HTML5课程教案

版本号：HTML5 1.0

密 级：受控文档

课

程

标

准

化

2017年5月2日

1.文档属性

|  |  |
| --- | --- |
| 文档属性 | 内容 |
| 项目/任务名称： |  |
| 项目/任务编号： |  |
| 文档名称： |  |
| 文档编号： |  |
| 文档状态： |  |
| 作 者： | 实训部开发组 |
| 文档评审通过日期： |  |
| 评审负责人签字： |  |
| 参考模板： |  |

2.文档变更过程

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 修正日期 | 修正人 | 描述 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# HTML 基础

## 章节目标

* 了解HTML基本概念
* 了解HTML标签和元素
* 认识HTML基本结构

## HTML 概述

HTML（HyperText Markup Language的缩写），即超文本链接标记语言。它是在互联网发布超文本文件(通常所说的网页)的通用语言。所谓超文本，就是它可以加入图片、声音、动画、影视等内容，每一个HTML文档都是一种静态的网页文件，这个文件里面包含了HTML标记，这些标记并不是一种程序语言，它只是一种排版网页中资料显示位置的标记语言。每个标记的符号都是一条命令、它告诉浏览器如何显示文本。这些标记均由"<"和">"符号以及一个字符串组成。而浏览器的功能是对这些标记进行解释，显示出文字、图像、动画、播放声音。这些标记符号用“<标记名字属性>”来表示。一个HTML文件的后缀名是.htm或者是.html。

* 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 <b> 和 </b>
* 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

例：<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。

HTML元素(HTML Element)用来标记文本，表示文本的内容。比如body, p, title就是HTML元素。

HTML元素用Tag表示 ，Tag通常是成对出现的，比如<body></body>。起始的叫做Opening Tag，结尾的就叫做Closing Tag。

例：HTML 元素: <p>这是一个段落</p>

HTML元素可以拥有属性。属性可以扩展HTML元素的能力。

比如你可以使用一个bgcolor属性，使得页面的背景色成为红色，就像这样：

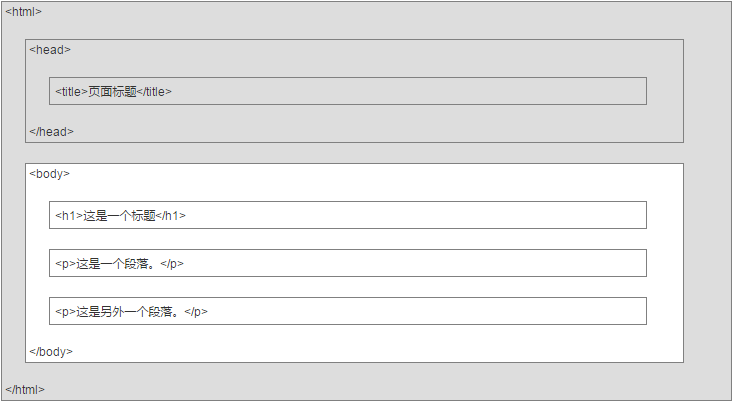
<body bgcolor="red">

属性通常由属性名和值成对出现，就像这样：name="value"。上面例子中的bgcolor就是name,value就是red，属性值一般用双引号标记起来。属性通常是附加给HTML的Opening Tag，而不是Closing Tag。

## HTML 基本结构

一个HTML文档是由一系列的元素和标记组成。元素名不区分大小写，HTML用标记来规定元素的属性和它在文件中的位置，HTML超文本文档分文档头和文档体两部分，在文档头里，对这个文档进行了一些必要的定义，文档体中才是要显示的各种文档信息。

下面是一个可视化的HTML页面结构:



注意：只有 <body> 区域 (白色部分) 才会在浏览器中显示。

示例解释:

这个文件的第一个Tag是<html>，这个Tag告诉你的浏览器这是HTML文件的头。文件的最后一个Tag是</html>，表示HTML文件到此结束。

在<head>和</head>之间的内容，是head信息。head信息是不显示出来的，你在浏览器里看不到。但是这并不表示这些信息没有用处。比如你可以在Head信息里加上一些关键词，有助于搜索引擎能够搜索到你的网页。

在<title>和</title>之间的内容，是这个文件的标题。你可以在浏览器最顶端的标题栏看到这个标题。

在<body>和</body>之间的信息，是正文。

在<h1>和</h1>之间的文字，表示定义最大的标题。

HTML文件看上去和一般文本类似，但是它比一般文本多了Tag，比如<html>，<h1>,<p>等，通过这些Tag，可以告诉浏览器如何显示这个文件。

## HTML版本

从初期的网络诞生后，已经出现了许多HTML版本:



## <!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">

## HTML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。

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <title>网页标题</title>  </head>  <body>  <h1>我的第一个标题</h1>  <p>我的第一个段落。</p>  </body>  </html> |

## HTML书写规范

### 使用小写元素名

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。

<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。

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> |

推荐使用:

|  |
| --- |
| <section>  <p>这是一个段落。</p>  </section> |

### 关闭所有 HTML 元素

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> |

推荐使用:

|  |
| --- |
| <section>  <p>这是一个段落。</p>  <p>这是一个段落。</p>  </section> |

### 使用小写属性名

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：

|  |
| --- |
| <body BGCOLOR="red"> |

推荐使用:

|  |
| --- |
| <body bgcolor="red"> |

### 属性值加引号

我们推荐属性值加引号：

如果属性值包含值，则必须使用引号 混合样式绝对不好 加引号的值更易阅读。

## HTML注释

注释的内容不在窗口中显示

|  |
| --- |
| <!-- 这是一段注释 -->  <p>这是一个段落。</p>  <!-- 记得在此处添加信息 --> |

注释：在开始标签中有一个惊叹号，但是结束标签中没有。浏览器不会显示注释，但是能够帮助记录您的 HTML 文档。可以利用注释在 HTML 中放置通知和提醒信息。

# HTML 元素和属性

## 章节目标

* 掌握基本元素
* 掌握基本属性

## HTML 元素



\* 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

### 嵌套的HTML元素

HTML 文档由嵌套的 HTML 元素构成。

【示例2.2-1】 HTML嵌套文档实例

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title></title>  </head>  <body>  <p>这是第一个段落</p>  </body>  </html> |

以上实例包含了三个 HTML 元素。

示例解析：

**<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）。

### 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。

<br> 就是没有关闭标签的空元素（<br> 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 <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**

## HTML 属性

### HTML属性语法

* 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"。

### HTML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：

|  |
| --- |
| <a href="http://www.baidu.com">这是一个链接</a> |

### HTML属性常用引用值

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:

|  |
| --- |
| <div style="background-image:url('a.jpg');"></div> |

### HTML基本属性

下面列出了适用于大多数 HTML 元素的属性：

|  |  |
| --- | --- |
| **·** | **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) |

下面这些元素不拥有基本属性。

|  |  |
| --- | --- |
| html、head | 文档和头部基本结构 |
| title | 网页标题 |
| base | 网页基准信息 |
| meta | 网页元信息 |
| param | 元素参数信息 |
| script、style | 网页的脚本和样式 |

