目录

[元素 2](#_Toc9278676)

[嵌套元素 2](#_Toc9278677)

[块级元素和内联元素 2](#_Toc9278678)

[空元素 2](#_Toc9278679)

[属性 3](#_Toc9278680)

[布尔属性 3](#_Toc9278681)

[标记文本 3](#_Toc9278682)

[<head> 3](#_Toc9278683)

[<meta> 3](#_Toc9278684)

[添加自定义图标 5](#_Toc9278685)

[在HTML中应用CSS和JS 5](#_Toc9278686)

[为文档设定主语言 5](#_Toc9278687)

[列表<list>元素： 6](#_Toc9278688)

[3. 描述列表<dl> 6](#_Toc9278689)

[图像<img>元素 7](#_Toc9278690)

[链接<a> 7](#_Toc9278691)

[块级链接 7](#_Toc9278692)

[HTML中的空白 7](#_Toc9278693)

[引用 8](#_Toc9278694)

[块引用 8](#_Toc9278695)

[行内引用 8](#_Toc9278696)

[引文 8](#_Toc9278697)

[引用特殊字符 8](#_Toc9278698)

[HTML注释 9](#_Toc9278699)

[重点强调 9](#_Toc9278700)

[上标和下标 10](#_Toc9278701)

[展示计算机代码 10](#_Toc9278702)

[标记时间和日期 10](#_Toc9278703)

[统一资源定位符（URL）与路径（path） 10](#_Toc9278704)

[文档片段 11](#_Toc9278705)

[电子邮件链接 11](#_Toc9278706)

[文档的基本组成区段（section） 13](#_Toc9278707)

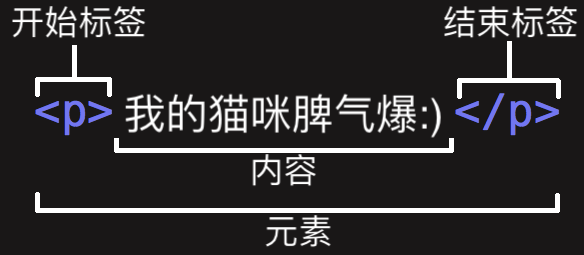
[无语义元素 13](#_Toc9278708)

[换行与水平分割线 14](#_Toc9278709)

HTML超文本标记语言，HyperText Mark Language

HTML是标记（markup或tag）语言，由一系列 “**元素**（elements）”组成，如：<head><title><body><header><footer><article><section><p><div><span><img><aside><audio><canvas><datalist><details><embed><nav><output><progress><video>等，不区分大小写或混合形式

# 元素



**<p>段落标签**，p为元素名称。

嵌套元素

如：<p>我的猫咪脾气<strong>爆</strong></p>

(**<strong>**标签使中间内容字体**加粗**)

(**<em>**标签使中间内容字体变成**斜体**强调)

块级元素和内联元素

**块级元素**（如：<p><li>等）在页面中以块的形式展现——与前后内容分开，独立成行。通常用于展示页面上结构化的内容。一个以block形式展现的块级元素不会被嵌套进内联元素中，但可以嵌套在其他块级元素中。

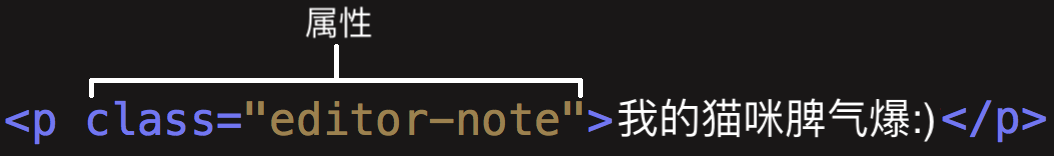
**内联元素**（如：<a><em><strong>）通常在块级元素中并包裹文档内容的一小部分，不会导致文本换行。

空元素

不包含**内容**和**结束标签**，如<img>嵌入图像，没有元素内容和</img>结束标签：<img src=”images/firefox-icon.png” alt=”测试图片”>

<img src="images/firefox-icon.png" alt="测试图片">

# 属性



元素**属性**（Attibute）应该包含：

1. 空格符
2. ②**属性名称**（这里是class），接上等号
3. ③由引号包围的**属性值**

**class**属性给元素赋予一个识别的**名字**（id），如：<p class=”id”>我家猫咪脾气爆：）</p>

布尔属性

即**没有值**的属性，他们只能有跟它的属性名一样的属性值。

例如**disabled**属性，标记表单输入使之变为不可用（变灰色），此时用户**不能输入**任何数据。

<input type="text" disabled="disabled">

可简写为

<input type="text" disabled>

# 标记文本

**<!DOCTYPE html>**：声明文档类型。

<html>元素，包裹整个页面，是一个**根元素**。

<head>元素放置的内容不是展现给用户的，例如面向搜索引擎的搜索关键字（keywords）、页面描述、CSS样式表和字符编码声明等。作用是包含一些页面的**元数据**（描述数据的数据 如<meta><title>）。

