高职高专计算机类专业“十二五”规划教材

JavaScript语言与Ajax应用

主　编　董　宁
副主编　陈　丹　袁晓曦
主　审　曹　静

实例丰富，内容充实
涉及大量实例介绍，涵盖JavaScript语言的每个领域

讲解通俗，步骤详细
各实例均以通俗易懂的语言描述，并配实例源代码

由浅入深，逐步讲解
以JavaScript与Ajax应用为核心，层层展开，环环相套

体现新技术
紧跟JavaScript语言的发展，讲解Web应用开发的主流技术

资源全免费
为方便读者使用，可免费下载书中实例的源文件和电子教案

中国水利水电出版社
www.waterpub.com.cn

高职高专计算机类专业“十二五”规划教材

JavaScript 语言与 Ajax 应用

主　编　董　宁

副主编　陈　丹　袁晓曦

主　审　曹　静

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书系统介绍 JavaScript 语言与 Ajax 应用的相关技术，主要内容包括：JavaScript 基本概念与开发环境的选择、面向对象程序设计、文档对象模型、事件处理、浏览器对象模型、JavaScript 库、动画效果、Ajax 技术和表单验证等。本书逻辑严密，实例丰富，内容翔实，可操作性强。

本书可作为高职院校或大专院校相关专业的教材，也可作为 Web 应用前台开发人员的参考书，还可以作为各类计算机培训班的教材。

本书免费提供全部程序的源文件和电子教案，读者可以从中国水利水电出版社网站以及万水书苑下载，网址为：http://www.waterpub.com.cn/softdown/或 http://www.wsbookshow.com。

图书在版编目（CIP）数据

JavaScript语言与Ajax应用 / 董宁主编. -- 北京 : 中国水利水电出版社, 2011.7
高职高专计算机类专业“十二五”规划教材
ISBN 978-7-5084-8719-9

Ⅰ. ①J… Ⅱ. ①董… Ⅲ. ①JAVA语言－程序设计－高等职业教育－教材②计算机网络－程序设计－高等职业教育－教材 Ⅳ. ①TP312②TP393.09

中国版本图书馆CIP数据核字(2011)第115730号

策划编辑：杨庆川　责任编辑：宋俊娥　加工编辑：刘晶平　封面设计：李　佳

书　名	高职高专计算机类专业“十二五”规划教材 JavaScript 语言与 Ajax 应用
作　者	主　编　董　宁 副主编　陈　丹　袁晓曦 主　审　曹　静
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经　售	全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	北京泽宇印刷有限公司
规　格	184mm×260mm　16 开本　13.5 印张　330 千字
版　次	2011 年 7 月第 1 版　2011 年 7 月第 1 次印刷
印　数	0001—3000 册
定　价	24.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社营销中心负责调换

前　言

JavaScript 是一种运行在浏览器中的脚本语言。随着 Web 2.0 和 Ajax 成为主流，JavaScript 语言已经被推到了 Web 应用开发的中心位置，熟练掌握 JavaScript 语言是 Web 应用开发人员的必备技能。本书的目标是将它作为大学生学习 Web 应用开发的核心课程之一，学习该课程前需要掌握 HTML、CSS 和服务器端程序开发的相关内容。

本书不仅包含 JavaScript 语言与 Ajax 技术的各种概念和理论知识，而且对多种知识的综合运用进行了详细讲解。本书知识点系统连贯，逻辑性强；重点难点突出，利于组织教学；在内容安排上注意承上启下，由简到繁，循序渐进地讲述 JavaScript 语言，从基本概念到面向对象编程、从 JavaScript 库的使用到 Ajax 技术都进行了详细阐述，并进行了细致的实例讲解。

本书是作者在多年的教学实践和科学研究的基础上，参阅大量国内外相关教材后，几经修改而成的。主要特点如下：

（1）实例丰富，内容充实。本书中使用大量实例来介绍 JavaScript 语言，几乎涉及 JavaScript 语言的每一个领域。

（2）讲解通俗，步骤详细。本书中的每个示例都是以通俗易懂的语言描述，并配以示例源代码帮助读者更好地掌握 JavaScript 语言。

（3）由浅入深，逐步讲解。本书按照由浅入深的顺序，循序渐进地介绍了 JavaScript 语言与 Ajax 应用的相关知识。各个章节在编写的时候都是层层展开，环环相套的。

（4）内容紧跟 JavaScript 语言技术的发展。本书中介绍的 JavaScript 语言编程技术与 Ajax 技术都是目前 Web 应用开发中使用的主流技术。

（5）本书配有全部程序的源文件和电子教案。为方便读者使用，书中全部实例的源代码及电子教案均免费赠送给读者。

本书循序渐进地介绍了 JavaScript 语言开发相关的各方面知识，包括开发环境的选择、JavaScript 语法、面向对象程序设计、文档对象模型、事件处理、浏览器对象模型、JavaScript 库、动画效果、Ajax 技术和表单验证。同时本书还介绍了大量 JavaScript 的开发经验，对使用中的重点难点进行了专门的讲解。

