

A Programmer's Guide on JavaScript

动态网页编程 实例手册

策划 / WISBOOK 海洋智慧图书

编著 / 张长富 黄中敏

- ◆ 在标题栏和状态栏中显示跑马灯效果
- ◆ 利用框架构造菜单及其对应内容
- ◆ 构造常用“调查表”表单
- ◆ 文本框、单选框、复选框、密码框、选项列表框的应用
- ◆ 检测用户输入数据的有效性
- ◆ 用幻灯片形式显示图像
- ◆ 动态更换图像、图像翻转效果、随机更换图像
- ◆ 实现增强鼠标指针功能
- ◆ 构造用户访问计数器
- ◆ 嵌入方式样式表、行内方式样式表、外部方式样式表应用
- ◆ 控制重叠对象前后位置
- ◆ 实现下拉菜单和即时提示
- ◆ 更多精彩内容请见本书……



本书源代码下载

<http://www.wisbook.com/down/JavaScript.exe>



海洋出版社

A Programmer's Guide on JavaScript

动态网页编程 实例手册

策划 / WISBOOK 海洋智慧图书
编著 / 张长富 黄中敏

海洋出版社

北京

内 容 简 介

这是一部关于如何用 JavaScript 语言来进行动态网页编程的实例手册。

本书内容：全书由 16 章和 3 个附录构成。主要内容包括 JavaScript 功能简介、编程环境、数据类型和变量、与用户交互的基本方法、运算符、条件和循环语句、函数、创建和使用自定义对象、JavaScript 核心对象、浏览器对象模型、访问文档内容、事件响应与编程、使用 Cookie 存储网页状态、活用正则表达式、动态 HTML、JavaScript 和 XML。附录 A 为 JavaScript 中常用对象的属性、方法和事件，附录 B 为 JavaScript 预定义颜色表，附录 C 为自测练习题。

本书特点：全书以示例教学方式来组织内容，将 JavaScript 编程的基础知识有机地与控制鼠标特效、导航栏中的下拉菜单、设置浏览器状态条、复杂的动态菜单和表单的自动填充等 200 多个程序示例相结合，内容丰富、讲解细致、由浅入深，集趣味性、直观性、可操作性于一体，帮助读者快速掌握动态网页编程的方法和技巧。

读者对象：从事动态网页编程的广大程序员的必备手册，高等院校网页编程专业配套教材。

说明：书中所有示例均已上机调试过，源代码及每章练习题答案可以登录海洋出版社智慧图书网站 (<http://www.wisbook.com/download/JavaScript.exe>) 免费下载。

图书在版编目(CIP)数据

JavaScript 动态网页编程实例手册/张长富，黄中敏编著. —北京：海洋出版社，2005.8
ISBN 7-5027-6344-9

I. J… II. ①张…②黄… III. JavaScript 语言—程序设计—手册 IV. TP312-62

中国版本图书馆 CIP 数据核字 (2005) 第 065746 号

总 策 划：WISBOOK

责任编辑：黄梅琪 王红卫 钱晓彬

责任校对：肖新民

责任印制：肖新民 阎秋华

排 版：海洋计算机图书输出中心 晓阳

出版发行：海洋出版社

地 址：北京市海淀区大慧寺路 8 号 (716 房间)
100081

经 销：新华书店

技术支持：www.wisbook.com/bbs

发 行 部：(010) 62132549 62112880-878

62174379 (传真) 86489673 (小灵通)

网 址：www.wisbook.com

承 印：北京时事印刷厂

版 次：2005 年 8 月第 1 版

2006 年 8 月北京第 2 次印刷

开 本：787mm×1092mm 1/16

印 张：31 彩插 1 页

字 数：705 千字

印 数：5001~7000 册

定 价：46.00 元

本书如有印、装质量问题可与发行部调换

写在前面的话

当前我国正向现代化、信息化、工业化的国家大步迈进,人才资源自然是最为重要的资源。社会各行业、工业企业等部门人才短缺、特别是技能型人才严重短缺,在某种程度上已经影响和妨碍了现代化建设的发展。近年来,我国的职业教育已日益被经济建设所依赖,技能型人才需求存在巨大缺口,因此培养培训任务迫在眉睫。

为配合国家《2003—2007 教育振兴行动计划》,推动“职业教育与培训创新工程”的健康发展,中国计算机学会职业教育专业委员会根据中国计算机学会 2004 年召开的“第八届全国会员代表大会”的精神,努力为落实和推进“职业教育与培训创新工程”做出贡献。

培养培训软件技术专业领域技能型紧缺人才是职业教育的根本使命和当前的紧迫任务,目的就是要刻不容缓地把这些走进校园的学生培养成适合国家发展和企业需要的有用人才,培养他们成为有一技之长的劳动者和实用型人才,培养的目的主要是面向就业。

本套教材就是面对目前全国职业院校学生的现状和职业需求而编写的、颇具特色的实用培养培训教材,以配合教育部、劳动与社会保障部、国防科工委、信息产业部、交通部、卫生部联合颁发的《教育部等六部门关于职业院校制造业和现代服务业技能型紧缺人才培养培训工程的通知》而行动。通知中同时颁发了教育部办公厅和信息产业部办公厅《关于确定职业院校开展计算机应用与软件技术专业领域技能型紧缺人才培养培训工作的通知》,在这个通知中,颁发了《职业院校计算机应用与软件技术专业领域技能型紧缺人才培养指导方案》。

根据以上精神和指导方案,中国计算机学会职业教育专业委员会与海洋出版社海洋智慧图书有限公司,特组织北京、河北、大连、长春、唐山、武汉、深圳、肇庆和杭州等地主要职业院校负责人和一线教师,召开教材研讨会,相互交流经验,介绍需求,共同策划和编写了本套《全国计算机应用与软件技术专业领域技能型紧缺人才培养培训教材》,倾心奉献给全国广大的教师和学生,为国家“职业教育与培训创新工程”推波助澜,为满足社会巨大的人才培养需求做出应有的贡献!

