## HTML基本语法

## 目录

[第一章 HTML总览 7](#_Toc24871)

[1、HTML 基本文档 7](#_Toc28666)

[2、基本标签（Basic Tags） 8](#_Toc20253)

[3、文本格式化（Formatting） 8](#_Toc20434)

[4、链接（Links） 8](#_Toc28811)

[5、图片（Images） 9](#_Toc28233)

[6、样式/区块（Styles/Sections） 9](#_Toc15360)

[7、无序列表 9](#_Toc2547)

[8、有序列表 9](#_Toc5887)

[9、定义列表 9](#_Toc30368)

[10、表格（Tables） 9](#_Toc16674)

[11、框架（Iframe） 10](#_Toc27739)

[12、表单（Forms） 10](#_Toc13726)

[13、实体（Entities） 10](#_Toc30591)

[第二章 HTML介绍 10](#_Toc7567)

[1.HTML 简介 10](#_Toc23094)

[（1） 什么是HTML? 11](#_Toc498)

[（2）HTML 标签 11](#_Toc29465)

[第三章 HTML 基本语法 11](#_Toc8771)

[1、HTML 图像 11](#_Toc6617)

[2、换行符br 12](#_Toc15626)

[3、HTML 标签对大小写不敏感 12](#_Toc10767)

[4、HTML 属性 12](#_Toc16868)

[5、html标题 12](#_Toc24780)

[6、HTML 注释 13](#_Toc25471)

[7、HTML 与 XHTML 之间的差异 13](#_Toc18608)

[8、HTML5 <progress> 标签 13](#_Toc25653)

[9、HTML 文本格式化 13](#_Toc14221)

[10、HTML 的各种标签 14](#_Toc30269)

[（1）文本格式化标签 14](#_Toc20660)

[（2）HTML "计算机输出" 标签 14](#_Toc31660)

[（3）HTML 引文, 引用, 及标签定义 14](#_Toc4575)

[11、HTML 链接- id 属性 15](#_Toc30560)

[12、链接HTML 15](#_Toc17982)

[（1）创建图片链接: 15](#_Toc13423)

[（2）无边框的图片链接: 15](#_Toc3754)

[（3）没有下划线的链接 15](#_Toc12801)

[（4）跳出框架 15](#_Toc4246)

[（5）锚点链接：定位到页面中的任意位置 16](#_Toc16126)

[13、HTML <head> 16](#_Toc10623)

[第四章 HTML样式实例 17](#_Toc19519)

[1、 背景颜色 17](#_Toc13033)

[2、字体, 字体颜色 ，字体大小 17](#_Toc28955)

[3、文本对齐方式 17](#_Toc808)

[4、内部样式表 17](#_Toc15750)

[5、外部样式表 18](#_Toc31110)

[第五章 HTML <label> 标签 18](#_Toc3627)

[第六章 HTML 图像 19](#_Toc6276)

[1、图像标签（ <img>）和源属性（Src） 19](#_Toc31250)

[2、HTML 图像- Alt属性 20](#_Toc1869)

[3、HTML 图像- 设置图像的高度与宽度 20](#_Toc28382)

[6、 图片标签的相对路径 20](#_Toc23442)

[(1) 相对路径 20](#_Toc30638)

[(2) 绝对路径 20](#_Toc22371)

[第七章HTML表格 20](#_Toc19474)

[1、表格简介 21](#_Toc25291)

[2、表格属性 21](#_Toc1835)

[3、HTML 表格表头 22](#_Toc8885)

[4、HTML 表格标签 22](#_Toc652)

[（1）没有边框的表格 22](#_Toc31858)

[（2）垂直标题 22](#_Toc22634)

[（3）单元格间距 23](#_Toc26037)

[（4）带标题的表格 23](#_Toc8110)

[（5）跨列或跨行的表格 23](#_Toc15338)

[（6） 合并单元格 24](#_Toc31151)

[第八章 HTML列表 24](#_Toc986)

[1、HTML无序列表 24](#_Toc18492)

[2、HTML 有序列表 25](#_Toc31784)

[3、HTML 自定义列表 25](#_Toc32531)

[4、 HTML 列表标签 25](#_Toc10990)

[6、不同类型的有序列表示例 26](#_Toc25783)

[7、嵌套列表：缩进格式，可以嵌套多个ul 26](#_Toc11712)

[第九章 HTML的<div> 和<span> 27](#_Toc9254)

[1、HTML 区块元素 27](#_Toc28260)

[2、HTML的内联元素 27](#_Toc24967)

[3、HTML <div> 元素 27](#_Toc17398)

[4、HTML <span> 元素 27](#_Toc5332)

[第十章 HTML表单和输入 27](#_Toc30378)

[1、表单简介 27](#_Toc23816)

[2、 HTML 表单 - 输入元素 28](#_Toc10946)

[（1）input的控件与属性 28](#_Toc4550)

[3、HTML表单--输入类型 29](#_Toc13332)

[（1） 文本域（Text Fields） 29](#_Toc12101)

[（2） 密码字段 29](#_Toc7890)

[（3） 单选按钮（Radio Buttons） 30](#_Toc24429)

[（4） 复选框（Checkboxes） 30](#_Toc11253)

[（5） 提交按钮(Submit) 30](#_Toc29070)

[（6） 下拉列表 30](#_Toc23632)

[（7）创建按钮 31](#_Toc2539)

[第十一章 HTML框架 31](#_Toc27912)

1、iframe 语法 31

[2、iframe - 设置高度与宽度 31](#_Toc2264)

3、iframe - 移除边框 ...............................................................................................................31

[4、使用 iframe 来显示目标链接页面 32](#_Toc23349)

[第十二章 HTML颜色、脚本与实体字符 32](#_Toc22301)

[1、HTML颜色 32](#_Toc5218)

[2、HTML 脚本 32](#_Toc7050)

[（1）HTML <script> 标签 32](#_Toc22557)

[（2）HTML<noscript> 标签 33](#_Toc14516)

[3、HTML 字符实体 33](#_Toc24228)

[（1）HTML 实体 33](#_Toc27597)

[（2）不间断空格 33](#_Toc6594)

[（3）结合音标符 33](#_Toc18325)

[（4）HTML字符字体 34](#_Toc26485)

[第十三章 HTML统一资源定位器 34](#_Toc14563)

[1、URL定义 34](#_Toc14520)

[2、常见的 URL Scheme 35](#_Toc14309)

[3、URL 字符编码 35](#_Toc6774)

[4、HTML 速查列表 35](#_Toc31126)

[5、HTML框架 35](#_Toc28783)

[第十四章 HTML5的简介 35](#_Toc5417)

[1、什么是 HTML5? 35](#_Toc31065)

[2、<canvas> 新元素 36](#_Toc19231)

[3、新多媒体元素 36](#_Toc14976)

[4、新表单元素 36](#_Toc24371)

[5、新的语义和结构元素 36](#_Toc30152)

[第十五章 HTML5 Canvas 37](#_Toc16400)

[1、什么是 canvas? 38](#_Toc27248)

[2、创建一个画布（Canvas） 38](#_Toc22411)

[3、使用 JavaScript 来绘制图像 38](#_Toc5877)

[4、Canvas 坐标 38](#_Toc8756)

[5、Canvas - 路径 38](#_Toc19183)

[6、Canvas - 文本 39](#_Toc746)

[7、Canvas - 渐变 39](#_Toc23238)

[第十六章 HTML5 SVG 40](#_Toc5507)

[1、什么是SVG？ 40](#_Toc160)

[2、SVG优势 40](#_Toc2302)

[3、SVG 圆形 41](#_Toc32690)

