。

提示和注释

提示： 被 `` 元素包含的文本，您可以使用 CSS 对它定义样式，或者使用 JavaScript 对它进行操作。

HTML 4.01 与 HTML5 之间的差异

NONE.

全局属性

`` 标签支持 [HTML 的全局属性](#)。

事件属性

`` 标签支持 [HTML 的事件属性](#)。

[HTML `<source>` 标签](#)

[HTML <strike> 标签](#)

点我分享笔记

[HTML 标签](#)

[HTML 标签](#)

HTML <strike> 标签 - HTML5 不支持

实例

给文本添加删除线：

```
<p>Version 2.0 is <strike>not yet available!</strike> now available!</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <strike> 标签。

标签定义及使用说明

HTML5 不支持 <strike> 标签。请用 [](#) 标签代替。

在 HTML 4.01 中，<strike> 元素 [已废弃](#)。
<strike> 定义加删除线文本。

标准属性

在 HTML 4.01 中，<strike> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<s> 和 <strike> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本

ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <strike> 标签](#)

[HTML <style> 标签](#)

HTML `` 标签

实例

对文档中的文本进行格式化：

```
<strong>加粗文本</strong>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `` 标签。

标签定义及使用说明

`` 标签是一个短语标签，用来定义计算机程序的样本重要的文本。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 **CSS**，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
<code></code>	呈现为被强调的文本。
<code></code>	定义重要的文本。

<dfn>	定义一个定义项目。
<code>	定义计算机代码文本。
<samp>	定义样本文本。
<kbd>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<var>	定义变量。您可以将此标签与 <pre> 及 <code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中， 标签定义加粗的被强调的文本，在 HTML 5 中， 标签定义重要的文本。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程：[HTML 文本格式化](#)

[HTML <strike> 标签](#)

[HTML <style> 标签](#)

点我分享笔记

[HTML 标签](#)

[HTML <sub> 标签](#)

HTML <style> 标签

实例

在 HTML 文档中使用 <style> 元素：

```
<!DOCTYPE html> <html> <head> <meta charset="utf-8"> <title>菜鸟教程(runoob.com)</title> <style
type="text/css"> h1 {color:red;} p {color:blue;} </style> </head> <body> <h1>这是一个标题</h1> <p>这是一个段落。<
/p> </body> </html>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <style> 标签。

标签定义及使用说明

<style> 标签定义 HTML 文档的样式信息。

在 <style> 元素中，您可以规定在浏览器中如何呈现 HTML 文档。

每个 HTML 文档能包含多个 <style> 标签。

提示和注释

提示：如需链接外部样式表，请使用 <link> 标签。

提示：如需学习更多有关样式表的知识，请阅读我们的 [CSS 教程](#)。

注释：如果没有使用 "scoped" 属性，则每一个 <style> 标签必须位于 head 头部区域。

HTML 4.01 与 HTML5 之间的差异

"scoped" 属性是 HTML 5 中的新属性，它允许我们为文档的指定部分定义样式，而不是整个文档。

如果使用 "scoped" 属性，那么所规定的样式只能应用到 style 元素的父元素及其子元素。

属性

New: HTML5 中的新属性。

| 属性 | 值 | 描述 |
|-----------------------|--------------------|----------------|
| media | <i>media_query</i> | 为样式表规定不同的媒体类型。 |

| | | |
|-----------------------------------|----------|-------------------------------------|
| scoped New | scoped | 如果使用该属性，则样式仅仅应用到 style 元素的父元素及其子元素。 |
| type | text/css | 规定样式表的 MIME 类型。 |

全局属性

<style> 标签支持 [HTML 的全局属性](#)。

事件属性

<style> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML CSS](#)

HTML DOM 参考手册: [Style 对象](#)

[HTML 标签](#)

[HTML <sub> 标签](#)

点我分享笔记

[HTML <style> 标签](#)

[HTML <summary> 标签](#)

HTML <sub> 标签

实例

下标文本：

```
<p>这个文本包含 <sub>下标</sub>文本。</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <sub> 标签。

标签定义及使用说明

<sub> 标签定义下标文本。下标文本将会显示在当前文本流中字符高度的一半为基准线的下方，但是与当前文本流中文字的字体和字号都是一样的。下标文本能用来表示化学公式，比如 H₂O。

提示：请使用 <sup> 标签定义上标文本。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

<sub> 标签支持 [HTML 的全局属性](#)。

事件属性

<sub> 标签支持 [HTML 的事件属性](#)。

[HTML <style> 标签](#)

[HTML <summary> 标签](#)

点我分享笔记

[HTML <sub> 标签](#)

[HTML <sup> 标签](#)

HTML <summary> 标签

实例

使用 <summary> 元素：

```
<details> <summary>Epcot Center</summary> <p>Epcot is a theme park at Walt Disney World Resort featuring exciting attractions, international pavilions, award-winning fireworks and seasonal special events.</p></details>
```

尝试一下 »

浏览器支持

表格中的数字表示支持该元素的第一个浏览器版本号。

| 元素 | | | | | |
|-----------|------|------|------|-----|------|
| <summary> | 12.0 | 79.0 | 49.0 | 6.0 | 15.0 |

标签定义及使用说明

<summary> 标签为 <details> 元素定义一个可见的标题。 当用户点击标题时会显示出详细信息。

HTML 4.01 与 HTML5 之间的差异

<summary> 标签是 HTML5 中的新标签。

提示和注释

注释： <summary> 元素应该是 <details> 元素的第一个子元素。

全局属性

<summary> 标签支持 [HTML 的全局属性](#)。

事件属性

<summary> 标签支持 [HTML 的事件属性](#)。

[HTML <sub> 标签](#)