<meta>元素

<meta charset=”**utf-8**”>包含于<head>元素中，设置文档使用utf-8**字符集编码**。

包含了name和content特性：

name指定了meta元素的类型，说明该元素包含了什么类型的信息。

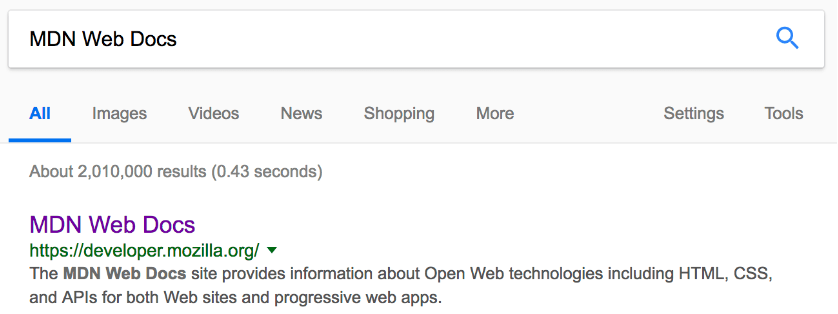
content指定了实际的元数据内容。

例一：

<meta name="author" content="Chris Mills">

<meta name="description" content="The MDN Learning Area aims to provide ….">

1. author指定作者
2. **description**指定关于页面内容的关键字描述，也被使用在搜索引擎现实的结果页中。



以上标题内容为<title>元素，下面的小字是description的content。

例二：

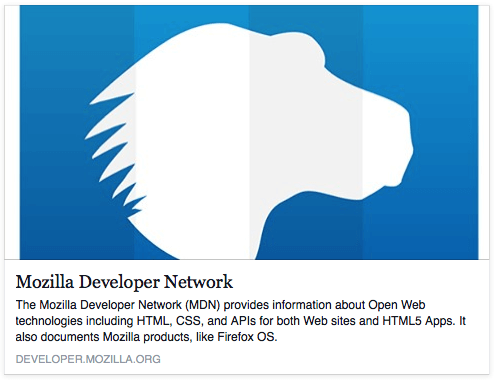
<meta property="og:image" content="https://developer.cdn.mozilla.net/static/img/opengraph-logo.dc4e08e2f6af.png">

<meta property="og:description" content="The Mozilla Developer Network (MDN) provides

information about Open Web technologies including HTML, CSS, and APIs for both Web sites

and HTML5 Apps. It also documents Mozilla products, like Firefox OS.">

<meta property="og:title" content="Mozilla Developer Network">



添加自定义图标

<link rel="shortcut icon" href="favicon.png" type="image/x-icon">

用于浏览器打开的标签页中和书签面板中的书签页面中



在HTML中应用CSS和JS

<link rel="stylesheet" href="my-css-file.css">

rel=“stylesheet”表示这是文档的样式表，href包含样式表路径

<script src="my-js-file.js"></script>

<script>部分既可放在<head>中，也可以放在<body>中

脚本可以直接放入<script>元素中，也可以指向外部脚本文件。

**<title>**元素设置页面的标题，出现在浏览器上,是面向搜索引擎的搜索关键字，也作为建议的书签名。

<body>元素包含期望让用户在访问页面时看到的内容

**标题**<heading>元素**，<h1>-<h6>**六个级别依次变小，不要使用标题元素来加大、加粗字体，要保持页面结构清晰，标题整洁，不要发生标题级别跳跃。

**段落**<paragraph>元素，**<p>**

为文档设定主语言

可以添加lang属性到HTML开始标签中

<html lang="en-US">

也可以将文档的分段设置为不同语言，如：

<p>Japanese example: <span lang="jp">ご飯が熱い。</span>.</p>

**<span>元素**

无语义，可以包裹CSS或JS内容

<span style="font-size: 32px; margin: 21px 0;">顶级标题</span>

# **列表<list>元素**：

1. **无序列表（Unordered List）**中项目的顺序并不重要，就像购物列表。用一个 [<ul>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/ul) 元素包围。
2. **有序列表（Ordered List）**中项目的顺序很重要，就像烹调指南。用一个 [<ol>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/ol) 元素包围。

