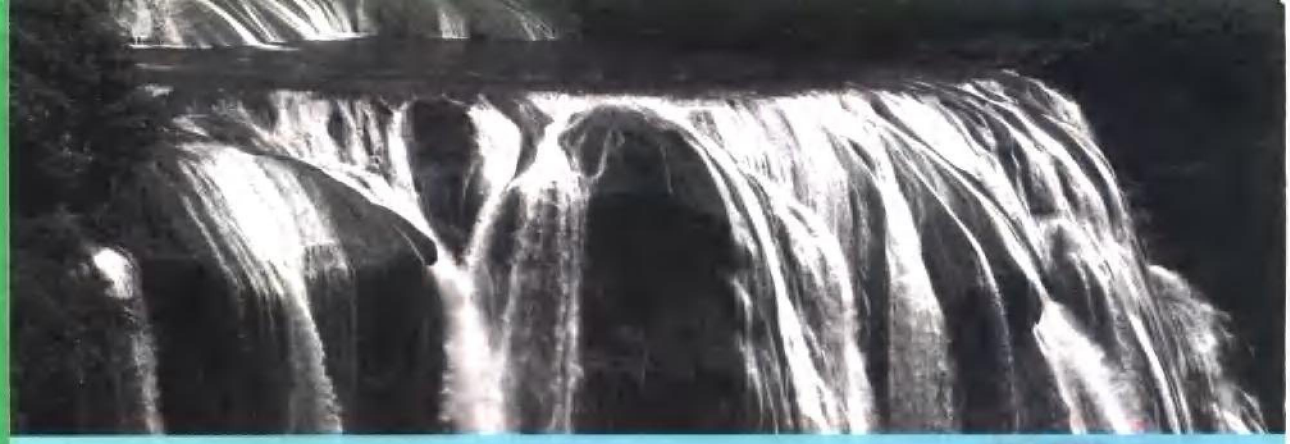

深入学习：JavaScript 开发与实例

〔美〕R. Allen Wyke　Jason D. Gilliam　Chalton Ting 著

陈建春　于大明　郑顺义　等译

- 一本包含了详细代码的首选参考书
- 覆盖了JavaScript 1.4、ECMAScript 和 JScript 5
- 内含具有详细注释、有商业质量的代码示例
- 附有完整、深入的 JavaScript 参考内容

Pure JavaScript

SAMS

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
URL: http://www.phei.com.cn

深入学习:JavaScript 开发与实例

Pure JavaScript

[美]R. Allen Wyke　Jason D. Gilliam　Chalton Ting 著

陈建春　于大明　郑顺义　等译

電子工業出版社

Publishing House of Electronics Industry

内 容 提 要

本书是介绍在网页上和网站上开发、测试和使用 JavaScript 解决方案的全方位的参考手册，介绍了 JavaScript 的所有对象，并提供了大量有价值的示例代码。本书分为 3 个部分，包括概念、技术和语法的参考。第一部分介绍了 JavaScript 及其部分细节；第二部分介绍 JavaScript 的优点和强有力的特性；第三部分是本书的主体部分，介绍了对程序员最有用的信息——按对象排列的参考材料，每一个对象的介绍包括与之相关的属性、方法和事件处理器，每一条都有支持的语言和浏览器版本、语法、说明和示例程序。本书是进行 JavaScript 程序设计必备的、全面的参考手册。

图书在版编目(CIP)数据

深入学习：JavaScript 开发与实例/(美)威克(Wyke, R. A.)著；陈建春等译. —北京：电子工业出版社，2000.4

书名原文：Pure JavaScript

ISBN 7-5053-5894-4

Ⅰ.深… Ⅱ.①威… ②陈… Ⅲ.Java 语言-软件开发 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2000) 第 05572 号

书　　名：**深入学习：JavaScript 开发与实例**

原 书 名：Pure JavaScript

著　　者：[美]R. Allen Wyke　Jason D. Gilliam　Chalton Ting

译　　者：陈建春　于大明　郑顺义　等

责任编辑：窦　昊

排版制作：电子工业出版社计算机排版室监制

印 刷 者：**北京天竺颖华印刷厂**

出版发行：电子工业出版社　URL：http://www.phei.com.cn

北京市海淀区万寿路 173 信箱　邮编 100036

经　　销：各地新华书店

开　　本：787×1092　1/16　印张：76　字数：1897 千字

版　　次：2000 年 4 月第 1 版　2000 年 4 月第 1 次印刷

书　　号：ISBN 7-5053-5894-4
TP·3063

印　　数：4000 册　　**定价：120.00 元**

版权贸易合同登记号　图字：01-1999-3119

译者的话

JavaScript是简单易学的Web编程语言，它以强大的功能、良好的接口能力和接近自然语言的语法而深受广大程序设计者的喜爱。

本书是有经验的JavaScript程序员的首选参考书，它包含了JavaScript窗体、窗口以及层的精确介绍，也包括了对JavaScript概念和组件的进一步介绍。