[HTML <sup> 标签](#)

点我分享笔记

[HTML <summary> 标签](#)

[HTML <table> 标签](#)

HTML <sup> 标签

实例

上标文本：

```
<p>这个文本包含 <sup>上标</sup> 文本。</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<sup>` 标签。

标签定义及使用说明

`<sup>` 标签定义上标文本。上标文本将会显示在当前文本流中字符高度的一半为基准线的上方，但是与当前文本流中文字的字体和字号都是一样的。上标文本能用来添加脚注，比如 `WWW[1]`。

提示： 请使用 `<sub>` 标签定义下标文本。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<sup>` 标签支持 [HTML 的全局属性](#)。

事件属性

`<sup>` 标签支持 [HTML 的事件属性](#)。

[HTML <summary> 标签](#)

[HTML <table> 标签](#)

点我分享笔记

[HTML <sup> 标签](#)

[HTML <tbody> 标签](#)

HTML <table> 标签

实例

一个简单的 HTML 表格，包含两列两行：

```
<table border="1"> <tr> <th>Month</th> <th>Savings</th> </tr> <tr> <td>January</td> <td>$100</td> </tr> <tr> <td>February</td> <td>$80</td> </tr> </table>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 <table> 标签。

标签定义及使用说明

<table> 标签定义 HTML 表格

一个 HTML 表格包括 <table> 元素，一个或多个 [<tr>](#)、[<th>](#) 以及 [<td>](#) 元素。

<tr> 元素定义表格行，<th> 元素定义表头，<td> 元素定义表格单元。

更复杂的 HTML 表格也可能包括 <caption>、<col>、<colgroup>、<thead>、<tfoot> 以及 <tbody> 元素。

HTML 4.01 与 HTML5 之间的差异

在 HTML5 中，仅支持 "border" 属性，并且只允许使用值 "1" 或 ""。

属性

| 属性 | 值 | 描述 |
|-----------------------------|---|--|
| align | left
center
right | HTML5 不支持。HTML 4.01 已废弃。规定表格相对周围元素的对齐方式。 |
| bgcolor | <i>rgb(x,x,x)</i>
<i>#xxxxxx</i>
<i>colorname</i> | HTML5 不支持。HTML 4.01 已废弃。规定表格的背景颜色。 |
| border | 1
"" | HTML5 不支持。规定表格单元是否拥有边框。 |
| cellpadding | <i>pixels</i> | HTML5 不支持。规定单元边沿与其内容之间的空白。 |
| cellspacing | <i>pixels</i> | HTML5 不支持。规定单元格之间的空白。 |
| frame | void
above
below | HTML5 不支持。规定外侧边框的哪个部分是可见的。 |

| | | |
|-------------------------|---|----------------------------|
| | hsides
lhs
rhs
vsides
box
border | |
| rules | none
groups
rows
cols
all | HTML5 不支持。规定内侧边框的哪个部分是可见的。 |
| summary | <i>text</i> | HTML5 不支持。规定表格的摘要。 |
| width | <i>pixels</i>
% | HTML5 不支持。规定表格的宽度。 |

全局属性

<table> 标签支持 [HTML 的全局属性](#)。

事件属性

<table> 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[表格中的标题](#)

本例演示如何显示表格标题。

[空单元格](#)

本例演示如何使用 " " 处理没有内容的单元格。

[带有标题的表格](#)

本例演示一个带标题(caption)的表格。

[表格内的标签](#)

本例演示如何在其他的元素内显示元素。

[跨行或跨列的表格单元格](#)

本例演示如何定义跨行或跨列的表格单元格。

相关文章

HTML 教程: [HTML 表格](#)

HTML DOM 参考手册: [Table 对象](#)

HTML 表格生成器: [在线 HTML 表格生成器](#)

[HTML <sup> 标签](#)

[HTML <tbody> 标签](#)

[HTML <table> 标签](#)

[HTML <td> 标签](#)

HTML <tbody> 标签

实例

帶有 <thead>、<tfoot> 和 <tbody> 元素的 HTML 表格：

```
<table border="1">
<thead>
<tr>
<th>Month</th>
<th>Savings</th>
</tr>
</thead>
<tfoot>
<tr>
<td>Sum</td>
<td>$180</td>
</tr>
</tfoot>
<tbody>
<tr>
<td>January</td>
<td>$100</td>
</tr>
<tr>
```

```
<td>February</td>
<td>$80</td>
</tr>
</tbody>
</table>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<tbody>` 标签。

标签定义及使用说明

`<tbody>` 标签用于组合 HTML 表格的主体内容。

`<tbody>` 元素应该与 [<thead>](#) and [<tfoot>](#) 元素结合起来使用，用来规定表格的各个部分（主体、表头、页脚）。

通过使用这些元素，使浏览器有能力支持独立于表格表头和表格页脚的表格主体滚动。当包含多个页面的长的表格被打印时，表格的表头和页脚可被打印在包含表格数据的每张页面上。

`<tbody>` 标签必须被用在以下情境中：作为 `<table>` 元素的子元素，出现在 `<caption>`、`<colgroup>` 和 `<thead>` 元素之后。

提示和注释

注释： `<tbody>` 元素内部必须包含一个或者多个 `<tr>` 标签。

提示： `<thead>`、`<tbody>` 和 `<tfoot>` 元素默认不会影响表格的布局。不过，您可以使用 CSS 来为这些元素定义样式，从而改变表格的外观。

HTML 4.01 与 HTML5 之间的差异

在 HTML 5 中，不再支持 HTML 4.01 中 <tbody> 标签的任何属性。

属性

| 属性 | 值 | 描述 |
|-------------------------|--|---|
| align | right
left
center
justify
char | HTML5 不支持。定义 <tbody> 元素中内容的对齐方式。 |
| char | <i>character</i> | HTML5 不支持。规定 <tbody> 元素中内容根据哪个字符来对进行文本对齐。 |
| charoff | <i>number</i> | HTML5 不支持。规定 <tbody> 元素中内容的第一个对齐字符的偏移量。 |
| valign | top
middle
bottom
baseline | HTML5 不支持。规定 <tbody> 元素中内容的垂直对齐方式。 |

全局属性

<tbody> 标签支持 [HTML 的全局属性](#)。

事件属性

<tbody> 标签支持 [HTML 的事件属性](#)。

[HTML <table> 标签](#)

[HTML <td> 标签](#)

点我分享笔记

[HTML <tbody> 标签](#)

[HTML <textarea> 标签](#)

HTML <td> 标签

实例

一个简单的 HTML 表格，带有两个单元格：