列表的**每个项目用一个列表项目元素<li>包围**

<p>这里聚集着</p>

<ul>

<li>技术人员</li>

<li>思考者</li>

<li>建造者</li>

</ul>

<ul><ol>

**嵌套列表（Nesting Lists）** 可在有序中嵌套无序，或者在无序中嵌套有序。

1. 描述列表<dl>

<dl>

<dt>内心独白</dt>

<dd>只面向观众，而其他角色不会听到。</dd>

<dt>语言独白</dt>

<dd>观众和其他角色都可以听到。</dd>

<dt>旁白</dt>

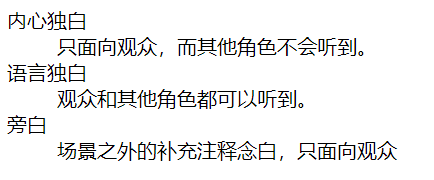
<dd>场景之外的补充注释念白，只面向观众</dd>

</dl>

**<dl>**闭合标签

其中每一项用**<dt>**(description term)元素闭合

每个描述用**<dd>**(description description)元素闭合。



浏览器会在描述部分(dd)和描述术语(dt)之间产生缩进，一个<dt>可以有多个<dd>。

# 链接<a>

<a href="http://www.mozilla.org/zh-CN/about/manifesto/">Mozilla 宣言</a>

其中Mozilla宣言是链接显示字样

**<a>**元素,a是anchor（锚）的缩写，它使被标签包裹的内容成为一个**超链接**

其可以添加的属性：

**href**：声明超链接的web地址。

**title**：如：title=“Mozilla”当鼠标悬浮时，标题作为提示信息出现（这里为Mozilla）。

**target**：目标target属性用于显示连接的浏览上下文。target="\_blank"将在**新标签页**中显示链接（打开另一个网页标签）。如果你希望在目前标签页显示链接，只需忽略这个属性。

块级链接

将块级元素转换为链接，如图像，把图像元素放到<a></a>标签中间

<a href="https://www.mozilla.org/en-US/">

<img src="mozilla-image.png" alt="mozilla logo that links to the mozilla homepage">

</a>

HTML中的空白

以下两个代码等价：

<p>狗 狗 很 呆 萌。</p>

<p>狗 狗 很

呆 萌。</p>

无论使用多少空白（空格或者换行），HTML解释器会将连续出现的空白字符减少为**一个**单独的空格符。

# 引用

块引用

一个块级内容（一个或多个段落，列表等）从其他地方被引用，用**<blockquote>**元素包裹引用内容，并且在**<cite>**属性里用URL来指向资源。浏览器会默认增加**缩进**

<blockquote cite="https://developer.mozilla.org/en-US/docs/Web/HTML/Element/blockquote">

<p>The <strong>HTML <code>&lt;blockquote&gt;</code> Element</strong> (or <em>HTML Block

Quotation Element</em>) indicates that the enclosed text is an extended quotation.</p>

</blockquote>

行内引用

与块引用工作方式相同，但使用的是**<q>**元素，浏览器默认增加**引号**，用于不需要分段的短引用。

引文

如果不使用JS和CSS，浏览器不会显示cite内容，要把<cite>元素放到引用元素旁，如在链接中引用。

<a href="https://developer.mozilla.org/en-US/docs/Web/HTML/Element/q">

<cite>MDN q page</cite></a>

引文默认的字体样式为**斜体**。

引用特殊字符

如<,>,”,’和&都是特殊字符，是html语法的一部分，其在文本中的引用方法：以&开始，以；结束。如：&<; 引用<符号

<p>HTML 中用 <p> 来定义段落元素。</p>

<p>HTML 中用 &lt;p&gt; 来定义段落元素</p>

**&lt;**和**&gt;**等价字符引用代替了角括号<和>。

| **原义字符** | **等价字符引用** |
| --- | --- |
| < | &lt; |
| > | &gt; |
| " | &quot; |
| ' | &apos; |
| & | &amp; |

HTML注释

注释内容用特殊的记号**<!--** 和 **-->** 包括起来。如：

<!-- <p>我在注释内！</p> -->

# 重点强调

**斜体强调 <em>**(emphasis)元素，也可用<span><i>元素

**粗体强调 <strong>** ，或者<span><b>元素

也可将<em><strong>嵌套在其他标签中

**下划线强调** <u>元素

<p>

总有一天我会改掉写<u>措字</u>的毛病。

</p>

**缩略语** **<abbr>**包裹一个缩略语或缩写，并用title属性提供解释，当鼠标移动到项目上时，出现提示

<p>我们使用 <abbr title="超文本标记语言（Hypertext Markup Language）">HTML</abbr> 来组织网页文档。</p>