[4、SVG 与 Canvas两者间的区别 41](#_Toc25953)

[第十七章 HTML5 拖放和地理定位 41](#_Toc11109)

[1、拖放介绍 41](#_Toc14037)

[（1）定义 41](#_Toc29693)

[（2）设置元素为可拖放 42](#_Toc31099)

[（3）拖动什么 - ondragstart 和 setData() 42](#_Toc27470)

[（4）放到何处 - ondragover 42](#_Toc31071)

[（5）进行放置 - ondrop 42](#_Toc11616)

[2、HTML5 Geolocation（地理定位） 42](#_Toc7549)

[第十八章 HTML5 Video(视频) 43](#_Toc1722)

[1、HTML5 (视频)- 如何工作 43](#_Toc17527)

[2、视频格式与浏览器的支持 43](#_Toc27730)

[3、HTML5 <video> - 使用 DOM 进行控制 44](#_Toc10554)

[第十九章 HTML5 Audio(音频) 45](#_Toc8770)

[HTML5 Audio - 如何工作 45](#_Toc7699)

[第二十章 HTML5 新Input 类型 45](#_Toc16337)

[1、Input 类型: color 45](#_Toc7926)

[2、Input 类型: date 46](#_Toc31210)

[3、Input 类型: datetime 46](#_Toc20731)

[4、Input 类型: datetime-local 46](#_Toc27054)

[5、Input 类型: email 46](#_Toc18704)

[6、Input 类型: month 47](#_Toc13)

[7、Input 类型: number 47](#_Toc19582)

[8、Input 类型: range 47](#_Toc4043)

[9、Input 类型: search 48](#_Toc1398)

[10、Input 类型: tel 48](#_Toc31088)

[11、Input 类型: time 48](#_Toc9086)

[12、Input 类型: url 48](#_Toc17555)

[13、Input 类型: week 48](#_Toc12293)

[第二十一章 HTML5 表单元素 48](#_Toc15441)

[1、HTML5 <datalist> 元素 49](#_Toc15220)

[2、HTML5 <keygen> 元素 49](#_Toc25128)

[3、HTML5 <output> 元素 49](#_Toc11417)

[4、HTML5 表单属性 50](#_Toc27636)

[第二十二章 HTML5 语义元素 50](#_Toc14521)

[1、HTML5 <section> 元素 50](#_Toc288)

[2、HTML5 <article> 元素 50](#_Toc7372)

[3、HTML5 <nav> 元素 50](#_Toc18771)

[4、HTML5 <aside> 元素 50](#_Toc27277)

[5、HTML5 <footer> 元素 51](#_Toc15007)

[6、HTML5 <figure> 和 <figcaption> 元素 51](#_Toc16567)

[第二十三章HTML5 Web 存储 52](#_Toc22540)

[1、localStorage 和 sessionStorage  52](#_Toc19426)

[2、localStorage 对象 52](#_Toc24271)

[3、sessionStorage 对象 52](#_Toc18159)

[第二十四章 HTML5 Web SQL 数据库 52](#_Toc10056)

[1、核心方法 52](#_Toc2502)

[2、打开数据库 52](#_Toc11936)

[3、执行查询操作 52](#_Toc5697)

[4、插入数据 53](#_Toc27453)

[第二十五章 HTML5 WebSocket 53](#_Toc23906)

## **第一章 HTML总览**

## **1、HTML 基本文档**

<!DOCTYPE html>

<html>

<head>

<title>文档标题</title>

</head>

<body> 可见文本... </body>

</html>

## **2、基本标签（Basic Tags）**

<h1>最大的标题</h1>

<h2> . . . </h2>

<h3> . . . </h3>

<h4> . . . </h4>

<h5> . . . </h5>

<h6>最小的标题</h6>

<p>这是一个段落。</p> <br> （换行） <hr> （水平线） <!-- 这是注释 -->

## **3、文本格式化（Formatting）**

<b>粗体文本</b>

<code>计算机代码</code>

<em>强调文本</em>

<i>斜体文本</i>

<kbd>键盘输入</kbd>

<pre>预格式化文本</pre>

<small>更小的文本</small>

<strong>重要的文本</strong>

<abbr> （缩写）

<address> （联系信息）

<bdo> （文字方向）

<blockquote> （从另一个源引用的部分）

<cite> （工作的名称）

<del> （删除的文本）

<ins> （插入的文本）

<sub> （下标文本）

<sup> （上标文本）

## **4、链接（Links）**

普通的链接：

<a href="http://www.example.com/">链接文本</a>

图像链接： <a href="http://www.exle.com/"><img src="URL" alt="替换文本"></a>

邮件链接： <a href="mailto:webmaster@example.com">发送e-mail</a>

书签： <a id="tips">提示部分</a> <a href="#tips">跳到提示部分</a>

## **5、图片（Images）**

<img loading="lazy" src="URL" alt="替换文本" height="42" width="42">

## **6、样式/区块（Styles/Sections）**

<style type="text/css">

h1 {color:red;}

p {color:blue;}

</style>

<div>文档中的块级元素</div>

<span>文档中的内联元素</span>

## **7、无序列表**

<ul>

<li>项目</li>

<li>项目</li>

</ul>

## **8、有序列表**

<ol>

<li>第一项</li>

<li>第二项</li>

</ol>

## **9、定义列表**

<dl>

<dt>项目 1</dt>

<dd>描述项目 1</dd>

<dt>项目 2</dt>

<dd>描述项目 2</dd>

</dl>

## **10、表格（Tables）**

<table> <tr>

<th>表格标题</th>

<th>表格标题</th>

</tr>

<tr>

<td>表格数据</td>

<td>表格数据</td>

</tr> </table>

## **11、框架（Iframe）**

<iframe src="demo\_iframe.htm"></iframe>

## **12、表单（Forms）**

<form action="demo\_form.php" method="post/get">

<input type="text" name="email" size="40" maxlength="50">

<input type="password"> <input type="checkbox" checked="checked">

<input type="radio" checked="checked"> <input type="submit" value="Send"><input type="reset"> <input type="hidden">