这些元素一般位于文档投部区域，用来标识网页元信息。

### HTML语言属性

语言属性主要用来定义元素的语言类型，包括两个属性。

|  |  |
| --- | --- |
| lang | 定义元素的语言代码或编码 |
| dir | 定义文本的方向，包括ltr和rtl取值，分别表示从左向右和从右向左。 |

下面这些元素不拥有语言语义属性。

|  |  |
| --- | --- |
| frameset、frame、iframe | 网页框架结构 |
| br | 换行标识 |
| hr | 结构装饰线 |
| base | 网页基准信息 |
| param | 元素参数信息 |
| script | 网页的脚本 |

### HTML内容属性

内容属性定义元素包含内容的附加信息，这些信息对于元素来说具有重要补充作用，避免元素本身包含信息不全而被误解，包括5个属性。

|  |  |
| --- | --- |
| alt | 定义元素的替换文本 |
| title | 定义元素的提示文本 |
| longdesc | 定义元素包含内容的大段描述信息 |
| cite | 定义元素包含内容的引用信息 |
| datetime | 定义元素包含内容的日期和时间 |

alt和title是两个常用的属性，分别定义元素的替换文本和提示文本。

<a href=”URL” title=”提示文本”>超链接</a>

<img src=”URL” alt=”替换文本” title=”提示文本” />

替换文本并不是用来提示的，不是为图像提供额外说明信息的。相反，title属性才负责为元素提供额外说明信息。

当图像无法显示时，必须准备替换的文本来替换无法显示的图像，因此alt属性只能用在img、area和input元素中。对于input元素，alt属性用来替换提交按钮的图片。

<input type=”image” src=”URL” alt=”替换文本” />

## HTML 标题

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

<h1> 定义最大的标题。 <h6> 定义最小的标题。

【示例 2】使用HTML标题标签

|  |
| --- |
| <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> |

**注释:** 浏览器会自动地在标题的前后添加空行。

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 段落

### HTML段落

HTML 可以将文档分割为若干段落,HTML 段落是通过标签 <p> 来定义的。

【示例 2.5-1】使用HTML段落标签

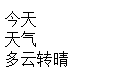
|  |
| --- |
| <p>春眠不觉晓，</p>  <p>处处闻啼鸟。</p>  <p>夜来风雨声，</p>  <p>花落知多少。</p> |

**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）

### HTML 换行

如果希望在不产生一个新段落的情况下进行换行（新行），请使用 <br /> 标签。

【示例 2.5-2】使用HTML换行标签



|  |
| --- |
| <p>  今天<br/>天气<br/>多云转晴  </p> |

<br /> 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 水平线

<hr> 标签在 HTML hr 元素可用于分隔内容。

【示例 2.5-3】在页面中创建水平线

|  |
| --- |
| <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> |

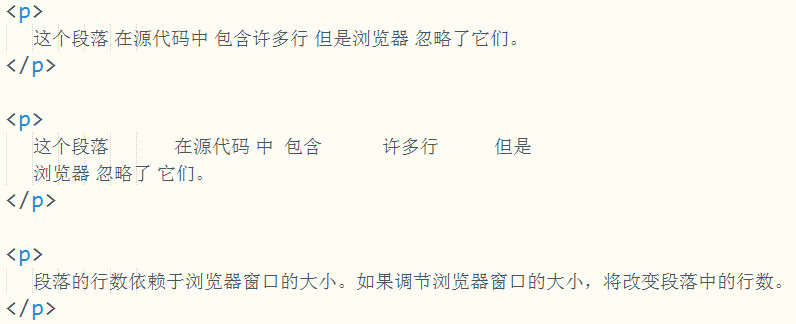
### HTML 输出

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，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。

【示例 2.5-4】如何在页面中插入多个空格



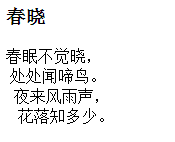
使用特殊符号&nbsp;可以解决上述问题。

**常见特殊符号**

|  |  |
| --- | --- |
| **特殊符号** | **命名实体** |
| **空格** | **&nbsp;** |
| © | **&copy;** |
| **>** | **&gt;** |
| **<** | **&lt;** |

注意：HTML5中的“>”和“<”可以直接输入

【课堂练习 2.5-1】使用HTML完成下图效果。



## HTML 文本格式化

**HTML格式化标签**

TML 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。

**HTML文本格式化标签**

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

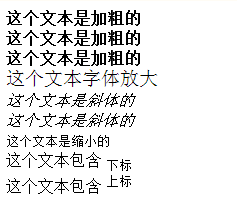
**HTML“计算机输出”标签**

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

**HTML 引文, 引用, 及标签定义**

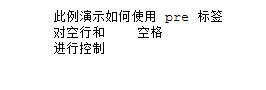
|  |  |
| --- | --- |
| **标签** | **描述** |
| [<abbr>](http://www.runoob.com/tags/tag-abbr.html) | 定义缩写 |
| [<address>](http://www.runoob.com/tags/tag-address.html) | 定义地址 |
| [<bdo>](http://www.runoob.com/tags/tag-bdo.html) | 定义文字方向 |
| [<blockquote>](http://www.runoob.com/tags/tag-blockquote.html) |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

【示例 2.6-1】文本格式化。



|  |
| --- |
| <b>这个文本是加粗的</b><br />  <strong>这个文本是加粗的</strong><br />  <big>这个文本字体放大</big><br />  <em>这个文本是斜体的</em><br />  <i>这个文本是斜体的</i><br />  <small>这个文本是缩小的</small><br /> 这个文本包含  <sub>下标</sub><br /> 这个文本包含  <sup>上标</sup> |

【示例 2.6-2】预格式文本。

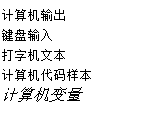


|  |
| --- |
| <pre>  此例演示如何使用 pre 标签  对空行和 空683C  进行控制  </pre> |

pre 元素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**。而文本也会呈现为**等宽字体**。

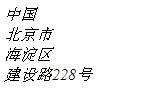
可以导致段落断开的标签（例如标题、<p> 和 <address> 标签）**绝不能**包含在 <pre> 所定义的块里。尽管有些浏览器会把段落结束标签解释为简单地换行，但是这种行为在所有浏览器上并不都是一样的。

【示例 2.6-3】计算机输出。



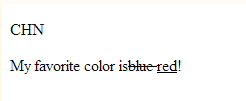
|  |
| --- |
| <code>计算机输出</code><br />  <kbd>键盘输入</kbd><br>  <tt>打字机文本</tt><br />  <samp>计算机代码样本</samp><br/>  <var>计算机变量</var> |

【示例 2.6-4】输出地址。



|  |
| --- |
| <address>  北京市<br/>  海淀区<br/>  建设路228号  </address> |

【示例 2.6-5】缩写和删除、插入文字。



|  |
| --- |
| <p>  <!--缩写-->  <abbr title="China">CHN</abbr>  </p>  <p>My favorite color is  <!--删除文字-->  <del>blue </del>  <!--插入文字-->  <ins>red</ins>!  </p>` |