本书由董宁担任主编，陈丹、袁晓曦担任副主编，曹静担任主审，谢日星、罗炜、刘洁、张宇、肖奎、李汉桥参加编写，董宁、陈丹统编全稿。

读者在阅读本书的过程中，如果觉得有疑问或不妥之处，请与编者联系，帮助我们改正提高，编者将不胜感激。

编　者

2011 年 5 月

目　　录

第 1 章　JavaScript 基础

JavaScript 是一种基于对象的脚本编程语言，从本章可以了解到 JavaScript 是如何产生的，以及从它产生到如今涵盖各种特性的具体实现方法，本章还介绍了 JavaScript 的具体应用及 JavaScript 脚本语言的开发环境。

- JavaScript 和客户端脚本编程的起源
- 在 Web 页面中使用 JavaScript 的方法
- 编写和调试 JavaScript 的几种常用工具

1.1　JavaScript 的历史与现状

1.1.1　JavaScript 的发展

当 Internet 的普及越来越广泛时，对于开发客户端脚本的需求也逐渐增大。此时，大部分 Internet 用户仅仅通过 28.8kbit/s 的 Modem 来连接到网络，即使这时网页已经不断地变得更大和更复杂。使用户痛苦的是，仅仅为了简单的表单有效性验证，就要与服务器进行多次的往返交互。设想一下，用户填写完一个表单，单击“提交”按钮，等待 30 秒后，看到的却是一条告诉你忘记了填写一个必要的字段信息。那时正处于技术革新最前沿的 Netscape，开始考虑开发一种客户端语言来处理简单的问题。

当时为 Netscape 工作的 Brendan Erich 开始着手为 1995 年发行的 Netscape Navigator 2.0 开发一个称之为 LiveScript 的脚本语言，目的是同时在浏览器和服务器上使用它。Netscape 与 Sun 公司联手完成了 LiveScript 的开发。就在 Netscape Navigator 2.0 即将正式发布前，Netscape 将 LiveScript 更名为 JavaScript，主要是为了利用 Java 这个 Internet 时髦词汇。Netscape 的这一决定也实现了当初的意图，JavaScript 从此变成了 Internet 的必备组件。

因为 JavaScript 1.0 的成功，Netscape 在 Navigator 3.0 中又发布了 JavaScript 1.1 版本。恰巧那个时候，微软决定进军浏览器市场，发布了 IE 3.0b 并搭载了一个 JavaScript 的克隆版，叫做 JScript（这样命名是为了避免与 Netscape 潜在的许可纠纷）。微软步入浏览器领域的这重要一步推动了 JavaScript 的进一步发展。在微软进入后，开发了 3 种不同的 JavaScript 版本，即 Netscape Navigator 3.0 中的 JavaScript、IE 中的 JScript 及 CEnvi 中的 ScriptEase。与其他编程语言不同的是，JavaScript 并没有一个标准来统一其语法或特性，而这 3 种不同的版本恰恰存在这个问题，随着用户担心的增加，这个语言标准化显然已经势在必行。

1997 年，JavaScript 1.1 作为一个草案提交给 ECMA（欧洲计算机制造商协会）。第 39 技术委员会（TC39）被委派来“标准化一个通用、跨平台、中立于厂商的脚本语言的语法和语义”。由来自 Netscape、Sun、微软、Borland 和其他一些对脚本编程感兴趣的公司的程序员组成的 TC39 锤炼出了 ECMA-262，该标准定义了称为 ECMAScript 的全新脚本语言。1998 年，该标准成为了国际 ISO 标准（ISO/IEC 16262）。这个标准至今仍然处于发展之中。2005 年 12 月，ECMA 发布了 ECMA-357 标准（ISO/IEC 22537），这一标准主要增加了对扩展标记语言 XML 的有效支持。从此，Web 浏览器就开始致力于（虽然有着不同成都的成功和失败）将 ECMAScript 作为 JavaScript 实现的基础。

1.1.2 JavaScript 在 HTML 中的作用

HTML 超文本标记语言可用来制作所需的 Web 网页，通过 HTML 中符号的描述就可以实现文字、表格、声音、图像、动画等多媒体信息的检索。然而单纯采用 HTML 技术存在一定的缺陷，那就是它只能提供一种静态的信息资源，缺乏动态效果。这里所说的动态效果分为两种：一种是客户端的动态效果，就是所看到的 Web 页面是活动的，可以处理各种事件，如鼠标移动时图片会有翻转效果等；另一种是客户端与服务器端的交互产生的动态效果。实现客户端的动态效果，JavaScript 无疑是一件适合的工具。JavaScript 的出现，使得信息和用户之间不仅只是一种显示和浏览的关系，而是实现了一种实时的、动态的、交互性的表达能力。从而基于 CGI 静态的 HTML 页面可提供动态实时信息，并对客户操作进行反应的 Web 页面所取代。JavaScript 脚本正是为满足这种需求而编写的语言。