<select>

<option>苹果</option>

<option selected="selected">香蕉</option>

<option>樱桃</option>

</select> <textarea name="comment" rows="60" cols="20"></textarea>

</form>

## **13、实体（Entities）**

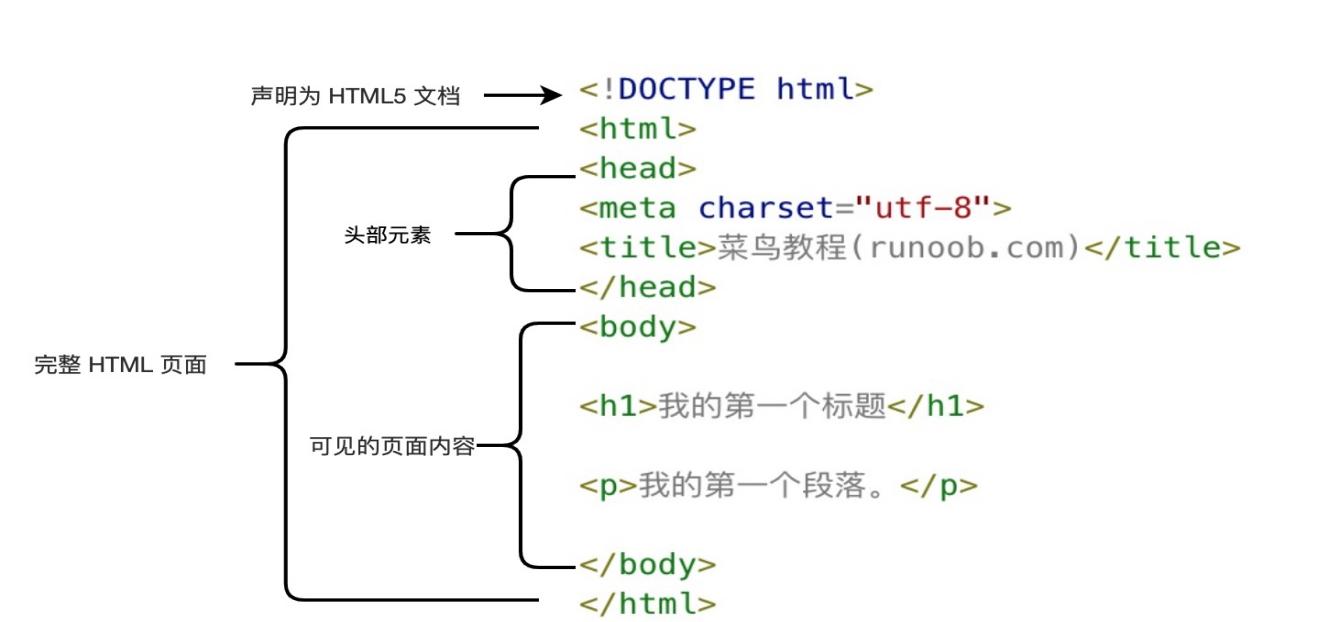
&lt; 等同于 <

&gt; 等同于 >

&#169; 等同于 ©

## **第二章 HTML介绍**

# 1.HTML 简介



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">。

<body> 元素包含了可见的页面内容。<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。

<!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：

1. **什么是HTML?**

HTML 是用来描述网页的一种语言，指的是超文本标记语言

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

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

**（2）HTML 标签**

HTML 标记标签通常被称为 HTML 标签 (HTML tag)。

HTML 标签是由标签是开始标签，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

## **第三章 HTML 基本语法**

# 1、HTML 图像

HTML 图像是通过标签 <img> 来定义的.

实例:

<img src="1.jpg" width="150" height="100" />

PS：路径可以为桌面，可以为直接图片名称。

# 2、换行符br

<br> 就是没有关闭标签的空元素（<br> 标签定义换行）。在开始标签中添加斜杠，比如 <br />，是关闭空元素的正确方法，HTML、XHTML 和 XML 都接受这种方式。即使 <br> 在所有浏览器中都是有效的，但使用 <br /> 其实是更长远的保障。

# 3、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。

# 4、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"。[其他标准属性参考表](https://kdocs.cn/l/ceuPit5abKcY)

|  |  |
| --- | --- |
| **属性** | **描述** |
| [accesskey](https://m.runoob.com/tags/att-global-accesskey.html" \o "HTML Global accesskey 属性) | 设置访问元素的键盘快捷键。 |
| [class](https://m.runoob.com/tags/att-global-class.html" \o "HTML Global class 属性) | 规定元素的类名（classname） |
| [contenteditable](https://m.runoob.com/tags/att-global-contenteditable.html" \o "HTML Global contenteditable 属性) | 规定是否可编辑元素的内容。 |
| [contextmenu](https://m.runoob.com/tags/att-global-contextmenu.html" \o "HTML contextmenu 属性) | 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-\*](https://m.runoob.com/tags/att-global-data.html) | 用于存储页面的自定义数据 |
| [dir](https://m.runoob.com/tags/att-global-dir.html" \o "HTML dir 属性) | 设置元素中内容的文本方向。 |
| [draggable](https://m.runoob.com/tags/att-global-draggable.html" \o "HTML draggable 属性) | 指定某个元素是否可以拖动 |
| [dropzone](https://m.runoob.com/tags/att-global-dropzone.html" \o "HTML dropzone 属性) | 指定是否将数据复制，移动，或链接，或删除 |
| [hidden](https://m.runoob.com/tags/att-global-hidden.html" \o "HTML hidden 属性) | hidden 属性规定对元素进行隐藏。 |
| [id](https://m.runoob.com/tags/att-global-id.html" \o "HTML id 属性) | 规定元素的唯一 id |
| [lang](https://m.runoob.com/tags/att-global-lang.html" \o "HTML lang 属性) | 设置元素中内容的语言代码。 |
| [spellcheck](https://m.runoob.com/tags/att-global-spellcheck.html" \o "HTML spellcheck 属性) | 检测元素是否拼写错误 |
| [style](https://m.runoob.com/tags/att-global-style.html" \o "HTML style 属性) | 规定元素的行内样式（inline style） |
| [tabindex](https://m.runoob.com/tags/att-global-tabindex.html" \o "HTML tabindex 属性) | 设置元素的 Tab 键控制次序。 |
| [title](https://m.runoob.com/tags/att-global-title.html" \o "HTML title 属性) | 规定元素的额外信息（可在工具提示中显示） |
| [translate](https://m.runoob.com/tags/att-global-translate.html" \o "HTML translate 属性) | 指定是否一个元素的值在页面载入时是否需要翻译 |

# 5、html标题

<hr> 标签在 HTML 页面中创建水平线，hr 元素可用于分隔内容。

<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>

结果：



html标题：字体大小降低

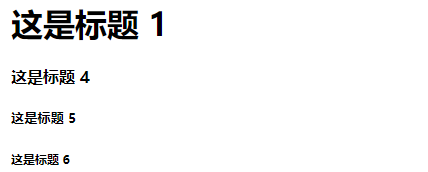
<h1>这是标题 1</h1>

<h4>这是标题 4</h4>

<h5>这是标题 5</h5>

<h6>这是标题 6</h6>

结果：



# 6、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。

Eg:<!-- 这是一个注释 -->

# 7、HTML 与 XHTML 之间的差异

xmlns 属性在 XHTML 中是必需的，但在 HTML中不是。HTML 4.01 与 HTML5之间的差异，HTML5 中，增加了一个新属性：manifest。

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。

# 8、HTML5 <progress> 标签

标记"下载进度"：

<progress value="22" max="100"></progress>

# 9、HTML 文本格式化

HTML 使用标签 <b>("bold") 与 <i>("italic") 对输出的文本进行格式, 如：粗体or斜体，通常标签 <strong> 替换加粗标签 <b> 来使用, <em> 替换 <i>标签使用。然而，这些标签的含义是不同的：<b> 与<i> 定义粗体或斜体文本。  
<strong> 或者 <em>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。

# 10、HTML 的各种标签

# （1）文本格式化标签

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<b>](https://m.runoob.com/tags/tag-b.html) | 定义粗体文本 |
| [<em>](https://m.runoob.com/tags/tag-em.html) | 定义着重文字 |
| [<i>](https://m.runoob.com/tags/tag-i.html) | 定义斜体字 |
| [<small>](https://m.runoob.com/tags/tag-small.html) | 定义小号字 |
| [<strong>](https://m.runoob.com/tags/tag-strong.html) | 定义加重语气 |
| [<sub>](https://m.runoob.com/tags/tag-sub.html) | 定义下标字 |
| [<sup>](https://m.runoob.com/html/m/tags/tag-sup.html) | 定义上标字 |
| [<ins>](https://m.runoob.com/tags/tag-ins.html) | 定义插入字 |
| [<del>](https://m.runoob.com/tags/tag-del.html) | 定义删除字 |

# (2)HTML "计算机输出" 标签

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<code>](https://m.runoob.com/tags/tag-code.html) | 定义计算机代码 |
| [<kbd>](https://m.runoob.com/tags/tag-kbd.html) | 定义键盘码 |
| [<samp>](https://m.runoob.com/tags/tag-samp.html) | 定义计算机代码样本 |
| [<var>](https://m.runoob.com/tags/tag-var.html) | 定义变量 |
| [<pre>](https://m.runoob.com/tags/tag-pre.html) | 定义预格式文本 |