整套书的编写宗旨

- 三符合:符合教育部教学大纲、符合市场技术潮流、符合职业院校专业课程需要。
- 技术新、任务明、步骤细致、实用性强,专为技能型紧缺人才量身定制。
- 软件功能与具体范例操作紧密结合,边讲解边动手,学习轻松,上手容易。
- 三适应:适应新的教学理念、适应学生水平现状、适应用人标准要求。

整套书的特色

- 理论精练够用、任务明确具体、技能实操落实,活学活用。

教材编委会

前 言

在今天的互联网领域中, JavaScript 可以说无处不在。如果你在 Google 搜索引擎 (www.google.com) 中输入关键词 “JavaScript”, 那么会得到数以万计个查询结果。据了解, 现在的互联网中大约有 25% 的页面包含 JavaScript 代码, 使用 JavaScript 能完成最基本的编程效果, 比如控制鼠标特效、导航栏中的下拉菜单、设置浏览器状态条、复杂的动态菜单和表单的自动填充等。JavaScript 脚本语言在 Web 浏览器环境中的统治地位不可动摇。

JavaScript 成功的主要原因在于它的易用性。例如, 一条 JavaScript 语句就可以生成一个有趣的浏览器特效, 建立一个完全交互的浏览器环境也仅需要几行代码。

时下, JavaScript 与 C++ 和 Visual Basic 一样被广泛应用。例如, 微软在它的 .NET 开发环境中包含了 JavaScript 2.0 的一个版本, 称作 JScript.NET。JScript.NET 几乎可以访问所有的 .NET 框架类, 并能够创建运行在你的计算机上的窗口应用程序。事实上, 要使用的下一代文字处理程序或许完全由 JavaScript 代码来编写。

笔者在开发一些项目时, 经常有学生对要求学好 JavaScript 感到困惑: “不是已经有 Dreamweaver 了吗? 还要学 JavaScript 干什么?” 我的回答是, 虽然已经有 Dreamweaver 了, 但是请打开它的对话框看一看, 做动态 Web 页面时需要什么, 尤其是制作适应多种浏览器的网页时需要什么? 经过一段时间的实践, 他们终于明白了, 要成为一个 Web 页面制作高手, 掌握 JavaScript 几乎是不二法门。

在为初学者选择 JavaScript 参考书时, 也让笔者颇费心思。市场上有各种各样关于 JavaScript 的图书, 有些书籍太注重于技术层面, 缺少图样和示例; 有些书籍则只有示例, 而缺少语言要点的讲解, 不太适合初学者的阅读和使用。这让笔者萌发了一种将两者的长处结合起来编写一本适合初学者学习的参考书的动机。编写本书的目的是期望读者学会编写程序, 而不是学会粘贴程序, 帮助初学者迅速掌握 JavaScript 的核心内容和技术要素, 并应用到实际项目开发中。

本书由张长富和黄忠敏编写, 在编写过程中得到了众多专家和好友的支持, 参与本书编写、整理、资料搜集工作的有李匀、刘民、张建安、杨文保、孙兵、马静静、姚铂、崔立模、陈聪、马向群、王志宏等。享受政府特殊津贴的高级工程师张福林先生对本书提出了许多指导性的意见和建议, 在此致以衷心的感谢! 另外, 还要特别感谢海洋出版社的编辑吴清平、王红卫、黄梅琪等, 是他们的辛勤劳动才让本书得以面世。由于作者水平有限, 热忱欢迎读者批评指正, 可将对本书的意见或建议发送至 chfuzh@263.net 信箱。

书中所有示例均已上机调试过, 源代码及每章练习题答案可以登录海洋出版社智慧图书网站 (<http://www.wisbook.com/down/JavaScript.exe>) 免费下载。

编 者

目 录

第 1 章 JavaScript 简介	1
1.1 什么是 JavaScript	1
1.2 JavaScript 功能简介	3
1.3 JavaScript 的版本以及支持它的浏览器	3
1.3.1 版本情况	3
1.3.2 JavaScript 的发展简史	4
1.3.3 JavaScript 的标准化	6
1.4 JavaScript 和事件	6
1.5 在哪里编写 JavaScript 代码	7
1.6 使用包含在外部文件中的 JavaScript 代码	9
1.7 如何兼容支持和不支持 JavaScript 的浏览器	10
1.7.1 使用 HTML 注释	10
1.7.2 使用<noscript>标记	12
1.8 练习题	13
第 2 章 JavaScript 基础	14
2.1 JavaScript 代码的编写和执行	14
2.2 语法规则	15
2.2.1 大小写	15
2.2.2 代码书写格式	16
2.2.3 保留字	16
2.2.4 注释	16
2.2.5 分号和语句结束符	17
2.2.6 <script>标记	17
2.3 动态生成页面内容	19
2.3.1 字符串及其拼接方法	19
2.3.2 document 对象的 write()和 writeln()方法	20
2.4 代码的调试方法与技巧	21
2.4.1 错误类型	22
2.4.2 调试工具	22
2.4.3 预防错误的方法	24
2.5 练习题	25
第 3 章 数据类型和变量	26

