PROGRAMMER TO PROGRAMMER™

Beginning JavaScript 3rd Edition

JavaScript 入门经典

(第3版)

- 畅销红皮书
- JavaScript 经典教程
- 涵盖各种新技术

(美) Paul Wilton
Jeremy McPeak 著
施宏斌 译

wrox™
WILEY
清华大学出版社

JavaScript 入门经典

(第 3 版)

(美) Paul Wilton
Jeremy McPeak 著
施宏斌 译

清华大学出版社
北 京

Paul Wilton，Jeremy McPeak
Beginning JavaScript, 3rd Edition
EISBN：978-0-470-05151-1

图书在版编目(CIP)数据

JavaScript 入门经典(第 3 版)/(美)威尔顿(Wilton, P.)，(美)麦可匹克(McPeak, J.)　著；施宏斌　译.
—北京：清华大学出版社，2009.2
书名原文：Beginning JavaScript, 3rd Edition
ISBN 978-7-302-19419-4
Ⅰ. J…　Ⅱ. ①威…②麦…③施…　Ⅲ. JAVA 语言—程序设计　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2009)第 010676 号

责任编辑：王　军　郑雪梅
装帧设计：孔祥丰
责任校对：胡雁翎
责任印制：孟凡玉
出版发行：清华大学出版社　　地　　址：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　邮　　编：100084
社　总　机：010-62770175　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京密云胶印厂
装 订 者：三河市金元印装有限公司
经　　销：全国新华书店
开　　本：185×260　印　张：47　字　数：1144 千字
版　　次：2009 年 2 月第 1 版　　印　　次：2009 年 2 月第1 次印刷
印　　数：1～4000
定　　价：98.00 元

前　言

JavaScript 是一种脚本语言，它可以增强静态 Web 应用的功能，从而为 Web 页面提供动态的、个性化的内容，通过 JavaScript 还可以与用户进行交互。JavaScript 提升了用户访问站点时的用户体验，增强了网站对用户的吸引力。现在，令人炫目的下拉菜单、滚动的文字和动态的内容已经广泛应用于各种 Web 站点，这一切都是通过 JavaScript 来实现的。各种主流的现代浏览器都支持 JavaScript，实际上 JavaScript 语言已经成为客户端 Web 开发的首选脚本语言。另外，JavaScript 语言也可以应用于 Web 之外的其他场合，例如 Windows 系统中的自动管理任务。

本书的宗旨在于介绍使用 JavaScript 进行开发的基础知识，即 JavaScript 是什么，JavaScript 代码是如何运行的，以及使用 JavaScript 能够实现哪些功能等。本书将先介绍 JavaScript 语言的基本语法，然后再介绍如何使用 JavaScript 创建功能强大的 Web 应用程序。读者无须为没有编程经验而担心，在本书中将详细地介绍编写程序的相关知识。学习 JavaScript 是通向程序设计世界的一道大门，通过对本书中基础知识的学习和理解，就可以进一步学习编程世界中的新知识和其他高级技术。

本书读者对象

为了最好地汲取本书中的知识，读者应该对 HTML 有所了解，并知道如何创建静态的 Web 页面。除此之外，读者无须具备任何编程基础。

本书同样适合于具有编程经验的读者，并把这些读者引导到 Web 程序设计的世界中。某些读者可能具备一定的计算机知识和程序设计的概念，但对 Web 技术却缺乏了解。

另外，如果读者已经具备了编写程序的背景，并对计算机知识和 Web 技术有所了解。那么本书可以作为一个进入 Web 应用程序开发世界的快速通道。

对于所有的读者，希望本书物有所值。

本书内容

在本书中，我们将详细介绍什么是 JavaScript，以及 JavaScript 的基础语法。本书将详细介绍程序设计的基础概念，包括 JavaScript 语言的数据、数据类型、以及选择语句和循环语句等结构化程序设计的概念。

在学习了 JavaScript 语言的基础知识之后，本书将介绍 JavaScript 中的一个重要概念——对象。JavaScript 提供了很多内建对象，如 Date 对象和 String 对象等，这些内建对象为程序设计带来了很多好处，例如利用内建对象可以管理复杂的数据类型，并简化 JavaScript 应用程序的设计。本书还将介绍如何使用 JavaScript 操作浏览器提供的对象，如 form 对象、window 对象或其他的控件元素对象。使用这些知识，就可以创建具有专业水准的 Web 应用，并与用户进行交互。