# （3）HTML 引文, 引用, 及标签定义

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<abbr>](https://m.runoob.com/tags/tag-abbr.html) | 定义缩写 |
| [<address>](https://m.runoob.com/tags/tag-address.html) | 定义地址 |
| [<bdo>](https://m.runoob.com/tags/tag-bdo.html) | 定义文字方向 |
| [<blockquote>](https://m.runoob.com/tags/tag-blockquote.html) | 定义长的引用 |
| [<q>](https://m.runoob.com/tags/tag-q.html) | 定义短的引用语 |
| [<cite>](https://m.runoob.com/tags/tag-cite.html) | 定义引用、引证 |
| [<dfn>](https://m.runoob.com/tags/tag-dfn.html) | 定义一个定义项目。 |

## 

## **11、HTML 链接- id 属性**

id 属性可用于创建一个 HTML 文档书签。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。

## **12、链接HTML**

链接是通过标签 <a> 来定义的。

实例<a href ="https://www.runoob.com">这是一个外部链接</a>

内部链接：在同一级的网页文件，href=”gfthyr.html”

（1）创建图片链接:

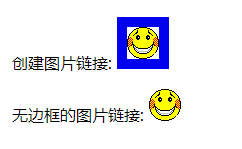
<a href="https://www.runoob.com/html/html-tutorial.html">

<img border="10" src="smiley.gif" alt="HTML" width="32" height="32"></a>

（2）无边框的图片链接:

<a href="https://www.runoob.com/html/html-tutorial.html">

<img border="0" src="smiley.gif" alt="HTML" width="32" height="32"></a>



（3）没有下划线的链接

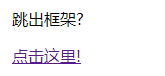
<a href="https://www.runoob.com/" style="text-decoration:none;">访问 runoob.com!</a>



（4）跳出框架

可以是跳转到别的页面或者跳转到空白页面，看链接的属性。

<a href="https://www.runoob.com/" target="\_top">点击这里!</a>



（5）锚点链接：定位到页面中的任意位置

（5.1）在链接文本的href属性值上，设置属性值为#名字，如href=”#edrf”

（5.2）找到目标位置标签，里面添加一个id 属性=刚才的名字，如<h3 id=”#edrf”> 第二季<h3/>。



## **13、HTML <head>**

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。

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<head>](https://m.runoob.com/tags/tag-head.html" \t "https://m.runoob.com/html/_blank) | 定义了文档的信息 |
| [<title>](https://m.runoob.com/tags/tag-title.html" \t "https://m.runoob.com/html/_blank) | 定义了文档的标题 |
| [<base>](https://m.runoob.com/tags/tag-base.html" \t "https://m.runoob.com/html/_blank) | 定义了页面链接标签的默认链接地址 |
| [<link>](https://m.runoob.com/tags/tag-link.html" \t "https://m.runoob.com/html/_blank) | 定义了一个文档和外部资源之间的关系 |
| [<meta>](https://m.runoob.com/tags/tag-meta.html" \t "https://m.runoob.com/html/_blank) | 定义了HTML文档中的元数据 |
| [<script>](https://m.runoob.com/tags/tag-script.html" \t "https://m.runoob.com/html/_blank) | 定义了客户端的脚本文件 |
| [<style>](https://m.runoob.com/tags/tag-style.html" \t "https://m.runoob.com/html/_blank) | 定义了HTML文档的样式文件 |

可以添加在头部区域的元素标签为: <title>, <style>, <meta>, <link>, <script>, <noscript> 和 <base>。

<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。

示例：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">

## **第四章 HTML样式实例**

# 1、 背景颜色

背景色属性（background-color）定义一个元素的背景颜色：

<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>



# 2、字体, 字体颜色 ，字体大小

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

<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>

**3、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：

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

**4、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>

**5、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，你就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>

|  |  |
| --- | --- |
| **值** | **描述** |
| alternate | 链接到该文档的替代版本（比如打印页、翻译或镜像）。 |
| author | 链接到该文档的作者。 |
| help | 链接到帮助文档。 |
| icon | 导入表示该文档的图标。 |
| license | 链接到该文档的版权信息。 |
| next | 表示该文档是集合中的一部分，且集合中的下一个文档是被引用的文档。 |
| prefetch | 规定应该对目标资源进行缓存。 |
| prev | 表示该文档是集合中的一部分，且集合中的上一个文档是被引用的文档。 |
| search | 链接到针对文档的搜索工具。 |
| stylesheet | 要导入的样式表的 URL。 |

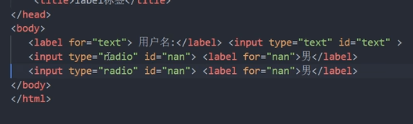
## **第五章 HTML <label> 标签**

<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 for=”sex” >男</label>

<input type=”radio” name=”sex” id=”sex”>

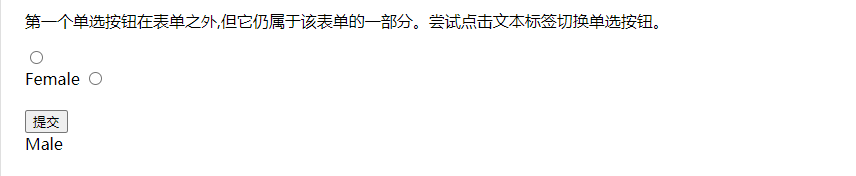


<label> 标签的 for 属性应当与相关元素的 id 属性相同：

|  |  |  |
| --- | --- | --- |
| **属性** | **值** | **描述** |
| [for](https://m.runoob.com/tags/att-label-for.html) | *element\_id* | 规定 label 与哪个表单元素绑定。 |
| [form](https://m.runoob.com/tags/att-label-form.html)**New** | *form\_id* | 规定 label 字段所属的一个或多个表单。 |

Eg:

<form action="demo\_form.html" id="form1">  
<input type="radio" name="sex" id="male" value="male"><br>  
<label for="female"> Female </label>  
<input type="radio" name="sex" id="female" value="female">  
<input type="submit" value="提交"></form>  
<label for="male" form="form1">Male</label>



<style>标签的属性值

|  |  |  |
| --- | --- | --- |
| **属性** | **值** | **描述** |
| [media](https://m.runoob.com/tags/att-style-media.html) | *media\_query* | 为样式表规定不同的媒体类型。 |
| [scoped](https://m.runoob.com/tags/att-style-scoped.html) | 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](https://m.runoob.com/tags/att-style-type.html) | text/css | 规定样式表的 MIME 类型。 |

## **第六章 HTML 图像**

**1、图像标签（ <img>）和源属性（Src）**

在 HTML 中，图像由<img> 标签定义。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：

<img src="*url*" alt="*some\_text*">

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](https://www.runoob.com/images/pulpit.jpg" \t "https://m.runoob.com/html/_blank)。

**2、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。

<img src="boat.gif" alt="Big Boat">

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。

**3、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。

<img src="pulpit.jpg" alt="Pulpit rock" width="304" height="228">

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。

**4、设置图像边框粗细 (border)**

**5、提示文本(title)鼠标放到图片上的提示文本**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<img>](https://m.runoob.com/tags/tag-img.html) | 定义图像 |
| [<map>](https://m.runoob.com/tags/tag-map.html) | 定义图像地图 |
| [<area>](https://m.runoob.com/tags/tag-area.html) | 定义图像地图中的可点击区域 |