本书除了包括在其他绝大多数参考书中可找到的简单介绍和简短的语法摘录外，还对JavaScript每个对象及每个对象的属性、方法以及事件处理器提供了实用、详细的示例代码。这不仅能帮助对语法的理解，还为确定如何以及为何选用特定的对象或方法，提供了相关帮助。

本书还详细介绍了Netscape Navigator、Microsoft Internet Explore和Opera等浏览器支持的JavaScript的完整参考，介绍了针对各个浏览器中的功能扩充。

本书介绍了JavaScripte的所有对象，并提供了有商业价值的示例代码，是进行JavaScript程序设计的必备参考书，适合从JavaScript学习者到有经验程序员的各层次技术人员使用。

本书的翻译是集体劳动的结果。主要翻译校对人员有陈建春、于大明、郑顺义、魏芳、柴雪松、徐学卫、孙卓，陈建春最后统一审校了全书。另外参加人员还有梁青槐、贺为、郭文军、张海燕、刘建峰、高震宇、马骏、周华国、石红兵等。

作 者 简 介

Jason D. Gilliam 是 Research Triangle Park, NC 公司的 KOZ.com 软件工程师。他编写过许多企业内部网的网页和 CGI 程序,以及大量 C++ GUI 应用程序和数据库连接程序。他在北卡来罗那州立大学获得计算机工程学士学位。工作之余,他编制了基于 Windows 的声音播放程序,并使用电脑创作音乐。

Chalton Ting 来自弗吉尼亚州的 Vienna, 是 Lucent Technologies 的软件工程师。他为处在领先地位的因特网公司开发网络电话解决方案。他用 Java、JavaScript、CGI、CORBA 和 HTML 技术开发了许多应用程序和网页。他从北卡来罗那州立大学获得计算机和电子工程学士学位。

R. Allen Wyke 是 Engage Technologies 的系统集成咨询部的执行经理。他为许多大公司开发了 Intranet 网页和因特网网站。他与别人合作编写了《The Perl 5 Programmer's Reference》和《The Official Netscape Navigator 4 Book》,并对两本关于 HTML 的著作《The HTML 4 Programmer's Reference》和《HTML Publishing for the Internet, 2nd Edition》的再版作出了贡献。他还完成了另外一本关于因特网资源的著作,是专为大学毕业生编写的。

谨以此书献给

谨以此书献给我的妻子 Deena,她是最棒的! 在写作这本书时,你一直鼓励我、支持我! 我爱你!

——Jason D. Gilliam

谨以此书献给我的全家:John、Alice、Angela、Melissa 和 Olivia。谢谢你们对我的爱和支持。我爱你们!

——Charlton Ting

谨以此书献给我的姐妹:Sandra、Valerie 和 Evelyn。她们是我生命的灵感,开阔了我的眼界,支持我能想到的每一件事情。我爱你们!

——R. Allen Wyke

致　　谢

我要感谢 TIPS 出版公司的 Bob Kern,以及我的合作者 Allen 和 Chuck。感谢他们为这本书所作出的努力。我还要感谢约稿编辑(Acquisitiions Editor)Randi Roger 和开发编辑(Development Editor)Scott Meyers,感谢他们的努力工作。感谢 Sams 的每一个帮助出版此书的工作人员。

我还要感谢"午餐哥们",感谢他们的耐心聆听和鼓励的话语。

——Jason D. Gilliam

我要感谢 TIPS 出版公司的 Bob Kern 为本书的出版所作出的艰苦努力。我还要感谢我的合作者 Allen 和 Jason,感谢他们的努力工作、奉献精神和鼓励。此外,我还要感谢约稿编辑(Acquisitiions Editor)Randi Roger 和开发编辑(Development Editor)Scott Meyers,感谢他们的努力工作。感谢 Sams 的每一个帮助出版此书的工作人员。

我还要感谢那些在艰苦时刻为我提供无私帮助的所有朋友:Mike、Crolyn、Monty、Theresa、John、Blanke、Stacey、医生和我可能忘了提到的人。你们这帮家伙是最好的朋友。

——Charlton Ting

就出版这方面来说,我感谢 TIPS 出版公司的 Bob Kern,我的合作者 Jason 和 Chuck。感谢他们的职业精神、努力工作和为此书的提议和写作所作出的全面帮助。我还要感谢 Randi Roge,一位非凡的约稿编辑,还有 Scott Meyers ,他组织此书并使我们能够集中精力工作。感谢 Sams 的每一个帮助出版此书的工作人员,他们总是想把此书做成最好的。