3.1 数据类型	26
3.1.1 基本数据类型	26
3.1.2 复合数据类型	31
3.2 变量	31
3.2.1 变量名称	32
3.2.2 变量声明和变量初始化	32
3.2.3 变量类型的动态变化	34
3.2.4 变量的作用域	35
3.3 避免错误与释放空间	35
3.3.1 避免或减少错误的途径	35
3.3.2 空间释放问题	36
3.4 练习题	36
第 4 章 与用户交互的基本方法	37
4.1 显示信息对话框的 alert()方法	37
4.2 显示确认选择对话框的 Confirm()方法	39
4.3 显示提示对话框的 Prompt()方法	40
4.4 练习题	42
第 5 章 运算符	43
5.1 运算符和表达式概述	43
5.2 赋值运算符	44
5.3 算术运算符	44
5.3.1 基本算术运算符	44
5.3.2 各种赋值运算符	45
5.3.3 增量/减量运算符	47
5.3.4 字符串拼接运算符	48
5.4 关系运算符	49
5.5 逻辑运算符	52
5.5.1 逻辑与运算符&&	53
5.5.2 逻辑或运算符 	54
5.5.3 逻辑非运算符!	55
5.6 位运算符	56
5.7 其他运算符	58
5.8 运算符的优先级和结合方式	59
5.9 数据类型转换	62
5.9.1 基本数据类型转换	62

5.9.2 提取整数的 parseInt()方法.....64	8.2.1 使用构造函数创建对象.....106
5.9.3 提取浮点数的 parseFloat()方法.....65	8.2.2 对象的属性和方法.....108
5.9.4 用于执行语句和计算表达式的 eval()方法.....66	8.3 定义对象.....109
5.10 练习题.....68	8.3.1 new()运算符和 Object()构造函数.....109
第6章 条件和循环语句.....70	8.3.2 通过函数创建对象.....110
6.1 条件语句.....70	8.3.3 定义对象方法.....111
6.1.1 if...else 条件语句.....70	8.4 定义文字量对象.....114
6.1.2 Switch 条件语句.....74	8.5 使用和访问对象.....116
6.2 循环语句.....77	8.5.1 使用 with 关键字访问对象成员.....116
6.2.1 while 语句.....77	8.5.2 使用 for...in 循环操作对象.....117
6.2.2 do...while 语句.....79	8.6 练习题.....119
6.2.3 for 语句.....80	第9章 JavaScript 核心对象.....121
6.2.4 for...in 语句.....81	9.1 数组对象.....121
6.2.5 控制循环执行过程——break 和 continue 语句.....83	9.1.1 声明数组.....121
6.2.6 循环的嵌套.....84	9.1.2 使用 for 循环为数组元素赋值.....123
6.2.7 循环的优化.....86	9.1.3 创建数组的同时为数组元素赋初值.....125
6.3 练习题.....87	9.1.4 使用字符串作数组下标.....126
第7章 函数.....89	9.2 创建多维数组.....127
7.1 什么是函数.....89	9.3 数组的属性与方法.....129
7.2 定义函数和使用函数.....89	9.3.1 数组的属性.....130
7.3 函数参数的传递.....92	9.3.2 数组的方法.....131
7.4 在多个位置调用函数.....93	9.4 使用 Math 对象完成数学计算.....138
7.4.1 在事件响应中调用函数.....93	9.4.1 数值的截断运算.....139
7.4.2 通过链接调用函数.....95	9.4.2 生成随机数.....141
7.5 函数中变量的作用域与函数的返回值.....96	9.4.3 其他常用数学函数.....142
7.5.1 函数中变量的作用域.....96	9.5 运用 Date 对象操作日期和时间.....143
7.5.2 函数的返回值.....97	9.5.1 Date 对象的方法.....144
7.6 函数的高级用法——递归与嵌套.....99	9.5.2 日期时间的运算.....147
7.6.1 递归函数.....99	9.5.3 扩展 Date 对象的功能.....148
7.6.2 函数的嵌套.....102	9.6 基本数据类型的对象表示.....150
7.7 编写和使用函数过程中常见的错误.....104	9.6.1 Number 对象.....150
7.8 练习题.....105	9.6.2 Boolean 对象.....153
第8章 创建和使用自定义对象.....106	9.6.3 String 对象与字符串处理.....155
8.1 对象的概念.....106	9.7 函数对象 (Function 对象).....161
8.2 对象创建与对象的属性和方法.....106	9.8 练习题.....163

第 10 章 浏览器对象模型	164
10.1 浏览器对象模型 (Browser Object Model)	164
10.2 navigator 对象	165
10.2.1 navigator 对象的属性和方法	165
10.2.2 检测浏览器及其特性	166
10.3 管理浏览器窗口	168
10.3.1 window 对象的属性与方法	168
10.3.2 打开和关闭窗口	170
10.3.3 移动和缩放窗口	173
10.3.4 定时完成指定任务	175
10.3.5 滚动文档内容	179
10.3.6 运用框架	181
10.3.7 创建菜单	185
10.4 location 对象	189
10.5 history 对象	191
10.6 screen 对象	193
10.7 练习题	195
第 11 章 访问文档内容	196
11.1 文档对象模型 (DOM)	196
11.1.1 文档对象模型概述	196
11.1.2 document 对象的层次结构	197
11.1.3 document 对象的属性和方法	198
11.2 表单 (form) 应用	204
11.2.1 表单基础	204
11.2.2 form 对象	208
11.2.3 使用名称引用 form 对象及输入域对象	211
11.2.4 使用表单输入域	213
11.2.5 检测用户输入数据的有效性	230
11.3 链接编程	235
11.3.1 URL 的形式和链接中调用 JavaScript 代码	235
11.3.2 链接对象 links	236
11.4 图像编程	238