# **图片标签的相对路径**

1. 相对路径

定义：以引用文件所在位置为参考基础，而建立出的目录路径。

这里简单来说，图片相对于HTML页面的位置

|  |  |  |
| --- | --- | --- |
| 相对路径分类 | 符号 | 说明 |
| 同一级路径 |  | 图像文件位于HTML文件同一级 如<img src=”baidu.gif”/> |
| 下一级路径 | / | 图像文件位于HTML文件下一级如<img src=”image/baidu.gif”/> |
| 上一级路径 | ../ | 图像文件位于HTML文件上一级 如<img src=”../baidu.gif”/> |

1. 绝对路径

定义：是指目录下的绝对位置，直接到达目标位置，通常是从盘符开始的路径。

例如：“D:\web\img\logo.gif”或完整的网络地址”http://www/itcast.com”

# **第七章HTML表格**

# **1、表格简介**

表格由 <table> 标签来定义。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）。

字母 td 指表格数据（table data），即数据单元格的内容。

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。

<table border="1"> <tr>

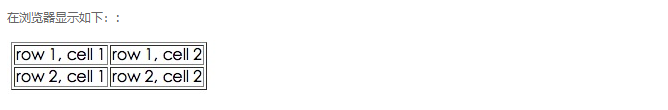
<td>row 1, cell 1</td>

<td>row 1, cell 2</td> </tr>

<tr> <td>row 2, cell 1</td>

<td>row 2, cell 2</td> </tr>

</table>



## 

Td单元格

Tr行标签

Table

# **2、表格属性**

|  |  |  |
| --- | --- | --- |
| 属性名 | 属性值 | 描述 |
| align | Left/center/right | 规定表格相对周围元素的对齐方式 |
| border | 1或”” | 规定表格单元是否有边框，默认为””。表示没有边框 |
| cellpadding | 像素值 | 规定单元边沿与其内容之间的空白，默认为1像素 |
| cellspacing | 像素值 | 规定单元格之间的空白，默认2像素 |
| width | 像素值或百分比 | 规定表格的宽度 |

## <table align=”center” cellpadding=”1”cellspacing=”0” border=”5” width=”500”height=”500”>

## **3、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：

Eg:<table border="1"> <tr>

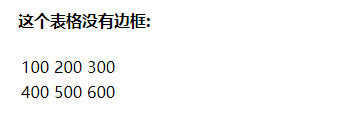
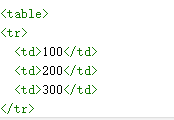
<th>Header 1</th>

<th>Header 2</th> </tr>

## **4、HTML 表格标签**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<table>](https://m.runoob.com/tags/tag-table.html) | 定义表格 |
| [<th>](https://m.runoob.com/tags/tag-th.html) | 定义表格的表头 |
| [<tr>](https://m.runoob.com/tags/tag-tr.html) | 定义表格的行 |
| [<td>](https://m.runoob.com/tags/tag-td.html) | 定义表格单元 |
| [<caption>](https://m.runoob.com/tags/tag-caption.html) | 定义表格标题 |
| [<colgroup>](https://m.runoob.com/tags/tag-colgroup.html) | 定义表格列的组 |
| [<col>](https://m.runoob.com/tags/tag-col.html) | 定义用于表格列的属性 |
| [<thead>](https://m.runoob.com/tags/tag-thead.html) | 定义表格的页眉 |
| [<tbody>](https://m.runoob.com/tags/tag-tbody.html) | 定义表格的主体 |
| [<tfoot>](https://m.runoob.com/tags/tag-tfoot.html) | 定义表格的页脚 |

# （1）没有边框的表格

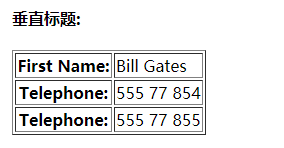


# （2）垂直标题

<table border="1"><tr>

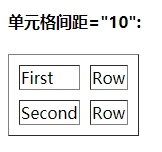
<th>First Name:</th>

<td>Bill Gates</td></tr>



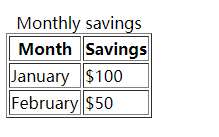
# （3）单元格间距

<table border="1" cellspacing="10">



（4）带标题的表格

<caption>Monthly savings</caption>



（5）跨列或跨行的表格

单元格跨两列: colspan=”合并单元格的个数”

<table border="1"><tr>

<th>Name</th>

<th colspan="2">Telephone</th>

</tr> <tr> <td>Bill Gates</td> <td>555 77 854</td> <td>555 77 855</td>

</tr></table>

单元格跨两行: rowspan=”合并单元格的个数”

<table border="1"><tr>

<th>First Name:</th>

<td>Bill Gates</td>

</tr> <tr><th rowspan="2">Telephone:</th> <td>555 77 854</td> </tr>

<tr> <td>555 77 855</td> </tr>

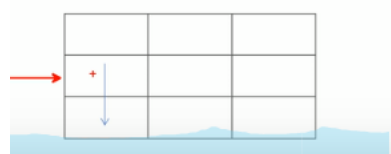
</table>

1. 合并单元格

目标单元格：（写合并代码）

跨行：最上侧单元格为目标单元格，写合并代码

跨列：最左侧单元格为目标单元格，写合并代码



合并单元格三步曲：

1. 先确定是跨行还是跨列合并；
2. 找到目标单元格，写上合并方式=合并的单元格数量，比如：<td colspan=’2’></td>；
3. 删除多余单元格。

# **第八章 HTML列表**

## **1、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 <ul> 标签

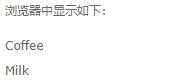
<ul>  
<li>Coffee</li>  
<li>Milk</li>  
</ul>

## 

## **2、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 <ol> 标签。每个列表项始于 <li> 标签。 列表项使用数字来标记。

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



**3、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。

## **HTML 列表标签**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<ol>](https://m.runoob.com/tags/tag-ol.html" \t "https://m.runoob.com/html/_blank) | 定义有序列表 |
| [<ul>](https://m.runoob.com/tags/tag-ul.html" \t "https://m.runoob.com/html/_blank) | 定义无序列表 |
| [<li>](https://m.runoob.com/tags/tag-li.html" \t "https://m.runoob.com/html/_blank) | 定义列表项 |
| [<dl>](https://m.runoob.com/tags/tag-dl.html) | 定义列表 |
| [<dt>](https://m.runoob.com/tags/tag-dt.html" \t "https://m.runoob.com/html/_blank) | 自定义列表项目 |
| [<dd>](https://m.runoob.com/tags/tag-dd.html" \t "https://m.runoob.com/html/_blank) | 定义自定列表项的描述 |

**5、无序列表示例**

圆点列表：

<ul style="list-style-type:disc">

<li>Apples</li>

<li>Bananas</li>

</ul>

<ul style="list-style-type:circle">

<ul style="list-style-type:square">

**6、不同类型的有序列表示例**

大写字母列表：

<ol type="A">

<li>Apples</li>

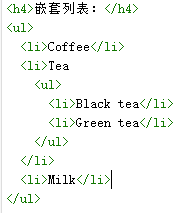
<li>Bananas</li>

</ol>

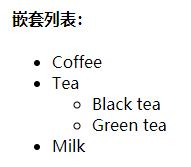
<ol type="a">

<ol type="i">

**7、嵌套列表：缩进格式，可以嵌套多个ul**



结果：



|  |  |  |
| --- | --- | --- |
| 标签名 | 定义 | 说明 |
| <ul></ul> | 无序标签 | 里面只包含li,没有顺序，使用较多，li里面可以包含任何标签 |
| <ol></ol> | 有序标签 | 里面只包含li,有顺序，使用较少，li里面可以包含任何标签 |
| <dl></dl> | 自定义标签 | 里面只包含dt和dd,dt和dd里面可以放任何标签 |