哪怕对于一个程序设计方面的专家，随着代码长度的增加，错误也在所难免，JavaScript 编写的代码亦是如此。本书介绍了一些常见的语法错误和逻辑错误，还介绍了如何发现这些错误，以及如何使用 Microsoft 脚本调试器。本书还介绍了如何处理漏网之鱼的错误，并确保这些错误不会对用户的最终体验造成不良的影响。

随后，本书将介绍一些 JavaScript 的高级主题，例如如何使用 cookie，以及如何使用 DHTML 和 XML 来使 Web 页面变得活泼生动起来等。最后，本书还介绍了远程脚本和 Ajax 的概念，这是一种相对较新的令人激动的 Web 开发技术。远程脚本和 Ajax 技术允许 HTML 页面中的 JavaScript 代码直接与服务器进行通信，最有用的就是通过 Ajax 查询服务器上的数据库，以获取相应信息而无须刷新整个页面。Google 工具栏正是一个非常成功的 Ajax 技术案例。如果在浏览器中安装了 Google 工具栏，只须在 Google 工具栏中输入搜索关键字，Google suggestion 将立即提供相应的搜索建议，这些建议正是通过查询 Google 搜索数据库获取的。

对于本书中介绍的每一个新概念，都将以相应的实例加以说明。这些实例可以对所学的 JavaScript 原理进行实践和练习，以巩固所学的知识。本书各章的结尾都包含了一些习题，在附录中则提供了这些习题的参考答案。

在本书的前半部分中，将创建一个复杂的应用实例——“在线小测试”程序，该程序将演示如何将 JavaScript 应用在实际问题中。

如何使用本书

JavaScript 代码是纯文本的，因此要创建 JavaScript 程序，只须使用一个文本编辑器即可，例如 Windows 中的记事本，或者其他的文本编辑器即可。

另外，为了测试本书中的 JavaScript 代码，还需要一个支持较新版本 JavaScript 的浏览器。可以使用 IE 6 及以上版本的浏览器，或者 Firefox 1.5 以上的浏览器进行测试。本书中的代码在以上两种浏览器中都进行了详细的测试。大部分现代浏览器都支持 JavaScript，本书第 12 章、第 13 章中部分实例的代码被指定兼容特定的浏览器，以演示 DHTML 和 DOM 的脚本编程技术。除此之外，本书中的大部分代码都是跨浏览器兼容的，如果存在不能跨浏览器兼容的情况，本书将特别作出明确的说明。

如何下载本书的示例代码

在读者学习本书中的示例时，可以手工输入所有的代码，也可以使用本书附带的源代码文件。本书使用的所有源代码都可以从本书合作站点 www.wrox.com 或 www.tupwk.com.cn/downpage 上下载。登录到站点 www.wrox.com，使用 Search 框或使用书名列表就可以找到本书，接着单击本书细目页面上的 Download Code 链接，就可以获得所有的源代码。

提示：

许多图书的书名都很相似，所以通过 ISBN 查找本书是最简单的，本书的英文原版的 ISBN 是 978-0-470-05151-1。

在下载了代码后，只需用自己喜欢的解压缩软件对它进行解压缩即可。另外，也可以进入 www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果您在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者节省时间、避免阅读和学习受挫，当然，这还有助于提供更高质量的书籍。请给 wkservice@vip.163.com 发电子邮件，我们就会检查您的信息，如果是正确的，就把它发送到该书的勘误表页面上，或在本书的后续版本中采用。

要在网站上找到本书的勘误表，可以登录 www.wrox.com，通过 Search 框或书名列表查找本书，然后在本书的细目页面上，单击 Book Errata 链接。

p2p.wrox.com

P2P 邮件列表是为作者和读者之间的讨论而建立的。读者可以在 p2p.wrox.com 上加入 P2P 论坛。该论坛是一个基于 Web 的系统，用于传送与 Wrox 图书相关的信息和相关技术，与其他读者和技术用户交流。该论坛提供了订阅功能，当论坛上有新贴子时，会给您发送您选择的主题。Wrox 作者、编辑和其他业界专家和读者都会在这个论坛上进行讨论。

