McGraw-Hill 北京科海培训中心

▶ 数据库技术专业丛书

SQL Server 2000 Web 应用开发指南

[美] Craig Utley 著
宫丽杰 译
熊桂熹 审校

清 华 大 学 出 版 社

需要全本请访问 www.ertongbook.com

(京)新登字 158 号

著作权合同登记号:01-2001-3280

内 容 提 要

本书为构造、部署、管理使用 SQL Server 2000 的大型 Web 应用解决方案提供了独一无二的技术指导。书中内容涉及如何创建 ASP,使用 ADO 访问数据库,直接使用 Visual InterDev 工具来构造数据驱动的网页而无需编程。书中通过大量实用的范例,介绍如何使用 ASP 及 COM 组件来开发 N 层应用的知识,利用 COM+组件服务如何增强应用的可扩展性和提高速度。同时本书还讨论了加密技术及应用安全性。

利用本书的内容,可以最大化地挖掘 SQL Server 2000 上的 Web 应用功能,是一本 Web 和电子商务应用开发者的优秀参考书。

SQL Server™ 2000 Web Application Developer's Guide

Copyright ©2001 by McGraw-Hill.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publishe .

本书中文简体字版由美国 McGraw-Hill 公司授权清华大学出版社和北京科海培训中心出版。未经出版者书面允许不得以任何方式复制或抄袭本书内容。

版权所有,盗版必究。

本书封面贴有 McGraw-Hill 激光防伪标签,无标签者不得进入各书店。

书　名:SQL Server 2000 Web 应用开发指南

作　者:Craig Utley

译　者:宫丽杰

出版者:清华大学出版社(北京清华大学校内,邮编 100084)

印刷者:北京朝阳科普印刷厂印刷

发行者:新华书店总店北京科技发行所

开　本:787×1092 1/16　印张:23.75　字数:578 千字

版　次:2001 年 10 月第 1 版　　2001 年 10 月第 1 次印刷

印　数:0001～5000

书　号:ISBN 7-302-04949-1/TP・2787

定　价:39.00 元

简　介

本书主要讲述如何使用Microsoft SQL Server作为数据库引擎来开发Web应用。Web应用涉及的范围很广，其中很多技术在本书中都有介绍，而且比较重要的部分有深入的讲解，如IIS对象和ActiveX数据对象(ADO)等。

Web应用开发的学习步骤

本书讨论的内容非常广，因此我们用一张图来说明开发Web应用究竟需要掌握哪些知识。

图1中是创建Web应用时必须学习的几个方面，它们之间有一定的顺序关系。首先要了解的是HTML，这是因特网的语言，用来向最终用户显示页面。

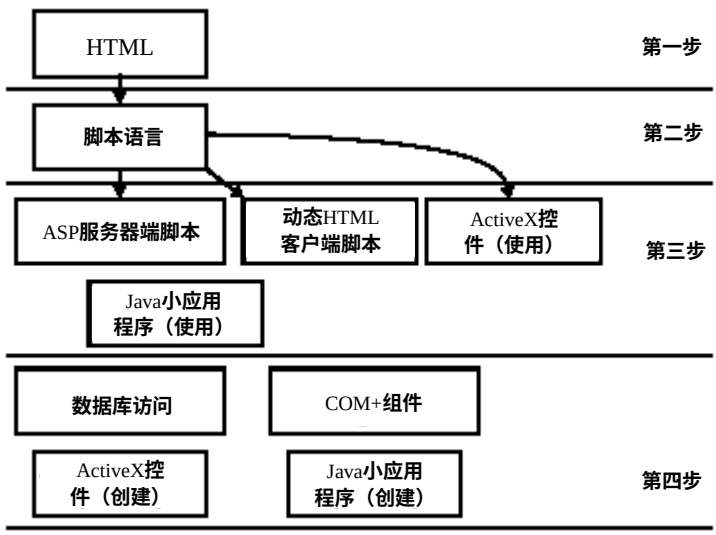


图 1　开发 Web 应用的步骤路标

注意：也可以用XML显示数据。XML和HTML一样是纯文本的，但要做到页面显示至少还需多一步的处理。因而本书中假定用HTML来显示文本，用XML来传送或显示数据。

第二步要学习的是脚本语言，通常是VBScript和JavaScript(也经常叫做JScript,或ECMAScript)。许多Web开发者同时使用VBScript和JavaScript。用脚本语言可以完成几个重要的功能。首先，可以创建客户端脚本及使用DHTML(动态HTML，某种意义上可称为客户端脚本的高级格式)，在客户端运行一部分代码。用这种方法可得到某些很强大的功