JavaScript 是一种基于对象和事件驱动并具有安全性能的脚本编写语言，它采用小程序段的方式实现编程，像其他脚本语言一样，JavaScript 同样也是一种解释性语言，它提供了一个简易的开发过程。它的基本结构形式与 C、C++、Visual Basic、Delphi 十分类似。但它不像这些语言一样，需要先编译，而是在程序运行过程中被逐行地解释。在 HTML 基础上，使用 JavaScript 可以开发交互式 Web 网页，它是通过嵌入或调入标准 HTML 语言中实现的。JavaScript 与 HTML 标识结合在一起，实现在一个网页中链接多个对象，与网络客户交互作用，从而可以开发客户端的应用程序，其作用主要体现在以下几个方面。

（1）动态性。JavaScript 是动态的，它可以直接对用户或客户输入做出响应，无须经过 Web 服务程序。它对用户的响应是采用以事件驱动的方式进行的。所谓事件驱动，就是指在主页中执行了某种操作所产生的动作，称为“事件”。比如按下鼠标、移动窗口、选择菜单等都可以视为事件。当事件发生后，可能会引起相应的事件响应。

（2）跨平台。JavaScript 是依赖于浏览器本身，与操作环境无关，只要能运行浏览器的计算机，并支持 JavaScript 的浏览器就可以正确执行。

（3）相对安全性。JavaScript 是客户端脚本，通过浏览器解释执行。它不允许访问本地的硬盘，并且不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效地防止数据的丢失。

（4）节省客户端与服务器端的交互时间。随着 WWW 的迅速发展。有许多服务器提供的服务要与客户端进行交互，如确定用户的身份、服务的内容等，这项工作通常由 CGI/Perl 编写相应的接口程序与用户进行交互来完成。很显然，通过网络与用户的交互过程，一方面增大了网络的通信量，另一方面影响了服务器的服务性能。服务器为一个用户运行一个 CGI 时，需要一个进程为它服务，这就要占用服务器的资源（如 CPU 服务、内存耗费等），如果用户填

表出现错误，交互服务所占用的时间就会相应增加。被访问的热点主机与用户交互越多，服务器的性能影响就越大。而 JavaScript 是一种基于客户端浏览器的语言，用户在浏览中填表、验证的交互过程只是通过浏览器对调入 HTML 文档中的 JavaScript 源代码进行解释执行来完成的，即使是必须调用 CGI 的部分，浏览器只需将用户输入验证后的信息提交给远程的服务器，这便大大减少了服务器的开销。

1.1.3　Ajax

Ajax 即 Asynchronous JavaScript and XML（异步 JavaScript 和 XML），Ajax 并非缩写词，而是由 Jesse James Gaiiett 创造的名词，是指一种创建交互式网页应用的网页开发技术。Ajax 描述了把 JavaScript 和 Web 服务器组合起来的编程范型，JavaScript 是 Ajax 的核心技术之一，在 Ajax 技术架构中起着不可替代的作用。Ajax 是一种 Web 应用程序开发的手段，它采用客户端脚本与 Web 服务器交换数据，所以不必采用中断交互的完整页面刷新，就可以动态地更新 Web 页面。使用 Ajax 技术可以不必刷新整个页面，只需对页面的局部进行更新，而且还可以节省网络宽带，提高网页加载速度，从而缩短用户等待时间，改善用户的操作体验。

1.2　JavaScript 的运行

1.2.1　JavaScript 代码的装载与解析

当一个 HTML 页面被装载时，它会装载并解析这一过程中遇到的任何 JavaScript。Script 标签可以出现在文档的 head 中，也可以出现在 body 中。如果有指向外部 JavaScript 文件的链接，它会先装载该链接，再继续解析页面。嵌入第三方的脚本时，如果远程服务器因负担过重而无法及时返回文件，就有可能导致页面的装载时间显著变长。

代码解析是浏览器取得代码并将其转化成可执行代码的过程。这个过程的第一步是检查代码的语法是否正确，如果不正确，过程就会立即失败。如果一个包含语法错误的函数被运行，就很可能得到一条错误消息，显示函数还没定义。当浏览器确认代码合法之后，它会解析 script 块中所有的变量和函数。如果要调用的函数来自其他 script 块或者其他文件，就要确保它在当前 script 元素之前装载。

1.2.2　在 HTML 页面中嵌入 JavaScript

JavaScript 的脚本包含在 HTML 中，已成为 HTML 文档的一部分，与 HTML 标识相结合，构成动态的、能够交互的网页。

1. 引入 JavaScript 脚本代码到 HTML 文档中

如果要把一段 JavaScript 插入 HTML 页面，就需要使用 script 标签（同时使用 type 属性来定义脚本语言）。这样，<script type="text/javascript">和</script>就可以告诉浏览器 JavaScript 从何处开始，到何处结束。浏览器载入嵌有 JavaScript 脚本的 HTML 文档时，能自动识别 JavaScript 脚本代码起始标记和结束标记，并将其间的代码按照解释 JavaScript 脚本代码的方法加以解释，然后将解释结果返回 HTML 文档并在浏览器窗口显示。

【例 1-1】将 JavaScript 脚本嵌入到 HTML 文档中。

```
<html>
```

```
<head>
  <title>JavaScript Test</title>
</head>
<body>
  <center>
    <script language="JavaScript" type="text/javascript">
      document.write("Hello World!");
    </script>
  </center>
</body>
</html>
```