**标记联系方式 <address>**标记的是编写html者的联系方式

<address>

<p>Page written by <a href="../authors/chris-mills/">Chris Mills</a>.</p>

</address>

上标和下标

**<sup>上标<sub>下标**

C<sub>8</sub>H<sub>10</sub>N<sub>4</sub>O<sub>2</sub>

x<sup>2</sup>

输出结果C8H10N4O2和 x2

展示计算机代码

**<code>**：标记计算机通用代码

**<pre>**：保留空格和换行（通常是代码块）

**<var>**：标记具体变量名

**<kbd>**：用于标记键盘输入（如ctrl）

**<samp>**：用于标记计算机程序的输出

标记时间和日期

**<time>元素**

<time datetime="2016-01-20">2016年1月20日</time>

# 统一资源定位符（URL）与路径（path）

**指向当前目录**

<a href="contacts.html">contacts page</a>

**指向子目录**

<a href="projects/index.html">project homepage</a>

**指向上级目录**（返回上一个目录级用两个英文点号.. 表示）

<a href="../pdfs/project-brief.pdf">project brief</a>

文档片段

超链接除了可以链接到文档外，也可以链接到HTML文档的特定部分（即**文档片段**）这要借助元素的**id**属性。如，链接到一个特定的标题

<h2 id="Mailing\_address">Mailing address</h2>

在URL结尾使用哈希符号（#）指向它，如：

<a href="contacts.html#Mailing\_address">mailing address</a>

链接同一份文档时，链接文档的另一部分：

<a href="#Mailing\_address">company mailing address</a>

**绝对URL**：指向web上的绝对位置定义的位置，包括协议和域名，总是指向确定的相同位置

**相对URL**：指向链接文件的相关位置

尽可能使用相对链接，链接到非html资源（需要下载的资源），留下提示

可使用下载**download**属性来提供一个默认的保存文件名，如：

<a href="https://download.mozilla.org/?product=firefox-latest-ssl&os=win64&lang=en-US"

download="firefox-latest-64bit-installer.exe">

Download Latest Firefox for Windows (64-bit) (English, US)

</a>

电子邮件链接

<a href="mailto:nowhere@mozilla.org">Send email to nowhere</a>

点击此链接打开一个新的电子邮件发送信息

**mailto:**后面跟链接邮件地址，若后面为空，电子邮件窗口也会被打开，只是没有收件人的地址信息，这在“分享”链接时有用

除了电子邮件地址，您还可以提供其他信息。事实上，任何标准的邮件头字段可以被添加到你提供的邮件URL。 其中最常用的是主题(subject)、抄送(cc)和主体(body) (这不是一个真正的头字段，但允许您为新邮件指定一个短内容消息)。 每个字段及其值被指定为查询项。

下面是一个包含cc、bcc、主题和主体的示例：

<a href="mailto:nowhere@mozilla.org?cc=name2@rapidtables.com&bcc=name3@rapidtables.com&amp;subject=The%20subject%20of%20the%20email &amp;body=The%20body%20of%20the%20email">

Send mail with cc, bcc, subject and body

</a>

**注意:** 每个字段的值必须是URL编码的。 也就是说，不能有非打印字符（不可见字符比如制表符、换行符、分页符）和空格 [percent-escaped](http://en.wikipedia.org/wiki/Percent-encoding). 同时注意使用问号（?）来分隔主URL与参数值，以及使用&符来分隔mailto:中的各个参数。 这是标准的URL查询标记方法。阅读 [The GET method](https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Sending_and_retrieving_form_data#The_GET_method) 以了解哪种URL查询标记方法是更常用的。

# 文档的基本组成区段（section）

**标题栏** **<header>**如果是<body>的子元素就是网站的全局页眉，如果是<article><section>子元素则是这些部分特有的页眉，且不同于<h>

<header> <!-- 本站所有网页的统一主标题 -->

<h1>聆听电子天籁之音</h1>

</header>

**导航栏 <nav>**其中不应包含二级链接等内容

<nav> <!-- 本站统一的导航栏 -->

<ul>

<li><a href="#">主页</a></li>

<li><a href=”#”>作品</a><li>

</ul>

<form> <!-- 搜索栏是站点内导航的一个非线性的方式。 -->

<input type="search" name="q" placeholder="要搜索的内容">

<input type="submit" value="搜索">

</form>

</nav>

**主内容 <main>** 存放每个页面独有内容，每个页面只能用一次，且直接位于<body>中，还可以有各种子内容区段，可用**<article>**(包裹一篇文章，与其他内容无关)**<section>**(组织页面使其按功能分块)和**<div>**等元素

**侧边栏 <aside>**

**页脚 <footer>**

无语义元素

**<span>**内联无语义元素，会将内容稍稍隔开

**<div>**块级无语义元素，如购物车组件

<div class="shopping-cart">

<h2>购物车</h2>

<ul>

<li>

<p><a href=""><strong>银耳环</strong></a>：$99.95.</p>

<img src="../products/3333-0985/" alt="Silver earrings">

</li>

</ul>

<p>售价：$237.89</p>

</div>

换行与水平分割线

**<br>换行**

**<hr>水平分割线**，两个元素都没有结束标签

# 图像<img>元素

这是一个**空元素**（不需要包含文本内容或闭合标签）

<img src="images/firefox-icon.png" alt="测试图片" width="400" height="341">

**src**(source)为图像文件路径的地址属性，**alt**为替换文字属性即“描述文本”，描述图像内容，用于当图像不能被用户看见时显示，**width**和**height**如果图片没有显示，你会注意到浏览器会为要显示的图片留下一定的空间，但如果没有保持正确大小，图片可能看起来会扭曲，因此应该使用CSS而不是HTML。

