21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

Javascript 程序设计教程

Javascript Programming

李林 施伟伟 编著

- 在实践中练习巩固所学知识
- 内容涵盖JavaScript高级应用
- 从Web标准角度讲解编程技术

精品系列

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

Javascript 程序设计教程

Javascript Programming

李林 施伟伟 编著

精品系列

人 民 邮 电 出 版 社

北 京

图书在版编目（CIP）数据

Javascript 程序设计教程 / 李林，施伟伟编著. —北京：人民邮电出版社，2008.5
21 世纪高等学校计算机规划教材. 精品系列
ISBN 978-7-115-17744-5

Ⅰ. J… Ⅱ. ①李…②施… Ⅲ. JAVA 语言—程序设计—高等学校—教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2008）第 029055 号

内 容 提 要

JavaScript 是目前 Web 客户端开发的主要编程语言。本书通过基础知识与应用示例相结合的方式，对 JavaScript 编程技术进行了讲解。主要内容包括：JavaScript 概述、JavaScript 基础、JavaScript 面向对象编程、正则表达式、字符串处理、浏览器对象模型（BOM）、DOM 基础、事件处理模型、JavaScript 控制页面样式、JavaScript 中的 XML 编程、JavaScript 与服务器的通信、JavaScript 与插件。

本书内容丰富，注重实际编程与开发能力的培养。对于每个知识点，本书都提供了丰富实例；对于每段程序代码，本书都提供了详尽的注释。本书可作为高等院校计算机科学与技术、计算机应用、网络工程、软件工程等专业 JavaScript 程序设计、动态网页制作等课程的教材，也可作为相关培训班的教学用书。

21 世纪高等学校计算机规划教材——精品系列

Javascript 程序设计教程

◆ 编　　著　李　林　施伟伟
责任编辑　蒋　亮

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：16.5
字数：432 千字　　2008 年 5 月第 1 版
印数：1 – 3 000 册　　2008 年 5 月北京第 1 次印刷

ISBN 978-7-115-17744-5/TP

定价：27.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

出版者的话

计算机应用能力已经成为社会各行业最重要的工作要求之一，而计算机教材质量的好坏会直接影响人才素质的培养。目前，计算机教材出版市场百花争艳，品种急剧增多，要从林林总总的教材中挑选一本适合课程设置要求、满足教学实际需要的教材，难度越来越大。

人民邮电出版社作为一家以计算机、通信、电子信息类图书与教材出版为主的科技教育类出版社，在计算机教材领域已经出版了多套计算机系列教材。在各套系列教材中涌现出了一批被广大一线授课教师选用、深受广大师生好评的优秀教材。老师们希望我社能有更多的优秀教材集中地呈现在老师和读者面前，为此我社组织了这套“21 世纪高等学校计算机规划教材——精品系列”。

“21 世纪高等学校计算机规划教材——精品系列”具有下列特点。

（1）前期调研充分，适合实际教学需要。本套教材主要面向普通本科院校的学生编写，在内容深度、系统结构、案例选择、编写方法等方面进行了深入细致的调研，目的是在教材编写之前充分了解实际教学的需要。

（2）编写目标明确，读者对象针对性强。每一本教材在编写之前都明确了该教材的读者对象和适用范围，即明确面向的读者是计算机专业、非计算机理工类专业还是文科类专业的学生，尽量符合目前普通高等教学计算机课程的教学计划、教学大纲以及发展趋势。

（3）精选作者，保证质量。本套教材的作者，既有来自院校的一线授课老师，也有来自 IT 企业、科研机构等单位的资深技术人员。通过他们的合作使老师丰富的实际教学经验与技术人员丰富的实践工程经验相融合，为广大师生编写出适合目前教学实际需求、满足学校新时期人才培养模式的高质量教材。

（4）一纲多本，适应面宽。在本套教材中，我们根据目前教学的实际情况，做到“一纲多本”，即根据院校已学课程和后续课程的不同开设情况，为同一科目提供不同类型的教材。

（5）突出能力培养，适应人才市场要求。本套教材贴近市场对于计算机人才的能力要求，注重理论技术与实际应用的结合，注重实际操作和实践动手能力的培养，为学生快速适应企业实际需求做好准备。

（6）配套服务完善，共促提高。对于每一本教材，我们在教材出版的同时，都将提供完备的 PPT 课件，并根据需要提供书中的源程序代码、习题答案、教学大纲等内容，部分教材还将在作者的配合下，提供疑难解答、教学交流等服务。