## HTML 链接

### HTML链接介绍

HTML 使用超级链接与网络上的另一个文档相连。几乎可以在所有的网页中找到链接。点击链接可以从一张页面跳转到另一张页面。

<a href="index.html">本文本</a> 是一个指向本网站中的一个页面的链接。</p>

<p><a href="//www.baidu.com/">本文本</a> 是一个指向万维网上的页面的链接。</p>

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

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。

【示例 2.7-1】使用HTML链接访问百度首页

|  |
| --- |
| <a href="http://www.baidu.com">访问百度</a> |

**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。

【示例 2.7-2】使用HTML创建图片链接



|  |
| --- |
| <p>  创建图片链接:<br>  <a href="http://www.baidu.com">  <img src=img/logo.jpg alt="百度"/>  </a>  </p> |

**注意：请始终将正斜杠添加到子文件夹。**

假如这样书写链接：href="http://www.baidu.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="http://www.baidu.com/html/"。

### Target属性

使用 target 属性，你可以定义被链接的文档在何处显示。

**定义和用法**

<a> 标签的 target 属性规定在何处打开链接文档。

如果在一个 <a> 标签内包含一个 target 属性，浏览器将会载入和显示用这个标签的 href 属性命名的、名称与这个目标吻合的框架或者窗口中的文档。如果这个指定名称或 id 的框架或者窗口不存在，浏览器将打开一个新的窗口，给这个窗口一个指定的标记，然后将新的文档载入那个窗口。从此以后，超链接文档就可以指向这个新的窗口。

下面的这行会在新窗口打开文档：

<a href=“http://www.baidu.com/” target="\_blank">

访问百度

</a>

**特殊的目标**

有 4 个保留的目标名称用作特殊的文档重定向操作：

* **\_blank**

浏览器总在一个新打开、未命名的窗口中载入目标文档。

* **\_self**

这个目标的值对所有没有指定目标的 <a> 标签是默认目标，它使得目标文档载入并显示在相同的框架或者窗口中作为源文档。这个目标是多余且不必要的，除非和文档标题 <base> 标签中的 target 属性一起使用。

* **\_parent**

这个目标使得文档载入父窗口或者包含来超链接引用的框架的框架集。如果这个引用是在窗口或者在顶级框架中，那么它与目标 \_self 等效。

* **\_top**

这个目标使得文档载入包含这个超链接的窗口，用 \_top 目标将会清除所有被包含的框架并将文档载入整个浏览器窗口。

**提示：**这些 target 的所有 4 个值都以下划线开始。任何其他用一个下划线作为开头的窗口或者目标都会被浏览器忽略，因此，不要将下划线作为文档中定义的任何框架 name 或 id 的第一个字符。

**属性值**

|  |  |
| --- | --- |
| **值** | **描述** |
| \_blank | 在新窗口中打开被链接文档。 |
| \_self | 默认。在相同的框架中打开被链接文档。 |
| \_parent | 在父框架集中打开被链接文档。 |
| \_top | 在整个窗口中打开被链接文档。 |
| framename | 在指定的框架中打开被链接文档。 |

### ID属性

id属性可用于创建在一个HTML文档书签标记。

**提示:** 书签是不以任何特殊的方式显示，在HTML文档中是不显示的，所以对于读者来说是隐藏的。

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>

【示例 2.7-3】在当前页面链接到指定位置

|  |
| --- |
| <p>  <a href="#C3">查看章节3</a>  <a href="#C8">查看章节8</a>  </p>  <h2>章节1</h2>  <p>这边显示该章节的内容</p>  <h2>章节2</h2>  <p>这边显示该章节的内容</p>  <h2><a id="C3">章节3</a></h2>  <p>这边显示该章节的内容</p>  <h2>章节4</h2>  <p>这边显示该章节的内容</p>  <h2>章节5</h2>  <p>这边显示该章节的内容</p>  <h2>章节6</h2>  <p>这边显示该章节的内容</p>  <h2>章节7</h2>  <p>这边显示该章节的内容</p>  <h2><a id="C8">章节8</a></h2>  <p>这边显示该章节的内容</p>  <h2>章节 9</h2>  <p>这边显示该章节的内容……</p> |

同样使用name属性也可以实现上述效果。

### Title属性

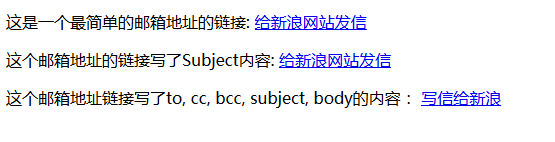
使用 title 属性，可以让鼠标悬停在超链接上的时候，显示该超链接的文字注释。例:

<a href="http://www.baidu.com/" title="最好用的搜索引擎">百度</a>

### 链接到email地址

在网站中，你经常会看到“联系我们”的链接，一点击这个链接，就会触发你的邮件客户端，比如Outlook Express，然后显示一个新建mail的窗口。用<a>可以实现这样的功能。

【示例 2.7-4】链接到email地址



|  |
| --- |
| <p>  这是一个最简单的邮箱地址的链接:  <a href="mailto:info@sina.com">给新浪网站发信</a>  </p>  <p>  这个邮箱地址的链接写了Subject内容:  <a href="mailto:info@sina.com?subject=Hello">给新浪网站发信</a>  </p>  <p>  这个邮箱地址链接写了to, cc, bcc, subject, body的内容：  <a href="mailto:info@sina.com?cc=webmaster@vip.sina.com&bcc=media@sina.com&subject=I%20like%20your%20site&body=真是个好站点！">写信给新浪</a>  </p> |

## HTML 头部

<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>, and <base>。

### HTML<title>元素

title> 标签定义了不同文档的标题。

<title> 元素:

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

### 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：

<head>

<base href="http://www.baidu.com/" target="\_blank">

</head>

【示例 2.8-1】使用 <base> 定义页面中所有链接默认的链接目标地址

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>HTML base元素</title>  <base href="http://news.baidu.com/" target="\_blank">  </head>  <body>  <a href="sports">百度体育新闻</a>  </body>  </html> |

### 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：

|  |
| --- |
| <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：

|  |
| --- |
| <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。

【示例 2.8-5】使用<meta>为搜索引擎定义关键词、描述内容、作者、定时刷新页面。

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <!--为搜索引擎定义关键词:-->  <meta name="keywords" content="HTML,CSS,JavaScript"/>  <!--为网页定义描述内容:-->  <meta name="description" content="学习前端知识">  <!--定义网页作者-->  <meta name="author" content="xiaobian"/>  <!--每5秒当前页面:-->  <meta http-equiv="refresh" content="5"/>  <title>使用meta为搜索引擎定义关键词、描述内容、作者、定时刷新页面。</title>  </head>  <body>    </body>  </html> |

### HTML<script>元素

<script>标签用于加载脚本文件，如： JavaScript。

HTML头部标签整理如表：



## HTML 图像

### 图像标签<img>和源属性(src)

在 HTML 中，图像由<img> 标签定义。

<img> 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”，来源。源属性的值是图像的 URL 地址。alt指“alternative”，替代内容。

定义图像的语法:

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

URL 指存储图像的位置。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。

### 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。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。

### 设置图像的高度和宽度

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:

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

**提示:**指定图像的高度和宽度的一个很好的习惯。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。

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。

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。

### Align属性

用align属性，可以改变图片的垂直(居上、居中、居下)对齐方式。

【示例 2.9-1】 使用align属性排列图片

|  |
| --- |
| <p><img src="img/smiley.gif" alt="笑脸" width="30" height="30" align="top">对齐方式:top</p>  <p><img src="img/smiley.gif" alt="笑脸" width="30" height="30" align="middle">对齐方式:middle</p>  <p><img src="img/smiley.gif" alt="笑脸" width="30" height="30" align="bottom">对齐方式:bottom</p>  <p><b>注意:</b>在HTML 4中 align 属性已废弃，HTML5 已不支持该属性，可以使用 CSS 代替。</p> |

【示例 2.9-2】 使用CSS “float”属性排列图片

|  |
| --- |
| <p>  <img src="img/smiley.gif" alt="笑脸" style="float:left" width="32" height="32"> 一个带图片的段落，图片浮动在这个文本的左边。  </p>  <p>  <img src="img/smiley.gif" alt="笑脸" style="float:right" width="32" height="32"> 一个带图片的段落，图片浮动在这个文本的右边。  </p>  <p><b>注意:</b> 在这里我们使用了 CSS "float" 属性，在HTML 4中 align 属性已废弃，HTML5 已不支持该属性，可以使用 CSS 代替。</p> |

### 图片链接

【示例 2.9-3】 将一个图片作为一个超链接

这个示例演示了如何将一个图片作为一个超链接，即点击一个图片，可以连接到其它文件。

|  |
| --- |
| <a href="http:www.baidu.com/">  <img src="img/smiley.gif" alt="百度" width="32" height="32" />  </a> |

## HTML区块

### HTML块级元素

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>

### HTML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>

### 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 <span>元素

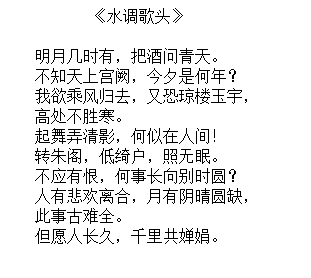
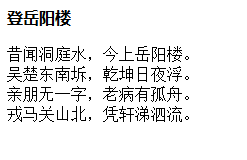
HTML <span> 元素是内联元素，可用作文本的容器

<span> 元素也没有特定的含义。

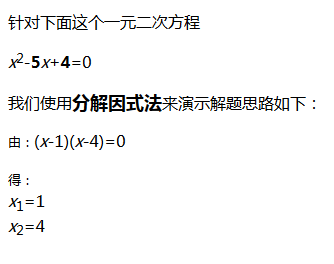
当与 CSS 一同使用时，<span> 元素可用于为部分文本设置样式属性。

## 本章作业

1.使用HTML完成下图效果。



2.使用HTML格式化标签完成下图效果。



3.使用HTML设置一个图片超链接。(图片可自由选择)效果如图：



* + 其中链接网址为: <http://china.nba.com/>
  + 链接目标在新窗口中打开
  + 设置图片的相应宽度和高度
  + 定义标签的替换文本
  + 定义标签的提示文本

# HTML表格

## 章节目标

* 了解表格结构
* 掌握创建表格、设置表格属性

## 表格

和常用的excel画表格一样，你可以使用HTML创建表格。

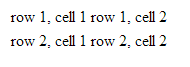
表格由 <table> 标签来定义。每个表格均有若干行（由 <tr> 标签定义），每行被分割为若干单元格（由 <td> 标签定义）。字母 td 指表格数据（table data），即数据单元格的内容。数据单元格可以包含文本、图片、列表、段落、表单、水平线、表格等等。

基本的表格语法如下：

【示例 3-1】 基本表格

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <meta charset="UTF-8">  <title>基本表格</title>  </head>  <body>  <table ><!--表格标签-->  <tr><!--行标签-->  <td>row 1, cell 1</td><!--单元格标签-->  <td>row 1, cell 2</td>  </tr>  <tr>  <td>row 2, cell 1</td>  <td>row 2, cell 2</td>  </tr>  </table>    </body>  </html> |

在浏览器中显示的结果如下：



注意：通常表格<table></table>包括行 <tr></tr>行里面包括单元格 <td></td>。内容一定要放入最底层的标签里面

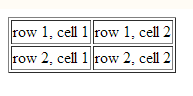
## Border属性

在缺省情况下，如果不设置表格的边界，表格是不显示边界的。

【示例 3-2】加边框的表格

|  |
| --- |
| <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> |

在浏览器中显示的结果如下：



## 表格表头

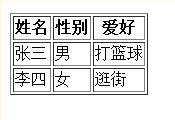
表格的表头使用 <th> 标签进行定义。

大多数浏览器会把表头显示为粗体居中的文本：

【示例 3-3】带表头的表格

|  |
| --- |
| <table border="1">  <th>姓名</th>  <th>性别</th>  <th>爱好</th>  <tr>  <td>张三</td>  <td>男</td>  <td>打篮球</td>  </tr>  <tr>  <td>李四</td>  <td>女</td>  <td>逛街</td>  </tr>  </table> |

在浏览器中运行结果如下:



## 空的单元格

如果表格的单元格<td></td>之间没有内容，那么这个单元格的边界是不会被显示出来的，尽管整个表格已设置边界值。要显示这个单元格的边界，可以插入一个&nbsp;空格符。

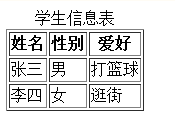
## 带标题的表格

表格的标题使用 <caption> 标签进行定义。

【示例 3-4】如何用<caption>在一个表格上加上标题。

|  |
| --- |
| <table border="1">  <caption>学生信息表</caption>  <th>姓名</th>  <th>性别</th>  <th>爱好</th>  <tr>  <td>张三</td>  <td>男</td>  <td>打篮球</td>  </tr>  <tr>  <td>李四</td>  <td>女</td>  <td>逛街</td>  </tr>  </table> |

在浏览器中运行结果如下：



## 合并单元格

HTML中单元格可以横向或者纵向合并。

横向合并，通俗的说就是左右合并，通过<th>或<td>标签的colspan属性，例如colspan="2"，意思就是横向合并2个单元格。纵向合并，就是说的上下合并。

纵向合并是通过<th>或<td>标签的rowspan属性完成的，比如rowspan="2"，就是纵向合并2行。

【示例 3-5】横向和纵向合并单元格

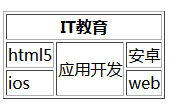
效果如下图所示：



具体代码如下：

|  |
| --- |
| <table border="1" >  <tr>  <th colspan="3">商品类目</th>  </tr>  <tr>  <td rowspan="2">虚拟</td>  <td>移动</td>  <td>联通</td>  </tr>  <tr>  <td>充值卡</td>  <td>彩票</td>  </tr>  <tr>  <td rowspan="2">服装</td>  <td>T恤</td>  <td>衬衫</td>  </tr>  <tr>  <td>西装</td>  <td>Polo衫</td>  </tr>  </table> |

【课堂练习】使用rowspan、colspan完成下图效果。

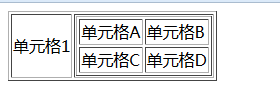


## 表格嵌套

【示例 3-6】嵌套表格

|  |
| --- |
| <table border="1">  <tr>  <td>  单元格1  </td>  <td>  <table border="1">  <tr>  <td>单元格A</td>  <td>单元格B</td>  </tr>  <tr>  <td>单元格C</td>  <td>单元格D</td>  </tr>  </table>  </td>  </tr>  </table> |