```
<table border="1">
<tr>
<td>Cell A</td>
<td>Cell B</td>
</tr>
</table>
```

尝试一下 »

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 `<td>` 标签。

标签定义及使用说明

`<td>` 标签定义 HTML 表格中的标准单元格。

HTML 表格有两种单元格类型：

- 表头单元格 - 包含头部信息（由 `<th>` 元素创建）
- 标准单元格 - 包含数据（由 `<td>` 元素创建）

`<th>` 元素中的文本通常呈现为粗体并且居中。

`<td>` 元素中的文本通常是普通的左对齐文本。

提示和注释

提示：如果需要将内容横跨多个行或列，请使用 `colspan` 和 `rowspan` 属性！

HTML 4.01 与 HTML5 之间的差异

HTML 5 中不再支持 HTML 4.01 中的某些属性。

属性

| 属性 | 值 | 描述 |
|--------------------------------|---|---------------------------------------|
| <u>abbr</u> | <i>text</i> | HTML5 不支持。规定单元格中内容的缩写版本。 |
| <u>align</u> | left
right
center
justify
char | HTML5 不支持。规定单元格内容的水平对齐方式。 |
| <u>axis</u> | <i>category_name</i> | HTML5 不支持。对单元格进行分类。 |
| <u>bgcolor</u> | <i>rgb(x,x,x)</i>
<i>#xxxxxx</i>
<i>colorname</i> | HTML5 不支持。HTML 4.01 已废弃。规定单元格的背景颜色。 |
| <u>char</u> | <i>character</i> | HTML5 不支持。规定根据哪个字符来进行内容的对齐。 |
| <u>charoff</u> | <i>number</i> | HTML5 不支持。规定对齐字符的偏移量。 |
| <u>colspan</u> | <i>number</i> | 规定单元格可横跨的列数。 |
| <u>headers</u> | <i>header_id</i> | 规定与单元格相关联的一个或多个表头单元格。 |
| <u>height</u> | <i>pixels</i>
<i>%</i> | HTML5 不支持。HTML 4.01 已废弃。
设置单元格的高度。 |
| <u>nowrap</u> | nowrap | HTML5 不支持。HTML 4.01 已废弃。 |

| | | |
|-------------------------|-------------------------------------|-----------------------------------|
| | | 规定单元格中的内容是否折行。 |
| rowspan | <i>number</i> | 设置单元格可纵跨的行数。 |
| scope | col
colgroup
row
rowgroup | HTML5 不支持。定义将表头单元格与数据单元格相关联的方法。 |
| valign | top
middle
bottom
baseline | HTML5 不支持。规定单元格内容的垂直排列方式。 |
| width | <i>pixels</i>
% | HTML5 不支持。HTML 4.01 已废弃。规定单元格的宽度。 |

全局属性

<td> 标签支持 [HTML 的全局属性](#)。

事件属性

<td> 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[表格中的标题](#)

本例演示如何显示表格标题。

[空单元格](#)

本例演示如何使用 " " 处理没有内容的单元格。

[带有标题的表格](#)

本例演示一个带标题(caption)的表格。

[表格内的标签](#)

本例演示如何在其他的元素内显示元素。

[跨行或跨列的表格单元格](#)

本例演示如何定义跨行或跨列的表格单元格。

相关文章

HTML 教程: [HTML 表格](#)

HTML DOM 参考手册: [Td 对象](#)

[HTML <tbody> 标签](#)

[HTML <textarea> 标签](#)

点我分享笔记

[HTML <td> 标签](#)

[HTML <template> 标签](#)

HTML <textarea> 标签

实例

一个 HTML 文本区域：

```
<textarea rows="10" cols="30">  
我是一个文本框。  
</textarea>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <textarea> 标签。

标签定义及使用说明

<textarea> 标签定义一个多行的文本输入控件。

文本区域中可容纳无限数量的文本，其中的文本的默认字体是等宽字体（通常是 Courier）。

可以通过 cols 和 rows 属性来规定 textarea 的尺寸大小，不过更好的办法是使用 CSS 的 height 和 width 属性。

HTML 4.01 与 HTML5 之间的差异

HTML5 增加了一些新的属性。

属性

New: HTML5 中的新属性。

| 属性 | 值 | 描述 |
|-------------------------------|----------------|-------------------------|
| <u>autofocus</u> New | autofocus | 规定当页面加载时，文本区域自动获得焦点。 |
| <u>cols</u> | <i>number</i> | 规定文本区域内可见的宽度。 |
| <u>disabled</u> | disabled | 规定禁用文本区域。 |
| <u>form</u> New | <i>form_id</i> | 定义文本区域所属的一个或多个表单。 |
| <u>maxlength</u> New | <i>number</i> | 规定文本区域允许的最大字符数。 |
| <u>name</u> | <i>text</i> | 规定文本区域的名称。 |
| <u>placeholder</u> New | <i>text</i> | 规定一个简短的提示，描述文本区域期望的输入值。 |

| | | |
|--|---------------|--------------------------|
| readonly | readonly | 规定文本区域为只读。 |
| required New | required | 规定文本区域是必需的/必填的。 |
| rows | <i>number</i> | 规定文本区域内可见的行数。 |
| wrap New | hard
soft | 规定当提交表单时，文本区域中的文本应该怎样换行。 |