11.4.1 HTML 图像标记概述	238
11.4.2 使用 JavaScript 操作 image 对象	239
11.5 练习题	246
第 12 章 事件响应与编程	247
12.1 什么是事件和事件处理程序	247
12.2 使用代码触发事件	251
12.3 响应窗口或框架的事件	253
12.3.1 响应 onLoad 和 onUnload 事件	253
12.3.2 响应 onFocus 和 onBlur 事件	256
12.3.3 使用 focus() 和 blur() 方法改变输入焦点	258
12.4 响应鼠标事件	260
12.4.1 使用鼠标事件	260
12.4.2 实现图像反转	263
12.4.3 显示幻灯片	264
12.5 响应键盘事件	266
12.6 响应链接事件	269
12.7 响应表单输入域的事件	270
12.7.1 按钮事件处理	271
12.7.2 输入域的 onClick 事件	273
12.7.3 表单元事件和事件方法的联合应用	274
12.7.4 响应 onFocus 和 onBlur 事件	276
12.7.5 响应 onChange 事件	277
12.7.6 运用 onSubmit 事件	279
12.8 出错处理——响应 onError 事件	282
12.9 深入控制事件——event 对象	283
12.9.1 srcElement 和 type 属性	284
12.9.2 clientX 和 clientY 属性	286
12.9.3 offsetX 和 offsetY 属性	287
12.9.4 button、altKey、ctrlKey、shiftKey 属性	288
12.9.5 keyCode 属性	290
12.10 练习题	293

第 13 章 使用 Cookie 存储网页状态.....295

- 13.1 什么是 Cookie295
- 13.2 Cookie 的组成297
 - 13.2.1 Cookie 的语法格式297
 - 13.2.2 Cookie 属性详解297
- 13.3 使用 Cookie298
 - 13.3.1 Cookie 对象299
 - 13.3.2 创建 Cookie299
 - 13.3.3 读取 Cookie304
 - 13.3.4 删除 Cookie305
- 13.4 Cookie 的高级用法307
 - 13.4.1 构造操作 Cookie 的函数库.....308
 - 13.4.2 检测 Cookie 是否可用311
 - 13.4.3 显示用户访问网页的次数313
 - 13.4.4 在网页之间传递数据315
- 13.5 练习题318

第 14 章 活用正则表达式.....320

- 14.1 什么是正则表达式320
 - 14.1.1 正则表达式的概念320
 - 14.1.2 正则表达式的用途321
- 14.2 创建正则表达式322
 - 14.2.1 使用文字量方法创建正则表达式.....322
 - 14.2.2 使用构造函数 RegExp()创建正则表达式.....322
 - 14.2.3 正则表达式对象的方法323
 - 14.2.4 正则表达式对象的属性326
- 14.3 使用正则表达式的字符串对象的方法328
 - 14.3.1 match()方法329
 - 14.3.2 replace()方法330
 - 14.3.3 search()方法331
 - 14.3.4 split()方法332
- 14.4 运用正则表达式的元字符335
 - 14.4.1 正则表达式元字符、元符号分类列表.....335
 - 14.4.2 圆点 (.) 元字符337
 - 14.4.3 字符类匹配338
 - 14.4.4 表示重复次数的元字符的使用.....344

- 14.4.5 使用定位元字符在指定位置进行匹配350
- 14.4.6 多重选择替换匹配.....354

14.5 正则表达式在用户输入有效性检查方面的应用355

- 14.5.1 检测邮政编码.....355
- 14.5.2 检测保单号码.....357
- 14.5.3 删除多余字符.....358
- 14.5.4 检查电子邮件地址的有效性360

14.6 练习题.....362**第 15 章 动态 HTML**.....363

- 15.1 什么是动态 HTML363
- 15.2 定义和使用样式表.....363
 - 15.2.1 CSS 的概念364
 - 15.2.2 使用样式表.....364
 - 15.2.3 常用的样式表属性.....368
- 15.3 样式表的不同定义方式和运用.....377
 - 15.3.1 嵌入方式样式表.....377
 - 15.3.2 行内方式样式表.....379
 - 15.3.3 外部方式样式表.....380
 - 15.3.4 创建和使用样式类.....384
 - 15.3.5 创建和使用 ID 选择符386
 - 15.3.6 使用标记控制局部文字格式388
 - 15.3.7 继承选择符.....389
 - 15.3.8 元素定位和分层.....392
 - 15.3.9 使用<div>标记构造文档骨架396
 - 15.3.10 控制重叠对象的前后位置.....398
- 15.4 使用 JavaScript 代码操作和改变样式399
 - 15.4.1 深入了解文档对象模型 (DOM)400
 - 15.4.2 使用 W3C DOM 访问 HTML 元素404
 - 15.4.3 style 对象407
 - 15.4.4 DOM 与事件处理414