能，但对客户端的浏览器会有限制。

用脚本语言还可以构建服务器端脚本。它在服务器端运行，对客户端浏览器无所苛求。因此它比DHTML还要强大。本书大多数情况下使用服务器端脚本，语言是VBScript, 这并不是说VB Script比JavaScript更优秀，只是它更经常被使用而已。

脚本语言使客户端ActiveX控件和Java Applet更能发挥作用。有时候可以不使用脚本，直接创建ActiveX控件和Java Applet。不过使用脚本时，可以与这些对象进行交互。

掌握了HTML和脚本语言后，就可以创建自己的Web应用程序了，还可以加进某些高级特性，如数据库访问，以及调用COM/COM+组件。可以创建自己的组件，用开发语言(不是脚本语言)创建ActiveX控件和Java Applet。

本 书 结 构

通过图1给出的路标，你会发现创建Web应用需要掌握许多知识。在编写本书时，我就向自己提出了这样的问题：是否假设读者已了解了HTML?是否要求他们已掌握了VBScript?读者是否应该熟悉SQL Server 2000?读者是否曾经有过编写COM组件的经历?

由于本书是数据库专业丛书的一部分，因此假设读者对SQL Server已有一定的了解(本书尽量降低对此的要求)，我假定读者已经会使用SQL Server Enterprise Manager来创建数据库、设置权限和浏览节点。本书关于SQL Server的使用讲述是很少的，主要涉及的是如何访问SQL Server，而不是创建表和数据库。

本书假设读者已熟悉HTML，不熟悉的读者请参考附录E。尽管附录E不是HTML的完全参考，但它包括了最主要的一些内容：HTML的工作方式、HTML表单和HTML表。对本书中的大部分例子，参考附录E已足够了。

本书还假设读者熟悉VBScript，对此也有一个附录F，其中讲述的是VBScript的最基本概念。

ASP占据了本书相当大的比例。第2章和第4章详细讲述了ASP的内容。第3章介绍Visual InterDev，这是ASP的开发工具。其他各章中也都涉及到ASP。

动态HTML只在第12章稍有提及。因为DHTML在浏览器方面有诸多限制，微软和Netscape的浏览器对DHTML的实现方式各异，因此很难写出适合多种浏览器使用的DHTML。而且浏览器的种类和版本对DHTML都有影响。

第5章到第9章着重讨论了数据库访问。第10章到第14章中也不同程度地涉及到数据库访问。COM+组件对于开发可伸缩的Web应用很重要，不管是数据库访问时的性能，还是一般的系统性能。这些内容在第13章和第14章中讨论。

最后，第15章讲述安全性，虽然在图1的路标中未提及，但也是很重要的一部分。

软 件

本书假定读者使用微软技术来构建Web应用。本书中使用Windows 2000，不过其中的大部分例子也可运行在Windows NT、Windows 9x及Windows Me上。

本书使用的数据库是SQL Server 2000 Enterprise Edition,运行在Windows 2000 Advanced Server上。Windows 2000上集成了IIS 5.0，这是微软的Web服务器。若使用Windows 9x或Windows Me，能使用的Web服务器是个人Web服务器，其管理工具等可能与本书中的描述会有差别，而且第15章安全性部分的介绍不适用于这些系统。

本书使用Visual InterDev 6.0，Service Pack 4。不使用这个工具也可以完成本书大部分的例子，除了第2章、第10章和第11章。

不管Visual Studio.NET会怎么改变现在的Visual InterDev的工作方式，但ADO不会变，而ASP的工作方式也不会变，所以仍需要了解和掌握，因此本书中的内容对Visual Studio.NET仍是有用的。