全局属性

<textarea> 标签支持 [HTML 的全局属性](#)。

事件属性

<textarea> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML DOM 参考手册: [Textarea 对象](#)

[HTML <td> 标签](#)

[HTML <template> 标签](#)

点我分享笔记

[HTML <textarea> 标签](#)

[HTML <tfoot> 标签](#)

HTML <template> 标签

实例

使用 <template> 标签在页面加载时该标签中的内容不会显示，加载后可以使用 JavaScript 来显示它：

```
<button onclick="showContent()">显示隐藏内容</button> <template> <h2>logo</h2>  </template> <script> function showContent() { var  
temp = document.getElementsByTagName("template")[0]; var clon = temp.content.cloneNode(true);  
document.body.appendChild(clon); } </script>
```

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该元素的第一个浏览器的版本号。

| 元素 | | | | | |
|------------|------|------|------|-----|------|
| <template> | 26.0 | 13.0 | 22.0 | 8.0 | 15.0 |

标签定义及使用说明

<template> 标签定义在页面加载时隐藏的一些内容，该标签中的内容可以稍后使用 JavaScript 呈现。

如果您有一些需要重复使用的 HTML 代码，则可以使用 <template> 设置为公用的模板。

更多实例

实例

实例中的每个数组元素都使用一个新的 div 元素来填充网页。每个 div 元素的 HTML 代码都在 template 元素内：

```
<template> <div class="myClass">我喜欢: </div> </template> <script> var myArr = ["Google", "Runoob", "Taobao",  
"Wiki", "Zhihu", "Baidu"]; function showContent() { var temp, item, a, i; temp =  
document.getElementsByTagName("template")[0]; item = temp.content.querySelector("div"); for (i = 0; i <  
myArr.length; i++) { a = document.importNode(item, true); a.textContent += myArr[i];  
document.body.appendChild(a); } } </script>
```

[尝试一下 »](#)

实例

查看浏览器是否支持 template 标签：

```
if (document.createElement("template").content) { document.write("您的浏览器支持 template 标签! "); } else {  
document.write("您的浏览器不支持 template 标签! "); }
```

[尝试一下 »](#)

全局属性

<time> 标签支持 [HTML 的全局属性](#)。

[HTML <textarea> 标签](#)

[HTML <tfoot> 标签](#)

点我分享笔记

[HTML <template> 标签](#)

[HTML <th> 标签](#)

HTML <tfoot> 标签

实例

带有 <thead>、<tfoot> 和 <tbody> 元素的 HTML 表格：

```
<table border="1"> <thead> <tr> <th>Month</th> <th>Savings</th> </tr> </thead> <tfoot> <tr> <td>Sum</td>
<td>$180</td> </tr> </tfoot> <tbody> <tr> <td>January</td> <td>$100</td> </tr> <tr> <td>February</td>
<td>$80</td> </tr> </tbody> </table>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<tfoot>` 标签。

标签定义及使用说明

`<tfoot>` 标签用于组合 HTML 表格的页脚内容。

`<tfoot>` 元素应该与 `<thead>` 和 `<tbody>` 元素结合起来使用，用来规定表格的各个部分（页脚、表头、主体）。

通过使用这些元素，使浏览器有能力支持独立于表格表头和表格页脚的表格主体滚动。当包含多个页面的长的表格被打印时，表格的表头和页脚可被打印在包含表格数据的每张页面上。

`<tfoot>` 标签必须被用在以下情境中：作为 `<table>` 元素的子元素，出现在 `<caption>`、`<colgroup>` 和 `<thead>` 元素之后，`<tbody>` 和 `<tr>` 元素之前。

提示和注释

注释：`<tfoot>` 元素内部必须包含一个或者多个 `<tr>` 标签。

提示：`<thead>`、`<tbody>` 和 `<tfoot>` 元素默认不会影响表格的布局。不过，您可以使用 CSS 来为这些元素定义样式，从而改变表格的外观。

HTML 4.01 与 HTML5 之间的差异

在 HTML 5 中，不再支持 HTML 4.01 中 `<tfoot>` 标签的任何属性。

属性

属性	值	描述
<code>align</code>	right	HTML5 不支持。定义 <code><tfoot></code> 元素中内容的对齐方式。

	left center justify char	
char	<i>character</i>	HTML5 不支持。规定 <tfoot> 元素中内容根据哪个字符来对进行文本对齐。
charoff	<i>number</i>	HTML5 不支持。规定 <tfoot> 元素中内容的第一个对齐字符的偏移量。
valign	top middle bottom baseline	HTML5 不支持。规定 <tfoot> 元素中内容的垂直对齐方式。

全局属性

<tfoot> 标签支持 [HTML 的全局属性](#)。

事件属性

<tfoot> 标签支持 [HTML 的事件属性](#)。

[HTML <template> 标签](#)

[HTML <th> 标签](#)

点我分享笔记

[HTML <tfoot> 标签](#)

[HTML <thead> 标签](#)

HTML <th> 标签

实例

一个简单的 HTML 表格，带有两个表头单元格和两个数据单元格：