在本套教材的策划组织过程中，我们获得了来自清华大学、北京大学、人民大学、浙江大学、吉林大学、武汉大学、哈尔滨工业大学、东南大学、四川大学、上海交通大学、西安交通大学、电子科技大学、西安电子科技大学、北京邮电大学、中国林业大学等院校老师的大力支持和帮助，同时获得了来自信息产业部电信研究院、联想、华为、中兴、同方、爱立信、摩托罗拉等企业和科研单位的领导或技术人员的积极配合。在此，人民邮电出版社向他们表示衷心的感谢。

我们相信，“21 世纪高等学校计算机规划教材——精品系列”一定能够为我国高等院校计算机课程教学做出应有的贡献。同时，对于工作欠缺和不妥之处，欢迎老师和读者提出宝贵的意见和建议。

前 言

JavaScript 是 Web 开发的重要语言，特别是在 Ajax 技术出现之后，JavaScript 编程技术在 Web 开发中得到越来越广泛的应用。Web 2.0 以及富客户端应用的流行，使得 Web 前端开发已经成为一个新兴的开发方向，专业的 Web 前端开发工程师更是供不应求，而 JavaScript 则是他们必备的技术之一。

JavaScript 语言是 1995 年由 Netscape 公司与其浏览器产品一同发布的，距今已有 10 多年的发展历史。有人说 JavaScript 是世界上最被误解的语言，这种说法不无道理。很多人认为 JavaScript 的功能无非是做一些简单的数据验证和客户端的特效，然而实际上 JavaScript 所能做的远远超过这些，通过本书的学习读者将会逐步认识到这一点。

在开始本书的学习之前，读者应当具备 HTML 的基本知识以及某一门高级编程语言（如 C/C++、Java、C#等）的使用经验。对于 JavaScript 编程将涉及的其他技术，如 CSS、XML 等，本书的相关章节将会介绍相关基础知识。

掌握一门语言最好的方法就是实践。本书的着眼点是将基础的理论知识讲解和实践应用相结合，使读者快速掌握 JavaScript 编程技术。全书共分 12 章，在大多数章节中，将首先介绍相关的基础知识，然后重点讲解相关的实例。

第 1、2 章对 JavaScript 的基础知识包括 JavaScript 的发展历史、相关标准、基本语法等进行了介绍。

第 3 章对 JavaScript 的面向对象特性进行了介绍，读者可以从这一章中了解如何在 JavaScript 中实现基本的面向对象特性，如封装、继承、多态等。

第 4、5 章分别讲解了 JavaScript 的字符串操作和正则表达式的相关知识。

第 6 ~ 9 章是本书的重点，分别讲解了 BOM 模型、DOM 基础知识、事件处理和 DOM 样式编程，这些都是在 Web 开发中最为常用的技术。

第 10 章介绍了如何使用 JavaScript 进行 XML 编程。

第 11 章也是本书的重点内容，介绍了各种使用 JavaScript 与服务器端进行通信的技术，如隐藏框架、远程脚本技术，还对 Ajax 技术的基本原理进行了介绍。

第 12 章介绍了 JavaScript 与各种嵌入式对象的交互方法，包括 ActiveX 控件、Flash 等。

本书由李林、施伟伟执笔编写。陈宏伟、刘锋、徐红、张蓓、范锡琴、朱琪和任献红等参与本书编辑、修改和整理等工作，在此向他们致以诚挚的谢意。书中所有例题和相关代码都经调试通过。本书还制作了多媒体课件，供老师教学参考使用。

由于时间仓促和作者的水平有限，书中错误和不妥之处在所难免，敬请读者批评指正。

编 者

2008 年 2 月

目 录

第 1 章 JavaScript 概述

在 Web 时代的早期，编写 HTML 相当简单，在网页上只是进行简单的呈现和表单提交；随着 Internet 的发展，人们对网页的期望值逐渐提高，他们希望对网页有更大的控制能力。此外，此时单纯的 HTML 已经不足以满足用户与应用程序进行交互的需求，因为在客户端就可以进行的验证工作也必须提交给服务器端，再由服务器将结果返回给客户端，这对于用户而言增加了不必要的等待，给服务器也造成了不必要的开销。

Netscape 发明了 JavaScript 语言，它提供了控制浏览器和在客户端与用户交互的方法，弥补了简单 HTML 页面的不足。自从诞生之日起，JavaScript 经历了多次的演化，我们将在本章中介绍 JavaScript 的演化过程。在本章中，我们还将讨论 JavaScript 是什么、它可以做什么、它不能做什么、它与 Web 标准之间的关系，以及 JavaScript 的相关开发工具。

1.1 JavaScript 是什么

JavaScript 语言是一门解释型的脚本语言，在 Web 开发中通常应用它来增强网页与应用程序间的交互。Netscape 公司首先在 Netscape Navigator 浏览器中实现了 JavaScript，随后其他的浏览器也相继推出了各自版本的 JavaScript，并且与标准化组织 ECMA 一起制定了 ECMAScript 语言规范。目前，ECMAScript 第 3 版是 ECMAScript 的最新版本，也是 JavaScript 的工业标准。