在 http://p2p.wrox.com 上有许多不同的论坛，帮助读者阅读本书，在读者开发自己的应用程序时，也可以从这个论坛中获益。要加入这个论坛，需执行下面的步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接。

(2) 阅读其内容，单击 Agree 按钮。

(3) 提供加入论坛所需的信息及愿意提供的可选信息，单击 Submit 按钮。

(4) 然后就会收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

提示：

不加入 P2P 也可以阅读论坛上的信息，但只有加入论坛后，才能发送自己的信息。

加入论坛后，就可以发送新信息，回应其他用户的贴子。可以随时在 Web 上阅读信息。如果希望某个论坛给自己发送新信息，可以在论坛列表中单击该论坛对应的 Subscribe to this Forum 图标。

对于如何使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作原理，以及许多针对 P2P 和 Wrox 图书的常见问题的解答。要阅读 FAQ，可以单击任意 P2P 页面上的 FAQ 链接。

目　录

第 1 章

Web 与 JavaScript 概述

第 1 章是本书的开篇，在本章中将介绍什么是 JavaScript、它有什么样的功能，以及开发 JavaScript 程序所需要的工具。了解这些基础知识之后，在本书其余章节中将逐步介绍如何使用 JavaScript 创建功能强大的 Web 站点。

对于学习编程的人来说，动手实践就是最好的学习方式。从本章开始，本书将带领你使用 JavaScript 创建大量非常实用的程序。在这一章中，我们将创建第一个 JavaScript 程序代码。

此外，在本书中，我们将展示一个名为“在线小测试”的 JavaScript Web 应用程序的完整开发过程，并通过一步一步地详细讲解来加深对如何创建 Web 应用程序的理解。在本章的末尾，我们将展示开发完成之后的“在线小测试”程序，并讨论一下该程序的设计思路。

1.1 JavaScript 简介

本小节简要介绍什么是 JavaScript、JavaScript 的历史、它的工作原理以及使用 JavaScript 能实现哪些有用的功能。

1.1.1 什么是 JavaScript

在购买本书之前，你也许已经知道 JavaScript 是一种计算机语言。但是，什么是计算机语言呢？简而言之，计算机语言就是一系列的指令，这些指令告诉计算机如何做事情。这些指令可以广泛地包含各种操作，包括显示文本信息、移动图片、或者请求用户输入信息等。通常，计算机自上而下地处理这些指令，这些指令也称为代码。简单地说，计算机阅读你编写的代码，计算出你想执行的操作，然后再执行这些操作。实际执行代码的过程被称为“运行代码”或“执行代码”。

在自然语言中，为了冲一杯速溶咖啡，可以采用如下的步骤，也可认为这是一种指令或代码：

(1) 将咖啡放入杯子中。

(2) 在烧水壶中装上水。

(3) 用烧水壶烧水。

(4) 如果水已经煮沸，则将水倒入咖啡杯，否则继续等待水煮沸。

(5) 品尝咖啡。

在冲咖啡的步骤中，你从第一行(指令 1)开始执行，接着执行第二行(指令 2)，然后是下一行，直到结束。这个执行步骤与大多数计算机语言的执行非常类似，JavaScript 代码的执行亦是如此。但是在某些情况下，你可能需要改变代码执行的顺序，或者跳过某些语句，在第 3 章中将对此做进一步的讲解。JavaScript 是一种解释型语言，而不是一种编译型语言。但解释型语言与编译型语言的含义是什么呢？

事实上，计算机并不真正理解 JavaScript。计算机需要某种解释程序来解释 JavaScript 代码并将其转换成计算机能理解的东西。因此，JavaScript 是一种解释型语言。本质上，计算机只能理解机器码，机器码实际上是由二进制数字组成的字符系列(即 0,1 的字符系列)。当浏览器遇到 JavaScript 时，就将 JavaScript 代码传递给一个称之为“解释器”的程序，解释器将 JavaScript 代码解释为计算机能够理解的机器码。这有点类似于请一个翻译将英语翻译为西班牙语。值得注意的是，解释和转换发生在代码运行时，并且每次运行都需要重复地进行解释和转换。解释型语言不仅包括 JavaScript，VBScript 也是一种解释型语言。