需要全本请访问 www.ertongbook.com

目　录

第1部分　用ASP进行Web应用开发

第1章　Web应用介绍 1

1.1　什么是Web应用 1

1.1.1　Web服务器和HTML 1

1.1.2　客户端脚本编程 4

1.1.3　服务器端脚本编程 6

1.2　微软的IIS和ASP 9

1.3　N层Web应用 11

1.4　小结 12

第2章　第一个ASP 13

2.1　准备创建ASP 13

2.1.1　IIS和虚拟目录 13

2.2　创建第一个ASP 16

2.2.1　ASP引擎如何处理代码 18

2.2.2　浏览目录和缺省文档 20

2.3　ASP应用程序 21

2.3.1　include文件 22

2.4　小结 24

第3章　Visual InterDev介绍 25

3.1　Visual InterDev 25

3.1.1　安装Visual InterDev 25

3.1.2　安装服务器端调试组件 26

3.2　使用Visual InterDev 29

3.2.1　在VI里浏览 31

3.2.2　在VI里创建第一个页面 33

3.2.3　理解VI如何处理文件 35

3.2.4　为工程添加已有的内容 37

3.2.5　调试Web应用程序 38

3.2.6　使用布局和主题 42

3.2.7　第一个数据驱动的Web页 45

3.3　小结 46

第4章 使用ASP对象 47
4.1 维护状态——Application对象和Session对象 47
4.1.1 Application对象 47
4.1.2 Session对象 48
4.1.3 Global.asa 52
4.2 Web应用程序的用户和环境 54
4.2.1 Response对象 54
4.2.2 Request对象 55
4.2.3 cookie 56
4.2.4 Server对象 57
4.3 综合运用 58
4.4 小结 65

第2部分 服务器端数据库访问

第5章 ADO介绍：建立连接 66
5.1 ADO介绍 66
5.1.1 微软数据访问简介 66
5.1.2 一个ADO例子 68
5.2 使用ADO对象模型 70
5.2.1 ADO常量 70
5.2.2 Connection对象 71
5.3 小结 85

第6章 ADO：Command和Recordset 86
6.1 Command对象 86
6.1.1 执行语句 86
6.1.2 使用存储过程 88
6.2 Recordset对象 99
6.2.1 打开记录集 99
6.2.2 修改记录 104
6.2.3 显示记录集的更好的方法 108
6.3 小结 110

第7章 通过Web访问SQL Server 2000数据仓库 111
7.1 数据仓库技术简介 111
7.1.1 怎样查看数据 112
7.1.2 用微软的Cube Browser查看立方体数据 114
7.2 用ADO MD访问立方体数据 116

7.2.1 用Catalog对象查看立方体结构 117
7.2.2 查看查询结果 123
7.3 小结 129

第8章 在Web上使用英语查询 130
8.1 理解英语查询 130
8.1.1 英语查询做什么 130
8.2 创建英语查询工程 130
8.2.1 第一次测试程序 133
8.2.2 修改模型 135
8.2.3 编译应用程序 139
8.3 在Web应用程序中使用英语查询 139
8.3.1 增加错误处理 142
8.4 在分析服务中使用英语查询 145
8.5 小结 149

第9章 高级数据访问 150
9.1 数据成形 150
9.1.1 一个关联的典型的记录集 150
9.1.2 层次型记录集 151
9.1.3 总计层次 160
9.2 非连接记录集 162
9.3 XML 166
9.3.1 什么是XML 166
9.3.2 SQL Server 2000和XML 168
9.3.3 ADO和XML 171
9.3.4 使用XML的意义 173
9.4 小结 174

第3部分 微软数据访问工具和客户端访问

第10章 Visual InterDev数据工具 175
10.1 用Visual InterDev创建连接 175
10.1.1 Visual InterDev的数据视图工具 178
10.2 增加一个命令 180
10.3 把数据加入页面 182
10.3.1 设计时控件 183
10.3.2 显示数据 185
10.3.3 关于DTC 192

10.4 小结 193

第11章 DTC编程 194

11.1 编写代码来处理DTC 194

11.1.1 修改SQL语句（方法一） 196

11.1.2 修改SQL语句（方法二） 196

11.1.3 SQL Builder更有趣的一面 199

11.1.4 从记录集获取字段 202

11.2 创建“事件驱动”页面 203

11.2.1 建立“事件驱动”网页 203

11.2.2 增强“事件驱动”页面 206

11.2.3 事件的工作原理 207

11.3 小结 208

第12章 客户端编程以及Visual InterDev的未来 209

12.1 用Visual InterDev进行客户端编程 209

12.1.1 VI的动态HTML能力 209

12.1.2 远程数据服务(RDS) 211

12.2 未来的ASP+ 214

12.2.1 ASP+简介 214

12.2.2 ASP+小结 220

12.3 Visual Studio.NET初窥 220

12.4 小结 225

第4部分 创建N层Web应用

第13章 用ASP和COM组件构建N层应用 226

13.1 理解N层应用 226

13.1.1 两层系统 226

13.1.2 N层系统 228

13.2 理解并创建组件 229

13.2.1 理解COM 229

13.3 创建组件 232

13.3.1 开始创建NWindWeb组件 233

13.4 在Web应用程序中使用组件 240

13.4.1 创建Web页 241

13.4.2 意义 244

13.5 小结 245

第14章 COM+组件服务 246
14.1 对组件服务的需求 246
14.1.1 基础结构支持 246
14.1.2 探索组件服务 246
14.2 使用组件服务 252
14.2.1 扩展性 252
14.2.2 编程效率 260
14.2.3 事务 264
14.3 分发 272
14.3.1 分发应用程序 272
14.3.2 在客户机注册 273
14.4 小结 273