在浏览器中运行结果如下：



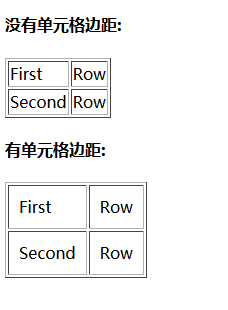
## 单元格边距(Cell padding)

cellpadding属性用来设置单元格内容与单元格边界之间的距离。

【示例 3-7】使用cellpadding设置单元格边距

|  |
| --- |
| <h4>没有单元格边距:</h4>  <table border="1">  <tr>  <td>First</td>  <td>Row</td>  </tr>  <tr>  <td>Second</td>  <td>Row</td>  </tr>  </table>  <h4>有单元格边距:</h4>  <table border="1" cellpadding="10">  <tr>  <td>First</td>  <td>Row</td>  </tr>  <tr>  <td>Second</td>  <td>Row</td>  </tr>  </table> |

在浏览器中运行结果如下:



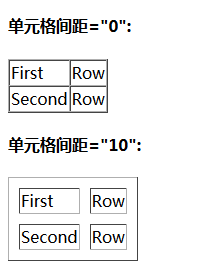
## 单元格间距(Cell spacing)

cellspacing这个属性设置单元格之间的距离。

【示例3-8】使用cellspacing设置单元格间距

|  |
| --- |
| <h4>单元格间距="0":</h4>  <table border="1" cellspacing="0">  <tr>  <td>First</td>  <td>Row</td>  </tr>  <tr>  <td>Second</td>  <td>Row</td>  </tr>  </table>  <h4>单元格间距="10":</h4>  <table border="1" cellspacing="10">  <tr>  <td>First</td>  <td>Row</td>  </tr>  <tr>  <td>Second</td>  <td>Row</td>  </tr>  </table> |

在浏览器中运行结果如下:



## 属性设置

常用的表格<table>属性包括以下：

* 表格的边框border=”数值” 默认是0
* 表格的宽度 width=”数值”
* 表格的高度height=”数值”（w3c推荐不建议使用高度）
* 表格的居中方式 align=”水平对齐方式” 取值 left center right
* 表格中的内容和单元格之间的距离 cellpadding=”数值” 默认数值是2像素
* 表格中单元格和单元格之间的距离 cellspacing=”数值” 默认数值是2像素
* 表格的背景颜色 bgcolor=”颜色值” 例如 bgcolor=”red”
* 表格的背景图片 background=”图片的地址”（背景图片不能含有中文）---修饰的作用
* 表格的合并边框线 rules=”all” (w3c不推荐使用，css样式来实现)
* 表格的边框颜色 bordercolor=”颜色值”例如 bordercolor=”#ff0000”(不建议使用)

常用的单元格的属性如下：

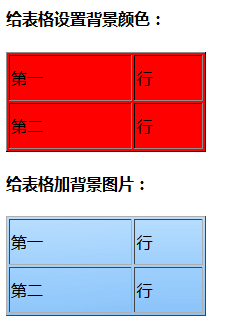
* Bgcolor 背景颜色
* Background 背景图片 例如 background=”images/XX.jpg”
* Width 单元格的宽度 例如 width=”300”
* Height 单元格的高度 例如height=”40”
* Align 内容在单元格中的水平对齐方式 align=”left/center/right”
* Valign 内容在单元格中垂直对齐方式 valign =”top/middle/bottom”;

下面用示例来演示表格和单元格属性。

**1.设置表格的背景颜色和背景图片**

在表格中使用bgcolor属性设置表格的背景颜色，使用background属性为表格添加背景图片。

【示例 3-9】设置表格的背景颜色和背景图片。



|  |
| --- |
| <h4>给表格设置背景颜色：</h4>  <table width="200px" height="100px" border="1" bgcolor="red">  <tr>  <td>第一</td>  <td>行</td>  </tr>  <tr>  <td>第二</td>  <td>行</td>  </tr>  </table>  <h4>给表格加背景图片：</h4>  <table border="1" width="200px" height="100px" background="img/demo.jpg">  <tr>  <td>第一</td>  <td>行</td>  </tr>  <tr>  <td>第二</td>  <td>行</td>  </tr>  </table> |

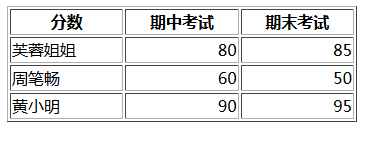
**提示：**设置单元格的背景颜色和背景图片和设置表格使用的属性一样。

**2.单元格内容的对齐方式**

示例教你如何用align属性设置单元格的对齐方式。

【示例 3-10】使用align属性设置单元格对齐方式。

效果如下图所示:



代码如下所示:

|  |
| --- |
| <table width="350" border="1">  <tr>  <th align="center">分数</th>  <th align="center">期中考试</th>  <th align="center">期末考试</th>  </tr>  <tr>  <td align="left">芙蓉姐姐</td>  <td align="right">80</td>  <td align="right">85</td>  </tr>  <tr>  <td align="left">周笔畅</td>  <td align="right">60</td>  <td align="right">50</td>  </tr>  <tr>  <td align="left">黄小明</td>  <td align="right">90</td>  <td align="right">95</td>  </tr>  </table> |

## 本章作业

使用HTML表格制作一份个人简历。

# HTML 列表

## 章节目标

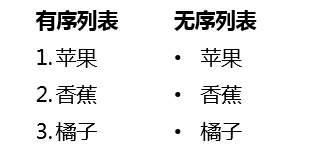
* 了解表格结构
* 掌握创建表格、设置表格属性

## 列表介绍

HTML有三种列表形式：有序列表(Ordered List)；无序列表(Unordered List)；定义列表(Definition List)。

HTML的列表元素是一个由列表标签封闭的结构，包含的列表项由<li></li>组成。 无序列表顾名思义，无序列表就是列表结构中的列表项没有先后顺序的列表形式。大部分网页应用中的列表均采用无序列表，其列表标签采用<ul></ul>，有序列表顾名思义，有序列表就是列表结构中的列表项有先后顺序的列表形式，从上到下可以有各种不同的序列编号，如1、2、3或a、b、c等。

如图所示：



## 有序列表

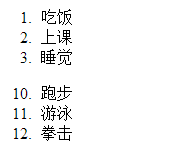
有序列表中,每个列表项前有数字，表示顺序。

有序列表由<ol>开始，每个列表项由<li>开始。

【示例 4-1】有序列表。

|  |
| --- |
| <ol>  <li>吃饭</li>  <li>上课</li>  <li>睡觉</li>  </ol>  <ol start="10">  <li>跑步</li>  <li>游泳</li>  <li>拳击</li>  </ol> |

在浏览器中查看结果如下：



**提示：**其中start 属性规定有序列表的开始点。

## 无序列表

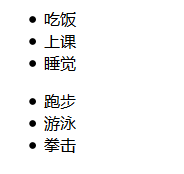
无序列表不用数字标记每个列表项，而采用一个符号标志每个列表项，比如圆黑点。

无序列表由<ul>开始，每个列表项由<li>开始。

【示例 4-2】无序列表

|  |
| --- |
| <ul>  <li>吃饭</li>  <li>上课</li>  <li>睡觉</li>  </ul>  <ul>  <li>跑步</li>  <li>游泳</li>  <li>拳击</li>  </ul> |