# **第九章 HTML的<div> 和<span>**

## **1、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>, <ul>, <table>

## **2、HTML的内联元素**

内联元素在显示时通常不会以新行开始。实例: <b>, <td>, <a>, <img>

## **3、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。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。

Eg: <div align="center" style="border:1px solid red"></div>

## **4、HTML <span> 元素**

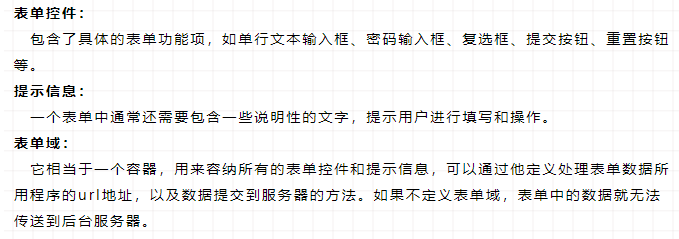
HTML <span> 元素是内联元素，可用作文本的容器。

<span> 元素也没有特定的含义。当与 CSS 一同使用时，<span> 元素可用于为部分文本设置样式属性。

|  |  |
| --- | --- |
| **值** | **描述** |
| left | 左对齐内容 |
| right | 右对齐内容 |
| center | 居中对齐内容 |
| justify | 对行进行伸展，这样每行都可以有相等的长度（就像在报纸和杂志中） |

# **HTML表单和输入**

**1、表单简介**



表单域是一个包含表单元素的区域。

在HTML标签中，<form>标签用于定义表单域，以实现用户信息的手机和传递。

<form>会把它范围内的表单元素信息提交给服务器。

Eg:<form action=’url地址’ method=’提交方式’ name=’表单域名称’>各种表单元素控件</form>

常用属性

|  |  |  |
| --- | --- | --- |
| 属性 | 属性值 | 作用 |
| action | url地址 | 用于指定接收并处理表单数据的服务器程序的url地址 |
| method | get/post | 用于设置表单数据的提交方式，其取值为get或post |
| name | 名称 | 用于指定表单的名称，以区分到一个页面中的多个表单域 |

HTML 表单用于收集用户的输入信息。

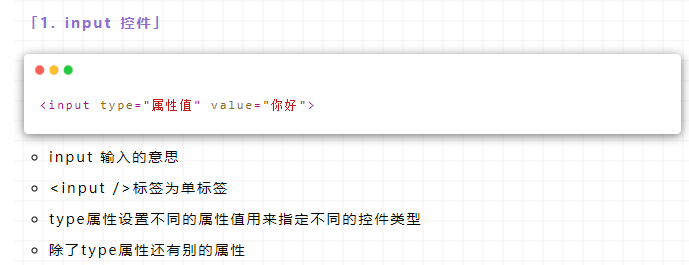
HTML 表单表示文档中的一个区域，此区域包含交互控件，将用户收集到的信息发送到 Web 服务器。

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



## **HTML 表单 - 输入元素**

（1）input的控件与属性





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。

Checked： 规定input元素首次加载时应当被选中

Maxlength：规定输入字段的最大长度。

Reset： 可以还原初始状态

File： 上传文件。<input type=”file”>

**3、HTML表单--输入类型**

1. 文本域（Text Fields）

定义：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。（输入内容较多时），可以定义多行文本。

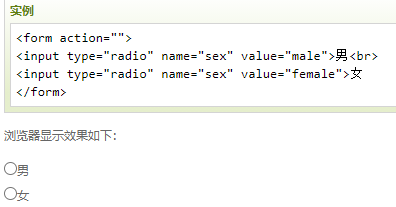
<textarea rows="10" cols="30">我是一个文本框。</textarea>

## 密码字段

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

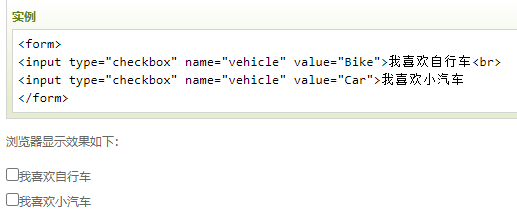
1. 单选按钮（Radio Buttons）

定义：需要有相同的name属性，<input type="radio"> 标签定义了表单的单选框选项。



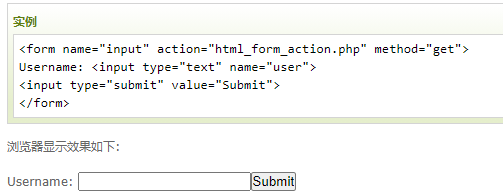
## 复选框（Checkboxes）

## 定义：<input type="checkbox"> 定义了复选框。复选框可以选取一个或多个选项。



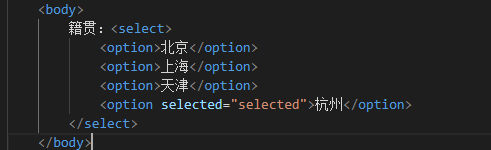
## 提交按钮(Submit)

## 定义：<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。



1. 下拉列表

定义：select，在option中定义 selected=”selected”时，当前项为默认选中项





（7）创建按钮

<form action="">

<input type="button" value="Hello world!">

</form>

# **HTML框架**

1. **iframe语法**

定义：该URL指向不同的网页。

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

## **2、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%")。

<iframe loading="lazy" src="demo\_iframe.htm" width="200" height="200"></iframe>



**3、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:

<iframe src="demo\_iframe.htm" frameborder="0"></iframe>

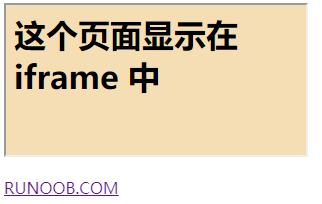


**4、使用 iframe 来显示目标链接页面**

## 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。

<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>



# **HTML颜色、脚本与实体字符**

**1、HTML颜色**

色板：[https://m.runoob.com/html/html-colors.html](https://kdocs.cn/l/ccJFBg9Sa00Z)

HTML 颜色由红色、绿色、蓝色混合而成。

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）。

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。[https://m.runoob.com/html/html-colornames.html](https://kdocs.cn/l/cpb9ryqqDo9f)

**2、HTML 脚本**

## （1）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。

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

## （2）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。

Eg:

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

<noscript>抱歉，你的浏览器不支持 JavaScript!</noscript>

<p>不支持 JavaScript 的浏览器会使用 &lt;noscript&gt; 元素中定义的内容（文本）来替代。</p>

**3、HTML 字符实体**

## （1）HTML 实体

在 HTML 中，某些字符是预留的。在 HTML 中不能使用小于号（<）和大于号（>），这是因为浏览器会误认为它们是标签。如果希望正确地显示预留字符，我们必须在 HTML 源代码中使用字符实体（character entities）。 字符实体类似这样：&#entity\_number; 如需显示小于号，我们必须这样写：&lt; 或 &#60; 或 &#060;