最后,我要感谢"浣熊",你使我度过了 6 个月的快乐时光。

——R. Allen Wyke

请与我们联系

作为读者,你是我们最重要的评论家,我们想知道你的意见,从而知道哪些做得好,哪些还需要改进,以及你希望我们出版哪方面的书。

你可以给我们发传真、电子邮件或直接给我们写信,以便让我知道你是否喜欢此书,我们如何才能把书写得更好。

请注意,我不会就此书的任何题目提供技术上的帮助。由于我收到的信件太多,我可能无法回答每一封来信。

如果来信,请写上这本书的题目和作者以及你的名字、电话或传真。我将认真阅读你的评论,并与此书的作者、编辑共享。

传真:317 - 581 - 4770

电子邮件: internet _ sams@mcp.com

地址:

Mark Taber

Associate Publisher

Sams Publishing

201 West 103rd Street

Indianapolis, IN 46290 USA

本书的介绍

欢迎使用本书！这是一本 JavaScript 程序员编写、面向 JavaScript 程序员的书。它是一本介绍在网页和网站上开发、测试和使用 JavaScript 解决方案的全方位的参考手册。

本书并非想教人如何编程，而是提供 JavaScript 的细枝末节以便程序员能够使用。JavaScript 在过去几年里一直在进步，现在它已经达到了许多新的领域，这些都在本书中一一介绍。本书分为 3 个部分，包括概念、技术和语法的参考。每一个都是学习和使用语言的有意义的一步。

本书的第一部分是“JavaScript 总览”，起到了为使用其他语言的程序员使用 JavaScript 的桥梁作用。许多时候，程序员买为初学者编写的 JavaScript 的书，因为他们不需要知道如何编写程序，但是他们需要知道 JavaScript 的细节。一旦他们知道这些细枝末节，语法就简单了。此部分为在编程上的这样的转移提供了必要的信息。

本书的第二部分“JavaScript 编程”，介绍了 JavaScript 的优点和强有力的特性，讨论了使用语言和 JavaScript 编译环境的一些问题。程序员可以学到如何在实际应用中使用 JavaScript，还将学到有关浏览器的问题和如何处理 Web 信息。程序员也将学到如何在一个小应用程序里使用 Java 函数，如何使用服务器端的 JavaScript 开发因特网、Intranet 和 extranet。当你学完这一部分之后，就可以开始编程了。

第三部分“JavaScript 的对象参考手册”是本书的主体部分，包括一些对于现在的 JavaScript 程序员最有用的信息——按对象排列的参考材料。在每一个对象之下都详细介绍了与它相联系的属性、方法和事件处理器，每一条都有自己的示例，每一条都说明了版本信息、浏览器和服务器的支持。这部分分为 4 章：第 6 章是 JavaScript 的标准对象和语法；第 7 章是用户端和浏览器的脚本对象；第 8 章主要是服务器端 JavaScript 对象（主要面向 Netscape Enterprise Servers）；第 9 章是 JavaScript 面向 Microsoft 活动脚本接口。另外还包括几个附录，在附录中包括关于浏览器和语法的版本支持的快速参考表，另外一个附录“JavaScript 资源”中，包含了 JavaScript 的网络资源。除此之外，还包括另外的文件和可能遇到的编程问题的帮助信息。

这就是本书的所有内容。JavaScript 的新程序员，欢迎你们来到 JavaScript 世界。对于那些想有一本好的参考书的人来说，我们希望本书是你能找到的资源最丰富、内容最新的参考书。

R. Allen Wyke
Jason Gilliam
Chalton Ting

第一部分　JavaScript 总览

第1章　什么是JavaScript

最初，人们用汇编语言进行程序设计，后来开始使用像sed、awk、Perl等脚本语言。到了20世纪80年代末和90年代初，因特网引发了一场技术革命，人们只要一个Modem就能和整个世界相联。随着因特网用户迅猛增加，网络浏览器的功能急待增强。

HTML虽然有其自身的优点，但在拓展网络功能方面稍有欠缺，这样就促使使用服务器端的程序来处理来自站点的开发人员所需要的动态页面。

这些程序使得Web开发人员能够增强站点的功能并处理用户提交的信息。当用户发出不正确或者不完整的信息时，CGI(通用网关接口)必须产生响应，以通知用户重新发送请求。这就使得服务器和浏览器之间来来回回发送大量数据，然而，从整体上来说，与用户所享受到的功能相比，这个代价很小。

很明显，客户端应该配有智能程序，以便用它来完成CGI的一些功能以及错误检查，从而减少用户花在连接服务器上的时间。这样还能使一些处理工作改由浏览器完成，从而减轻网站的处理负荷，提高网站的整体处理能力。实际上，创建这种能够在浏览器环境下而不是在服务器上运行的新的脚本语言的部分原因也正是为了增强客户端的功能，提高客户端的效率。这种语言可用来执行客户端的一些任务，如创建有效的动态页面内容，这样将把编程用在HTML出版之中。让我们来欢迎JavaScript的诞生吧！