JavaScript 通常内嵌在 HTML 页面中运行，与页面同时被浏览器加载和运行，在 HTML 页面中使用<script>标签引入 JavaScript 脚本。以下是一个简单的 JavaScript 应用示例。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>HTML Sample Page</title>
  <script type="text/javascript">
    alert("你好, JavaScript! ");
  </script>
</head>
<body>
  <h1>你好, JavaScript! </h1>
</body>
</html>
```

运行结果如图 1-1 所示。

图 1-1 简单 JavaScript 应用示例

除了直接在<script>标签内编写 JavaScript 脚本之外，还可以使用其 src 属性导入外部的 JavaScript 脚本文件，将以上示例中的<script>标签部分修改如下：

```
<script type="text/javascript" src="hello.js"><script>
```

其中 hello.js 是外部的 JavaScript 脚本，其内容为：

```
alert("你好, JavaScript! ");
```

页面运行的结果与图 1-1 中相同。这里需要特别指出的是：对于大多数 HTML 标签来说，如果标签内部为空，例如：

```
<input type="buton" value="click"></input>
```

可以简写为：

```
<input type="buton" value="click" />
```

在 XML 文件的编写中同样也可以采取类似的简写方式，但对于<script>标签来说，这种简写方式是不允许的。以下的代码是错误的，相应的 JavaScript 脚本将不会被正确的导入：