第15章 高级特性：扩展性和安全性 274
15.1 处理高负载 274
15.1.1 集群和网络负载平衡 274
15.1.2 会话状态和扩展性 275
15.1.3 负载检验 281
15.2 网站加密 281
15.2.1 加密和SSL 281
15.2.2 在IIS 5.0里建立SSL 285
15.3 应用级安全性 296
15.3.1 登录 296
15.3.2 COM+组件安全性 298
15.3.3 SQL Server 安全性 302
15.4 小结 303

第5部分 附 录

附录A IIS ASP对象模型 304
附录B ADO对象模型 311
附录C ADO扩展对象模型 320
附录D ADO多维对象模型 326
附录E HTML简介 332
附录F VBScript简介 349

第1部分　用ASP进行Web应用开发

第 1 章　Web 应用介绍

本书讨论的是以Microsoft SQL Server 2000为后端进行Web应用开发的技术。首先需要理解“Web应用”的意义以及如何利用微软的技术来构造Web应用程序。本章讨论的内容有HTML、IIS、ASP和n层开发，且假定读者已经熟悉了HTML（可以参见附录E）。

1.1　什么是Web应用

Web应用是指在Web上运行的应用程序。这听起来有些多余，不过请这样想想：许多网站实际上只是一个联机的目录，有时候也叫做“目录册”，它只是许多包含图片和描述文字的页面。说到构建应用程序时，我们指的是构建能做点什么的网站，可以输入信息，然后动态地做出应答。如果没有Web应用程序，我们就只可能写一些Web页，它能达到的交互效果跟一个目录册或杂志相比没什么两样。

Web应用覆盖了目前流行的电子商务的所有网站，在这些网站中，你可以浏览目录，同时可以选购物品放到购物车中，配置送货方式，然后付款，这个过程甚至连电话都不必打。许多网站提供实时的库存信息，这样访问者可及时了解到所需物品是否有货。另外还有一种B-to-B（business-to-business，商家对商家，与商家对客户对应的电子商务）应用。例如，A公司可以保存顾客的统计数据，其他公司就可以登录到A公司的网站，并利用他们的统计数据进行市场销路的报告分析。这就意味着这样的网站必须能够实时访问数据库，不仅可以读数据，而且还能够创建能反映出数据库的值的动态网页。动态网页的构建是Web应用的核心。

要创建Web应用，首先应该了解Web服务器和静态网页的工作原理。接下来我们就来看看如何用SQL Server数据库中的数据驱动的方式构建Web应用系统。

1.1.1　Web服务器和HTML

Web服务器的核心是非常简单的。客户端计算机连接到网络上，向某个特定的Web服务器请求某特定页面。该页面在Web服务器端就是存储在某物理分区上的一个文件。服务器定位到该网页后，把一份复制文件传送给请求的客户端计算机。如图1-1所示。

需要全本请访问 www.ertongbook.com

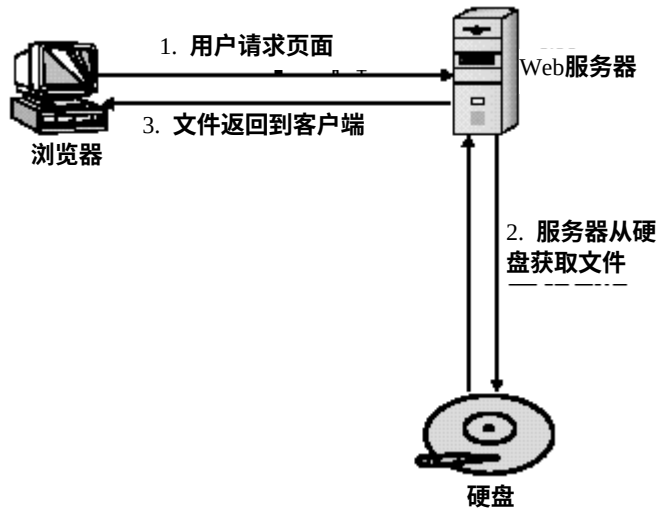


图 1-1 Web 服务器响应客户请求，返回一个页面的简图

客户端向Web服务器发的请求以及服务器返回给客户端的应答都遵循Internet的标准协议：HTTP（超文本传输协议）。HTTP传送客户端的请求，把它打包成Web服务器理解的格式。服务器应答时，返回给客户端的页面信息采用同样的HTTP协议。HTTP协议基于一个著名的网络协议——IP，即因特网协议。