1995年12月4日，Netscape公司与SUN公司合作，推出了JavaScript 1.0，当时称为LiveScript。完全不同于以前基于服务器的网络语言，它能够在Netscape Navigator 2.0浏览器中进行编译操作。作为一种解释性的语言，JavaScript定位在作为JAVA语言的一种补充，它允许开发者通过企业网与因特网来创建和定制应用程序。JavaScript给予开发者真正编写程序的能力，而不仅仅是HTML的格式数据。

除了开发者所喜欢的对客户端的控制之外，Netscape也实现了服务器端的JavaScript，这使得开发者能够在服务器端使用与他们在浏览器网页上相同的程序语言。扩展的数据库连接功能加进了该语言中(称为Livewire)，它允许开发者直接从数据库得到信息，并能更好地维护用户会话。JavaScript真正在简单的HTML和服务器端复杂的CGI程序之间建立起了一座桥梁，它为Web开发人员设计、实现和配置网络方案以及进行分发应用程序的整个处理过程，提供了一种通用的语言。

JavaScript脚本语言中的另外一种是Microsoft公司在其Internet Exploror 3.0浏览器中实现的语言，这种解释语言称为JScript。像Netscape公司一样，Microsoft也是在服务器端执行这种语言(JScript 2.0)。这种技术是Microsoft公司在推出ASP技术或者动态服务器网页(Active Server Pages)技术前后推出的。

JavaScript、JScript的区别以及什么是ECMAScript

JScript源于Netscape公司出版的JavaScript文档，它应该与JavaScript 1.0相同。但是Netscape公司在说明书中遗漏了一些特性，Microsoft公司又未能正确地模拟出来，所以当Microsoft发布JScript 1.0时就与JavaScript 1.0稍有不同。自从这些最初的浏览器发行以来，

JavaScript 被接受并渐成为一种标准，即欧洲计算机制造商联合会（ECMA）所称的 ECMAScript 1.0(ECMA-262)。因为这种标准化，人们认为 JavaScript 是 Netscape 公司对 ECMAScript 的实现，而 JScript 则是 Microsoft 公司的实现方法。

ECMAScript 1.0 于 1997 年 6 月正式发行，1998 年 4 月国际标准组织与国际电子技术协会(ISO/IEC16262)通过了这项标准。1998 年 6 月，ECMAScript 2.0 被 EMA 批准，正在得到其他标准组织的完全接受。

注释　因为 ECMAScript 是在 Netscape 公司的 JavaScript 之后被标准化，所以这本书将把它当作 JavaScript。

那么，对程序员来说，JavaScript 是什么呢？从本质上来说，JavaScript 是一种面向对象的、跨平台的、结构化的、多用途的语言，它使得程序员可以针对不同用户设计出各种类型的解决方案。它不仅具有在浏览器中添加网页渲染的功能，还能够执行 Netscape 和 Microsoft 网站服务器的服务器端的处理。

最近，JavaScript 也被包括在 Microsoft 的 Windows Scripting Host 中，这样使得程序员可以编写将在操作系统本身中执行的脚本。这种功能类似于 DOS 的批处理，但给予程序员更多的功能。也正是由于这种先进功能，使得该语言在计算机世界占有一席之地并得到进一步的发展。

除了可以执行 JavaScript 语言的这些环境的好处之外，它还提供安全保护措施，保护终端用户不受黑客的破坏。从年龄上来说，JavaScript 语言虽然还很年轻，但目前版本已非常成熟与强大。正是由于它的功能与多用途，使得它成为程序员的最佳选择。

既然我们已经知道了什么是 JavaScript，那现在应进一步了解它对程序员意味着什么。作为程序员，只知道战略上的一些词句并不能使一种语言有用。首先我们将看到 JavaScript 基于对象的特性。

1.1　基于对象的技术

阅读本书之前至少应使用过一种编程语言，即使只在大学里学过一学期这方面的课程也可以。进一步地说，我猜想你用的语言或是 C++、Java 或是 Perl 语言，其中 C++ 和 Java 语言属于面向对象(OO)的语言。从所有的对象都从核心 Java 语言类中派生这一点来说，Java 更是面向对象的语言。

对不熟悉所谓面向对象编程(OOP)的人来说，下面代码给出派生对象和类的概念。对象或类与它的不同特性和功能相联系，这定义了它所呈现的属性与状态。一旦这些对象或类被创建和定义，就可以生成新的实例——有时被称为“孩子”——它继承了与其“父”对象具有相同特征的能力。