```
<script type="text/javascript" src="hello.js" />
```

1.2 JavaScript 的发展历史

JavaScript 可以运行在多种宿主环境下，浏览器是其中最主要的一种，本书只讨论浏览器中的 JavaScript。实际上，JavaScript 的发展历史和浏览器的发展过程是密切相关的。

1.2.1 Netscape Navigator

JavaScript 首次亮相于 1995 年。由于当时的浏览器缺乏相应的处理机制，所有的数据验证都必须由服务器端来完成，因此用户常常会遇到少填写一、两个字段就不得不重新填写整个表单的情况。特别是当时的接入速率普遍为 28.8kbit/s，漫长的等待让用户十分厌烦，希望在客户端进行数据校验的呼声也越来越高。在这样的情况下，网景公司（Netscape）在其出品的浏览器软件 Netscape Navigator 2.0 中发布了 JavaScript 的 1.0 版本。推出 JavaScript 1.0 的主要目的是为了实现在客户端验证用户输入的信息，从而减少不必要的等待时间。

关于 JavaScript 的命名还有一段小插曲，它曾经用过的名字有 LiveWire、LiveScript，在正式发布前才改名为 JavaScript。JavaScript 1.0 的推出获得了极大的成功，从此 JavaScript 成为了进行 Web 开发所必备的一项技术，随后 Netscape 于 1996 年在 Netscape Navigator 3.0 中发布了 JavaScript 1.1。

1.2.2 Microsoft Internet Explorer

20 世纪 90 年代，微软公司（Microsoft）的 Internet Explorer（IE）是 Netscape Navigator 的主要竞争对手，它依靠其操作系统的绝对优势向 Netscape Navigator 发起了强大的攻势。从 Windows 95 OSR2 开始，IE 3.0 就成为所有 Windows 操作系统的默认浏览器。微软分别在 IE 3.0 的早期和后期版本中实现了 JScript 1.0 和 JScript 2.0，相当于 Netscape 的 JavaScript 1.0 和 JavaScript 1.1；在随后的 IE 4.0 中还实现了 JScript 3.0，相当于 Netscape 的 JavaScript 1.3。在很长一段时期内，IE 的市场占有率一直占有绝对优势，因此微软的 JScript 也成为了 JavaScript 的事实标准。

尽管如此，Netscape 一直没有放弃与微软的竞争，在 1997 ~ 2002 年间，它先后发布了 Netscape Navigator 的 4.0 ~ 4.8 版本。1998 年，Netscape 还宣布 Netscape Navigator 免费，并且公布了 Netscape

Communicator（包括 Netscape Navigator）的所有源代码，这也是 Mozilla 项目的开始。同年，Netscape 公司被 AOL 收购，之后 AOL 又被时代华纳公司收购，2003 年 7 月时代华纳宣布解散 Netscape 公司，从此 Netscape 公司成为历史。但是即便如此，在此之后 AOL 还是发布了 Netscape Navigator 7.2 ~ 8.x 版本，开发方面由加拿大的 Mercurial Communications 公司完成。

1.2.3　Mozilla Firefox

微软在赢得浏览器大战胜利之后，IE 在标准化方面却少有进展，对于 DOM 和 CSS 标准的支持一直没有改善，直至 IE 7.0 的推出，IE 对于 DOM 的支持还是 Level 1 和少数的 Level 2，对 CSS 的支持仍然存在着不少的缺陷，以致于开发人员不得不使用各种特殊技巧（Hack）来规避它们。

与 IE 形成鲜明对比的是 Mozilla Firefox。标签页浏览方式、良好的安全性和可扩展性、对标准的良好支持程度，使得 Firefox 赢得了众多用户（特别是 Web 开发人员）的青睐。Firefox 是基于 Gecko 引擎的浏览器，其内部使用的是名为 Spidermonkey 的 JavaScript 解释引擎，它是基于 C 语言的 JavaScript 实现，也是目前性能最好的 JavaScript 引擎。目前，Firefox 最新的稳定版本是 Firefox 2.0，它实现的 JavaScript 版本是 1.7。在未来的 Firefox 3.0 中，将会发布 JavaScript 2.0。Firefox 是当前对标准支持程度最高的浏览器，它支持全部的 DOM Level 1 和 Level 2，以及少数 Level 3；在 CSS 方面，它支持大部分的 CSS 2 和少部分的 CSS 3。