因此Web服务器的核心其实就是向请求浏览器传送文件，这一点是理解静态HTML（超文本标记语言）的关键：Web服务器并不对文件进行任何处理。换句话说，Web服务器对于静态网页起不到什么作用。但是，大多数Web服务器的确具有在传送网页之前执行数据处理的能力，这种页面处理能力要求Web服务器具有某些智能格式，这是本书讲述的主要内容。

但首先需要理解简单的静态HTML文件的处理过程，若文件没有处理就传送给客户端，那它是怎么显示的呢?这是浏览器的工作，目前有两种最被广泛使用的浏览器：微软的IE浏览器和Netscape的Navigator浏览器，另外还有如Opera，WebTV和AOL（America OnLine，美国在线）。浏览器用来解释HTML文件并在屏幕上显示出来。

起初HTML开发的目的是使文档内容可以在任何硬件平台上显示。其方法就是根据含义标记文档内容，然后使浏览器按平台允许的方式处理显示事项。用微软Word编辑器创建文档时，Word以二进制格式保存文档，但这种二进制格式在其他平台的程序中不一定被兼容。因此HTML的发明者希望能创建一种文档格式，既可以保存文本信息，又可以做到平台兼容。这就要求HTML必须是全文本的，不能有二进制信息。

注意：许多人都致力于类似HTML的语言开发，不过最后是Tim Berners-Lee和Robert Caillau发明了HTML。Berners-Lee和Caillau在欧洲粒子研究所（CERN），就是日内瓦的一个物理研究中心工作，他们发明了HTML，把它作为提供CERN里不同计算机上的平台无关的链接信息系统的途径。1991年，CERN创造了如今我们所熟悉的Web。

为实现全文本的要求，HTML以标签（tag）来保存文档的格式。Web服务器并不处理这些标签，而是由客户端的浏览器来解释。浏览器一般是与平台相关的，不过它们都以完全一样的方式解释HTML，基于万维网（WWW）协会（位于马萨诸塞州的波士顿，网址为http://www.w3.org）颁布的标准。这意味着每个浏览器都会把标签<H1>解释成再现一条更大和更粗的文本字体的命令。

HTML的目标已经讲过，比较简单，就是在屏幕上显示文本，在平台间保留文本的格式信息（粗体、斜体还是下划线等等），允许链接信息等。但还要指出一点，许多人可能从来没有想到，HTML不是一种编程语言。编程语言必须要有某些元素，如变量、循环和判断结构（如if语句）。HTML没有其中的任何一种。HTML是一种计算机语言，但其目的只是保存文本格式信息。它不是编程语言，因此不能像其他应用程序那样执行函数。本章的后面部分将介绍一些方法，可以在HTML里加入一些客户端脚本，从而使网页可以进行一些处理操作。但需要意识到这些脚本是以VBScript或JavaScript写成的，而非HTML本身。在客户端运行代码需要HTML和脚本语言的配合。同样在服务器端以ASP运行代码也需要HTML和脚本语言，这些内容在第2章将有更多介绍。

> 注意：本书中将使用JavaScript的称呼。需要了解的是JavaScript是Netscape公司开发的一种语言，微软有自己的版本，叫做JScript。JavaScript和JScript十分相似，但也存在一些差异。由于这些差异又开发了第三种语言叫ECMAScript，它得到一个标准团体的认可，两种平台都支持此语言。
> 本书将一直使用“JavaScript”这个称呼，但请注意这里它只是一个通用术语，代表着JavaScript、JScript或者ECMAScript。

有一个很重要的问题是不同的浏览器对同一段代码的翻译和显示并不总是相同的。首先，计算机的显示配置参数可能不同。在800×600的显示器上表现很好的页面可能在640×480显示器上不尽如人意。另外，不同机器上的色彩深度设置也可能不同，因而造成页面的色彩和图形没有按预期的效果显现。并且，不同系统中的缺省字体可能不同，则页面在不同机器上的显示也可能不同。

不仅仅是同一代码在不同浏览器上的显示有轻微差别，不同的浏览器开发商还对标准HTML做了适当的扩展。这样就造成了一段代码可能只能在一个浏览器上工作。例如，<BGSOUND>标签是微软增加的，使网页可以有背景音乐，该标签就只能工作于IE上。另外浏览器开发商也会给已有标签增加属性。