对于编译型语言，在程序运行之前，程序代码首先被转换成机器码，然后才实际运行，而且这个转换过程只需要执行一次。用于将程序员编写的代码转换成机器码的程序，就是通常所说的“编译器”。实际运行的代码就是编译器所产生的机器码。Visual Basic 和 C++ 语言都是编译型语言。如果将编译过程与现实世界做一下类比，这类似于让一个翻译将一份英文文档翻译成一份西班牙语的文档。如果英文文档没有发生变化，那么就可以一直使用翻译好的西班牙语的文档，而无须重新翻译。

此处，应该消除一个普遍的误解：JavaScript 并不是一种 Java 版的脚本语言。事实上，除了名字中都带有 Java 这个词外，二者并没有任何相同点。特别值得庆幸的是，相比 Java 语言，JavaScript 非常简单和易用。实际上，JavaScript 比大多数程序设计语言都要简单易学，但是它同样具有令人惊叹的强大功能。

1.1.2 JavaScript 与 Web

本书大部分实例的 JavaScript 代码都是包含在 Web 页面中，并由浏览器加载执行的。创建 Web 页只需要一个文本编辑器就可以了，例如 Windows 记事本。浏览页面则需要一个浏览器，例如 Firefox 或者 Internet Explorer，这些浏览器中都内置了 JavaScript 的解释器。

实际上，JavaScript 的第一个版本出现在 Netscape Navigator 2 浏览器中。最初，人们给 JavaScript 起的名字是 LiveScript。然而，由于当时 Java 技术正是如日中天，Netscape 公司决定将 LiveScript 命名为 JavaScript，以便使 JavaScript 更加引人注目。随着 JavaScript 的发展，微软公司决定在 IE 浏览器中加入微软品牌的 JavaScript，即 JScript。随后，Netscape、微软和其他公司都在自己最新的浏览器中不断发布 JavaScript 的最新版本。尽管这些不同公司、不同版本的 JavaScript 在很多方面是一致的，但是也存在一些差异，在编写程序时如果不小心的话，这些差异可能会导致一定的问题。本书前面几章所创建的代码适用于大多数浏览器，包括 Firefox 浏览器、Internet Explorer 浏览器以及 Netscape 浏览器。本书后面的章节将用到一些在 Firefox 1.5 或以上版本的浏览器、或者 Internet Explorer 6 和

Internet Explorer 7 浏览器中特有的新特性。在本章的末尾，将简要说明不同浏览器和不同版本的 JavaScript 可能导致的问题，以及如何处理这类问题。

本书提供了大量包含 JavaScript 代码的网页实例，它们可以存储在硬盘上，并可以用浏览器直接打开，就像打开普通的文件(例如文本文件)一样。但是，当浏览因特网上的网站时，浏览器却并不是这样加载页面的。因特网是一个巨大的计算机互连网络。网站是因特网上特定计算机提供的一种专门的服务，提供网站访问服务的计算机就是众所周知的 Web 服务器。

Web 服务器的最基本功能就是在硬盘上保存着大量的网页。通常，当另一台计算机上的浏览器向 Web 服务器请求一个保存在该服务器上的网页时，Web 服务器将从硬盘上载入该网页，并且通过一个称为超文本传输协议(Hypertext Transfer Protocol，HTTP)的专门的通信协议，将网页回传给发起请求的计算机。通常，我们将运行浏览器并发起请求的计算机称为客户机。客户机/服务器的关系有点像购物者与售货员的关系，当购物者进入商店并对售货员说：“请给我拿一件某样东西”。这时候，售货员将为购物者提供服务，找到购物者想要的东西并递给购物者。对于 Web 来说，运行浏览器的客户机就像一个购物者，而提供所请求网页服务的 Web 服务器就像一个售货员。