**注意：**搜索引擎也读取图像的文件名并把它们计入SEO。因此你应该给你的图片取一个描述性的文件名（dinosaur.jpg比img835.png好）。像[<img>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/img)和[<video>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/video" \o "HTML <video> 元素 用于在HTML或者XHTML文档中嵌入视频内容。" \t "_blank)这样的元素有时被称之为**替换元素**，因为这样的元素的内容和尺寸由外部资源（像是一个图片或视频文件）所定义，而不是元素自身。

**title**类似超链接

通过为图片搭配说明文字的方式来解说图片

<div class="figure">

<img src="/images/dinosaur\_small.jpg"alt="一只恐龙头部和躯干的骨架，它有一个巨大的头，长着锋利的牙齿。”>

<p>曼彻斯特大学博物馆展出的一只霸王龙的化石</p>

</div>

为将图片和文字联系起来，最好是下面这样：

<figure>

<img src="https://raw.githubusercontent.com/mdn/learning-area/master/html/multimedia-and-embedding/images-in-html/dinosaur\_small.jpg"

alt="一只恐龙头部和躯干的骨架，它有一个巨大的头，长着锋利的牙齿。"

<figcaption>曼彻斯特大学博物馆展出的一只霸王龙的化石</figcaption>

</figure>

**<figcaption>**文字描述 **<figure>**内容，<figure>可以是几张图片、代码、音视频、方程、表格或别的。

# 其他嵌入技术

<iframe>用来嵌入其他网页，例如来自在线视频提供商的视频

<iframe src="https://developer.mozilla.org/en-US/docs/Glossary"

width="100%" height="500" frameborder="0"

allowfullscreen sandbox>

<p> <a href="https://developer.mozilla.org/en-US/docs/Glossary">

Fallback link for browsers that don't support iframes

</a> </p>

</iframe>

1. **frameborder**：如果设置为1，则会在此框架和其他框架之间绘制边框，此为默认效果；0则删除边框，同css中的border:none
2. **allowfullscreen**：设置后允许设置全屏
3. **备选内容**：浏览器不支持时显示
4. **sandbox**：提高安全性设置（防黑客）

<embed><object>嵌入PDF,SVG甚至是Flash

|  | [**<embed>**](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/embed) | [**<object>**](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/object) |
| --- | --- | --- |
| 嵌入内容的[网址](https://developer.mozilla.org/en-US/docs/Glossary/URL" \o "URL: Uniform Resource Locator (URL) is a text string specifying where a resource can be found on the Internet.) | [src](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/embed#attr-src) | [data](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/object#attr-data) |
| 嵌入内容的准确[媒体类型](https://developer.mozilla.org/en-US/docs/Glossary/MIME_type) | [type](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/embed#attr-type) | [type](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/object#attr-type) |
| 由插件控制的框的高度和宽度（以CSS像素为单位） | [height](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/embed#attr-height) [width](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/embed#attr-width) | [height](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/object#attr-height) [width](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/object#attr-width) |
| 名称和值，将插件作为参数提供 | 具有这些名称和值的ad hoc属性 | 单标签[<param>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/param)元素，包含在内<object> |
| 独立的HTML内容作为不可用资源的回退 | 不支持（<noembed>已过时） | 包含在元素<object>之后<param> |

# 响应式图片(自适应图片)

不同尺寸

指创建可以自动更改其功能以适应不同屏幕尺寸，分辨率等的网页。如<picture>

<img srcset="elva-fairy-320w.jpg 320w,

elva-fairy-480w.jpg 480w,

elva-fairy-800w.jpg 800w"

sizes="(max-width: 320px) 280px,

(max-width: 480px) 440px,

800px"

src="elva-fairy-800w.jpg" alt="Elva dressed as a fairy">

**srcset**允许浏览器选择图像集，以及每个图像的大小，**图像的固有宽度**（以像素为单位）（480w）——注意到这里使用w单位，而不是你预计的px。这是图像的真实大小

**sizes**定义一组条件（如屏幕宽度），(max-width:480px)意思是“当视窗宽度是480像素或是更少”，440px：当条件为真，图像将填充的槽的宽度。

相同尺寸不同分辨率

srcset和x语法结合

<img srcset="elva-fairy-320w.jpg 1x,

elva-fairy-480w.jpg 1.5x,

elva-fairy-640w.jpg 2x"

src="elva-fairy-640w.jpg" alt="Elva dressed as a fairy">

根据设备像素选择图片加载

艺术方向

<img src="elva-800w.jpg" alt="Chris">

改用**<picture>**包含一些<source>

<picture>

<source type="image/svg+xml" srcset="pyramid.svg">

<source type="image/webp" srcset="pyramid.webp">

<img src="pyramid.png" alt="regular pyramid built from four equilateral triangles">