15.4.5 动态重叠的实现	417
15.4.6 实现下拉菜单和即时提示	419
15.5 练习题	424
第 16 章 JavaScript 和 XML	425
16.1 什么是 XML 和 XHTML	425
16.1.1 什么是 XML	425
16.1.2 什么是 XHTML	427
16.2 XML 基础	428
16.2.1 XML 概要知识	428
16.2.2 一个完整的 XML 示例	432
16.3 访问 XML 文档	436
16.4 使用 XML 数据岛	437
16.5 按名称访问 XML 文档中的元素	441
16.6 得到 XML 元素的属性值	442
16.7 用代码解析 XML 文档	443
16.8 处理 XML 文档事件	447
16.9 评估 XML 文档的有效性	449
16.10 应用 XSLT	451
16.11 练习题	456
附录 A JavaScript 中常用对象的属 性、方法和事件	457
附录 B JavaScript 预定义颜色表	464
附录 C 自测练习题	468
参考文献	476

第 1 章 JavaScript 简介

本章重点

- 什么是 JavaScript
- JavaScript 功能简介
- 支持 JavaScript 的浏览器
- JavaScript 中的事件
- 在哪里编写 JavaScript 代码
- 使用包含在外部文件中的 JavaScript 代码
- 如何兼容支持和不支持 JavaScript 的浏览器

1.1 什么是 JavaScript

JavaScript 是一种通用的脚本编程语言，也是一种基于对象（Object）和事件驱动（Event Driven）并具有安全性能的脚本语言。JavaScript 代码嵌入在 HTML 页面中，它把静态页面转变成支持用户交互并响应相应事件的活页面。现在，很难找到一个不包含 JavaScript 代码的商业站点页面，在其他类型的站点页面中，也可能找到 JavaScript 代码。图 1-1 是微软公司网站上的一个中文页面，查看它的源代码就会发现该页面中多处使用了 JavaScript 代码。



图 1-1 使用 JavaScript 代码创建的动态网页

JavaScript 最早称作 LiveScript, 由 Netscape Communications 和 Sun Microsystems 联合开发, 并与 Netscape Navigator 2.0 beta (测试) 版同时发行。JavaScript 是让静态 Web 页面动起来的一种编程语言。JavaScript 能够在许多平台上运行, 并独立于硬件。JavaScript 直接包含在常见的浏览器中, Netscape 和 Microsoft Internet Explorer 是支持 JavaScript 脚本语言的两个最常用的浏览器。在语法结构上, JavaScript 与 C、Java、Perl 编程语言十分相似, 比如, if、while 循环、for 循环语句的结构和意义与 Perl 语言中的对应语句几乎完全一样。JavaScript 是一种面向对象的解释性语言, 而不是像 C、C++、Visual Basic、Java 这样的编译语言, 因此也就不存在编译器的问题。

JavaScript 与 HTML 关系密切, 它们都由浏览器来处理。HTML 通过浏览器的组网库和图形子系统处理和展示, 而 JavaScript 程序则通过浏览器中内置的 JavaScript 解释器来执行。浏览器处理 Web 页面的一般过程为: 浏览器请求一个 Web 页面, Web 服务器则把整个页面的内容发送给浏览器, 包括 HTML 和 JavaScript 语句。之后, 浏览器加载 Web 页面, 在加载过程中, 浏览器读取 Web 页面的 HTML 语句, 并逐行进行翻译, 直到遇到 JavaScript 代码中的开始标记时, JavaScript 解释器接管页面的处理过程并继续进行处理, 到达 JavaScript 代码中的结束标记时, JavaScript 解释器将处理控制权交付出去, 浏览器继续处理 HTML 内容。

由浏览器处理的 JavaScript 称作客户端 JavaScript。虽然 JavaScript 主要用于客户端的脚本编程, 但这不是 JavaScript 的惟一用法, 它也能够用于除 Web 浏览器之外的其他环境中。比如, Netscape 创建了一种用于在服务器端编程的 JavaScript, 则它可以看成是像 Perl 或 ASP 一样的 CGI 语言。但是, 在本书中, 我们不讨论服务器端的 JavaScript, 而把精力集中在 JavaScript 中的一种最广泛的应用形式——客户端的 JavaScript 编程。

很多人将 JavaScript 误认为是 Java, 特别是初学者。事实上, JavaScript 不是 Java, 它们分别由两家不同的公司开发, JavaScript 由 Netscape 公司开发, Java 则由 Sun Microsystems 公司开发; Java 应用程序独立于 Web 页面, 或者说与 Web 页面无关, 而 JavaScript 程序则嵌入在 Web 页面中, 并且必须在浏览器中运行; Java 是一种强类型的语言, 而 JavaScript 中的数据类型则灵活多变; Java 语言中使用变量之前必须首先声明变量, 而 JavaScript 中各种类型的变量可以事先不声明而直接使用; Java 程序最终需要编译后才能运行, 而 JavaScript 程序则由内置在浏览器中的 JavaScript 引擎解释执行, 不需要进行编译。

同样, JavaScript 也不是 HTML, 但 JavaScript 代码可以嵌入在 HTML 文档中, 包含在 HTML 标记内。JavaScript 有自己的语法规则, 并且必须以符合这些规则的形式书写 JavaScript 代码。JavaScript 不能够理解 HTML 语言, 但能够在 JavaScript 的语句中包含 HTML 内容。在本书随后逐步深入的介绍中, 读者会深入体会 JavaScript 的这些特点。

JavaScript 是一种基于对象的编程语言, 而不是面向对象的语言。其原因在于 JavaScript 不支持其他面向对象语言所支持的一些面向对象的特征, 比如, JavaScript 不支持类继承机制, 它也没有私有 (Private)、保护 (Protected)、公有 (Public) 这样的概念, 而这些概念是面向对象语言 C++ 和 Java 的基本语言要素。