在例 1-1 的代码中除了 script 标记对之间的内容外，都是最基本的 HTML 代码，可见 script 标记对将 JavaScript 脚本代码封装并嵌入到 HTML 文档中。script 标记对将 JavaScript 脚本代码封装，同时告诉浏览器其间的代码为 JavaScript 脚本代码，然后调用 document 文档对象的 write()方法写字符串到 HTML 文档中。

下面重点介绍 script 标记的几个属性。

（1）language 属性：用于指定封装代码的脚本语言及版本，有的浏览器还支持 Perl、VBScript 等，所有浏览器都支持 JavaScript（当然，非常老的版本除外），同时 language 属性的默认值也为 JavaScript。

（2）type 属性：指定 script 标记对之间插入的脚本代码类型。

（3）src 属性：用于将外部的脚本文件内容嵌入到当前文档中，一般在较新版本的浏览器中使用，使用 JavaScript 脚本编写的外部脚本文件必须使用.js 为扩展名，同时在 script 标记对中不包含任何内容，具体如下：

```
<script language="JavaScript" type="text/javascript" src="Hello.js">
</script>
```

下面讨论<script>标记的 src 属性如何引入 JavaScript 脚本代码。

【例 1-2】改写例 1-1 的代码并保存为 1-2.htm。

```
<html>
  <head>
    <title>JavaScript Test</title>
  </head>
  <body>
    <center>
      <script language="JavaScript" type="text/javascript" src="1-2.js">
      </script>
    </center>
  </body>
</html>
```

同时再编辑以下代码并将其保存为 1-2.js：

```
document.write("Hello World!");
```

将 1-2.htm 和 1-2.js 文件放置于同一目录，在浏览器中打开 1-2.htm，显示结果与例 1-1 相同。

可见通过外部引入 JavaScript 脚本文件的方式，能实现同样的功能。同时具有以下优点：

（1）将脚本程序同现有页面的逻辑结构及浏览器结果分离。通过外部脚本，可以轻易实现多个页面共用完成同一功能的脚本文件，以便通过更新一个脚本文件内容达到批量更新的目的。

（2）浏览器可以实现对目标脚本文件的高速缓存，避免额外的由于引用同样功能的脚本代码而导致下载时间的增加。

与 C++语言等使用外部头文件（.h 文件等）相似，引入 JavaScript 脚本代码时使用外部脚本文件的方式符合结构化编程思想，但也有不利的一面，主要表现在以下几点：

（1）不是所有支持 JavaScript 脚本的浏览器都支持外部脚本，如 Netscape 2 和 Internet Explorer 3 及以下版本都不支持外部脚本。

（2）外部脚本文件功能过于复杂或由于其他原因导致的加载时间过长有可能导致页面事件得不到处理或者得不到正确的处理，程序员必须谨慎使用并确保脚本加载完成后其中的函数才被页面事件调用，否则浏览器报错。

综上所述，引入外部 JavaScript 脚本文件的方法是效果与风险并存，开发者应权衡优缺点以决定是将脚本代码嵌入到目标 HTML 文档中还是通过引用外部脚本文件的方式来实现相同的功能。

2. 嵌入 JavaScript 脚本代码的位置

JavaScript 脚本代码可放在 HTML 文档任何需要的位置。一般来说，可以在 head 标记和 body 标记之间按需要放置 JavaScript 脚本代码。

（1）在 head 标记之间放置。

放置在 head 标记之间的 JavaScript 脚本代码一般用于提前载入以响应用户的动作，一般不影响 HTML 文档的浏览器显示内容。以下是其基本文档结构：

```
<html>
  <head>
    <title>文档标题</title>
    <script language="javascript" type="text/javascript">
      //脚本语句…
    </script>
  </head>
  <body>
  </body>
</html>
```

（2）在 body 标记之间放置。

如果需要在页面载入时运行 JavaScript 脚本生成网页内容，应将脚本代码放置在 body 标记之间，可根据需要编写多个独立的脚本代码段并与 HTML 代码结合在一起。以下是其基本文档结构：

```
<html>
  <head>
    <title>文档标题</title>
  </head>
  <body>
    <script language="javascript" type="text/javascript">
      //脚本语句…
    </script>
      //html 语句
    <script language="javascript" type="text/javascript">
      //脚本语句…
```

```
    </script>
  </body>
</html>
```

（3）在两个标记对之间混合放置。

如果既需要提前载入脚本代码以响应用户的事件，又需要在页面载入时使用脚本生成页面内容，可以综合以上两种方式。以下是其基本文档结构：

```
<html>
  <head>
    <title>文档标题</title>
    <script language="javascript" type="text/javascript">
      //脚本语句…
    </script>
  </head>
  <body>
    <script language="javascript" type="text/javascript">
      //脚本语句…
    </script>
  </body>
</html>
```

在 HTML 文档中的哪个位置嵌入 JavaScript 脚本代码，应由其实际功能需求来决定。

1.3 JavaScript 的开发环境

由于 JavaScript 脚本代码是嵌入到 HTML 文档中被浏览器解释执行，所以开发 JavaScript 脚本代码并不需要特殊的编程环境，只需要普通的文本编辑器和支持 JavaScript 脚本的浏览器即可。