```
<table border="1"> <tr> <th>Month</th> <th>Savings</th> </tr> <tr> <td>January</td> <td>$100</td> </tr> </table>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 <th> 标签。

标签定义及使用说明

<th> 标签定义 HTML 表格中的表头单元格。

HTML 表格有两种单元格类型：

- 表头单元格 - 包含头部信息（由 `<th>` 元素创建）
- 标准单元格 - 包含数据（由 `<td>` 元素创建）

`<th>` 元素中的文本通常呈现为粗体并且居中。

`<td>` 元素中的文本通常是普通的左对齐文本。

提示和注释

提示： 如果需要将内容横跨多个行或列，请使用 `colspan` 和 `rowspan` 属性！

HTML 4.01 与 HTML5 之间的差异

HTML 5 中不再支持 HTML 4.01 中的某些属性。

属性

属性	值	描述
abbr	<i>text</i>	HTML5 不支持。 规定表头单元格中内容的缩写版本。
align	left right center justify char	HTML5 不支持。 规定表头单元格内容的水平对齐方式。
axis	<i>category_name</i>	HTML5 不支持。 对表头单元格进行分类。

<u>bgcolor</u>	<i>rgb(x,x,x)</i> <i>#xxxxxx</i> <i>colorname</i>	HTML5 不支持。HTML 4.01 已废弃。规定表头单元格的背景颜色。
<u>char</u>	<i>character</i>	HTML5 不支持。规定根据哪个字符来进行内容的对齐。
<u>charoff</u>	<i>number</i>	HTML5 不支持。规定对齐字符的偏移量。
<u>colspan</u>	<i>number</i>	规定表头单元格可横跨的列数。
<u>headers</u>	<i>header_id</i>	规定与表头单元格相关联的一个或多个表头单元格。
<u>height</u>	<i>pixels</i> <i>%</i>	HTML5 不支持。HTML 4.01 已废弃。规定表头单元格的高度。
<u>nowrap</u>	<i>nowrap</i>	HTML5 不支持。HTML 4.01 已废弃。规定表头单元格中的内容是否折行。
<u>rowspan</u>	<i>number</i>	规定表头单元格可纵跨的行数。
<u>scope</u>	<i>col</i> <i>colgroup</i> <i>row</i> <i>rowgroup</i>	规定表头单元格是否是行、列、行组或列组的头部。
<u>valign</u>	<i>top</i> <i>middle</i> <i>bottom</i> <i>baseline</i>	HTML5 不支持。规定表头单元格内容的垂直排列方式。
<u>width</u>	<i>pixels</i>	HTML5 不支持。HTML 4.01 已废弃。规定表头单元格的宽度。

	%	
--	---	--

全局属性

<th> 标签支持 [HTML 的全局属性](#)。

事件属性

<th> 标签支持所有 [HTML 的事件属性](#)。



尝试一下 - 实例

[表格中的标题](#)

本例演示如何显示表格标题。

[空单元格](#)

本例演示如何使用 " " 处理没有内容的单元格。

[带有标题的表格](#)

本例演示一个带标题(caption)的表格。

[表格内的标签](#)

本例演示如何在其他的元素内显示元素。

[跨行或跨列的表格单元格](#)

本例演示如何定义跨行或跨列的表格单元格。

相关文章

HTML 教程: [HTML 表格](#)

HTML DOM 参考手册: [Th 对象](#)

[HTML <tfoot> 标签](#)

[HTML <thead> 标签](#)

点我分享笔记

[HTML <th> 标签](#)

[HTML <time> 标签](#)

HTML <thead> 标签

实例

带有 <thead>、<tfoot> 和 <tbody> 元素的 HTML 表格:

```
<table border="1"> <thead> <tr> <th>Month</th> <th>Savings</th> </tr> </thead> <tfoot> <tr> <td>Sum</td>
<td>$180</td> </tr> </tfoot> <tbody> <tr> <td>January</td> <td>$100</td> </tr> <tr> <td>February</td>
<td>$80</td> </tr> </tbody> </table>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<thead>` 标签。

标签定义及使用说明

`<thead>` 标签用于组合 HTML 表格的表头内容。

`<thead>` 元素应该与 `<tbody>` 和 `<tfoot>` 元素结合起来使用，用来规定表格的各个部分（表头、主体、页脚）。

通过使用这些元素，使浏览器有能力支持独立于表格表头和表格页脚的表格主体滚动。当包含多个页面的长的表格被打印时，表格的表头和页脚可被打印在包含表格数据的每张页面上。

`<thead>` 标签必须被用在以下情境中：作为 `<table>` 元素的子元素，出现在 `<caption>`、`<colgroup>` 元素之后，`<tbody>`、`<tfoot>` 和 `<tr>` 元素之前。

提示和注释

注释：`<thead>` 元素内部必须包含一个或者多个 `<tr>` 标签。

提示：`<thead>`、`<tbody>` 和 `<tfoot>` 元素默认不会影响表格的布局。不过，您可以使用 CSS 来为这些元素定义样式，从而改变表格的外观。

HTML 4.01 与 HTML5 之间的差异

在 HTML 5 中，不再支持 HTML 4.01 中 `<thead>` 标签的任何属性。

属性

属性	值	描述
align	right left center justify char	HTML5 不支持。定义 <thead> 元素中内容的对齐方式。
char	<i>character</i>	HTML5 不支持。规定 <thead> 元素中内容根据哪个字符来对进行文本对齐。
charoff	<i>number</i>	HTML5 不支持。规定 <thead> 元素中内容的第一个对齐字符的偏移量。
valign	top middle bottom baseline	HTML5 不支持。规定 <thead> 元素中内容的垂直对齐方式。

全局属性

<thead> 标签支持 [HTML 的全局属性](#)。

事件属性

<thead> 标签支持 [HTML 的事件属性](#)。

[HTML <th> 标签](#)

[HTML <time> 标签](#)

点我分享笔记

[HTML <thead> 标签](#)

[HTML <title> 标签](#)

HTML <time> 标签

实例

如何定义时间和日期：

<p>我们在每天早上 <time>9:00</time> 开始营业。</p> <p>我在 <time datetime="2016-02-14">情人节</time> 有个约会。</p>

尝试一下 »

浏览器支持

表格中的数字表示支持该属性的第一个浏览器的版本号。

元素					
<time>	6.0	9.0	4.0	5.0	11.1

标签定义及使用说明

<time> 标签定义公历的时间（24 小时制）或日期，时间和时区偏移是可选的。
该元素能够以机器可读的方式对日期和时间进行编码，这样，举例说，用户代理能够把生日提醒或排定的事件添加到用户日程表中，搜索引擎也能够生成更智能的搜索结果。

HTML 4.01 与 HTML5 之间的差异

<time> 标签是 HTML5 中的新标签。

属性

New: HTML5 中的新属性。

属性	值	描述
datetime New	<i>datetime</i>	规定日期/时间。另一种方式，则是由元素的内容给定日期/时间。
pubdate	<i>datetime</i>	(该属性仍在被 WHATWG 和 W3C 组织设计和讨论中。) 指示 <time> 元素中的日期 / 时间是文档（或最近的前辈 <article> 元素）的发布日期

全局属性

<time> 标签支持 [HTML 的全局属性](#)。

事件属性

<time> 标签支持 [HTML 的事件属性](#)。

[HTML <thead> 标签](#)

[HTML <title> 标签](#)

点我分享笔记

[HTML <time> 标签](#)

[HTML <tr> 标签](#)

HTML <title> 标签

实例

为您的 HTML 文档定义标题：