当然, JavaScript 也不是惟一一种能够嵌套在 Web 页面中的编程语言, 微软开发的 VBScript 也是这样的语言, 不过, VBScript 目前只能在微软的 IE 浏览器中运行。微软实现的另一种脚本编程语言 JScript, 与 JavaScript 在核心内容上完全一样, 它们遵从相同的标准 (ECMA-Script), 也是一种能够嵌入在 Web 页面中的编程语言。

1.2 JavaScript 功能简介

JavaScript 程序用于检测用户的活动,并对用户操作做出反应,比如,当光标掠过某个链接时,该链接变为另一种颜色,就是对用户操作的一种反映形式。JavaScript 程序为 Web 站点提供了导航帮助、滚动消息、对话框、动态图像、购物车等。通过 JavaScript 程序,可以控制 Web 页面的外观,也能够评估用户输入的有效性,而这些操作都无需与服务器进行交互,从而减轻服务器的工作负荷。JavaScript 程序也能够检测用户计算机上是否安装了某些插件,并在需要时能够从相应的站点上下载所需的插件。

JavaScript 提供了字符串处理函数和正则表达式支持,利用它们,可以检查电子邮件、网页地址、信用卡号码、手机号码的有效性。JavaScript 是一种解释性的编程语言,它提供了变量、数据类型、条件语句、循环语句、函数、对象等一组基本的语言构件块。通过这些构件块,能够完成数学运算、操作日期和时间、分析字符串、操作数组和对象等。JavaScript 还能够写入和读取 cookie 值,并根据具体的 cookie 值动态地创建 HTML 页面。

总的来说,JavaScript 能够完成下述功能:

- 控制文档的外观和内容
- 控制浏览器的行为
- 与用户交互
- 使用 Cookie 读写客户端的状态
- 操作嵌入图像

当然,JavaScript 也不是万能的。它受限于与浏览器和 HTML 相关的任务,并在受限制的上下文环境中使用,因此,JavaScript 不具备独立语言(比如 C++、Java、Visual Basic 等)所拥有的一些必须特性。JavaScript 不具备下述特性:

- 除了能够格式化和显示 HTML 文档(包括图像、表格、框架、窗体、字体和其他用户界面元素)之外,JavaScript 并不具有任何图像处理能力。
- 出于安全方面的原因,客户端的 JavaScript 不允许读写文件。
- 除了能够让浏览器下载任何 URL 指定的内容外,JavaScript 不支持任何网络访问功能。
- JavaScript 不具有任何多线程能力。

1.3 JavaScript 的版本以及支持它的浏览器

我们知道,JavaScript 在浏览器中运行。每一个浏览器都有自己的 JavaScript 解释器,而同一个浏览器的 JavaScript 解释器也存在一些差别。JavaScript 本身也经历了从 1.0 到 1.5 的发展。本节中,我们将介绍 JavaScript 的各个版本以及支持它的浏览器。同时,为了方便读者了解 JScript,同时给出了 JScript 的版本发展历程以及支持它的浏览器。

1.3.1 版本情况

1995 年 12 月,就在 Navigator 2 正式发行之, Netscape 和 Sun 联合发布了这个后来称之为 JavaScript 的编程语言。JavaScript 的第一个版本称作 JavaScript 1.0。最初版本缺陷很多,已经被高版本取代。微软发布的与 JavaScript 兼容的脚本编程语言称作 JScript。表 1-1 列出了 JavaScript 版本及支持它的浏览器,表 1-2 列出了 JScript 版本及支持它的浏览器。

表 1-1 JavaScript 版本及支持它的浏览器

JavaScript 版本	所支持的浏览器
JavaScript 1.0	Netscape Navigator 2.0、 Internet Explorer 3.0
JavaScript 1.1	Netscape Navigator 3.0、 Internet Explorer 4.0
JavaScript 1.2	Netscape Navigator 4.0~4.05、 Internet Explorer 4.0
JavaScript 1.3	Netscape Navigator 4.06~4.7x、 Internet Explorer 5.0
JavaScript 1.5	Netscape Navigator 6.0 及更高版本、 Internet Explorer 5.5 及更高版本

注释：JavaScript 1.4 是服务器端的 JavaScript。

表 1-2 JScript 版本及支持它的浏览器

JScript 版本	所支持的浏览器
JScript 1.0	Internet Explorer 3
JScript 2.0	Internet Explorer 3
JScript 3.0	Internet Explorer 4
JScript 4.0	Internet Explorer 4
JScript 5.0	Internet Explorer 5
JScript 5.5	Internet Explorer 5

1.3.2 JavaScript 的发展简史

1. 早期的互联网和第一代浏览器

互联网由高级研究计划署 (ARPA) 提出建议, 于 1969 年建成, 当时称作 ARPANET, 这个最初的互联网由位于美国西南部连接在一起的四台主机组成。到了 1972 年, 美国大学中的许多计算机已经相互连接, 电子邮件、新闻组以及 telnet 协议也在开发过程中。到了上世纪的 70 年代和 80 年代, 人们开发了更多的网络互联协议, 此时的 ARPANET 也变成了 Internet, 它使用 TCP/IP 协议, 将大学校园网和军事网络连接起来。