在浏览器中查看结果如下：



## 自定义列表

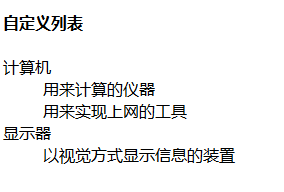
自定义列表不仅仅是一列项目，而是项目及其注释的组合。

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。

【示例 4-3】自定义列表

|  |
| --- |
| <h4>自定义列表</h4>  <dt>计算机</dt>  <dd>用来计算的仪器</dd>  <dd>用来实现上网的工具</dd>  <dt>显示器</dt>  <dd>以视觉方式显示信息的装置</dd> |

在浏览器中查看结果如下:



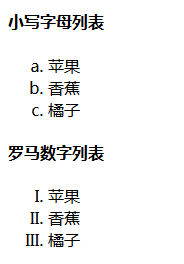
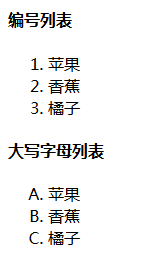
## 不同类型的有序列表

使用Type属性可以设置有序列表前的符号。

【示例 4-4】 设置不同类型的有序列表

|  |
| --- |
| <h4>编号列表</h4>  <ol>  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ol>  <h4>大写字母列表</h4>  <ol type="A">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ol>  <h4>小写字母列表</h4>  <ol type="a">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ol>  <h4>罗马数字列表</h4>  <ol type="I">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ol> |

在浏览器中运行结果如下：

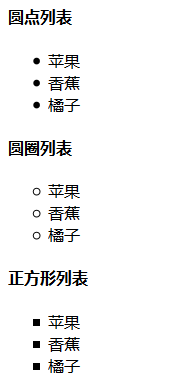


## 不同类型的无序列表

在 HTML 4中 ul 属性已废弃，HTML5 已不支持该属性，因此我们使用 CSS 代替来定义不同类型的无序列表如下：

|  |
| --- |
| <h4>圆点列表</h4>  <ul style="list-style-type: disc;">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ul>  <h4>圆圈列表</h4>  <ul style="list-style-type: circle;">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ul>  <h4>正方形列表</h4>  <ul style="list-style-type:square;">  <li>苹果</li>  <li>香蕉</li>  <li>橘子</li>  </ul> |

在浏览器中运行结果如下：



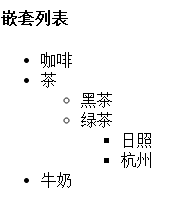
## 嵌套列表

列表里可以含有子列表。通常用这种嵌套的列表，反映层次较多的内容。

【示例 4-6】嵌套列表。

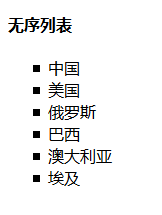
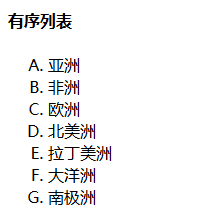
|  |
| --- |
| <h4>嵌套列表</h4>  <ul>  <li>咖啡</li>  <li>茶  <ul>  <li>黑茶</li>  <li>绿茶  <ul>  <li>日照</li>  <li>杭州</li>  </ul>  </li>  </ul>  </li>  <li>牛奶</li>  </ul> |

在浏览器中运行结果如下：

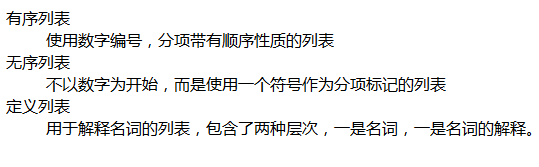


## 本章作业

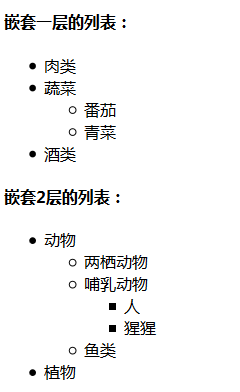
1.使用有序列表和无序列表实现下图效果：



2.使用自定义列表实现下图效果：



3.使用嵌套列表实现下图效果：



# HTML 框架

## 章节目标

* 了解HTML框架
* 掌握如何运用框架

## HTML框架介绍

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。常常我们使用在一个网页中某局域显示另外网页，即可使用iframe标签。

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%").

举例:

<iframe src="demo.html" width="200" height="200"></iframe>

## 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:

举例:

<iframe src="demo.html" frameborder="0"></iframe>

## 使用iframe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.

【示例 1】使用Ifram显示目标链接页面。

|  |
| --- |
| <iframe src="Demo.html" height="300px" width="500px" name="iframe\_a"></iframe>  <p><a href="http://www.baidu.com/" target="iframe\_a">www.baidu.com</a></p> |

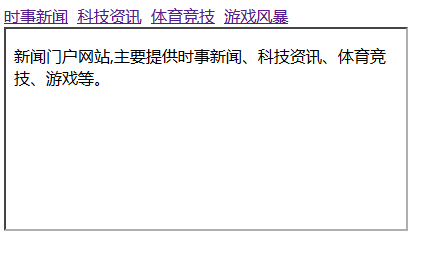
在浏览器中运行如下图：



注意： 因为 a 标签的 target 属性是名为 iframe\_a 的 iframe 框架，所以在点击链接时页面会显示在 iframe框架中。

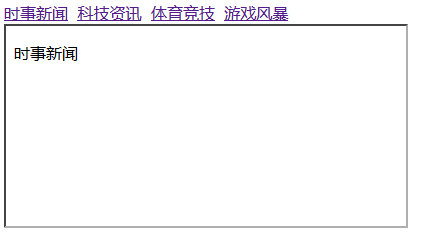
## 本章作业

使用iframe完成如下图效果：



分别点击时事新闻、科技资讯、体育竞技、游戏风暴超链接，分别显示对应的信息。

如：



# HTML 表单

## 章节目标

* 了解表单的结构和属性
* 掌握表单的具体应用

## 表单的结构

<form> 元素定义 HTML 表单：

语法:

|  |
| --- |
| <form 属性=”属性值”>  控件  </form> |

## 表单的属性

* **Name属性**

表单的识别名称。一个页面有多处表单，通过表单的识别名称来辨认页面提交的是哪个表单的数据。例如：

|  |
| --- |
| <form name=” Bill”> </form>  <form name=” game”> </form> |

* **Action属性**

处理表单数据程序的地址。例如：

|  |
| --- |
| <form action="index.html"> </form> |

* **Method属性**

把表单中的数据提交给服务器端的方法, Method最常用的两个取值是get和post，即method=”get/post”

|  |
| --- |
| <form action="index.html" method="get">  <form action="index.html" method="post"> |

其中get方法 默认的一种传递数据的方法

* 通过地址来传递表单中的数据
* 不能传递敏感的数据，密码
* 不能传递大量的数据，每次只能传递1024字节
* 不能上传附件

另外post 方法，不是通过地址栏传递数据，数据传给文件处理程序

* 相对安全
* 可以传递海量的数据
* 能上传附件。

那么何时使用GETFF1F

如果表单提交是被动的（比如搜索引擎查询），并且没有敏感信息。

当您使用 GET 时，表单数据在页面地址栏中是可见的：

search.aspx?name=join&age=25

注释：GET 最适合少量数据的提交。浏览器会设定容量限制。

何时使用POST?