## （2）不间断空格

## 浏览器总是会截短 HTML 页面中的空格。如果您在文本中写 10 个空格，在显示该页面之前，浏览器会删除它们中的 9 个。如需在页面中增加空格的数量，您需要使用 &nbsp; 字符实体。

## （3）结合音标符

## 发音符号是加到字母上的一个"glyph(字形)"。一些变音符号, 如 尖音符 (  ̀) 和 抑音符 (  ́) 。变音符号可以出现字母的上面和下面，或者字母里面，或者两个字母间。

## 变音符号可以与字母、数字字符的组合来使用。

示例：

|  |  |  |  |
| --- | --- | --- | --- |
| **音标符** | **字符** | **Construct** | **输出结果** |
| ̀ | a | a&#768; | à |
| ́ | a | a&#769; | á |
| ̂ | a | a&#770; | â |
| ̃ | a | a&#771; | ã |
| ̀ | O | O&#768; | Ò |
| ́ | O | O&#769; | Ó |
| ̂ | O | O&#770; | Ô |
| ̃ | O | O&#771; | Õ |

## （4）HTML字符字体

|  |  |  |  |
| --- | --- | --- | --- |
| **显示结果** | **描述** | **实体名称** | **实体编号** |
|  | 空格 | &nbsp; | &#160; |
| < | 小于号 | &lt; | &#60; |
| > | 大于号 | &gt; | &#62; |
| & | 和号 | &amp; | &#38; |
| " | 引号 | &quot; | &#34; |
| ' | 撇号 | &apos; (IE不支持) | &#39; |
| ￠ | 分 | &cent; | &#162; |
| £ | 镑 | &pound; | &#163; |
| ¥ | 人民币/日元 | &yen; | &#165; |
| € | 欧元 | &euro; | &#8364; |
| § | 小节 | &sect; | &#167; |
| © | 版权 | &copy; | &#169; |
| ® | 注册商标 | &reg; | &#174; |
| ™ | 商标 | &trade; | &#8482; |
| × | 乘号 | &times; | &#215; |
| ÷ | 除号 | &divide; | &#247; |

# **HTML统一资源定位器**

**1、URL定义**

## URL 是一个网页地址。URL可以由字母组成，如"runoob.com"，或互联网协议（IP）地址： 192.68.20.50。大多数人进入网站使用网站域名来访问，因为名字比数字更容易记住。

Web浏览器通过URL从Web服务器请求页面。当您点击 HTML 页面中的某个链接时，对应的 <a> 标签指向万维网上的一个地址。一个统一资源定位器(URL) 用于定位万维网上的文档。

**scheme://host.domain:port/path/filename/**

**2、常见的 URL Scheme**

以下是一些URL scheme:

|  |  |  |
| --- | --- | --- |
| **Scheme** | **访问** | **用于...** |
| http | 超文本传输协议 | 以 http:// 开头的普通网页。不加密。 |
| https | 安全超文本传输协议 | 安全网页，加密所有信息交换。 |
| ftp | 文件传输协议 | 用于将文件下载或上传至网站。 |
| file |  | 您计算机上的文件。 |

## **3、URL 字符编码**

URL 只能使用 [ASCII 字符集](https://m.runoob.com/tags/html-ascii.html)，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 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。

# **4、HTML 速查列表**

[https://m.runoob.com/html/html-quicklist.html](https://kdocs.cn/l/cj8gD3yDmE7Y)

**5、HTML框架**

图像插入，链接插入，整个页面的背景颜色，标题背景，表格插入（包括多行多列），选项框架，目标链接的转接页面。

**第十四章 HTML5的简介**

**1、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

**2、<canvas> 新元素**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<canvas>](https://m.runoob.com/tags/tag-canvas.html) | 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 |

## **3、新多媒体元素**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<audio>](https://m.runoob.com/tags/tag-audio.html) | 定义音频内容 |
| [<video>](https://m.runoob.com/tags/tag-video.html) | 定义视频（video 或者 movie） |
| [<source>](https://m.runoob.com/tags/tag-source.html) | 定义多媒体资源 <video> 和 <audio> |
| [<embed>](https://m.runoob.com/tags/tag-embed.html) | 定义嵌入的内容，比如插件。 |
| [<track>](https://m.runoob.com/tags/tag-track.html) | 为诸如 <video> 和 <audio> 元素之类的媒介规定外部文本轨道。 |

## **4、新表单元素**

|  |  |
| --- | --- |
| **标签** | **描述** |
| [<datalist>](https://m.runoob.com/tags/tag-datalist.html) | 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>](https://m.runoob.com/tags/tag-keygen.html) | 规定用于表单的密钥对生成器字段。 |
| [<output>](https://m.runoob.com/tags/tag-output.html) | 定义不同类型的输出，比如脚本的输出。 |

## **5、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>](https://m.runoob.com/tags/tag-article.html) | 定义页面独立的内容区域。 |
| [<aside>](https://m.runoob.com/tags/tag-aside.html) | 定义页面的侧边栏内容。 |
| [<bdi>](https://m.runoob.com/tags/tag-bdi.html) | 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>](https://m.runoob.com/tags/tag-command.html) | 定义命令按钮，比如单选按钮、复选框或按钮 |
| [<details>](https://m.runoob.com/tags/tag-details.html) | 用于描述文档或文档某个部分的细节 |
| [<dialog>](https://m.runoob.com/tags/tag-dialog.html) | 定义对话框，比如提示框 |
| [<summary>](https://m.runoob.com/tags/tag-summary.html) | 标签包含 details 元素的标题 |
| [<figure>](https://m.runoob.com/tags/tag-figure.html) | 规定独立的流内容（图像、图表、照片、代码等等）。 |
| [<figcaption>](https://m.runoob.com/tags/tag-figcaption.html) | 定义 <figure> 元素的标题 |
| [<footer>](https://m.runoob.com/tags/tag-footer.html) | 定义 section 或 document 的页脚。 |
| [<header>](https://m.runoob.com/tags/tag-header.html) | 定义了文档的头部区域 |
| [<mark>](https://m.runoob.com/tags/tag-mark.html) | 定义带有记号的文本。 |
| [<meter>](https://m.runoob.com/tags/tag-meter.html) | 定义度量衡。仅用于已知最大和最小值的度量。 |
| [<nav>](https://m.runoob.com/tags/tag-nav.html) | 定义导航链接的部分。 |
| [<progress>](https://m.runoob.com/tags/tag-progress.html) | 定义任何类型的任务的进度。 |
| [<ruby>](https://m.runoob.com/tags/tag-ruby.html) | 定义 ruby 注释（中文注音或字符）。 |
| [<rt>](https://m.runoob.com/tags/tag-rt.html) | 定义字符（中文注音或字符）的解释或发音。 |
| [<rp>](https://m.runoob.com/tags/tag-rp.html) | 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>](https://m.runoob.com/tags/tag-section.html) | 定义文档中的节（section、区段）。 |
| [<time>](https://m.runoob.com/tags/tag-time.html) | 定义日期或时间。 |
| [<wbr>](https://m.runoob.com/tags/tag-wbr.html) | 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>：着重字体
* <center>
* <dir>
* <font>
* <frame>：框架
* <frameset>
* <noframes>
* <strike>
* <tt>

## **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。

## **1、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。

## **2、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 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。

使用 style 属性来添加边框:

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



## **3、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：

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

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



## **4、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)

**5、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();

在canvas中绘制圆形, 我们将使用以下方法:

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

## **6、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

## 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);

## **7、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。

Eg:

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);



# **第十七章 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。

## **1、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

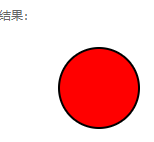
**2、SVG优势**

* 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

## **3、SVG 圆形**

<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>



<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>



## **4、SVG 与 Canvas两者间的区别**

SVG 是使用 XML 描述 2D 图形的语言。Canvas 通过 JavaScript 来绘制 2D 图形。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。

# **第十八章 HTML5 拖放和地理定位**

## **1、拖放介绍**

## （1）定义

## 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。

## （2）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">

## （3）拖动什么 - ondragstart 和 setData()

## ondragstart 属性调用了一个函数，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：

## （4）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。

*event*.preventDefault()