下面通过创建汽车对象来建立对对象的感性认识。部分汽车对象的特性可以是车门的数量、颜色和种类(如跑车或卡车)。此外，行使或停止这车辆的能力也可以作为描述特性。这种类型对象伪代码描述如下：

```
object vehicle(){
  // Characteristics of the vehicle
```

```
    num_doors;
    color;
    type;

    // Methods used to move and stop the truck. Note that the move()
    // method takes a direction as a property. This direction could
    // be something like forward, backward, left, or right.
    move(direction);
    stop();
}
```

汽车对象已经建立,下一步可以很容易地创建新的对象实例。如一辆红色 4 门汽车。还可以创建一辆黑色双门卡车。除了可以创建它们的实例外,还可以编程来改变实例的状态。这通过定义移动或停止两种状态来实现。

下面例子中的伪程序代码表示一辆正在行驶的黑色双门卡车。

```
// Create the new instance of the vehicle
myTruck = new vehicle();

// Define the type, number of doors and color
myTruck.doors = 2;
myTruck.color = "black";
myTruck.type = "truck";

// Define the "state" of the truck
myTruck.move(forward);
```

上述基本过程就是首先建立一个汽车对象的实例,然后给它加上特征,正是这些特征使它成为一辆特定的车辆的实例,即定义的卡车。

由这个例子可以看出,车辆对象的产生可以容易地创建更多的具有不同特征的车辆。在编程时,这里的"容易"意味着更少的代码——这正是所有程序员喜欢听到的。因为建立起对象,就可能创建继承它的特性的新的不同实例,而不需重新定义它们。通过这种特性,可以利用对象间的交叉特征。创建总的、主对象的思想能够通过它来派生出子类。

还可以进一步创建新的对象——不是实例——来继承父对象的特征,这样做可以从子类中派生只继承某些特征的子实例。可以定义一个子对象来只将父类的色彩特征传给任何一个子实例。正是这种对象特征使得你能够按这种模式编程。

下面程序代码描述了如何基于车辆对象来创建一个飞机对象。

```
// Create the new object that inherits the vehicle
object airplane(){

  // Inherit the vehicle object
  this = new vehicle();

  // Define the doors property, then assign it to the size
  // property of the plane object
  this.doors = "747"
  this.size = this.doors;

  // Assign the color and type of plane
```

```
    this.color = "silver";
    this.type = "American Airlines";

    // Define the "state" of the plane
    this.move(up);

    // Now that the object is created with the values, return the
    // object.
    return this;
}
```

并不是所有的语言都支持这个概念，也有其他的只基于这种概念的语言。这种概念无疑对语言有好处，但编写好的、有效的、模块化的代码并不要求有这个概念。JavaScript 就是应用这种概念的典型范例，但它并不是完全面向对象，它只是一种基于对象的语言。

在此详细谈论 OOP 超出了一本重点在于介绍 JavaScript 的书的范围，但对于热爱编程的人来说，值得进一步地探索。不妨去书店买一本有关 OOP 的书籍仔细阅读，或去网上 object central(http://www.objectcentral.com)得到关于 OOP 的资料和信息。

那么，基于对象的又是什么样子的呢？它与 OO 十分类似，只是不具有 OO 的所有功能与特征。对于一个基于对象的语言来说，能使用的继承性、范围、功能少得多。但这不应成为反对 JavaScript 的理由，因为它使得语言更容易学习且便于程序员的维护。学习面向对象的编程语言不是件容易的事，在得心应手之前会遇到许多令人头痛的问题。

通过允许创建自己的与对象类似的元件，以及通过在语言中赋予原型以新的特征来扩展核心对象，JavaScript 弥补了许多它在 OO 方面的局限。要知道这是如何实现的，请看下面的面向对象的 JavaScript。

1.1.1　面向对象的 JavaScript

在深入学习面向对象的 JavaScript 之前，先看一下在服务器端和客户端的 JavaScript 的内核和功能有什么不同。两者都有在自己的运行环境中特定的对象，因此，对象的初始化和创建在不同的时间发生。由于这个特征，应该从两个方面来看待 JavaScript：服务器端和客户端。

最低层次的客户端 JavaScript，是当一个页面被装载到浏览器中时创建的数个核心对象。除了这些核心对象，还有当在网页中有某些标签时所创建的派生对象。这些派生对象继承了父对象的不同特性，并允许脚本获得 HTML 标签的属性。

如果你计划进行深入编程，对 JavaScript 对象层次的理解就非常重要。必须很好地理解父与子对象如何相互作用与引用。为了帮助理解，图 1.1 清晰表明了基本的 JavaScript 对象在客户端的层次。