今天我们在使用 JavaScript 进行 Web 开发时，至少需要考虑兼容 Firefox 和 IE 两种浏览器。Firefox 对标准的支持程度更高，因此通常是 Web 开发人员首选的开发平台。

1.2.4　ECMAScript 标准

不同厂商所实现的 JavaScript 并不完全一致，这在很大程度上增加了开发人员的工作量，甚至有时候会导致一些概念上的混淆，因此在 1997 年 JavaScript 1.1 作为一个草案被提交到欧洲计算机制造商协会（ECMA），由 Netscape、微软、SUN、Borland 以及其他一些公司的程序员组成的 TC39 委员会根据这一草案最终制定出 ECMA-262 标准，该标准给出了 ECMAScript 脚本语言的定义。随后，国际标准化组织（ISO）和国际电工委员会（IEC）接纳 ECMAScript 为标准（ISO/IEC-1626），至此 ECMAScript 成为 JavaScript 的工业标准。

ECMAScript 从诞生至今共经历了 3 个版本。其中，ECMAScript 第 1 版对 JavaScript 1.1 的基本特性进行了标准化，并且增加了一些新特性，在 ECMAScript 第 1 版中没有对 switch 语句和正则表达式进行标准化。ECMAScript 第 2 版与第 1 版没有功能或者特性上的区别，只是增加了说明。与 ECMA 第 1 版和第 2 版相对应的浏览器实现是 Netscape 4.x 和 IE 4.0。ECMAScript 目前的最新版本是第 3 版，在这个版本中增加了 switch 语句、异常处理和正则表达式等特性，相应的浏览器实现版本是 Mozilla、Netscape 6 和 IE 5.5+。ECMAScript 的第 4 版正在开发过程中，它对应的将是未来的 JavaScript 2.0。

这里需要说明的是，ECMAScript 虽然源自 JavaScript，但是作为一个标准，JavaScript 并不是它的唯一实现。例如 Adobe 公司的 ActionScript 也是符合 ECMAScript 标准的脚本语言。如果用面向对象编程的概念来作个比喻，ECMAScript 是接口（Interface），那么 JavaScript、ActionScript 等语言则是实体类（Concrete Class）。ECMAScript 对实现语言的共性进行抽象，描述了脚本语言的语法、数据类型和对象等基本要素，而如何具体实现这些要素，ECMAScript 是不会对其进行规定的。每个具体的实现语言都会增加一些 ECMAScript 未曾定义的特性，这一点在开发兼容各

种浏览器的 Web 应用时需要特别注意，应该尽可能地使用 ECMAScript 中定义的方法和特性，而不要使用某种浏览器独有的特性。

Web 浏览器不是 ECMAScript 的唯一宿主环境，例如 Windows 操作系统可以通过 WScript 或者 CScript 来执行 JavaScript 脚本；Netscape 的应用服务器产品 Netscape Enterprise Server（NES）支持使用 JavaScript 进行服务器端开发，甚至与 Java 类进行交互。在本书中，主要考虑的宿主环境是 Web 浏览器，在这一领域最主要的两大 ECMAScript 解释引擎分别是微软的 JScript 和 Mozilla 的 Spidermonkey。根据相关的测试结果，Spidermonkey 是当今执行效率最高的 JavaScript 引擎，这也是在众多与 JavaScript 相关的性能测试报告中，Mozilla Firefox 的成绩会远远高于 IE 的主要原因。

1.3 JavaScript 与 Java

JavaScript 的名称给人们造成了很大的误解，很多初学者往往认为它与 Java 语言存在一些联系。实际上，JavaScript 与 Java 除了名称类似之外，没有任何共同点。

JavaScript 是一门脚本语言，通过浏览器的脚本引擎解释执行；Java 语言则是继 C、C++之后的又一种主流开发语言，Java 代码将首先编译为字节码（.class 文件），然后在 Java 虚拟机上将字节码编译为二进制代码运行。

JavaScript 语言绝大多数的应用在浏览器端，而 Java 语言的应用则相当广泛，从企业级服务器应用到移动设备都可以使用 Java 语言进行开发。在浏览器端，Java 可以用于创建 Java 小应用程序（Applet）。它可以实现用传统 HTML 很难实现的动画效果，Applet 在大多数情况下是独立运行的，但它允许通过外部脚本进行控制，本书的第 12 章将介绍如何使用 JavaScript 脚本操纵 Applet。

实际上，JavaScript 与另一门编程语言 C++的关系可能更加密切，浏览器的脚本解释引擎通常是使用 C++语言实现的。网页中编写的 JavaScript 代码最终将通过脚本解释引擎转化为 C++的指令执行。

