HTML 4.01 为
<script> 定义了下列 6 个属性。

□ async ：可选。表示应该立即下载脚本，但不应妨碍页面中的其他操作，比如下载其他资源或
等待加载其他脚本。只对外部脚本文件有效。异步脚本

□ charset ：可选。表示通过 src 属性指定的代码的字符集。由于大多数浏览器会忽略它的值，
因此这个属性很少有人用。

□ defer ：可选。表示脚本可以延迟到文档完全被解析和显示之后再执行。只对外部脚本文件有
效。IE7 及更早版本对嵌入脚本也支持这个属性。

2.1.2 延迟脚本

HTML 4.01 为<script>标签定义了 defer 属性。这个属性的用途是表明脚本在执行时不会影响页面的构造。也就是说，脚本会被延迟到整个页面都解析完毕后再运行。因此，在<script>元素中设置 defer 属性，相当于告诉浏览器立即下载，但延迟执行。

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example HTML Page</title>
    <script type="text/javascript" defer="defer" src="example1.js"></script>
    <script type="text/javascript" defer="defer" src="example2.js"></script>
  </head>
  <body>
    <!-- 这里放内容 -->
  </body>
</html>
```

在这个例子中，虽然我们把<script>元素放在了文档的<head>元素中，但其中包含的脚本将延迟到浏览器遇到</html>标签后再执行。HTML5 规范要求脚本按照它们出现的先后顺序执行，因此第一个延迟脚本会先于第二个延迟脚本执行，而这两个脚本会先于 DOMContentLoaded 事件（详见第 13 章）执行。在现实当中，延迟脚本并不一定会按照顺序执行，也不一定会在 DOMContentLoaded 事件触发前执行，因此最好只包含一个延迟脚本。

前面提到过，defer 属性只适用于外部脚本文件。这一点在 HTML5 中已经明确规定，因此支持 HTML5 的实现会忽略给嵌入脚本设置的 defer 属性。IE4～IE7 还支持对嵌入脚本的 defer 属性，但 IE8 及之后版本则完全支持 HTML5 规定的行为。

IE4、Firefox 3.5、Safari 5 和 Chrome 是最早支持 defer 属性的浏览器。其他浏览器会忽略这个属
性，像平常一样处理脚本。为此，把延迟脚本放在页面底部仍然是最佳选择。

□ language ：已废弃。原来用于表示编写代码使用的脚本语言（如 JavaScript 、JavaScript1.2
或 VBScript ）。大多数浏览器会忽略这个属性，因此也没有必要再用了。

□ src ：可选。表示包含要执行代码的外部文件。

□ type ：可选。可以看成是 language 的替代属性；表示编写代码使用的脚本语言的内容类型.

考虑到约定俗成和最大限度的浏览器兼容性，目前 type 属性的值依旧还是 text/javascript 。不过，这个属性并不是必需的，如果没有指定这个属性，则其默认值仍为 text/javascript 。

按照惯例，外部 JavaScript 文件带有.js 扩展名。但这个扩展名不是必需的，因为浏览器不会检查包含 JavaScript 的文件的扩展名。这样一来，使用 JSP、PHP 或其他服务器端语言动态生成 JavaScript 代码也就成为了可能。但是，服务器通常还是需要看扩展名决定为响应应用哪种 MIME 类型。如果不使用.js 扩展名，请确保服务器能返回正确的 MIME 类型。

动态生成 JavaScript 代码

2.2 嵌入代码与外部文件

在 HTML 中嵌入 JavaScript 代码虽然没有问题，但一般认为最好的做法还是尽可能使用外部文件来包含 JavaScript 代码。不过，并不存在必须使用外部文件的硬性规定，但支持使用外部文件的人多会强调如下优点。

- **可维护性**：遍及不同 HTML 页面的 JavaScript 会造成维护问题。但把所有 JavaScript 文件都放在一个文件夹中，维护起来就轻松多了。而且开发人员因此也能够在不触及 HTML 标记的情况下，集中精力编辑 JavaScript 代码。
- **可缓存**：浏览器能够根据具体的设置缓存链接的所有外部 JavaScript 文件。也就是说，如果有两个页面都使用同一个文件，那么这个文件只需下载一次。因此，最终结果就是能够加快页面加载的速度。
- **适应未来**：通过外部文件来包含 JavaScript 无须使用前面提到 XHTML 或注释 hack。HTML 和 XHTML 包含外部文件的语法是相同的。

2.3 文档模式

IE5.5 引入了文档模式的概念，而这个概念是通过使用文档类型（doctype）切换实现的。

2.4 <noscript>元素

早期浏览器都面临一个特殊的问题，即当浏览器不支持 JavaScript 时如何让页面平稳地退化。对这个问题的最终解决方案就是创造一个<noscript>元素，用以在不支持 JavaScript 的浏览器中显示替代的内容。这个元素可以包含能够出现在文档<body>中的任何 HTML 元素——<script>元素除外。包含在<noscript>元素中的内容只有在下列情况下才会显示出来：

- 浏览器不支持脚本；
- 浏览器支持脚本，但脚本被禁用。

符合上述任何一个条件，浏览器都会显示<noscript>中的内容。而在除此之外的其他情况下，浏览器不会呈现<noscript>中的内容。

请看下面这个简单的例子：

```
<html>
  <head>
    <title>Example HTML Page</title>
    <script type="text/javascript" defer="defer" src="example1.js"></script>
    <script type="text/javascript" defer="defer" src="example2.js"></script>
  </head>
  <body>
    <noscript>
      <p>本页面需要浏览器支持（启用）JavaScript。
    </noscript>
  </body>
</html>
```