但是，当你在浏览器中输入一个网址时，浏览器怎么知道该向哪一个 Web 服务器请求页面呢？对于商店来说，它有类似“某市中央大街 45 号”这样的地址，但是 Web 服务器却没有这样的地址标志。Web 服务器使用的是 IP(Internet Protocol，因特网协议)地址，IP 地址唯一标识了因特网上的主机。一个 IP 地址是由句点(“.”)分隔的 4 组数字组成，例如：127.0.0.1。

如果你经常在网上冲浪，你可能会对上述内容感到不解。因为 Web 站点通常具有 www.somewebsite.com 这样好记的名字，而不是 IP 地址。实际上，像 www.somewebsite.com 这样的域名，仅仅是实际 IP 地址的一个好记的名字而已，字符串形式的域名比起数字形式的 IP 地址来说更便于人们记忆。域名解析服务器可以将域名解析为实际的 IP 地址，一般情况下，域名解析服务器已经由因特网服务提供商(ISP)在因特网上建立了。

本书的最后将介绍如何通过一个 Step by Step 方式的向导来建立你自己的 Web 服务器。Web 服务器并不仅仅是一个能回传页面给客户机的简单机器，实际上，Web 服务器可以使用 JavaScript 来做一些简单的处理。本书后面将对此作相应的介绍。

另外，在本书中，我们将把 Internet Explorer 浏览器简称为 IE。

1.1.3　为什么选择 JavaScript

脚本语言不仅只有 JavaScript 一种，还有其他的脚本语言，如 VBScript 和 Perl 语言等。为什么我们偏偏选择 JavaScript 而不是其他呢？

选择 JavaScript 的主要原因在于 JavaScript 使用广泛，并且非常实用。两大主流的浏览器，IE 和 Firfox，都支持 JavaScript，大部分其他的现代浏览器也都支持 JavaScript。所以，当人们访问你的网站时，客户端的浏览器中都已经安装了某一版本的 JavaScript。当然，人们也可以通过设置浏览器的选项来禁用 JavaScript。

对于上面提到的其他脚本语言，例如 VBScript，它可以实现与 JavaScript 完全相同的

功能，但是只有 Windows 操作系统上的 IE 浏览器支持 VBScript。而 Perl 语言则根本不在 Web 浏览器中使用。

JavaScript 不仅能在网页中使用，它还有很多方面的用途。例如，JavaScript 还可用于 Windows 系统中的自动化计算机管理任务；在 Adobe Acrobat 的.pdf 文件中也可以使用一个简化版的 JavaScript，用于控制页面的显示，就像控制网页一样。也许对于哪一种脚本语言更有用或者更强大这样的问题，并没有唯一正确的答案。VBScript 几乎能完成绝大部分 JavaScript 能完成的工作，反之亦然。

1.1.4 JavaScript 的功能

通常，我们使用 JavaScript 与用户进行交互、获取用户输入的信息，以及对数据进行校验。例如，你可能会在页面上放置一个下拉列表作为菜单，并希望当用户在列表中选择一个项后，将用户重定向到与该选项所对应的页面。然而，下拉列表只是普通的 HTML 标记，无法完成这样的功能，这时候，要实现根据用户的选择来跳转页面的功能，就需要使用 JavaScript 脚本进行处理。表单(form)通常用于获取用户输入的信息，这是使用 JavaScript 与用户交互的典型场合。表单中的元素通常都是一些普通的 HTML，那么我们怎么才能实现对用户输入的数据进行检查呢？例如，在一个在线购物的网站中，有一个要求用户填写详细信用卡信息的表单，在成交之前，你就需要确认用户是否填入了这些详细信息。有时，还需要对输入数据的类型进行检查，例如，对于年龄，应该输入表示年龄的数值，而不是其他字符串。通过 JavaScript 脚本，我们就能实现这些检查。

使用 JavaScript 还可以实现各种各样的特殊效果。例如，当用户的鼠标经过一个图片时，将图片切换成另一幅图片，这是在网页中比较常见的一种特效。另外，也许你已经见过在浏览器状态栏(通常位于浏览器窗口的底部)中显示滚动信息的特效，或者在页面上显示滚动信息的特效，但它们是如何实现的呢？在本书中将用 JavaScript 展示这种特效的实现。当用户鼠标经过一个菜单项时，将该菜单下的子菜单展开，以提供一个列表供用户选择，这也是由 JavaScript 驱动的一种比较常见的特效。