1.4 JavaScript 可以做什么

本书仅讨论浏览器中的 JavaScript，它可以完成以下的任务：

（1）操纵浏览器对象，例如窗口的打开和关闭，浏览器类型和版本的检测等；

（2）操纵 DOM 树，这是 JavaScript 运用最为广泛的领域，JavaScript 可以实现对网页 DOM 元素的操作，支持 DOM 事件处理以及对网页的样式进行编程控制；

（3）通过 XMLHttpRequest 对象与服务器端进行异步通信，这是 Ajax 技术（Asynchronous JavaScript and XML）的核心；

（4）XML 编程，JavaScript 可以借助于 ActiveX 控件或者浏览器内置对象完成对 XML DOM 解析、XPath 查询和 XSLT 转换等工作；

（5）与浏览器插件的交互，常见的插件类型包括 ActiveX 控件、Flash 动画和 Java Applet。

在后续章节中，本书将逐步地向读者介绍如何使用 JavaScript 完成以上这些任务。

1.5　JavaScript 不能做什么

上一小节介绍了 JavaScript 可以做的事情，但是强大的功能往往是一把双刃剑，如果被人恶意滥用，将会产生很多安全性方面的问题，因此浏览器的设计者都会为浏览器定义一个安全沙箱，JavaScript 脚本能够实现的功能将被限制在这个沙箱之内，从而保证网站用户的安全。安全沙箱对 JavaScript 实现功能的限制包括：

（1）不能访问本网站所在域外的脚本和资源；

（2）不能操作客户端本地的文件（cookie 除外）；

（3）只能操纵由它自己打开的浏览器窗口；

（4）不能将浏览器窗口的尺寸设置得过小或者将窗口移出屏幕可视范围。

以上的限制是基于通常情况下的浏览器默认配置，有些浏览器允许用户对安全沙箱的限制进行定制，当前网络上流行的各种浏览器辅助工具也有可能对 JavaScript 所能实现的功能产生影响。

除了安全沙箱的限制之外，还有很多任务是 JavaScript 不能实现的，例如直接访问服务器端的数据和程序等。准确地讲，这些限制是由 Web 应用的架构所决定的，JavaScript 运行在浏览器端，只有通过提交表单、XMLHttpRequest 对象等间接的方式触发服务器端的相应程序，而不能直接操作服务器端对象。

1.6　JavaScript 与 Web 标准

Web 标准不是某一个标准，而是一系列标准的集合。网页主要由 3 部分组成：结构（Structure）、表现（Presentation）和行为（Behavior）。与之对应的标准也分为 3 个方面：结构化标准语言主要包括 XHTML 和 XML，表现标准语言主要包括 CSS，行为标准主要包括对象模型，如 W3C DOM 标准、ECMAScript 等。JavaScript 属于行为的范畴，它遵循了 ECMAScript 标准。

为什么要推广使用 Web 标准呢？最重要的原因之一是要结束 Web 开发领域由于不同浏览器类型和版本而导致的无序和混乱状态。很多情况下，某些网站只能使用特定的浏览器才能正常浏览（通常是 IE），这样就在无形中屏蔽了部分潜在的用户，而事实上非 IE 浏览器（特别是 Firefox）的市场占有率正在上升。如果要兼容各种浏览器类型和版本，Web 开发人员不得不为特定的浏览器编写单独的代码。直到本书写作时，我们仍然需要面对这样的事实，但是 Web 标准的普及已经成为一种趋势，因此从现在开始就编写符合 Web 标准的代码，无疑是明智之举。

那么，从 JavaScript 编程的角度来说，我们需要注意遵循哪些标准呢？首先是 DOM 标准，本书将会涉及 DOM 相关的理论和示例。在具体的应用场景，具体的项目中，我们可以使用非标准的方法，但是前提是作为专业的 Web 开发人员，应该清楚地了解哪些是 DOM 标准中定义的方法、哪些是个别浏览器特有的方法，以及使用它们可能带来什么问题、我们是否可以接受这些问题带来的后果等。

同样，对于 CSS 和 JavaScript 语言本身，都需要注意标准化的问题，不同浏览器对 CSS 和 JavaScript 脚本的支持程度不尽不同。

关于 Web 标准在实际开发工作中的实施，笔者的建议是应该首先熟知相关的标准规范，其次

借助相关开发工具的支持，最后是在兼容 DOM 标准的现代浏览器上进行开发和测试。

1.7 JavaScript 开发工具

1.7.1 集成开发环境（IDE）

“工欲善其事，必先利其器”。一款方便的开发工具会在很大程度上提升编码的效率，降低发生错误的可能性。评价 JavaScript 开发工具的优劣，通常会考虑以下的因素：

（1）代码提示功能；

（2）内置调试功能；

（3）对 HTML/CSS 的支持。

这里推荐使用的 JavaScript 开发工具是 Aptana（http://www.aptana.org），它是一套免费的 Web 集成开发环境，既可以与 Eclipse 集成，又可以单独运行，完全满足以上 3 点要求。不仅如此，它还具有一些其他的特性，包括：