JavaScript 脚本编程一般分为以下步骤：

（1）选择 JavaScript 语言编辑器编辑脚本代码。

（2）嵌入该 JavaScript 脚本代码到 HTML 文档中。

（3）选择支持 JavaScript 的浏览器浏览该 HTML 文档。

（4）如果错误则检查并修正源代码，重新浏览，此过程重复直至代码正确为止。

（5）处理不支持 JavaScript 脚本的情况。

1.3.1 编写 JavaScript 代码

由于 JavaScript 纯粹由文本构成，因此编写 JavaScript 代码可以用任何文本编辑器，也可以用编写 HTML 和 CSS 文件的任何程序，或者用像 Visual Studio 和 Eclipse 这样强大的集成开发环境。还可以使用开源的 Aptana Studio 作为 JavaScript 的编写工具，Aptana Studio 是一个集成式的 Web 应用程序开发环境，它不仅可以作为独立的程序运行，而且还可以作为 Eclipse 插件使用，其最广为人知的是它非常强悍的 JavaScript 编辑器和调试器。Aptana Studio 可以支持多种 Ajax 和 JavaScript 工具箱，包括 JavaScript 编辑和调试。

这里介绍如何使用 Aptana Studio 编写 JavaScript 脚本，启动 Aptana Studio 2.0 后，出现如图 1-1 所示的开发环境界面。