1989 年, Tim Berners-Lee 发明了 World Wide Web。他于 1990 年开发了世界上第一台浏览器, 称作 WorldWideWeb。1992 年, 浏览器已经成为使用 HTTP 协议、传输基于文本信息的世界性工具。1993 年, NCSA Mosaic 浏览器被开发出来, 它提供了在页面中嵌入图像、使用颜色、运用超链接以及文本的功能。Netscape 的第一个版本于 1994 年发行。

2. JavaScript 的诞生——第二代和第三代浏览器

Netscape 2 于 1996 年发布, 它提供了 Netscape 发明的一些全新技术, 其中最重要的技术是框架和 JavaScript。JavaScript 是 Brendan Eich 设计的一种编程语言, 它可以嵌入到 Web 页面中, 能够处理数字和修改窗体的内容。在开发演变过程中, JavaScript 最初叫 LiveWire, 之后改名叫 LiveScript。由于 JavaScript 的核心脚本语法与 Java 的语法十分类似, 因此在正式发行时将其命名为 JavaScript。JavaScript 引用窗体的方式、作为文档对象子对象的链接和锚点、以及作为父窗体子对象的输入一起构成了 DOM level 0。

同年, Netscape 将它的 JavaScript 语言提交欧洲计算机制造商协会 (ECMA) 进行标准化。ECMA 制定了 ECMAScript 标准, 该标准采用了 JavaScript 的核心语法, 但没有定义 DOM level 0 的所有方面。随着 Netscape 3 的发行, Netscape 发布了 JavaScript 1.1, 这个版本的 JavaScript

提供了改变图像位置的功能,这种显示技巧被广泛流行。

3. DHTML——第四代浏览器

1997 年发布 Netscape 4 时, Netscape 同时也发布了直到今天依然是最好的技术——DHTML。通过将专用的 HTML 标记与新的 W3C CSS 1 建议结合在一起, JavaScript 1.2 为程序员提供了改变文档组件 CSS 样式的手段。这种分层技术得到了广泛认可,从而诞生了 Layers DOM。

随着分层方法日益流行,微软决定强力加入竞争,并于 1997 年发布了 Internet Explorer 4。这个浏览器向程序员展示了灵巧的浏览器应该是什么样子。与 Layers DOM 模型不同, IE 4 的专用 DOM 模型引用文档中的任何部分,改变各种 CSS 样式,包括许多 CSS 2 声明。并且 IE 4 提供了其他许多扩展,比如过滤和过渡等。引用文档对象的新语法更加可靠、更加灵活,因此, W3C 决定将它的许多语法和组件采纳到即将发布的 DOM 建议中。

1998 年, Netscape 发行新版本浏览器,这个浏览器依然使用效果不佳的分层 DOM。尽管发布了新的 JavaScript 1.3,但 Netscape 没能修改许多令人讨厌的浏览器错误。Netscape 还发布了浏览器的源代码来平息人们对这个浏览器的抱怨。Netscape 希望利用开放源代码来生产它的下一个版本的浏览器。而开放源代码团体更理性,他们决定放弃不稳定且受限制的 Netscape 核心,然后根据 W3C DOM 指南完全重写浏览器。

4. W3C DOM——第五代浏览器

1999 年,微软发布了 Internet Explorer 5,这是第五代浏览器,它支持 W3C DOM 1 建议的大多数内容,并且支持 Internet Explorer 4 支持的专用技术。IE5 还支持 ECMAScript 的当前版本——ECMAScript 3——等价于 JavaScript 1.5 (JavaScript 1.4 是只用于 Netscape 服务器端的 JavaScript)。除此之外,它还使用进行一些核心扩展的 JavaScript 版本,即 JScript。

W3C DOM 比以前版本的 DOM 新增了许多功能。它给出了创建、删除、修改 HTML 元素的途径,并且这些操作可以在 Web 页面加载后完成。所有样式都可以修改、添加和删除。它甚至改变了检测事件的途径,任何事件都可以有多个事件处理函数。W3C DOM 还将文档表示为一个树形结构,HTML 元素是树的根,包含在它其中的每一个元素都是该元素的子对象。这种方式用更复杂、更逼真的表示方法来描述文档,进而能够访问文档的每一部分。

5. 新一代浏览器

2002 年发布的 Mozilla 浏览器基于开放的源代码 Gecko 展示了 SpiderMonkey JavaScript 引擎。它是最灵活的浏览器之一,遵从 W3C DOM 建议,为 Web 程序员提供了想做任何事情的能力。

随着微软发行 IE 6 浏览器,它依然保持着最大的市场份额。实际上,这个浏览器还存在很多缺陷。它的 CSS 支持力远远比不上 Gecko 引擎,通过对 JScript 进行细微的修改,给 Web 开发人员带来更多的问题。不管怎样,浏览器终于达成了一种共识——采用 W3C DOM 标准。

由于 Gecko 展示引擎的可移植性良好, Mozilla 的知名度也在不断提高,现在已经有 30 多种不同的浏览器采用了 Gecko 引擎,包括 Netscape 6 和 7。如果幸运的话,微软或许也会采用。绝大多数 Web 开发者希望所有的浏览器能够使用相同的 JavaScript 语法。W3C DOM 有助于达成这一目标。如果所有的浏览器厂商生产的浏览器都遵从 W3C 制定的规则,那么 Web