</picture>

**media**属性包含媒体条件，必须提供<img>元素及src和alt属性，在浏览器不支持<picture>时作为后备方案。 **type**属性使浏览器能立即拒绝其不支持的文件类型。可以在srcset和sizes中使用逗号分割的列表。

# 音频和视频

<video>

<video src="rabbit320.webm" controls>

<p>你的浏览器不支持 HTML5 视频。可点击<a href="rabbit320.mp4">此链接</a>观看</p>

</video>

**src** 指向视频资源

**controls控制组件**

**标签内段落** 作为**后备内容**，当浏览器不支持<video>标签时显示出来

<video controls width="400" height="400"

autoplay loop muted

poster="poster.png">

<source src="rabbit320.mp4" type="video/mp4">

<source src="rabbit320.webm" type="video/webm">

<p>你的浏览器不支持 HTML5 视频。可点击<a href="rabbit320.mp4">此链接</a>观看</p>

</video>

src从<video>中移除，放进单独的**<source>**中，浏览器播放第一个匹配的媒体。

**type**属性可选，浏览器会通过检查这个属性来迅速跳过那些不支持的格式，否则会尝试加载每一个文件。

**autoplay**使视频或音频立即播放

**loop**循环播放

**muted**默认关闭声音

**poster**属性指向一个图像URL，这个图像会在视频播放前显示，通常用于广告

**prelaod**用来缓冲较大文件：①**“none”**：不缓冲 ②**“auto”**：页面加载后缓存 ③**“metadata”**：仅缓冲文件元数据

width/height：视频边框拉伸，未被视频填充部分显示默认背景颜色。

<audio>

与<video>基本相同

<audio controls>

<source src="viper.mp3" type="audio/mp3">

<source src="viper.ogg" type="audio/ogg">

<p>你的浏览器不支持 HTML5 音频，可点击<a href="viper.mp3">此链接</a>收听。</p>

</audio>

不支持width/height属性，也不支持poster属性（没有视觉部件）

显示音轨文本

借助WebTT格式，**<track>**标签提供音频内容副本

**WebTT**是一个格式，用来编写文本文件，文件以.vtt后缀名保存文件，有以下三种类型：

**subtitle**添加翻译字幕

**captions**同步翻译对白

**timed descriptions**将文字转换为音频

WEBVTT

1

00:00:22.230 --> 00:00:24.606

第一段字幕

2

00:00:30.739 --> 00:00:34.074

第二段

引用WebTT文件：

<video controls>

<source src="example.mp4" type="video/mp4">

<source src="example.webm" type="video/webm">

<track kind="subtitles" src="subtitles\_en.vtt" srclang="en">

</video>

1. <track>标签需放在<audio><video>当中，<source>之后
2. **kind**属性来指明哪一种类型
3. **srclang**来告诉浏览器编写subtitles的语言

# 添加矢量图形

**矢量图形**：用算法定义尺寸较小却可以高度缩放，SVG格式

**位图**：用像素网格定义，bmp，png，jpg，gif格式

SVG是用于描述矢量图像的XML语言。如[<circle>](https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/circle) 和[<rect>](https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/rect" \o "rect元素是SVG的一个基本形状，用来创建矩形，基于一个角位置以及它的宽和高。它还可以用来创建圆角矩形。" \t "_blank)，

[<feColorMatrix>](https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/feColorMatrix)（使用变换矩阵转换颜色）[<animate>](https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/animate) （矢量图形的动画部分）和 [<mask>](https://developer.mozilla.org/zh-CN/docs/Web/SVG/Element/mask)（在图像顶部应用蒙版）

内联SVG

可以使用CSS交互（如:focus），只适用于在一个地方使用SVG

<svg width=”100%” height=”100%”>

<rect width="100%" height="100%" fill="green" />

<circle cx="100%" cy="100%" r="150" fill="blue" stroke=”black”/>

<polygon points="120,0 240,225 0,225" fill="pink"/>

<text x="50" y="100" font-family="Verdana" font-size="55"

fill="white" stroke="black" stroke-width="2">

Hello!

</text>

</svg>

外联SVG

**<img>**在src属性中引用，且需要一个heigth和width属性

<img

src="equilateral.svg"

alt="triangle with all three sides equal"

height="87px"

width="100px" />

* 无法使用JavaScript操作图像。
* 如果要使用CSS控制SVG内容，则必须在SVG代码中包含内联CSS样式。 （从SVG文件调用的外部样式表不起作用）

对于不支持SVG的浏览器，您可以从src属性引用PNG或JPG，并使用[srcset](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/img#attr-srcset)属性来引用SVG。 在这种情况下，仅支持浏览器将加载SVG - 较旧的浏览器将加载PNG：

<img src="equilateral.png" alt="triangle with equal sides" srcset="equilateral.svg">

还可以使用SVG作为CSS背景图像,

background: url("fallback.png") no-repeat center;

background-image: url("image.svg");

background-size: contain;