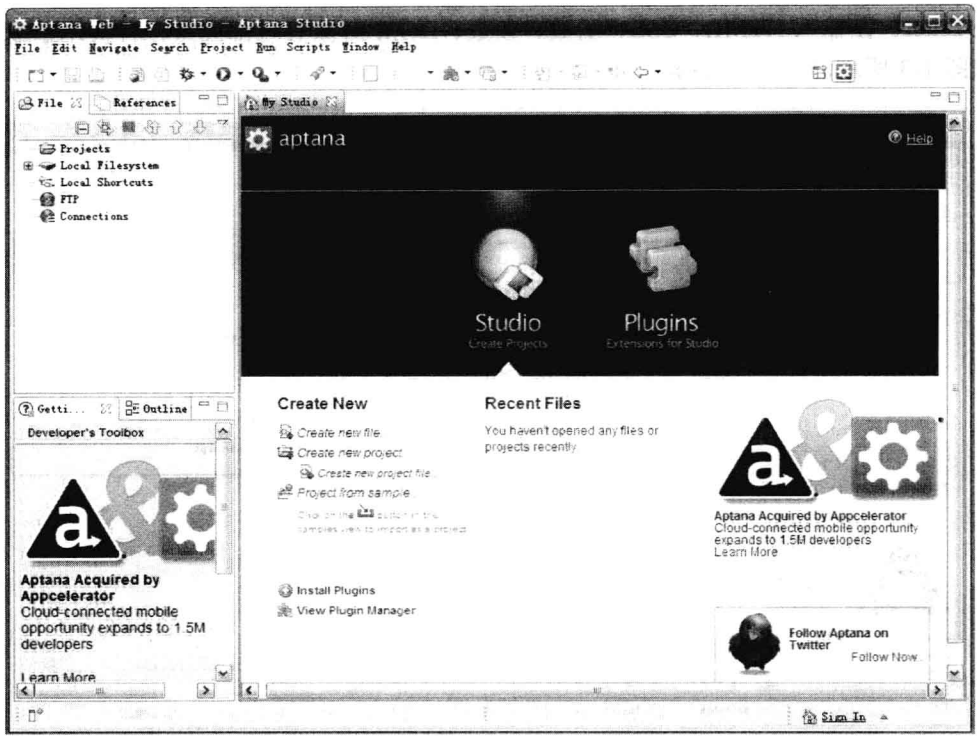

图 1-1　Aptana Studio 2.0 开发环境界面

选择 File→New→Untitled HTML File 菜单命令，新建一个空白的 HTML 文档，如图 1-2 所示。

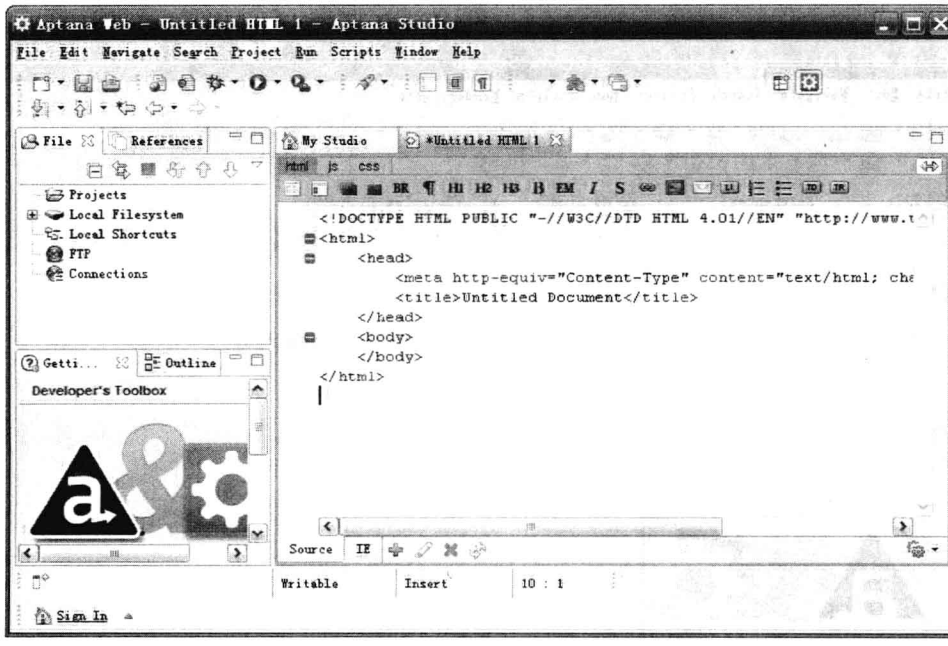

图 1-2　新建 HTML 文档

在 Aptana Studio 中编辑代码时，可以看到它的智能提示功能非常实用。它可以自动完成括号、引号的输入，无论是{}、[]、()还是引号；还有自动提示功能，输入选中的提示项的快

捷键是 Enter。在输入 HTML、CSS、函数、参数、DOM 对象等时都会有各种提示信息的信息头，如图 1-3 所示。

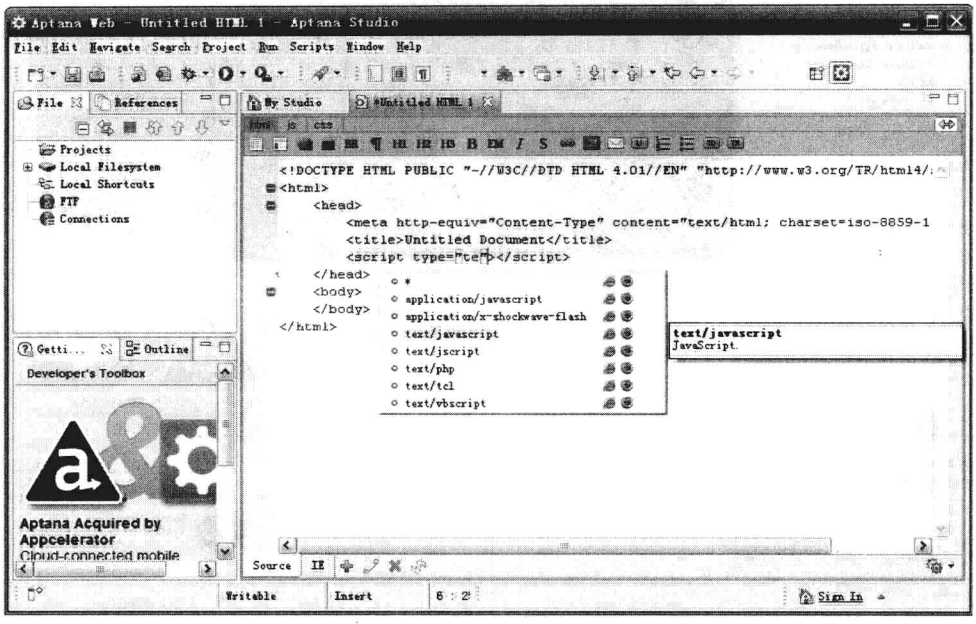

图 1-3　在 Aptana Studio 中编辑代码

从菜单中选择 File→New→Untitled JavaScript File 命令，创建一个 JavaScript 文件，编写例 1-2 中的 1-2.js 文件，如图 1-4 所示。