如果表单正在更新数据，或者包含敏感信息（例如密码）。

POST 的安全性更加，因为在页面地址栏中被提交的数据是不可见的。

## HTML表单元素

表单元素指的是不同类型的 input 元素、复选框、单选按钮、提交按钮等等。

表单最重要的作用是获取用户信息。这就需要在表单中加入表单项（控件），例如文本框、单选按钮等，常用的表单项如下表：

|  |  |
| --- | --- |
| **表单项** | **说明** |
| <input type=”text”> | 单行文本框 |
| <input type=”password”> | 密码文本框 |
| <input type=”submit”> | 提交按钮，将表单里的信息提交给表单里的action所指向的地址 |
| <input type=”image”> | 图片提交 |
| <input type=”reset”> | 重置按钮，重设表单内容 |
| <input type=”button”> | 普通按钮 |
| <input type=”hidden”> | 隐藏元素 |
| <input type=”radio”> | 单选按钮 |
| <input type=”checkhox”> | 复选框 |
| <input type=”file”> | 文件域 |
| <select>…</select> | 列表框 |
| <textarea>…</textarea> | 多行文本框 |

* **单行文本框**

单行文本框允许用户输入一些简短的单行信息，例如用户姓名、住址等。其基本语法如下所示：

|  |
| --- |
| <input type="text" name="名称" size="数值" value="预设内容" maxlength="数值" /> |

其中，

Name：设定文本框的名称，数据交互中会用到。

Size：设定此文本框显示的宽度。

Value：设定此文本框的预设内容。

Maxlength：设定此文本框输入的最大长度。

* **密码文本框**

密码文本框主要用于一些保密信息的输入，例如密码。其基本语法如下：

|  |
| --- |
| <input type="password" name="名称" size="数值" value="预设内容" maxlength="数值" /> |

密码文本框用法基本与单行文本框一样，其属性也类似于单行文本框，在此不予重复。

* **提交按钮**

通过提交按钮可以将表单里的信息提交给表单action所指向的文件地址。其基本语法如下所示：

|  |
| --- |
| <input type="submit" name="名称" value="预设内容" /> |

其中，

Name：设定提交按钮的名称。

Value：设定按钮上显示的文本。默认为“提交”。

* **图片提交**

“图片提交”作用和“提交按钮”相同，不同的是“图片提交”以一个图片作为表单的提交按钮，其中src属性表示图片的路径。其基本语法如下所示：

|  |
| --- |
| <input type="image" src="图片路径" name="名称" alt="替代文本" width="宽度" height="高度"/> |

其中,

Name:设定提交图片的名称。

Src：图像路径，一般使用相对路径

Alt：鼠标经过图像或者图像不显示时的替换文本。

Width：设定图像的宽度。

Height：设定图像的高度。

* **重置按钮**

重置按钮是表单中另外一个比较常用的按钮，其作用是重置用户填写的信息。其基本语法如下所示：

|  |
| --- |
| <input type="reset" name="名称" value="预设内容" /> |

其中，

Name:设定重置按钮的名称。

Value：设定按钮上显示的文本，默认为“重置”。

* **普通按钮**

表单中经常用到普通按钮，它没有默认的动作，有时需要利用JavaScript来做一些特殊的效果时使用。其基本语法如下所示：

|  |
| --- |
| <input type="button" name="名称" value="预设内容" /> |

除了按钮的显示文字外，可以在按钮上添加很多效果，特别是单击按钮后发生的事件等。

* **隐藏元素**

隐藏元素多用于在提交表单时向服务器传递一些不需要用户设定但程序必须的参数值。这在动态网页中的需求更加明显。基本语法如下：

|  |
| --- |
| <input type="hidden" name="名称" value="预设内容" /> |

隐藏元素一般位于<form></form>标签内，在表单提交时一同被发送给服务器端。

* **单选按钮**

单选按钮通常是给出一个选项供用户选择，一次只能从中选一个，应用单选按钮时要确定显示给用户的文字和不同选项的取值。其基本语法如下所示：

|  |
| --- |
| <input type="radio" name="名称" value="选项内容" checked="checked"/> |

其中,

Value:设定此单选按钮的选定值。

Checked：当该项默认被选中时设定，否则不设定。

* **复选框**

复选框通常是给出几个选项供用户选择，并且可以从中选择多个，使用复选框时也要确定显示给用户的文字和不同选项的取值。其基本语法如下所示：

|  |
| --- |
| <input type="checkbox" name="名称" value="选项内容" checked="checked"/> |

其中,

Value：设定此复选框的选定值。

Checked：当该项默认被选中时设定，否则不设定。

* **文件域**

文件域是用来填写文件路径，通过表单上传文件的地方。

|  |
| --- |
| <input type="file" name="名称" /> |

* **列表框**

列表框是表单中供用户选择的一个表单项，可以显示多个选项供选择，且用户能同时选择其中一个或多个。列表框中包含<option>标签。其基本语法如下：

|  |
| --- |
| <select name="名称" size="大小" multiple="multiple">  <option value="">给客户看的内容</option>  <option value="">给客户看的内容</option>  ……  </select> |

其中，

Name:设定下拉列表的名称。

Size：设定下拉列表显示选项的个数。

Multiple：设定此下拉列表可多选，如果为单选则省略该项。

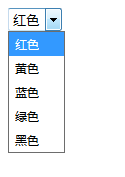
Value：<option>的属性，当选择该项时的值。

<select>中size属性指整个选项框内显示的选项个数，默认为1，当size值大于1时，列表框会改变形式，显示多个列表项。一般情况下，列表框都是用在单选的情况下，如果要使用多选的功能，只要加上multiple=”multiple”就可以。

【示例 6-1】用select元素添加颜色下拉列表:

|  |
| --- |
| <form>  <select name="color">  <option value="red">红色</option>  <option value="yellow">黄色</option>  <option value="blue">蓝色</option>  <option value="green">绿色</option>  <option value="black">黑色</option>  </select>  </form> |

在浏览器中运行效果如下：



<option> 元素定义待选择的选项。

列表通常会把首个选项显示为被选选项。

您能够通过添加 selected 属性来定义预定义选项。

* **多行文本框**

多行文本框用来输入较多的文字信息，常在新闻发布与论坛系统中用到。其基本语法如下所示：

|  |
| --- |
| <textarea name="名称" rows="文本框的显示行数" cols="文本框的显示列数"></textarea> |

其中，

Rows:文本框显示的行数。

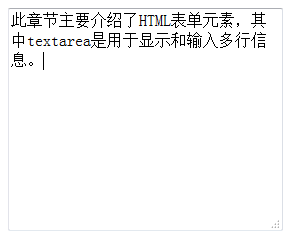
Cols：文本框显示的列数。

使用多行文本框主要是确定它的名称以及大小，当用户输入的文字超过显示容量时，多行文本框会自动产生滚动条。

【示例 6-2】使用<textarea> 元素。

|  |
| --- |
| <body>  <textarea name="remark" id="remarks" rows="10" cols="30">可以输入多行数据</textarea>  </body> |