这样我们就接触到了浏览器运作的另一个重要部分：浏览器被设计成可以忽略任何它所不认识的标签或属性。下面有一段代码：

```
<H1> Welcome to Northwind</H1>
  <CRAIG>This text is formatted with the Craig style</CRAIG>
```

结果怎样呢?根本没有<CRAIG>这个标签。浏览器看到这段代码时，只是简单地忽略了这个标签，把中间的文本当成正常的文本显示出来。浏览器希望标签可以指明文本的格式，当它不认得某个标签时，就会忽略掉它，而把中间部分的文本用缺省字体和大小显示。上面的代码显示情况见图1-2。

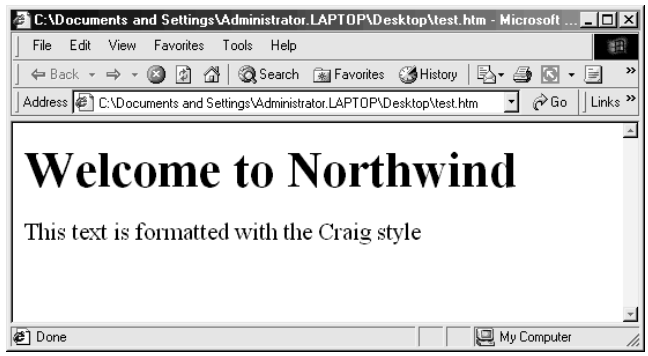


图 1-2 包含<CRAIG>标签的代码段

浏览器是相当宽容的，这是其优点，也是它的缺陷。优点之一就是客户端不会显示错误信息，而主要的缺陷就是开发者也看不到调试的出错信息。另外，HTML大小写不敏感以及标签不一定需要封闭（配对）的事实也造成了非常松散的结构，开发人员喜欢结构化，而HTML不强调好的结构化编码。网上有一些检验工具，如WWW协会的HTML Validation Server（http://Validator.w3.org）以及Netscape的Web Site Garage（http://Websitegarage.nets- cape.com）。

关于HTML的另外一个要点就是：一旦它提交到浏览器之后，若不刷新就不能有改动。所以要改变一个HTML页面，必须重新连到服务器上再请求一次页面。也就是说，页面文字一旦经浏览器解释后就不可能再修改了。这正是HTML被称为“静态”的理由。某些情况下，表格里的字段可以通过写一段客户端代码进行修改，也可以用动态HTML（DHTML）写一段可编程对象的代码。但是这些都需要某种客户端的脚本，而这些原来并不在HTML的内容里。而且并不是所有的浏览器都支持这样的风格。

1.1.2 客户端脚本编程

下面说一说客户端脚本编程。记住HTML不是一种编程语言，因此它本身不能做任何处理操作，不过我们可以在客户端脚本中处理某些动作。研究客户端脚本编程时有两点需要考虑：

- 客户浏览器是不是完全支持客户端脚本?
- 若支持，那它都支持什么语言的呢?

规则非常简单：VBScript为更多的开发者所熟悉，一般认为它更易于编程，但只运行在IE浏览器上。JavaScript 在IE和Navigator上都可运行。显然，出于应用范围更广泛的目标，应选择JavaScript。

> 注意：不要混淆JavaScript和Java Applet。Java Applet是预编译组件并被下载到客户端浏览器的Java虚拟机（JVM）中执行；JavaScript和VBScript不是预编译的，它们在客户端浏览器中被解释执行。

在学习HTML时，你应该会遇到HTML表格。表格允许用户输入信息，然后把这些信息传给服务器。服务器就可以对这些信息执行某些奇异的处理，比如添加进数据库等。经

常有人提问“怎么确认表格输入信息的有效性呢?”直接的HTML的回答很简单：不能。标准HTML没有提供检验表格字段中数据有效性的方法。就是说，若问用户的年龄，而用户输入的是“abc”，则“abc”被如实传送给服务器的年龄字段，在服务器上你必须自己检验信息的有效性。这不是什么坏事，不过请想一想：用户填写了表格，按下提交按钮，然后等待。数据通过Internet传到你的服务器，然后你进行数据有效性检查。若某个字段中的信息不正确，你需要创建一个页面返回给客户端，告诉客户错误信息。现在用户得到了一个错误信息，接着必须选择浏览器的后退按钮，返回到原来的表格，修改数据。整个过程大概要花去数秒钟。