```
<html> <head> <meta charset="utf-8"> <title>文档标题</title> </head> <body> 文档内容..... </body> </html>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <title> 标签。

标签定义及使用说明

`<title>` 标签定义文档的标题，在所有 HTML 文档中是必需的。

`<title>` 元素：

- 定义浏览器工具栏中的标题
 - 提供页面被添加到收藏夹时的标题
 - 显示在搜索引擎结果中的页面标题
-

提示和注释

注释：一个 HTML 文档中不能有一个以上的 `<title>` 元素。

提示：如果您遗漏了 `<title>` 标签，文档作为 HTML 是无效的。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

`<title>` 标签支持 [HTML 的全局属性](#)。

相关文章

HTML 教程：[HTML 头部](#)

[HTML <time> 标签](#)

[HTML <tr> 标签](#)

1 篇笔记 写笔记

1. Aa
946***834@qq.com

34

1.title 标签只能在 head 标签内出现；

- o 2.定义浏览器工具栏中的标题；
- o 3.浏览器中收藏夹内书签的名称是 title 的内容；
- o 4.title 的内容可以方便搜索引擎索引页面；
- o 5.从搜索引擎搜索到的内容的标题往往是网页 title 的内容；title 通常体现了网页的主题内容，所以记得一定要加上。

[Aa](#)

Aa
946***834@qq.com

7 年前 (2017-05-22)

[HTML <title> 标签](#)

[HTML <track> 标签](#)

HTML <tr> 标签

实例

一个简单的 HTML 表格，包含两列两行：

```
<table border="1">
<tr>
<th>Month</th>
<th>Savings</th>
</tr>
<tr>
<td>January</td>
<td>$100</td>
</tr>
</table>
```

[尝试一下 »](#)

(更多实例见页面底部)

浏览器支持



所有主流浏览器都支持 `<tr>` 标签。

标签定义及使用说明

`<tr>` 标签定义 HTML 表格中的行。

一个 <tr> 元素包含一个或多个 [<th>](#) 或 [<td>](#) 元素。

HTML 4.01 与 HTML5 之间的差异

在 HTML 5 中，不支持 <tr> 标签在 HTML 4.01 中的任何属性。

属性

属性	值	描述
align	right left center justify char	HTML5 不支持。定义表格行的内容对齐方式。
bgcolor	<i>rgb(x,x,x)</i> <i>#xxxxxx</i> <i>colorname</i>	HTML5 不支持。HTML 4.01 已废弃。规定表格行的背景颜色。
char	<i>character</i>	HTML5 不支持。规定根据哪个字符来进行文本对齐。
charoff	<i>number</i>	HTML5 不支持。规定第一个对齐字符的偏移量。
valign	top middle bottom baseline	HTML5 不支持。规定表格行中内容的垂直对齐方式。

全局属性

<tr> 标签支持 [HTML 的全局属性](#)。

事件属性

<tr> 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[表格中的标题](#)

本例演示如何显示表格标题。

[空单元格](#)

本例演示如何使用 " " 处理没有内容的单元格。

[带有标题的表格](#)

本例演示一个带标题(caption)的表格。

[表格内的标签](#)

本例演示如何在其他的元素内显示元素。

[跨行或跨列的表格单元格](#)

本例演示如何定义跨行或跨列的表格单元格。

相关文章

HTML 教程: [HTML 表格](#)

HTML DOM 参考手册: [Tr 对象](#)

[HTML <title> 标签](#)

[HTML <track> 标签](#)

点我分享笔记

[HTML <tr> 标签](#)

[HTML <tt> 标签](#)