正如在图表中描述的一样，所有客户端对象或是从 Window(窗口)对象或是从 navigator 对象派生出来。考虑到 JavaScript 是基于对象的语言，列出图 1.1 中的结构非常有意义。在给定页面上的所有对象都是在浏览器的窗口中形成，因此，所有 JavaScript 对象均为 Window 对象的后代。通过使用 Window 对象，程序员可以在页面中选择不同的窗格、文本、层、表格以及许多其他的对象和特性。

navigator 对象作为浏览器的元件，它是浏览器本身的一部分。这尤其是指安装的插件以

及与浏览器相关的 MIME(多重因特网邮件扩展)类型。使用 navigator 对象可以检测浏览器的版本、决定插件的安装以及决定与 MIME 类型有关的程序在系统中的注册。除此之外,还有许多可以获取其他浏览器属性的能力。

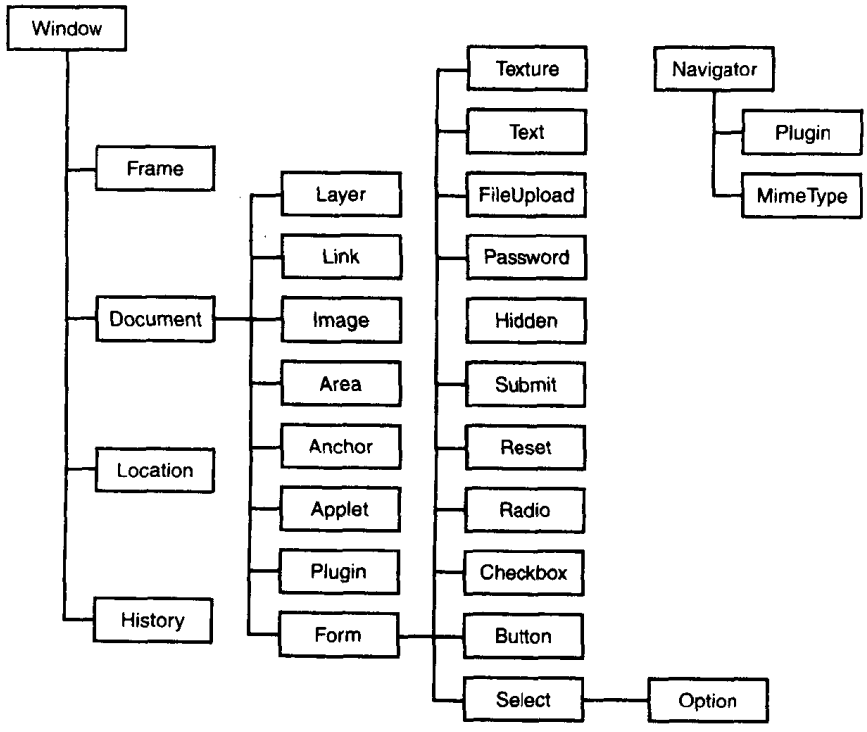

图 1.1 客户端 JavaScript 对象层次结构图

同客户端一样,服务器端 JavaScript 由几个可以派生出许多其他对象的核心对象组成。根对象是 Dbpool 和数据库(Database)对象,从中您可以创建数据库的链接、控制光标位置以及存储过程和运算结果。图 1.2 表示出特定服务器端对象的层次结构。

注释 服务器端和客户端 JavaScript 语言都是基于对象的,熟悉 Java 的程序员会发现,这与 Java 语言十分相似。尽管 JavaScript 没有众多的对象/类,但它们不论从结构上还是语言形式上讲都非常相近。

由于对象的层次,掌握一个网页上不同对象与元件最有效的方法是利用层次本身。如果想获得一网页上某个表格的特定文本字段,可以用下面的语法:

```
window.document.formName.textboxName.value
```

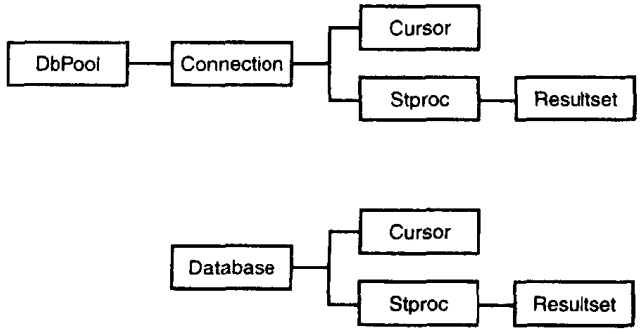

图 1.2　服务器端 JavaScript 对象层次结构图

程序员用 JavaScript 可以很容易地创建自己的对象或扩展语言已有的核心对象。具体解释如何创建自己的对象和一些例子在第 2 章"语言细节"中有详细解释。如果想了解更多的已有对象的扩展功能，可以在本书的参考章节中查找有原型属性的对象。

由于 JavaScript 是基于对象的，所以它自然就具有了使用模块化方法进行编程的许多优点。通过创建自己的对象和方法，可以更好地维护目前的代码，可以创建出可多次用于别的程序、实例的代码。当您能够一次创建并传递那些互不相同或者类似对象的特征时，为什么要写两遍事实上完全相同的代码呢？