特效仅仅是 JavaScript 功能的一方面，更具有实用价值的是 JavaScript 小应用程序。例如：在一个当铺的网站上，包含一个用 JavaScript 编写的典当计算器，或者在一个关于财务计划的网站上使用 JavaScript 计算税务清单。只须发挥一下想象力，JavaScript 就能为你创造出许多惊喜。

1.2 创建 JavaScript Web 应用程序所需的工具

要为 Web 应用程序创建 JavaScript 代码，仅需要一个简单的文本编辑器，例如 Windows 记事本，或者某种提供了行号、查找和替换等功能的更高级的文本编辑器。也可以使用功能强大的专用 HTML 编辑器，我们可以在 HTML 编辑器中编辑 HTML 的源文件，并在其中加入相应的 JavaScript 代码。有许多非常优秀的软件工具是专门针对 Web 应用程序的开发而设计的，例如 Adobe 公司的 Dreamweaver 8 就非常好用。但是，本书将着重介绍 JavaScript 本身，而不是某一特定的开发工具。在学习 JavaScript 基础知识的过程中，手工

编写代码往往比依赖于开发工具更有收获。在程序开发中，许多高级的内容和逻辑是开发工具本身不能解决的，亲自动手编写代码将有助于提高你对基础知识的理解。在深入掌握了 JavaScript 的知识后，使用开发工具可以提高开发效率，节省时间，并使你有更多的时间和精力去关注那些高级和有价值的内容。

浏览器用于浏览网页。如果你希望用户使用某种指定的浏览器来访问你的网站，则最好在这种浏览器环境下进行 JavaScript 开发。现在已经有几种不同版本的 JavaScript，且不同版本的浏览器所支持的 JavaScript 版本也不相同。虽然这些不同版本的 JavaScript 其核心是相同的，但是各版本之间具有不同的语言扩展。本书中的所有实例都已经在 Firefox 1.5 和 IE 6 、IE 7 浏览器中通过了调试。无论何处，如果出现了与这些浏览器不兼容的代码，本书都用文字对这些不兼容的代码的影响作了说明。

如果你使用的是 Windows 系统，则绝大部分情况下都已经安装了 IE 浏览器。如果未能安装，请访问如下网址以获取最新版本的 IE 浏览器：www.microsoft.com/windows/ie/default.mspx。

Firefox 浏览器则可以到 www.mozilla.com/firefox/all.html 下载。在安装 Firefox 时，最好选择自定义安装，这样可以选择安装哪些组件。特别值得安装的是 Developer Tools 组件，虽然它不是必须安装的，但对一个开发人员来说它是一个值得拥有的扩展工具。

即使你的浏览器支持 JavaScript，但是仍然可以通过浏览器的设置禁用 JavaScript 功能。在下一节的内容中，你将开始编写第一个 JavaScript 程序，所以首先应该检查一下你的浏览器是否允许使用 JavaScript。

对于 Firefox 浏览器，请从浏览器的 Tools 菜单下，选择 Options 子菜单。在弹出的窗口中，单击 Content 标签。在这个标签页中，请确认 Enable JavaScript 复选框已经被选中，如图 1-1 所示。