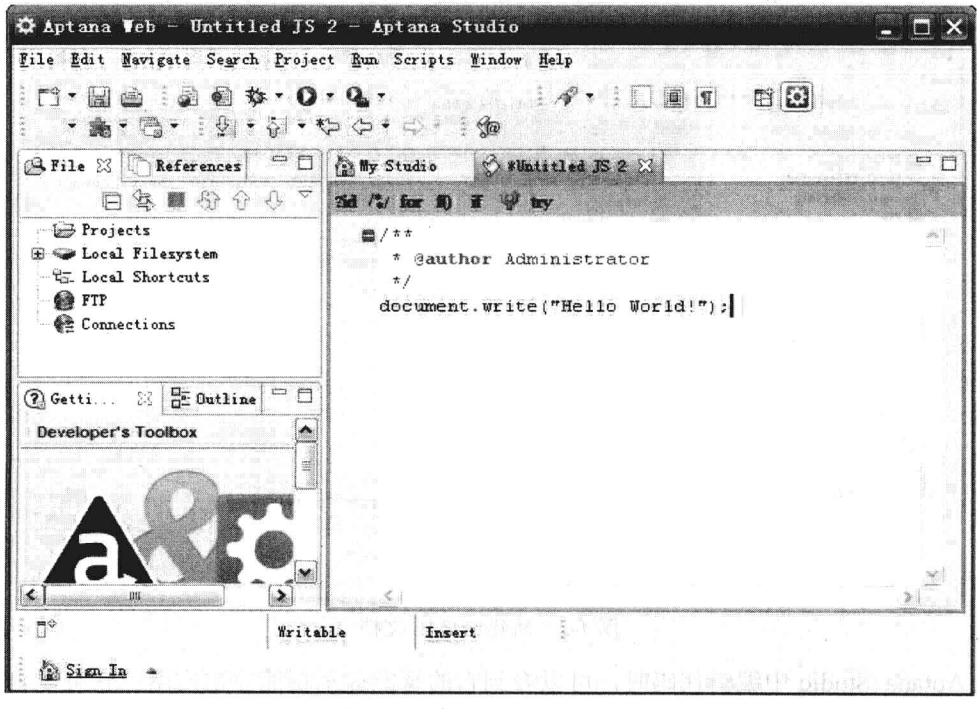

图 1-4　编写 JavaScript 文件

1.3.2　运行与调试 JavaScript 代码

运行和调试 JavaScript 代码的主要工具还是 Web 浏览器，主流的 Web 浏览器还会包含一些 JavaScript 调试程序。对于 JavaScript，Mozilla Firefox 是最适合开发用的浏览器之一。Mozilla Firefox Web 浏览器的插件 Firebug 是 Web 开发必不可少的，尤其是使用 JavaScript 和 Ajax 的 Web 开发，目前在 JavaScript 和 CSS 调试工具中位居首位。这个插件程序允许检查一个 Web 页面的所有元素、查看 Ajax 调用的结果及查看 CSS，所有这些都是实时的，可使得调试更加容易。

在 Firefox 中安装好 Firebug 后，可以在 Firefox 的“工具”菜单下的“附加组件”对话框中看到已安装的 Firebug 插件，如图 1-5 所示。

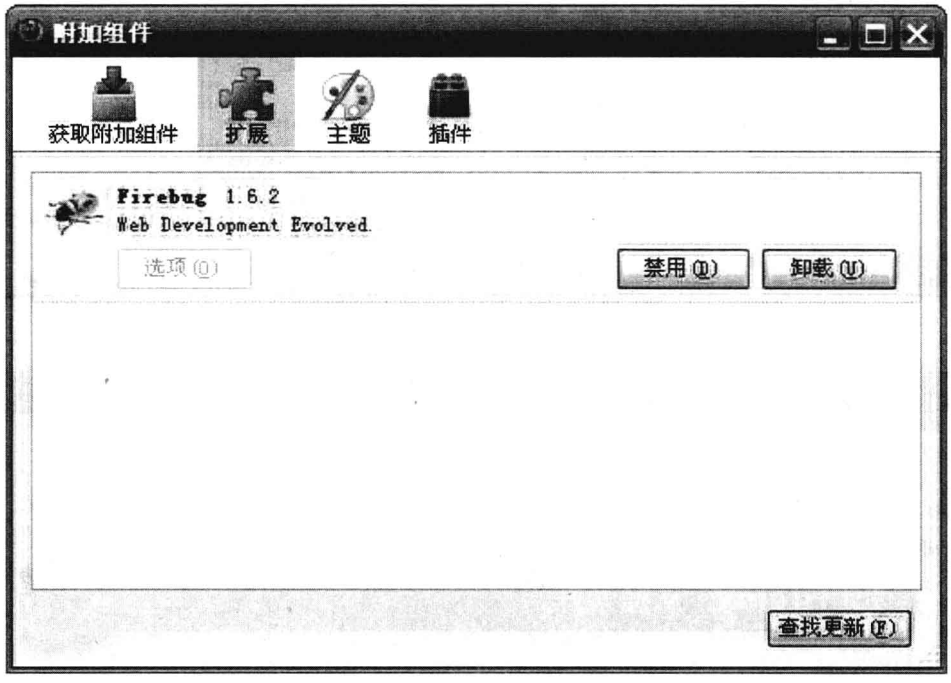

图 1-5　Firebug 插件

在 Firefox 浏览器窗口的右下角有一个小图标，单击这个图标即可打开 Firebug，如图 1-6 所示。从 Firebug 界面上的 HTML、CSS、脚本和 DOM 标签页可以观察到每个元素的状态，在 HTML 标签页中选择一个元素，右侧面板就会显示选中元素上应用的样式信息。还可以进行编辑修改，而修改结果会在 Firefox 窗口中实时显示出来，但是在这里进行的修改都是暂时的，刷新页面或关闭窗口之后就会丢失，而不会修改原始的文件。

在窗口的 Firebug 部分单击“控制台”标签页，如果控制台被禁用，可单击“控制台”标签页下的“启用”菜单打开并使用控制台，如图 1-7 所示。

JavaScript 错误信息、Ajax 调用、性能分析结果、命令行执行结果都会显示在控制台的界面上。在 Firebug 中没有必要使用警告语句或者页面记录，因为 Firebug 提供了很多手段将信息记录到 Firebug 控制台。其中，最常用的函数是 console.log()，它会将信息写入控制台，而不会干扰当前页面。下面将例 1-1 的结果改由控制台输出。