HTML <track> 标签

实例

带有字幕轨道（friday.vtt）的视频：

```
<video width="320" height="240" controls> <video controls src="/video/php/friday.mp4"> <track default  
kind="captions" srclang="en" src="/video/php/friday.vtt" /> 抱歉，您的浏览器不支持嵌入视频！ </video> </video>
```

[尝试一下 »](#)

浏览器支持

表格中的数字表示支持该标签的第一个浏览器版本号。

Element					
<track>	23.0	10.0	31.0	6.0	12.1

标签定义及使用说明

<track> 标签为媒体元素（比如 <audio> and <video>）规定外部文本轨道，也就是字幕，字幕格式有 WebVTT 格式（.vtt 格式文件）。这个元素用于规定字幕文件或其他包含文本的文件，当媒体播放时，这些文件是可见的。

HTML 4.01 与 HTML5 之间的差异

<track> 标签是 HTML5 中的新标签。

可选的属性

New: HTML5 中的新属性。

属性	值	描述
<u>default</u> New	default	规定该轨道是默认的。如果用户没有选择任何轨道，则使用默认轨道。
<u>kind</u> New	captions chapters descriptions	规定文本轨道的文本类型。

	metadata subtitles	
label New	<i>text</i>	规定文本轨道的标签和标题。
src New	<i>URL</i>	必需的。规定轨道文件的 URL。
srclang New	<i>language_code</i>	规定轨道文本数据的语言。如果 kind 属性值是 "subtitles", 则该属性是必需的。

全局属性

<track> 标签支持 [HTML 的全局属性](#)。

事件属性

<track> 标签支持 [HTML 的事件属性](#)。

[HTML <tr> 标签](#)

[HTML <tt> 标签](#)

点我分享笔记

[HTML <track> 标签](#)

[HTML <u> 标签](#)

HTML <tt> 标签 - HTML5 不支持

实例

定义打字机文本：

```
<p><tt>Teletype text</tt></p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <tt> 标签。

标签定义及使用说明

HTML5 不支持 <tt> t 标签。请用 CSS 代替。

<tt> 标签定义打字机文本。

提示和注释

提示：使用 CSS 来规定文档中的[文本类型](#)。

HTML 4.01 与 HTML5 之间的差异

HTML5 不支持 <tt> 标签。HTML 4.01 支持 <tt> 标签。

标准属性

在 HTML 4.01 中，<tt> 标签支持如下标准属性：

属性	值	描述
class	<i>classname</i>	规定元素的类名
dir	rtl ltr	规定元素中内容的文本方向
id	<i>id</i>	规定元素的唯一 id
lang	<i>language_code</i>	规定元素中内容的语言代码
style	<i>style_definition</i>	规定元素的行内样式
title	<i>text</i>	规定元素的额外信息
xml:lang	<i>language_code</i>	规定 XHTML 文档元素中内容的语言代码

如需完整的描述，请访问[标准属性](#)。

事件属性

在 HTML 4.01 中，<tt> 标签支持如下事件属性：

属性	值	描述
onclick	<i>script</i>	当鼠标被单击时执行脚本
ondblclick	<i>script</i>	当鼠标被双击时执行脚本
onmousedown	<i>script</i>	当鼠标按钮被按下时执行脚本
onmousemove	<i>script</i>	当鼠标指针移动时执行脚本
onmouseout	<i>script</i>	当鼠标指针移出某元素时执行脚本
onmouseover	<i>script</i>	当鼠标指针悬停于某元素之上时执行脚本
onmouseup	<i>script</i>	当鼠标按钮被松开时执行脚本
onkeydown	<i>script</i>	当键盘被按下时执行脚本
onkeypress	<i>script</i>	当键盘被按下后又松开时执行脚本
onkeyup	<i>script</i>	当键盘被松开时执行脚本

如需完整的描述，请访问[事件属性](#)。

[HTML <track> 标签](#)

[HTML <u> 标签](#)

点我分享笔记

[HTML <tt> 标签](#)

[HTML 标签](#)

HTML <u> 标签

实例

使用 <u> 标签为文本添加下划线：

```
<p>This is a <u>parragraph</u>.</p>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <u> 标签。

标签定义及使用说明

<u> 标签定义与常规文本风格不同的文本，像拼写错误的单词或者汉语中的专有名词。

提示和注释

提示：请尽量避免使用 <u> 为文本加下划线，用户会把它混淆为一个超链接。

注释：HTML5 规范建议开发者尽量使用其他元素替代 <u> 元素。

HTML 4.01 与 HTML5 之间的差异

在 HTML 中，<u>元素 [已废弃](#)（<u> 元素被用来定义下划线）。

HTML5 中重新定义了 <u> 元素，它定义与常规文本风格不同的文本，像拼写错误的单词或者汉语中的专有名词。

全局属性

<u> 标签支持 [HTML 的全局属性](#)。

事件属性

<u> 标签支持 [HTML 的事件属性](#)。

[HTML <tt> 标签](#)

[HTML 标签](#)

点我分享笔记

[HTML <u> 标签](#)

[HTML <var> 标签](#)

HTML [](#) 标签

实例

无序 HTML 列表:

 Coffee Tea Milk

尝试一下 »

(更多实例见页面底部)

浏览器支持

标签					
	Yes	Yes	Yes	Yes	Yes

标签定义及使用说明

 标签定义无序列表。

将 标签与 标签一起使用，创建无序列表。

提示和注释

提示：使用 CSS 为列表定义样式。

提示：使用 标签创建有序列表。

HTML 4.01 与 HTML5 之间的差异

在 HTML 4.01 中，"compact" 和 "type" 属性 [已废弃](#)。HTML5 则不支持这两个属性。

属性

属性	值	描述
compact	compact	HTML5 不支持。HTML 4.01 已废弃。规定列表呈现的效果比正常情况更小巧。
type	disc square circle	HTML5 不支持。HTML 4.01 已废弃。规定列表的项目符号的类型。