## （5）进行放置 - ondrop

当放置被拖数据时，会发生 drop 事件。ondrop 属性调用了一个函数，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

## **2、HTML5 Geolocation（地理定位）**

请使用 getCurrentPosition() 方法来获得用户的位置。

示例：

var x=document.getElementById("demo");

function getLocation() {

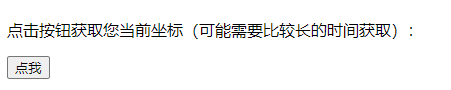
if (navigator.geolocation) {

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; }

结果：



# **第十九章 HTML5 Video(视频)**

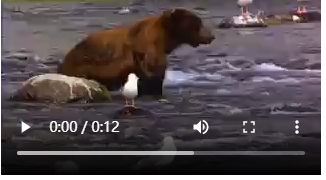
## **1、HTML5 (视频)- 如何工作**

<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>



<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。

## **2、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 |

## **3、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。

示例：

<div style="text-align:center">

<button onclick="playPause()">播放/暂停</button>

<button onclick="makeBig()">放大</button>

<button onclick="makeSmall()">缩小</button>

<button onclick="makeNormal()">普通</button>

<video id="video1" width="420">

<source src="mov\_bbb.mp4" type="video/mp4">

## <source src="mov\_bbb.ogg" type="video/ogg"> </video>

</div>

<script>

var myVideo=document.getElementById("video1");

function playPause(){

if (myVideo.paused)



myVideo.play();

else

myVideo.pause(); }

function makeBig(){

myVideo.width=560; }

function makeSmall(){

myVideo.width=320; }

function makeNormal(){

myVideo.width=420; }

</script>

# **第二十章 HTML5 Audio(音频)**

## **HTML5 Audio - 如何工作**

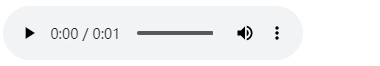
如需在 HTML5 中播放音频，你需要使用以下代码：

<audio controls>

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

<source src="horse.mp3" type="audio/mpeg">

</audio>



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

# **第二十一章 HTML5 新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。新的输入类型：color、date、datetime、datetime-local、email、month、number、range、search、tel、time、url、week。

## **1、Input 类型: color**

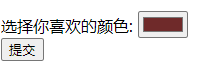
## 实例：从拾色器中选择一个颜色:

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

<form action="demo-form.php">选择你喜欢的颜色:

<input type="color" name="favcolor">

<input type="submit"></form>



## **2、Input 类型: date**

实例：定义一个时间控制器。

生日: <input type="date" name="bday">

<form action="demo-form.php">

生日: <input type="date" name="bday">

<input type="submit">

</form>



**3、Input 类型: datetime**

datetime 类型允许你选择一个日期（UTC 时间）。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">



**4、Input 类型: datetime-local**

datetime-local 类型允许你选择一个日期和时间 (无时区).

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"



## **5、Input 类型: email**

email 类型用于应该包含 e-mail 地址的输入域。

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">

## 

## **6、Input 类型: month**

month 类型允许你选择一个月份。

定义月与年 (无时区):

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

## **7、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

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



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 |
| 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 |

## **8、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。

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

  
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

**9、Input 类型: search**

search 类型用于搜索域，比如站点搜索或 Google 搜索。

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">

## **10、Input 类型: tel**

定义输入电话号码字段:

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

## **11、Input 类型: time**

time 类型允许你选择一个时间。定义可输入时间控制器（无时区）：

选择时间: <input type="time" name="usr\_time">

## **12、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。

定义输入URL字段:

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

## **13、Input 类型: week**

week 类型允许你选择周和年。定义周和年 (无时区):

选择周: <input type="week" name="week\_year">

# **第二十二章 HTML5 表单元素**

HTML5 有以下新的表单元素:

* <datalist>
* <keygen>
* <output>

## **1、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.

Eg:

<input list="browsers">

<datalist id="browsers">

<option value="Internet Explorer">

<option value="Firefox">

<option value="Chrome"> <option value="Opera"> <option value="Safari"> </datalist>



## **2、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。

Eg:

<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>



## **3、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

# **4、HTML5 表单属性**

## <input> height 和 width 属性

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">

# **第二十三章 HTML5 语义元素**

## **1、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。

<section>

<h1>WWF</h1>

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

</section>

## **2、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

<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>

**3、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。

<nav>

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

</nav>



## **4、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.

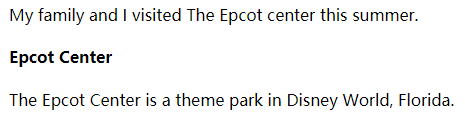
<p>My family and I visited The Epcot center this summer.</p>

<aside>

<h4>Epcot Center</h4>

<p>The Epcot Center is a theme park in Disney World, Florida.</p>

</aside>



## 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。

## **5、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

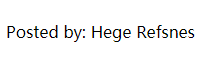
一个页脚通常包含文档的作者，著作权信息，链接的使用条款，联系信息等、文档中你可以使用多个 <footer>元素。

<footer>

<p>Posted by: Hege Refsnes</p>

<p><time pubdate datetime="2012-03-01"></time></p>

</footer>



## **6、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，则不应对文档流产生影响。

<figcaption> 标签定义 <figure> 元素的标题.

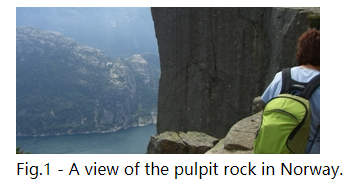
<figcaption>元素应该被置于 "figure" 元素的第一个或最后一个子元素的位置。

<figure>

<img loading="lazy" src="img\_pulpit.jpg" alt="The Pulpit Rock" width="304" height="228">

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

</figure>



**第二十四章HTML5 Web 存储**

## **1、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。

**2、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。

// 存储 localStorage.setItem("sitename", "菜鸟教程");

// 查找 document.getElementById("result").innerHTML = "网站名：" + localStorage.getItem("sitename");

## **3、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。

**第二十五章 HTML5 Web SQL 数据库**

## **1、核心方法**

以下是规范中定义的三个核心方法：

* 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。
* 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。
* executeSql：这个方法用于执行实际的 SQL 查询

**2、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。

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

openDatabase() 方法对应的五个参数说明：

**3、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)'); });

**4、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 "?"。**5、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');});

# **第二十六章 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。

笔记整理，方便大家查阅翻看，点击目录即可跳转至对应部分，省略了js的基础语法部分，一起加油！  
HTML基础入门(一)：https://blog.csdn.net/Augenstern\_QXL/article/details/115419453  
HTML5基础标签(二)：https://blog.csdn.net/augenstern\_qxl/category\_10946310.html?spm=1001.2014.3001.5482  
CSS基础入门(三)：https://blog.csdn.net/Augenstern\_QXL/article/details/115560532  
CSS核心盒模型浮动定位(四)：https://blog.csdn.net/Augenstern\_QXL/article/details/115560502  
CSS3提高(五)：  
https://blog.csdn.net/Augenstern\_QXL/article/details/115726577  
JavaScript基础之ES6(一)：  
https://blog.csdn.net/Augenstern\_QXL/article/details/115219073  
JavaScript基础之ES6(二)：  
https://blog.csdn.net/Augenstern\_QXL/article/details/115344398  
JavaScript提高之DOM(三)：  
https://blog.csdn.net/Augenstern\_QXL/article/details/115416921  
JavaScript提高之BOM（四）：  
<https://blog.csdn.net/Augenstern_QXL/article/details/115406408>