使用<iframe>嵌入SVG

<iframe src="triangle.svg" width="500" height="500" sandbox>

<img src="triangle.png" alt="Triangle with three unequal sides" />

</iframe>

若不支持iframe则显示img

# HTML表格

**<table>表格**标签，可在<table>中嵌套<table>

**<tr>**(table row)标签包裹**一行**的内容，**<td>**(table data)包容**单元格**，**<th>**包容**标题**

<tr>

<td>Hi, I'm your first cell.</td>

<td>I'm your second cell.</td>

<td>I'm your third cell.</td>

<td>I'm your fourth cell.</td>

</tr>

此为一行。第二行内容仍用<tr>标签继续包裹

单元格样式

利用标题中的**colspan**和**rowspan**属性，其属性值为没有单位的数值，数值决定宽度或高度**横跨几个单元格**

定义**整列**数据样式：**<col><colgroup>**；**<caption>**为表格添加**标题**

可以用**<thead><tbody><tfoot>**包裹<tr>使表格更加结构化

<table>

<caption>Dinosaurs in the Jurassic period</caption>

<colgroup>

<col>

<col style="background-color: yellow">

</colgroup>

<thead><tr>

<th>Data 1</th>

<th>Data 2</th>

</tr> </thead>

<tr>

<td>Calcutta</td>

<td>Orange</td>

</tr>

</table>

每个<col>定制每列样式，没有任何样式则添加一个空<col>元素

如果想把样式用在**每一列**，则需要包含**span**属性：

<colgroup>

<col style="background-color: yellow" span="2">

</colgroup>

在<th>元素中添加**scope**属性，来定义列或行标题

<th scope="colgroup">Clothes</th>

<th scope="rowgroup">Belgium</th>

<th scope="col">Trousers</th>

<th scope="row">Antwerp</th>



如表格所示，Clothes位于顶部，且定义了其他三个标题 scope="colgroup"

其他三个子标题scope="col"

替代scope的方法：为每个<th>属性添加一个不重复的id，在每个<td>元素添加包含<th>id属性的haders属性

<thead>

<tr>

<th id="purchase">Purchase</th>

<th id="location">Location</th>

<th id="date">Date</th>

</tr>

</thead>

<tbody>

<tr>

<th id="haircut">Haircut</th>

<td headers="location haircut">Hairdresser</td>

<td headers="date haircut">12/09</td>

……

# HTML表单

站点与程序交互的主要内容，由多个小部件组成，如文本字段（单行或多行），选择框，按钮，复选框或单选按钮

<form>

所有HTML表单都以一个<form>元素开始

* **action**属性定义了在提交表单时,应该把所收集的数据送给谁(/那个模块)(URL)去处理。.
* **method** 属性定义了发送数据的HTTP方法(“**get**”或“**post**”).

联系人表单<lable><input><textarea>

对应三个文本字段的标签<lable><input><textarea>元素

**<div>**使代码更容易样式化,将表单每个小部件用<div>划分，并用name属性命名

**<label>**元素的**for**属性引用相应的小部件id，将标签链接到表单小部件

**<input>**的**type**属性定义<input>的属性行为方式：

1. **text**：表示一个基本的单行文本字段，接受任何类型的文本输入
2. **email**：只接受格式良好的电子邮件地址单行文本字段

<input>是一个空元素，而**<textarea>**不是

如果要定义<input>的默认值（即文本框自动显示内容），需要使用value属性，<textarea>则放在开始和结束标签中间：

<input type="text" value="this element is filled " />

<textarea>this element is filled </textarea>

<button>

按钮元素：

<div class="button">

<button type="submit">Send your message</button>

</div>

<button>元素**type**属性，接受三个值中一个：

1. **submit**:单击按钮，将表单数据发送到<form>元素的action属性定义的网页
2. **reset:**将全部表单小部件回复默认值
3. **button：**不发生任何事

也可以使用<input type=”submit”>来生成按钮

向web服务器发送表单数据

<form>元素通过**action**和**method**属性定义发送数据的的位置和方式，并在表单每个小部件用**name**属性命名

<form action="/my-handling-form-page" method="post">

<div>

<label for="name">Name:</label>

<input type="text" id="name" name="user\_name" />

</div>

<div>

<label for="mail">E-mail:</label>

<input type="email" id="mail" name="user\_email" />

</div>

<div>

<label for="msg">Message:</label>

<textarea id="msg" name="user\_message"></textarea>

</div>

</form>

<fieldset><legend>元素

**<fieldset>**包裹具有相同目的的小部件组，在里面加上**<legend>**正式描述<fieldset>的用途。

<form>

<fieldset>

<legend>Fruit juice size</legend>

<p>

<input type="radio" name="size" id="size\_1" value="small">

<label for="size\_1">Small</label>

</p>

<p>

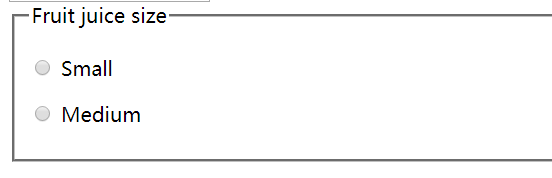
<input type="radio" name="size" id="size\_2" value="medium">