开发工作就会容易很多。Web 开发人员也就不用花费大量的时间、精力和金钱来解决浏览器之间的不兼容问题。

1.3.3 JavaScript 的标准化

为了提供一个生产 Web 页面的公司都可以遵从的 JavaScript 标准, ECMA (欧洲计算机制造商协会) 与 Netscape 联合制定了一个称之为 ECMAScript 的 JavaScript 标准。ECMAScript 基于核心 JavaScript, 并在所有支持这个标准的应用程序中具有相同的动作和行为。这个 ECMA 标准的第一个版本归档为 ECMA-262 规范。可以在 ECMA 或 Netscape DevEdge Web 站点上获取 ECMA-262 规范的联机文档或 PDF 文档。

1.4 JavaScript 和事件

使用 HTML 编写的 Web 页面是静态页面。HTML 可以用于创建按钮、单选框、复选框、列表框和各种表格, 但 HTML 本身不能对用户的操作做出反应。一般情况下, 浏览器将窗体信息打包在一起, 并发送给浏览器进行处理。但是, JavaScript 不是静态的, 而是动态的, 它能够和用户在客户端进行交互。例如, 当用户填写了一张表格, 单击了按钮、图像、链接, 或者在鼠标指针移过某个连接时, JavaScript 都能够对这些事件做出反应, 并与用户进行动态交互。比如说, 在把用户输入发送给服务器之前, JavaScript 能够检查用户输入的有效性。在本书后面的章节中, 我们将会详细讨论各种事件。在此之前, 应该对事件有一个清晰而正确的认识: 事件是 JavaScript 让页面活起来的驱动力, 所有的 JavaScript 代码都是对某个事件做出的操作和反应。

事件和 HTML 紧紧地结合在一起。在“代码 1”的示例中, 我们使用 <form> 标记和三个属性创建了一个窗体。Type 属性指定输入框的类型为按钮, 使用文字“button”指示; value 属性指定分配给这个按钮的值为“请单击此处”, 这个值也就是按钮的标题; 当用户在浏览器窗口中单击这个按钮时, 就会触发一个名称为 Click 的 JavaScript 事件, Click 的意思是“单击”。使用 onClick 这个符号代表发生单击事件时执行的 JavaScript 代码, 也称作事件处理程序。这里, 我们给 onClick 事件处理程序指定了一条语句, 用户单击按钮后, 就会执行这条语句。在这个示例中, 这条语句的功能是在一个消息框中显示“你刚刚单击了按钮”。图 1-2 所示是我们打开这个示例 HTML 文档时显示的页面, 图 1-3 所示是单击浏览器页面中的按钮后显示的消息框。

示例 1-1 一个简单的 JavaScript 代码

```
<html>
<head><title>Event</title></head>
<body>
  <form>
    <input type ="button"
      value = "请单击此处"
      onClick="alert('你刚刚单击了按钮') " >
  </form>
</body>
</html>
```

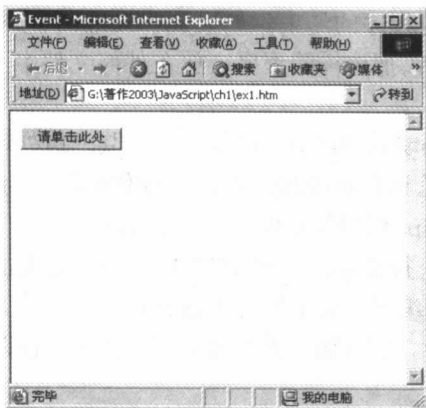


图 1-2 单击按钮时触发 onClick 事件



图 1-3 处理按钮的 onClick 事件

表 1-3 给出了 JavaScript 能够处理的一些常用事件。表格的第一列是事件处理程序 (Event Handler) 属性名, 这些名称都以 “on” 开始; 第二列是可以应用这些属性的 HTML 标记; 第三列给出了事件的触发时机, 也就是事件处理程序运行的时机。

表 1-3 一些常用事件

事件处理程序	HTML 标记	触发时机
onAbort		图像加载过程被中断时触发
onBlur	<body>和窗体元素	窗体或窗体元素失去键盘焦点时触发
onChange	窗体元素	用户修改了窗体元素的值后触发
onClick	所有元素	用户单击了类似按钮这样的窗体元素后触发
onError		加载图像过程中发生错误时触发
onFocus	<body>	窗体或窗体元素得到键盘焦点时触发
onLoad	<body>, <frameset>, , <iframe>, <object>	文档、图像或对象完成加载时触发
onMouseOut	所有元素	鼠标指针离开对象时触发
onMouseOver	所有元素	鼠标指针移过对象时触发
onSubmit	<form>	用户提交窗体时触发
onUnload	<body>, <frameset>	卸载文档时触发

1.5 在哪里编写 JavaScript 代码

要学好 JavaScript, 读者必须熟悉 HTML, 并知道如何创建 HTML 文档。虽然并不要求读者是 HTML 方面的专家, 但至少应该知道 HTML 文档的结构, 知道如何使用标记在 Web 浏览器中显示各种各样的内容。当开发出静态的 HTML 页面之后, 在这个页面中添加 JavaScript 语句就很简单了。

客户端的 JavaScript 程序嵌入在 HTML 文档中头标记<head>和</head>或体标记<body> 和</body>之间。许多开发者喜欢把 JavaScript 代码放置在<head>标记中, 这个位置也是存放函数定义和对象的理想场所。如果想让 JavaScript 代码输出的文本显示在文档中某个指定的位置, 那么应该把相应代码放置在<body>标记内。当然, 在一个 Web 页面中也可以定义多个 JavaScript 代码, 并把它们分别放置在<head>标记和<body>标记中。不管哪一种情况, JavaScript 程序都