（1）内置对流行 Ajax 开发框架的支持，如图 1-2 所示；

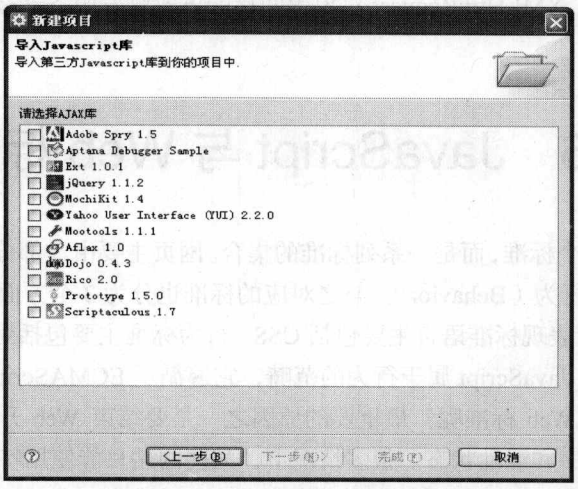

图 1-2　Aptana 对 Ajax 开发框架的支持

（2）Aptana 的代码提示功能将给出方法的浏览器兼容性和标准兼容性信息，如图 1-3 所示；

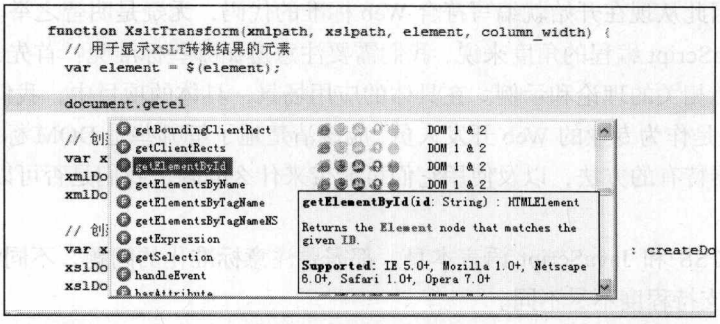

图 1-3　代码提示功能

（3）内置 Ajax 开发框架帮助文档，如图 1-4 所示；

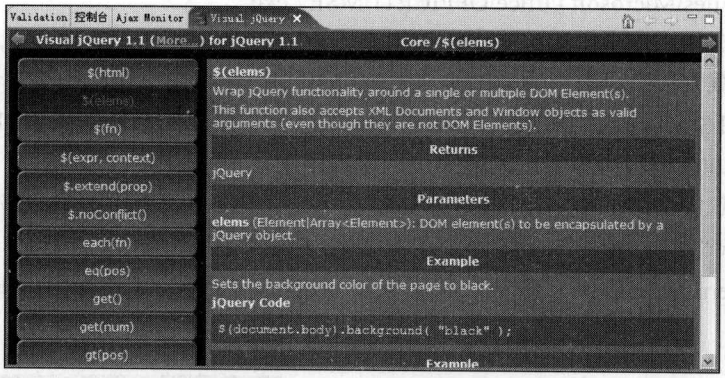

图 1-4　Ajax 开发框架帮助文档

（4）用户可以定制任务和宏，快速实现常用的功能，如图 1-5 所示。

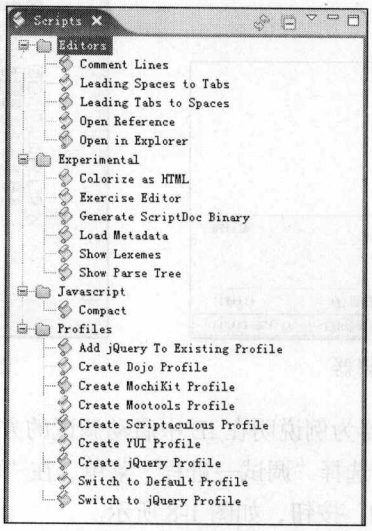

图 1-5　定制任务和宏

1.7.2　调试 JavaScript

当程序执行出现问题时，寻找问题出现原因的最有效方法就是进行跟踪调试，对于 JavaScript 代码而言也是一样的。当然，调试 JavaScript 代码与其他语言不同，一个浏览器兼容的 Web 应用，它的 JavaScript 代码必须保证能够在各种浏览器中正确运行，因此我们需要掌握在多种浏览器中调试 JavaScript 的方法。本书限于篇幅，不能对所有的 JavaScript 调试方法进行一一介绍，考虑到 IE 是用户数量最多的浏览器，FireFox 是标准兼容性最好的浏览器，下面就只介绍这两种浏览器的 JavaScript 调试方法。

1. IE 中调试 JavaScript

微软的很多产品都集成了脚本调试器，例如微软的办公软件 Office，在 Word 或者 Excel 的菜单中选择“工具→宏→Microsoft 脚本编辑器”，将会启动 Microsoft 脚本编辑器，如图 1-6 所示。

当然也可以到 Office 的安装目录下直接启动该程序，默认情况下，Microsoft 脚本编辑器的可

执行文件路径为：

C:\Program Files\Microsoft Office\OFFICE11\MSE7.exe

此外微软还提供了一个单独的脚本调试工具——Microsoft Script Debugger，用户可以到微软的官方网站上下载安装。再有，微软的集成开发环境 Visual Studio 同样也具备调试 JavaScript 脚本的功能，Visual Studio 6 中可以使用 Visual Interdev 工具进行脚本调试，Visual Studio .NET 2003 和 Visual Studio 2005 则将脚本调试功能集成到主程序中。

在 IE 中调试 JavaScript 之前，首先需要将 IE 的“禁用脚本调试”选项取消，具体的方法是在 IE 菜单中选择“工具→Internet 选项”，在 Internet 选项对话框中选择“高级”选项页，找到“禁用脚本调试（Internet Explorer）”选项，将其取消，如图 1-7 所示。最后单击“确定”按钮使选项的修改生效。

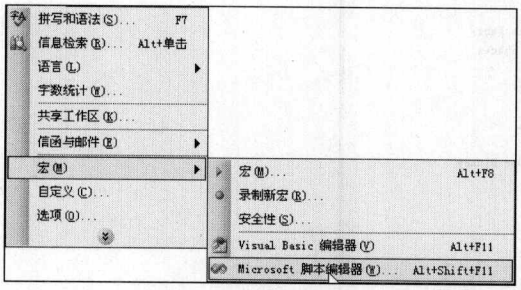

图 1-6　启动脚本编辑器

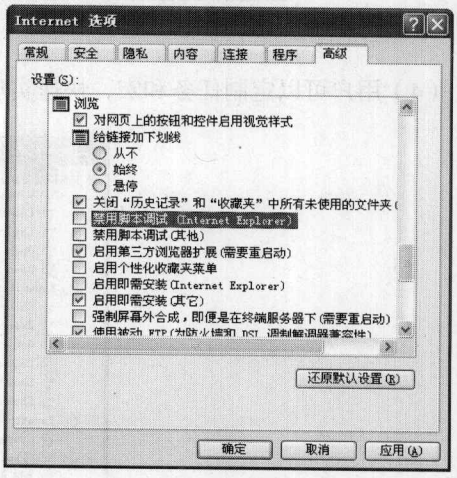

图 1-7　启动脚本编辑器

下面以 Microsoft 脚本编辑器为例说明在 IE 中启动调试的方法。在调试之前，需要在 IE 中打开相应的网页，在脚本编辑器中选择“调试→进程”菜单，在“可用进程”列表中选择 IE 的进程“IEXPLORE.EXE”，单击“附加”按钮，如图 1-8 所示。

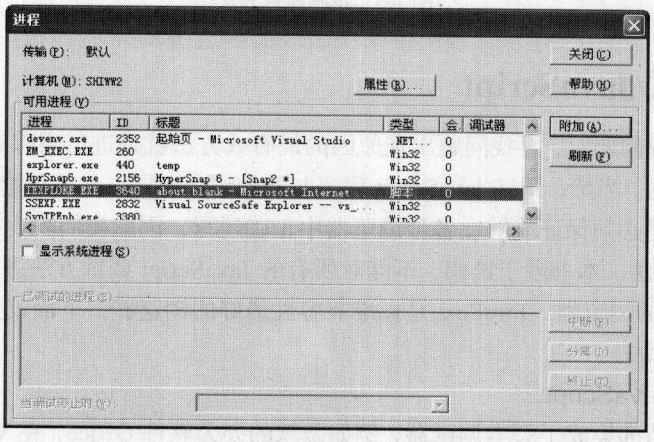

图 1-8　IE 中调试 JavaScript

然后，在弹出的对话框中选择“Script”，单击“确定”按钮，如图 1-9 所示。

将 IE 进程附加到调试进程中，在脚本编辑器中打开需要调试的网页或者 JavaScript 代码文件，设置断点后，即可进行 JavaScript 的调试。

如果觉得上面介绍的方式过于复杂，还可以通过在 JavaScript 脚本中添加 debugger 指令的方式自动启动 JavaScript 脚本调试器。下面是一段 JavaScript 代码，我们在其中添加了一行 debugger 代码，当程序执行到 debugger 这一行时，IE 会自动启动相应的脚本调试器，如果安装了多种脚本调试器，那么会提示要求选择调试器的类型，如图 1-10 所示。

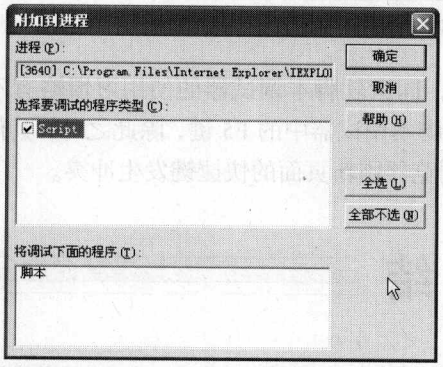

图 1-9　IE 中调试 JavaScript

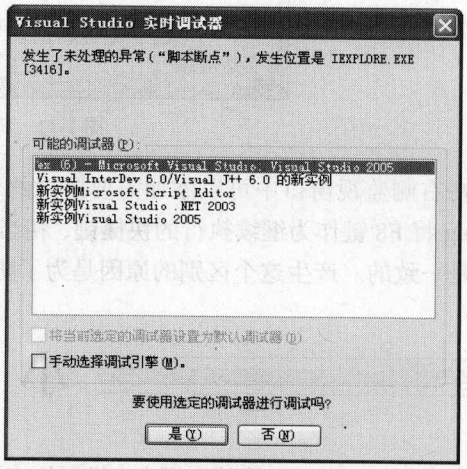

图 1-10　选择脚本调试器类型