<label for="size\_2">Medium</label>

</p>

</fieldset>

</form>



设置可点击标签

<form>

<p>

<label for="taste\_1">I like cherry</label>

<input type="checkbox" id="taste\_1" name="taste\_cherry" value="1">

</p>

</form>

即文字部分（I like cherry）可以点击

多个标签

将小部件和他的标签嵌套在一个<label>元素中

<div>

<label for="username">Name: <abbr title="required">\*</abbr></label>

<input id="username" type="text" name="username">

</div>

表单通用属性

| **属性名称** | **默认值** | **描述** |
| --- | --- | --- |
| autofocus | (*false*) | 自动具有输入焦点 |
| disabled | (*false*) | 用户不能与元素交互。元素将从包含的元素继承它的设置，  如[<fieldset>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/fieldset" \o "此页面仍未被本地化, 期待您的翻译!" \t "_blank);如果没有包含disabled属性集的元素，  那么就启用了元素。 |
| form |  | 小部件与之相关联的表单元素。属性值必需是同个文档中的[<form>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/form" \o "HTML <form> 元素 表示了文档中的一个区域，这个区域包含有交互控制元件，用来向web服务器提交信息。" \t "_blank)   属性的 id属性。 |
| name |  | 元素的名称;这是用于表单数据提交的。 |
| value |  | 元素的初始值。 |

文本输入域

**readonly：**用户不能修改输入值

**disabled：**输入值永远不会与表单数据的其余部分一起发送

**placeholder：**文本输入框出现的文本，简略描述输入框的目的

单行文本域<input>

<input type="email" id="email" name="email" multiple>

1. **multiple**属性可以让用户输入多个电子邮件地址（以逗号分隔）‘’

<input type="password" id="pwd" name="pwd">

1. **密码域**对输入文本没有任何特殊约束，但会模糊输入的值

<input type="search" id="search" name="search">

1. **搜索域**主要是文本框样式的区别（圆角框，并有×用于清除输入的值）

<input type="tel" id="tel" name="tel">

1. **电话号码域**

<input type="url" id="url" name="url">

1. **URL域**如果输入无效URL，浏览器报错

多行文本域<textarea>

允许用户输入包含回车的文本，只接受文本内容

其元素属性：

| **属性名** | **默认值** | **描述** |
| --- | --- | --- |
| [cols](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/textarea#attr-cols) | 20 | 文本控件的可见宽度，平均字符宽度。 |
| [rows](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/textarea#attr-rows) |  | 控制的可见文本行数。 |
| [wrap](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/textarea#attr-wrap) | soft | 表示控件是如何包装文本的。可能的值：hard 或 soft |

# 表单下拉内容

选择框<select>

**<option>**元素作为子元素

可以在<option>中用**selected**属性设为默认选择

<option>可以嵌套在<optgroup>元素中，创建组值，**label**属性为组值内容

<select id="simple" name="simple">

<optgroup label="fruits">

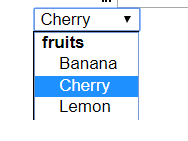
<option>Banana</option>

<option selected>Cherry</option>

<option>Lemon</option>

</optgroup>

</select>



多选选择框

将**multiple**属性添加到<select>元素，允许用户选择多个值（根据操作系统，如同时按下ctrl或cmd并点击多个值）

<select multiple id="multi" name="multi">

<option>Banana</option>

<option>Cherry</option>

<option>Lemon</option>

</select>

不再是下拉内容，而是显示在一个列表中

自动补全输入框

用**<datalist>**元素为表单小部件建议值，并用<option>元素来指定要显示的值

然后在文本域标签（通常是<input>）中使用**list**属性绑定

<label for="myFruit">What's your favorite fruit?</label>

<input type="text" name="myFruit" id="myFruit" list="mySuggestion">

<datalist id="mySuggestion">

<option>Apple</option>

<option>Banana</option>

<option>Blackberry</option>

<option>Blueberry</option>

<option>Lemon</option>

<option>Lychee</option>

<option>Peach</option>

<option>Pear</option>

</datalist>

可选中项

**复选框**使用type属性值**checkbox**

<input type="checkbox" checked id="carrots" name="carrots" value="carrots">

可包含**checked**属性使复选框在页面加载时自动被选中

# 高级表单部件

数字

<input type="number" name="age" id="age" min="1" max="10" step="2">

**step**属性指定增加和减少按钮更改小部件的值

滑块

<input type="range" name="beans" id="beans" min="0" max="500" step="10">

日期时间选择器

<input type="datetime-local" name="datetime" id="datetime">

<input type="month" name="month" id="month">

<input type="time" name="time" id="time">

<input type="week" name="week" id="week">

<label for="myDate">When are you available this summer?</label>

<input type="date" name="myDate" min="2013-06-01" max="2013-08-31" id="myDate">

图像按钮

<input type="image" alt="Click me!" src="my-img.png" width="80" height="30" />