在浏览器中运行结果如下:



## HTML Input属性

|  |  |
| --- | --- |
| **属性名** | **说明** |
| value | 输入字段的初始值。如:value="hello" |
| readonly | 输入字段为只读。readonly 属性不需要值。它等同于 readonly="readonly"。 |
| Disabled | 输入字段是禁用的。disabled 属性不需要值。它等同于 disabled="disabled"。 |
| size | size 属性规定输入字段的尺寸（以字符计）。如: size="40" |
| maxlength | 输入字段允许的最大长度。如: maxlength="10" |

## 表单应用

在实际应用中，经常会用到注册会员操作，下面我们就使用HTML表单完成会员注册的页面设计。

【示例 6-3】使用HTML表单完成会员注册页面。



源代码如下:

|  |
| --- |
|  |

## 本章作业

使用HTML表单完成下图效果：



# HTML5 简介

## 章节目标

* 了解什么是HTML5
* 了解HTML5 新特性

## 什么是HTML5

HTML5 是最新的 HTML 标准。

HTML5 是专门为承载丰富的 web 内容而设计的，并且无需额外插件。

HTML5 拥有新的语义、图形以及多媒体元素。

HTML5 提供的新元素和新的 API 简化了 web 应用程序的搭建。

HTML5 是跨平台的，被设计为在不同类型的硬件（PC、平板、手机、电视机等等）之上运行。

注释：在下面的章节中，您将学到如何“帮助”老版本的浏览器处理 HTML5。

## HTML5中新内容

HTML5 的新的文档类型（DOCTYPE）声明非常简单：

<!DOCTYPE html>

<meta charset="UTF-8">

HTML5 实例：

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

## HTML5新的属性语法

HTML5 标准允许 4 中不同的属性语法。

本例演示在 <input> 标签中使用的不同语法：

|  |  |
| --- | --- |
| **类型** | **示例** |
| Empty | <input type="text" value="John Doe" disabled> |
| Unquoted | <input type="text" value=John Doe> |
| Double-quoted· | <input type="text" value="John Doe"> |
| Single-quoted | <input type="text" value='John Doe'> |

在 HTML5 标准中，根据对属性的需求，可能会用到所有 4 种语法。

## HTML5新特性

HTML5的一些最有趣的新特性:

* 新的语义元素，比如 <header>, <footer>, <article>, and <section>。
* 新的表单控件，比如数字、日期、时间、日历和滑块。
* 强大的图像支持（借由 <canvas> 和 <svg>）
* 强大的多媒体支持（借由 <video> 和 <audio>）
* 强大的新 API，比如用本地存储取代 cookie。

新增语义元素如下所示：

|  |  |
| --- | --- |
| 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被删元素

以下 HTML 4.01 元素已从 HTML5 中删除：

* <acronym>
* <applet>
* <basefont>
* <big>
* <center>
* <dir>
* <font>
* <frame>
* <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。

### 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。

### 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 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：

<script>

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

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

ctx.fillStyle="#FF0000";

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

</script>

**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。

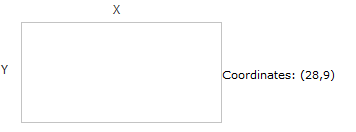
canvas 的左上角坐标为 (0,0)

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

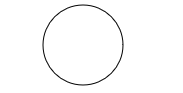


在canvas中绘制圆形, 我们将使用以下方法:

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

**实例**

使用 arc() 方法 绘制一个圆:



<script>

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

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

ctx.beginPath();

ctx.arc(95,50,40,0,2\*Math.PI);

</script>

## HTML5 表单元素

### HTML5 新的表单元素

TML5 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" name="browser">

<datalist id="browsers">

<option value="Internet Explorer">

<option value="Firefox">

<option value="Chrome">

<option value="Opera">

<option value="Safari">

</datalist>

### 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**

<form action="keygen.html" method="GET">

用户名:<input type="text" name="username">

加密:<keygen name="security">

<input type="submit">

</form>

### 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**

<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>

## HTML5 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

### autocomplete 属性

autocomplete 属性规定表单是否应该启用自动完成功能。

自动完成允许浏览器预测对字段的输入。当用户在字段开始键入时，浏览器基于之前键入过的值，应该显示出在字段中填写的选项。

注释：autocomplete 属性适用于 <form>，以及下面的 <input> 类型：text, search, url, telephone, email, password, datepickers, range 以及 color。

提示：在某些浏览器中，您可能需要手动启用自动完成功能。

<form action="" autocomplete="off">

用户名:<input type="text" name="username"><br>

密码：<input type="password"><br>

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

<input type="submit">

</form>

### autofocus 属性

autofocus 属性是一个 boolean 属性.

autofocus 属性规定在页面加载时，域自动地获得焦点。

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

### 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。

第一个提交按钮已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:

<input type="submit" formenctype="multipart/form-data">

### 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。

日期:<input type="date" name="day" max="2099-12-31"><br><br>

日期:<input type="date" name="day" min="2000-01-01"><br><br>

### 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:

上传多个文件:

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

### 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.

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）：

Code:<input type="text" name="country\_code" pattern="[A-Za-z]{3}" title="如:CHN">

### 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。

手机:<input type="text" name="tel" placeholder="请输入手机号">

### required 属性

equired 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）。

<form action="" method="POST">

<input type="text" required placeholder="请输入用户名"><br><br>

<input type="submit" value="提交">

</form>

## HTML5 语义元素

### 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 <span> - 无需考虑内容.

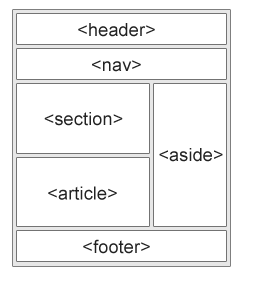
**语义**元素实例: <form>, <table>, and <img> - 清楚的定义了它的内容.

### 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>



### 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>俄罗斯世界杯</h1>

<p>耻辱！卫冕冠军德国0-2负韩国 首次小组赛出局</p>

</section>

<section>

<h1>习近平寄望青年建功新时代</h1>

<p>

青春，是人生最美的时光。青年，是国家的未来和民族的希望。

</p>

</section>

### article元素

<article> 标签定义独立的内容。

### 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</a>

<a href="#">CSS</a>

<a href="#">JavaScript</a>

<a href="#">JQuery</a>

<a href="#">BootStrap</a>

</nav>

### 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.

### 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.

<header>

<h1>慧与-济宁基地</h1>

<nav>

<a href="#">HTML</a>

<a href="#">CSS</a>

<a href="#">JavaScript</a>

<a href="#">JQuery</a>

<a href="#">BootStrap</a>

</nav>

</header>

### 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

<footer>

<p>慧与-国际软件人才及济宁产业基地</p>

</footer>

## 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。

### 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。

**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。

sessionStorage.username="zhangsan";

document.getElementById("result").innerHTML=sessionStorage.username;

实例解析：

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

移除 localStorage 中的 "lastname" :

localStorage.removeItem("username");

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);