```
// Flash 对象
var flash = getFlash("flash");
// 设置断点
debugger;
// 文本框输入值
var textValue = $("text").value;
// 设置 Flash 中 myVar 变量值，改变 Flash 中文本框的显示内容
flash.SetVariable("myVar", textValue);
```

在脚本调试器中调试 JavaScript 代码的功能同在 Visual Studio 中调试 C++、C#代码是完全一致的，使用过 Visual Studio 开发工具的读者应该很容易上手。

2. Firefox 中调试 JavaScript

Firefox 中有很多 Web 开发相关的插件都提供了调试 JavaScript 的功能，Firebug 是其中之一，它的功能相当强大，不仅仅可以作为 JavaScript 脚本调试器，还具备了 DOM 查看、CSS 可视化查看、HTTP 监控和 JavaScript 性能测试等多种功能，可以说 Firebug 是 Web 开发人员必备的 Firefox 插件。

Firebug 的官方网站是 http://www.getfirebug.com，在 Mozilla 的网站上也可以找到 Firebug 插件下载和安装的页面：https://addons.mozilla.org/en-US/firefox/addon/1843。安装 Firebug 之后，在 Firefox 状态栏的右下角会出现一个绿色的图标，单击该图标即可打开 Firebug 的主界面。

Firebug 同样支持 debugger 指令启动脚本调试器，图 1-11 所示为 Firebug 的 JavaScript 调试器界面。在 Firebug 界面右上方的搜索输入框中，可以输入需要查询的关键词，快速定位代码，还可以通过“#+行号”的方式直接定位到指定的代码行，例如输入“#40”时，左侧的代码查看区域将会定位到第 40 行的位置。单击代码查看区域中的行号，可以设置或者取消断点。