1.1.2　模块化程序设计

在 JavaScript 中以模块化的形式进行编程有 3 个关键的事情。在程序中使用这些项目可以使您创建可在工程间可以重复使用的代码。它们是

1. 创建您自己的对象
2. 定义通用函数来处理普遍的任务
3. 将可重用的代码存在外部的资源文件中(一般如 * .js 或 * .mocha 文件)

记住，和其他语言一样，好的注释文本常常是编程中最有益的方面。

因为如何创建您自己的对象将在第 2 章介绍，这里，先看一下处理普遍任务的函数。和其他语言一样，在许多实例中需要多次重复执行某些线程，许多时候只涉及到几个参数取不同的值，但过程是一样的。

例如，验证用户输入的日期。假定用户在一窗格中输入将提交给网络服务器作进一步处理的年月日。程序员所关心的问题之一是需要他输入的日期格式是 MM/DD/YYYY，其中月和日分别占两位数字，年占 4 位数字。

为了完成这个任务，可以建立一个单独的函数，在任何一个传来的数字前加零。这个函数能够简单地检验传来的值是否小于 10，若小于 10，则执行预加。通过在函数中定义这个处理，程序员可以使用相同的函数来进行月份和日期的验证。这避免了为每一个重写代码的麻烦。尽管这只是一个简单的例子，但却说明函数与代码重用的好处。

程序员还可以通过将他们的代码包含到 JavaScript 的外部资源文件中来使他们的程序具

有模式化的结构。这样,只需要书写一次代码,并将它们储存到单独的地方,在网页包括它们时,只要简单地引用资源文件即可。如果函数需要改动,只需要修改单独的函数文件而不是所有的使用它的文件。这样简单的事,却可以节省网页程序员几小时甚至几天的工作时间。

1.2 安全性

当今因特网发展的最大问题之一是安全性。如果没有安全防范措施,就不可能成功地开发任何应用程序,不论它是基于 Web 的还是 Web 基于的。可以这样说,一个程序的安全性能最终决定它对用户的价值。如果代码很容易被改动或被其他程序破坏,那么它将被拒绝使用。

因为 JavaScript 大部分直接在浏览器环境中被解释,故用户很容易受到恶意程序的破坏。因为浏览器运行于操作系统上,这意味着可以进入用户的文件系统。这样使得 JavaScript 程序可以利用浏览器中安全措施的漏洞进入文件系统。一旦程序员实现了这一点,许多事都可能发生,甚至可能获得访问私人文件的权利和将其删除的能力。这导致用户受到"黑客"的支配。

为 JavaScript 提供安全性,事实上有两方面,一方面在于编程人员的责任感,一个程序员必须保证用户执行的程序不是恶意的;另一方面在于用户本身,他们必须最终决定是否在他们的系统上运行一个 JavaScript 程序——这是指那些必须在浏览器上执行的程序。

由于这些潜在的破坏因素,在用 JavaScript 语言编程时,用户与程序员可以依赖不同的安全级别。正如已讨论的,安全措施一部分在于编程人员的责任感,而另外也包括在浏览器中设置允许用户控制他系统中运行的程序的措施。

1.2.1 安全措施

当 JavaScript 1.0 在 Navigator 2.0、Internet Explorer 3.0(JavaScript 1.0 称为 JScript 1.0)和 Opera 3.0 浏览器中发布的时候,仅有的真正的安全层是用户有关闭和打开浏览器的能力。浏览器本身控制 JavaScript 运行环境和它本身的安全措施。

在这个模型中,JavaScript 一旦运行起来,则完全由浏览器负责保护用户不受恶意程序的破坏。最初,这算是完全的安全措施。有人这样评论:"这是让专家来保护用户"。然而,想到就能做到,错用 JavaScript 的事开始涌现出来。

在网站上使用窗格时,一个似乎是潜在问题的事情发生了。既然窗格可在预定义的区域中载入独立文件,那么就可以从不同的域和服务器中载入不同的文件并展现给用户。当 JavaScript 文档从一个服务器传到另一个服务器时,参数就变得可查看和可修改,这样问题就产生了。

为了帮助用户避免这个问题,Navigator 2.0、Internet Explorer 3.0 和 Opera 3.0 共同执行了"同源政策"(Same Origin Policy)。这阻止了一个服务器发出的 JavaScript 程序获得来自其他服务器、端口、协议的文档的属性的权利,并且返回信息给它的服务器。

显然这个政策并不影响给定文件的所有元件,但它确实包括了一个核心设置。一个文档必须通过源检察的属性内容详见表 1.1。

表 1.1　Document 对象系列必须接受检查的项目和内容

检查对象	特性/方法
文件	读/写:anchors, applet, cookie, domain, elements, embeds, forms, lastModified, length, links, referrer, title, URL(用户程序设计语言),每个窗体样本,每个与 JavaScript 可在线连接的 Java 类。写:所有其他类
图像	lowsrc, src
层	src
位置	除了 location.x 和 location.y 的其他内容
窗口	find