但若使用客户端脚本，某些数据校验工作就可以在客户端进行。这样客户端将得到即时的反馈，而且是在同一页面上的，不必像上面那样用后退按钮返回。显然不是所有数据都能在客户端得到检验。有时候可能检验数据有效性需要访问数据库。通常我们并不直接从客户端访问数据库，尽管这一点可以办到（参见第12章）。

下面这个例子就是关于使用客户端脚本校验表格数据的。这是个过于简化的Web页面，包括一个表格，表格中有两个字段，若两个字段的数据输入都是正确的，此页面就被传送给服务器。否则通知用户进行修改。检查也很简单：在e-mail字段，只检查输入的字符串中是不是包含一个“@”符号。在生日字段，检查输入的是不是一个有效的时间。这段代码负责提交表格。图1-3是表格的处理效果。这里并没有真的提交表格，因为服务器端还没有作任何处理准备，另外还要注意这段代码是以VBScript写成的，因此只能在IE上运行。

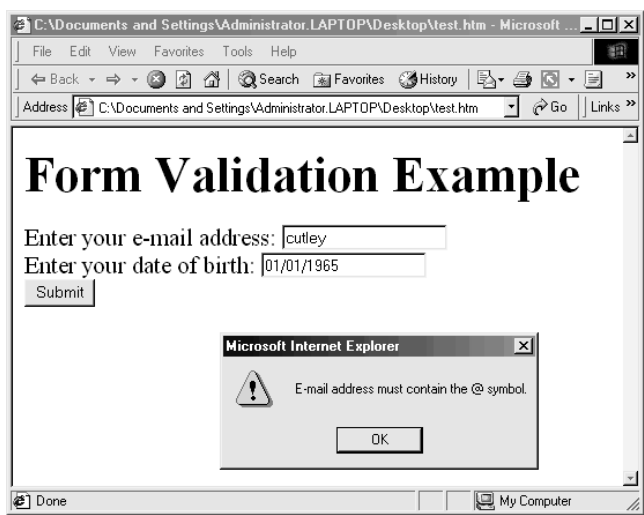


图 1-3　客户端脚本在表格提交前检查 e-mail 地址的有效性。这里代码检查到输入的 e-mail 地址不正确

```
<HTML>
    <H1>Form Validation Example</H1>
    <FORM name="frmSignUp" method="post">
        Enter your e-mail address: <INPUT name="email"><br>
        Enter your date of birth: <INPUT name="dob"><br>
        <INPUT type="button" name="cmdSubmit" value="Submit">
```

```
    </FORM>
    <SCRIPT language="VBScript">
        <!--
        Sub cmdSubmit_onClick ()
            If InStr (document.frmSignUp.email.value,"@") =0 Then
                alert "E-mail address must contain the @ symbol."
            Else
                If IsDate (document.frmSignUp.dob.value) then
                    alert "Data is fine,form will be submitted"
                Else
                    alert "Date of birth is not a valid date format."&_"Please
                re-enter."
                End If
            End If
        End Sub
        -->
    </SCRIPT>
</HTML>
```

还可以使用其他技术加强客户端功能，如使用 ActiveX**控件和**Java Applet**。这两种技术本身很不相同，但其最终效果类似，都是使客户端界面能够处理原本用**HTML**不能正常处理的操作。目前这两种技术的最大差别是：**ActiveX**只能运行在**IE**浏览器上，而**Java Applet在**IE**和**Navigator**上都可以运行，尽管其运行效果可能不尽相同。**

ActiveX**控件技术很流行，因为它在**Windows**世界中被大量采用，而且应用**Visual Basic、Visual C++**以及其他工具可以很容易地创建。但是**ActiveX**技术依赖于浏览器的能力。**ActiveX**控件在访问页面时被下载并在客户机上注册，然后一直保留在客户机上，这样用户再次**

访问页面时，ActiveX**控件已经在客户端存在并已注册，因此不必再次下载。**ActiveX**控件可以处理任何**Windows**程序能处理的操作：事件驱动、访问计算机硬件、读/写硬盘等等。因为这种访问硬盘的能力，许多人担心**ActiveX**技术的安全问题。的确，恶意的**ActiveX**控件将造成对用户机器的破坏。为防止出现这样的情况，提出了一些措施，如“认证码”技术，在第**15**章讨论安全性时我们还会详细介绍这一技术。**