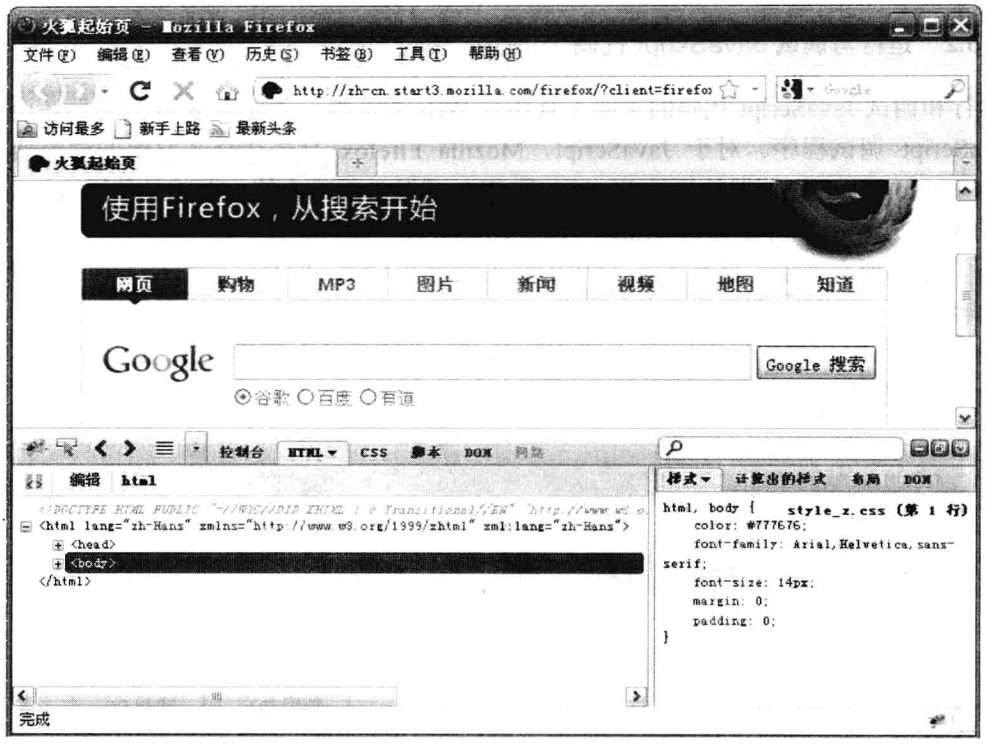

图 1-6 Firebug 中的 HTML 标签页

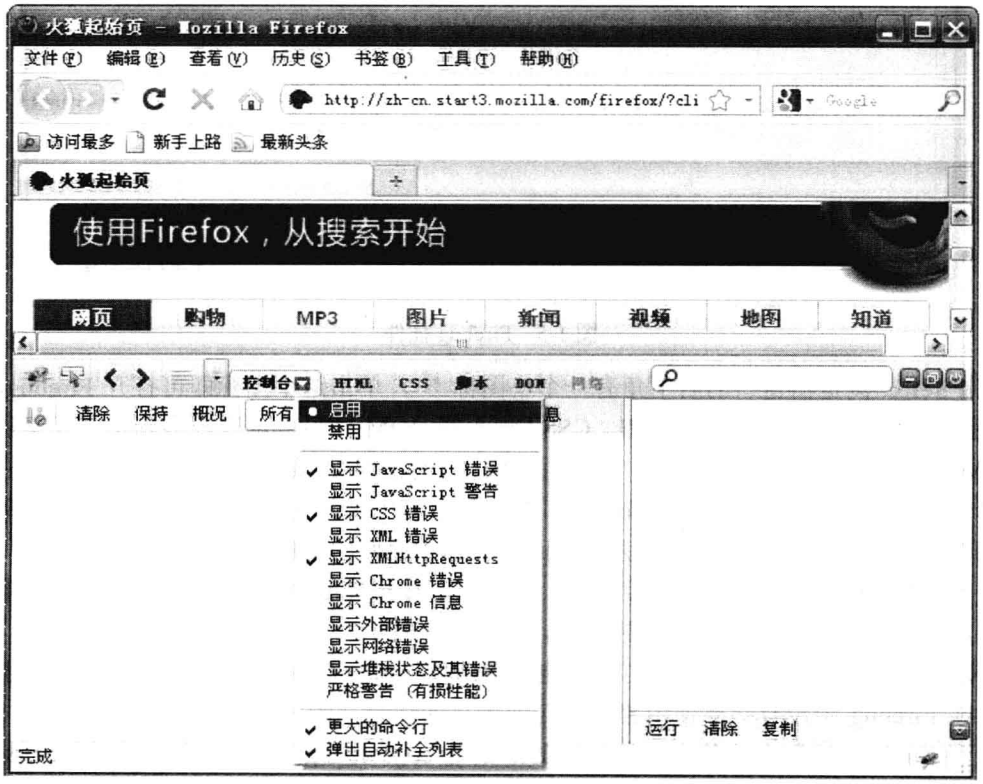

图 1-7 Firebug 控制台