有时人们希望脚本能够从同一域中的其他服务器中的网页中获取变量,这是该安全模式中的一个特例。但是通过定义,用户在 http://myscripts.purejavascript. com 的窗格中不能获取和上传 http://mydocs.purejavascript.com 中文档的属性。尽管它们域相同,但二者的网址(URL)不同。

程序员可以将文件的域属性加到当前域的后缀上,这样就可以获取相同域中不同服务器中网页的属性。接着上段所述的例子,用下面语句代码即可实现这种功能:

```
document.domain = "purejavascript.com" ;
```

设置这种属性使您能够访问域中的子域。

当 JavaScript 1.1 在 Navigator 3.0 发行的时候,Netscape 公司执行了一套更完全的叫"数据感染"(Data Taining)的安全方案。除了第一代 JavaScript 浏览器的完全模式外,"数据感染"允许用户或程序员设定他是否需要脚本获取其他服务器上文档的属性。如"数据感染"未启动(缺省),用户将得到不允许获取其他服务器文档属性的信息提示。

如用户想要一网页脚本有获取其他脚本或文档的属性时,可启动"数据感染"特性。这对安全是一个冒险,但如果在还有其他安全措施的企业网环境中,这也许是需要的。表 1.2 展示如何在不同的操作系统中完成该设置。

表 1.2　启动数据感染系统前对浏览器环境变量的设置

操作系统	环境变量	操作说明
Windows	NS_ENABLE_TAINT = 1	把该行写入 Windows 3.1x、Windows 95, Windows 98 版的 autoexec.bat。对 NT 可写入用户环境设定或 autoexec.bat
UNIX	NS_ENABLE_TAINT = 1	根据外壳决定是用 set env 还是 env 来设置这个变量
Macintosh	删去 NS_ENABLE_TAINT 前的两个 ASCII 码(两道斜杠//)	通过编辑浏览器应用程序中的具有类型 Envi 和数字 128 的源来实现。它应位于末端
OS/2	NS_ENABLE_TAINT = 1	在 config.sys 中设置

一旦这些参数被设置,将会影响到许多文档的属性,如表 1.3 显示缺省的被感染的文档对象的列表。

表 1.3 默认感染的文档对象

对象	感染的属性
document	cookie, domain, forms, lastModified, links, referrer, title, URL
Form	action, name
each Form instance	checked, defaultChecked, defaultValue, name, selected, selectedIndex, text, toString, value
history	current, next, previous, toString
image	name
Link	hash, host, hostname, href, pathname, port, protocol, search, toString
location	hash, host, hostname, href, pathname, port, protocol, search, toString
Option	defaultSelected, selected, text, value
Plugin	name
window	defaultStatus, name, status

作为程序员,您可以用 navigator.tainEnabled()方法来测试用户是否启动了"数据感染"。这个方法的示例,见第 7 章"客户端浏览器语法"。

用户除了有设定如何处理感染的能力,程序员还可以设定或感染那些没有经过用户允许则不能从一个脚本传到另一个脚本的对象或信息。这样规定以后,浏览器会弹出一个对话框,询问用户是否同意信息的传送。

更多的关于"数据感染"的内容详见 Netscape DevEdge 网站(http://developer.netscape.com)上的客户端 JavaScript 指南。其中"JavaScript 安全性"一章中有一整块内容("在 JavaScript 1.1 中使用数据感染")介绍 JavaScript 1.1 的安全与数据感染。

由于数据感染未能提供真正的 JavaScript 所需的安全模型,Netscape 公司在 JavaScript 1.2 中将它删去,代之以签名脚本,这是目前执行的最完整的安全模型。

1.2.2 签名脚本

签名脚本是 Netscape 公司执行的一种安全模型,它允许程序员在用户认证后得到不同的限制信息的权力。这个模型基于 Java 语言中的签名对象模型,用 LiveConnect 与 Java capabilities API 来执行它的功能。使用这个模型给了程序员非常明确的控制,即在用户的计算机上可以做什么,不可以做什么。

关于 Java capabilities API 的详情可在 Netscape 公司的 DevEdge 网站(http://developer.netscape.com/docs/manuals/signedobj/capabilities) 中找到。

使用这个模型时,可以给 JavaScript 外部资源文件(通过〈SCRIOT〉标签的 SRC 属性)、事件句柄、包含在网页中的代码做标记。实际的脚本标签操作通过 Netscape 公司的页面标签工具(Page Signer tool)完成,这个工具可在 http: //developer.netscape.com 得到。

页面标签工具允许建立 JAR(JAVA 文档)来包括程序员的安全证书和代码。浏览器遇到