全局属性

 标签支持 [HTML 的全局属性](#)。

事件属性

 标签支持 [HTML 的事件属性](#)。



尝试一下 - 实例

[列表小标记](#)

演示不同列表小标记。

[嵌套列表](#)

本例演示如何嵌套列表。

[嵌套列表 2](#)

本例演示更复杂的嵌套列表。

相关文章

HTML 教程：[HTML 列表](#)

[HTML 标签](#)

[HTML <video> 标签](#)

HTML **<var>** 标签

实例

对文档中的文本进行格式化：

```
<var>变量</var>
```

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 `<var>` 标签

标签定义及使用说明

`<var>` 标签是一个短语标签，用来定义变量。

提示：我们并不反对使用这个标签，但是如果您只是为了达到某种视觉效果而使用这个标签的话，我们建议您使用 **CSS**，这样可能会取得更丰富的效果。

所有短语标签：

标签	描述
<code></code>	呈现为被强调的文本。
<code></code>	定义重要的文本。
<code><dfn></code>	定义一个定义项目。
<code><code></code>	定义计算机代码文本。
<code><samp></code>	定义样本文本。
<code><kbd></code>	定义键盘文本。它表示文本是从键盘上键入的。它经常用在与计算机相关的文档或手册中。
<code><var></code>	定义变量。您可以将此标签与 <code><pre></code> 及 <code><code></code> 标签配合使用。

HTML 4.01 与 HTML5 之间的差异

无。

全局属性

<var> 标签支持 [HTML 的全局属性](#)。

事件属性

<var> 标签支持 [HTML 的事件属性](#)。

相关文章

HTML 教程: [HTML 文本格式化](#)

[HTML 标签](#)

[HTML <video> 标签](#)

点我分享笔记

[HTML <var> 标签](#)

[HTML <wbr> 标签](#)

HTML <video> 标签

实例

播放录像:

`<video width="320" height="240" controls> <source src="movie.mp4" type="video/mp4"> <source src="movie.ogg" type="video/ogg"> 您的浏览器不支持 video 标签。 </video>`

尝试一下 »

浏览器支持

表格中的数字表示支持该属性的第一个浏览器版本号。

Element					
<video>	4.0	9.0	3.5	4.0	10.5

标签定义及使用说明

<video> 标签定义视频，比如电影片段或其他视频流。
目前，<video> 元素支持三种视频格式：MP4、WebM、Ogg。

浏览器	MP4	WebM	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES

	从 Firefox 21 版本开始 Linux 系统从 Firefox 30 开始		
Safari	YES	NO	NO
Opera	YES 从 Opera 25 版本开始	YES	YES

- MP4 = MPEG 4 文件使用 H264 视频编解码器和 AAC 音频编解码器
- WebM = WebM 文件使用 VP8 视频编解码器和 Vorbis 音频编解码器
- Ogg = Ogg 文件使用 Theora 视频编解码器和 Vorbis 音频编解码器

视频格式的 MIME 类型

格式	MIME-type
MP4	video/mp4
WebM	video/webm
Ogg	video/ogg

HTML 4.01 与 HTML5 之间的差异

<video> 标签是 HTML5 的新标签。

提示和注释

提示：可以在 `<video>` 和 `</video>` 标签之间放置文本内容，这样不支持 `<video>` 元素的浏览器就可以显示出该标签的信息。

可选属性

New：HTML5 中的新属性。

属性	值	描述
autoplay New	autoplay	如果出现该属性，则视频在就绪后马上播放。
controls New	controls	如果出现该属性，则向用户显示控件，比如播放按钮。
height New	<i>pixels</i>	设置视频播放器的高度。
loop New	loop	如果出现该属性，则当媒介文件完成播放后再次开始播放。
muted New	muted	如果出现该属性，视频的音频输出为静音。
poster New	<i>URL</i>	规定视频正在下载时显示的图像，直到用户点击播放按钮。

preload <div>New</div>	auto metadata none	如果出现该属性，则视频在页面加载时进行加载，并预备播放。如果使用 "autoplay"，则忽略该属性。
src <div>New</div>	<i>URL</i>	要播放的视频的 URL。
width <div>New</div>	<i>pixels</i>	设置视频播放器的宽度。

全局属性

<video> 标签支持 [HTML 的全局属性](#)。

事件属性

<video> 标签支持 [HTML 的事件属性](#)。

[HTML <var> 标签](#)

[HTML <wbr> 标签](#)

点我分享笔记

[HTML <video> 标签](#)

HTML <wbr> 标签

实例

一段带有 Word Break Opportunity 的文本：

<p>学习 AJAX ，您必须熟悉 <wbr>Http<wbr>Request 对象。</p>

[尝试一下 »](#)

浏览器支持



所有主流浏览器都支持 <wbr> 标签，除了 Internet Explorer。

标签定义及使用说明

<wbr> (Word Break Opportunity) 标签规定在文本中的何处适合添加换行符。

提示：如果单词太长，或者您担心浏览器会在错误的位置换行，那么您可以使用 <wbr> 元素来添加 Word Break Opportunity（单词换行时机）。

HTML 4.01 与 HTML5 之间的差异

<wbr> 标签是 HTML5 中的新标签。

全局属性

<wbr> 标签支持 [HTML 的全局属性](#)。

事件属性

<wbr> 标签支持 HTML 的事件属性。

HTML <video> 标签

点我分享笔记

[illegible]