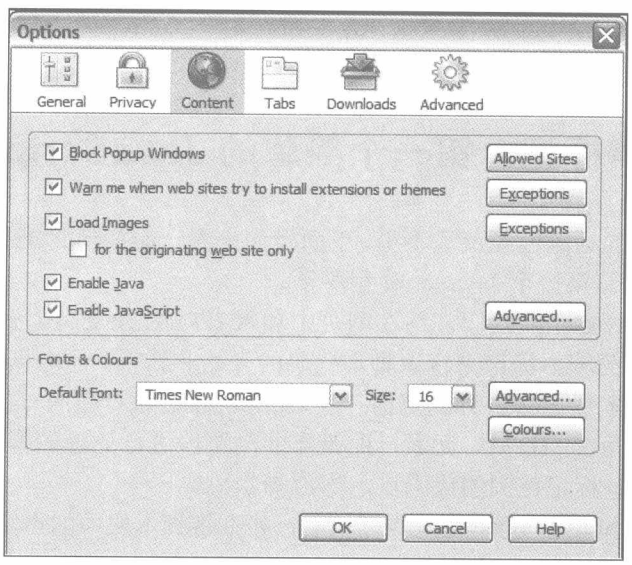

图　1-1

关闭或打开 IE 浏览器的脚本功能稍微有点麻烦。从 IE 浏览器的 Tools 菜单下，选择 Internet Options 子菜单。在弹出的窗口中，选择 Security 标签，对于 Internet 和 Local Intranet 两项，都可以进行自定义安全设置。要在浏览器中启用 JavaScript 功能，请在 Internet 和 Local Intranet 两项下，单击 Custom Level 按钮，在弹出的窗口中找到 Scripting 一节，将 Active Scripting 一项设置为 Enable。

下面我们再说明一下如何用浏览器来打开本书中的实例代码。实际上，对于本书中的代码，只需简单地从硬盘上打开代码文件就可以了。打开代码文件有很多种方法，比如在 IE 6 中，可以从浏览器的 File 主菜单下选择 Open 子菜单，单击 Browse 按钮，找到你所要打开的文件就可以了。类似地，在 Firefox 浏览器中，也是在 File 主菜单下选择 Open File 子菜单，然后找到你要打开的文件，再单击 Choose File 按钮即可。

对于新版本的 IE 7 浏览器，它具有一个新的菜单结构，菜单中没有 Open File 选项。你可以直接在浏览器的地址栏中输入盘符和冒号，例如，在地址栏中输入"C:"，以找到要打开的文件。另外，也可以将 IE 7 的菜单切换回低版本 IE 浏览器的经典菜单模式(如图 1-2 所示)。要切换到经典菜单模式，请在 IE 7 的 Tools 菜单下选择 Toolbars 子菜单，并选择 Classic Menu 选项。

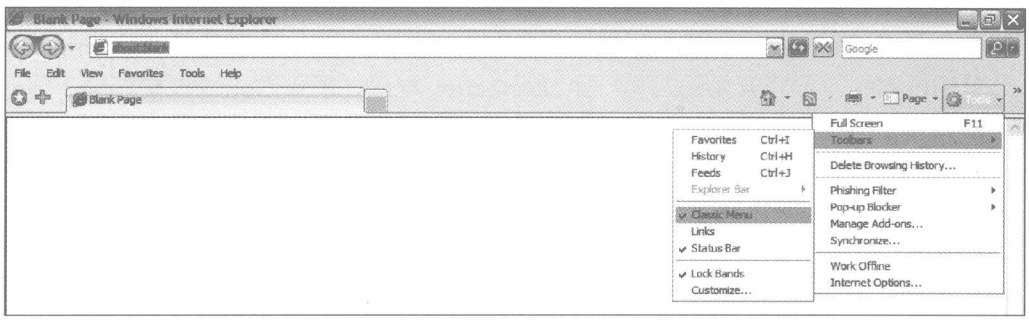

图 1-2

1.3 <script>标记：第一个简单的 JavaScript 程序

关于 JavaScript 的话题已经谈了不少，下面我们来看一看如何将 JavaScript 加入到网页中。马上你就可以编写第一个 JavaScript 程序了。

在网页中加入 JavaScript 代码，与在网页中加入 HTML 元素时需要使用成对的标记一样，也需要使用成对的脚本标记来标识脚本代码的开始和结束。这对标记就是<script>标记和</script>标记。脚本标记将告诉浏览器，在<script>标记和对应的</script>标记之间的代码块，并不是用于显示的 HTML 元素，而是需要浏览器进行处理的脚本代码。由<script>标记和对应的</script>标记包围的代码块，称为脚本块。

通常情况下，当浏览器遇到<script>标记后，并不将脚本块中的内容显示给用户，而是使用浏览器内建的 JavaScript 解释器来解释并运行脚本代码中的指令。虽然脚本代码中的指令有可能改变页面的显示方式，或者改变显示的内容，但是脚本自身的代码永远不会显示给用户。