Java Applet**可以是事件驱动的，并拥有更丰富的界面，但不能像**ActiveX**那样访问硬件。所以许多人认为**Java Applet**比**ActiveX**安全得多。另外它的应用更广泛，因为目前版本的**IE**和**Navigator**都支持它。有很多可用的**Java Applet**，例如股票行情表、倒计时器以及日历等。**

1.1.3 服务器端脚本编程

我们已经讨论了Web**服务器如何把一个静态的**HTML**流传送给浏览器进行解释，现在来看看服务器端如何动态创建网页。这么做的原因很明显：人们可能希望创建可以反映数据库数据、搜索结果以及实时信息等的页面。若没有**Web**服务器的动态创建网页的功能，就不可能有在线的购物车、库存清单或其他任何基于**Web**的应用系统。**

第一个动态创建网页的Web**服务器使用的是**CGI**技术（**Common Gateway Interchange**，通用网关交换）。**CGI**现今仍然通用，微软的**IIS（Internet Information Services，因特网信

息服务系统）可以像处理ASP一样处理CGI。CGI不是编程语言，它是一种技术，使我们能调用Web服务器上的已编译的程序和脚本。这种程序和脚本接受用户的输入数据，根据情况动态生成网页。CGI在基于UNIX的Web服务器上很常见，许多CGI程序是Perl脚本，从后缀名“ p1 ”可以看出来。

刚才提到，在IIS中可以用CGI，但IIS一般使用另外一种技术，它叫做活动服务器页面（Active Server Pages，ASP）。ASP使脚本可以直接嵌在HTML页中。这样，不像CGI，当动态创建网页时，不需要启动一个单独的服务器进程。ASP使脚本直接嵌在HTML页里，处理时可以只处理HTML的一部分。这种方法得到了广泛的应用，Java　Server Pages（JSP）就是用了非常类似的思想。

因为ASP在服务器端运行，因此脚本语言的选择由开发者确定。记住，使用客户端脚本时，必须要清楚浏览器是否支持脚本语言以及支持哪一种脚本语言。而使用服务端脚本时不必考虑客户端浏览器的能力问题，所有的代码都在服务器端处理，只把处理后的纯HTML传给客户端，因此客户端浏览器可以根本不支持任何脚本语言。实际上，若在客户端查看浏览器上的文档源代码，可以看到那就是静态的HTML，看不到任何用于创建网页的代码。

ASP提供了比HTML和客户端脚本复杂得多的应用环境。比如我们可能需要访问一台大型机上的数据或程序，这些程序处理大量的数据并返回处理结果。然后可能要把这些数据放入数据库进行进一步的处理，再生成最终页面返回给客户端。这样的应用过程如图1-4所示。这种过程，包括在大型机上运行程序、添加和查询数据库数据以及使用返回结果构成页面，已经超出了标准HTML和客户端脚本的能力范围，而支持服务器端脚本的智能Web服务器就可以做到，我们完全可以不去考虑客户端浏览器的能力。

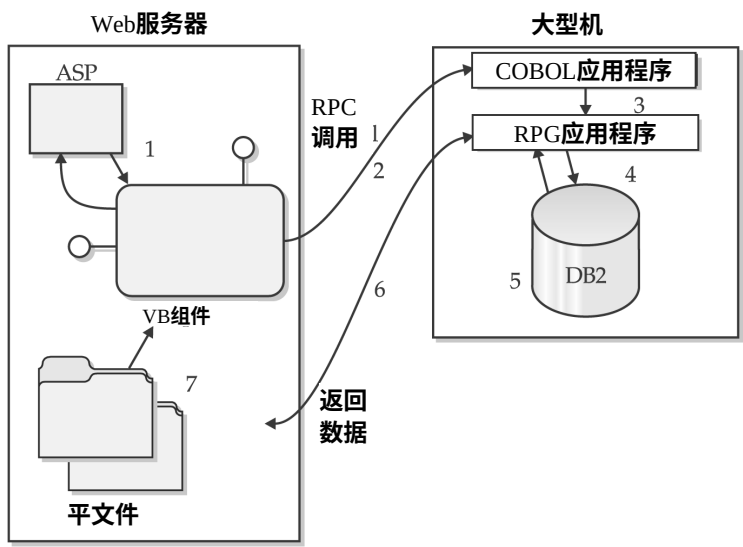


图 1-4　一个复杂的 Web 应用

若你在大型机环境下工作过，则会比较熟悉这一切。计算机最初发明出来的时候，其体积非常庞大，要占据成百上千平方英尺。所有的东西都在这一台机器上运行，人们通过

需要全本请访问 www.ertongbook.com