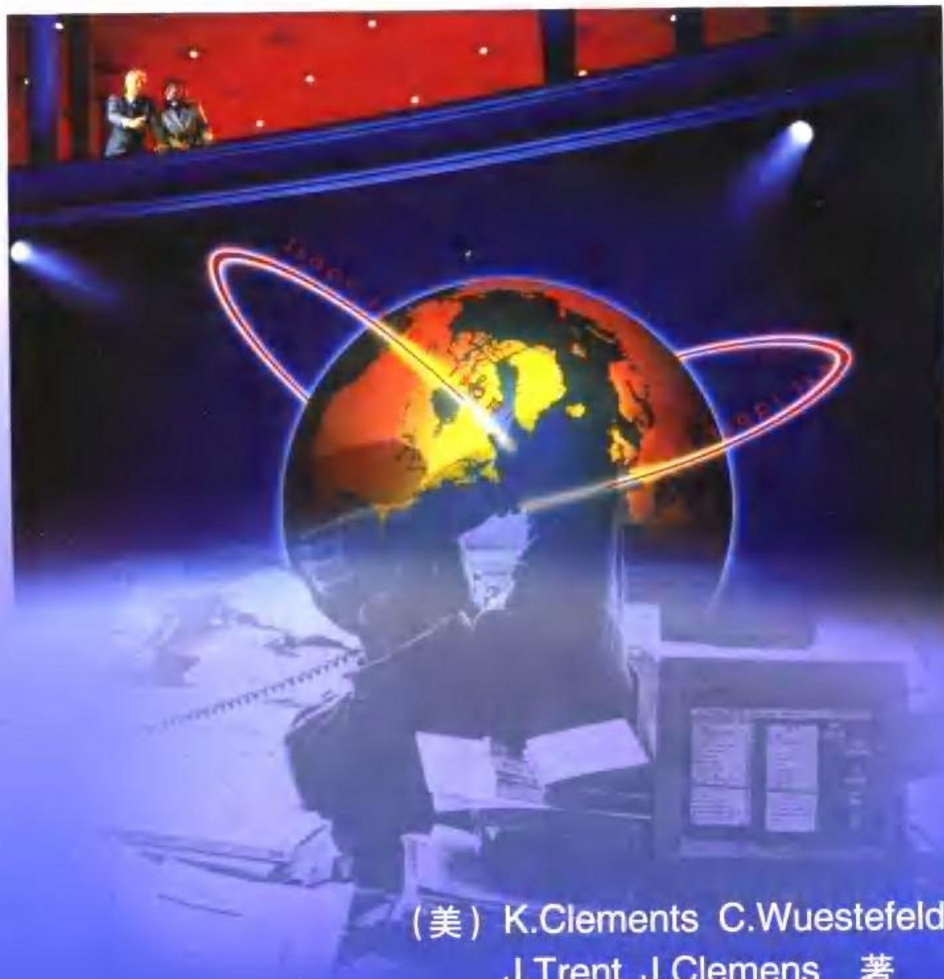




北京科海培训中心



(美) K.Clements C.Wuestefeld
J.Trent J.Clemens 著
朱玉山 王晓冬 译

ISAPI

实用技术指南

清华大学出版社

3/1/17

NRP

北京科海培训中心

ISAPI 实用技术指南

(美) K. Clements C. Wuestefeld

J. Trent J. Clemens 著

朱玉山 王晓冬 译

清 华 大 学 出 版 社

(京)新登字 158 号

著作权合同登记号:01-98-0884

内 容 提 要

本书由经验丰富的专家编写,全书针对具体任务、基于实际解决方案,详细讲述了完成 IS-API 编程的各个方面,内容包括:如何从数据库中提取多媒体文件,如何开发一个消息库,怎样通过过滤器增强日志,用 Delphi 编写与调试 ISAPI 程序。另外,本书通过一些生动的实例来证明方案的有效性,针对完成特定的任务提供了可编译运行的源代码,并详细介绍了如何有效利用过滤器的技巧来完成 ISAPI 应用程序,以实现功能强大的交互式 Web 站点的开发。通过本书你还能针对自己项目的特殊需要提出更合理的设计。

本书面向网络应用与开发人员。

Inside ISAPI

Copyright © 1997 by New Riders Publishing

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 publisher.

本书中文简体字版由美国西蒙与舒斯特公司授权科海培训中心和清华大学出版社。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 PRENTICE HALL 激光防伪标志,无标志者不得进入各书店。

书 名: ISAPI 实用技术指南

译 者: 朱玉山 王晓冬

出版者: 清华大学出版社(北京清华大学校内,邮编 100084)

印刷者: 北京门头沟胶印厂

发 行: 新华书店总店北京科技发行所

开 本: 16 印张: 24 字数: 580 千字

版 次: 1998 年 7 月第 1 版 1998 年 7 月第 1 次印刷

印 数: 0001~6000

书 号: ISBN 7-302-03053-7/TP·1628

定 价: 40.00 元

前言

欢迎使用《ISAPI 实用技术指南》一书,本书旨在帮助有经验的开发人员在 Microsoft 的 IIS(Internet 信息服务器)之上建立最新的交互式 Web 站点。

Internet 可能是当今世界上发展最快的技术领域了。就在不久以前,常为人们津津乐道的应用程序大都只提供静态展示,偶尔也会加入一些图片来点缀应用。

然而越来越难以满足网友的需要,他们提出了更高更复杂的要求,既希望程序能更吸引人,又具有更强的交互能力。同时,有这类要求的用户数目也在呈指数增长。这就要求新的应用程序必须在为更多的用户提供服务的同时还能保持同样的快速响应能力。

在这种情况下,程序员几乎无可适从。用户总是从已经超负荷运行的硬件中要求更多的功能,然而,应用平台及开发工具总是能够跟上新的变化步伐。特别是,服务器软件已变得越来越复杂,在提供简单的编程方式的同时还能带来优良的性能。

这些变化给开发人员带来了新的完全不同的问题。他们现在所能做的就是要紧跟新技术的步伐。当一个程序员对系统完全熟悉的时候,所有的规则可能又会再次改变。

最新的 Internet 服务器之一是 Microsoft 的 IIS。这一产品设置了新的性能标准,从而使它相对于其他同类 Web 服务器产品(包括 Netscape, O'Reilly, Process SoftWare 及其他等)具有更强的竞争能力。当然竞争的结果带给广大用户的则是更高质量、性能最佳的服务器(不管谁会从中取胜)。

IIS 同样带来了一个新的接口标准 ISAPI(Internet Sever Application Programming Interface),用来开发交互式 Web 应用程序。因此,对程序员而言,规则又不同了,他们又得重新去学一种新的方法。

作为一个开发人员,如果学习了 ISAPI,你可能会从中受益。这种新的架构能在 Web 应用开发方面带来更好的性能以及更大的灵活性。

读者对象

本书面向那些试图赶上这种最新技术潮流的开发人员。如果你打算在 Microsoft IIS 上构建新的应用程序,却又发现原有的很多经验在此已不再适用,那么你应该读一读本书了。

中级→高级 Windows 开发人员

本书的大部分内容面向有经验的开发人员。读者必须对下述编程概念较为熟悉:

- **Windows 编程:**许多 Windows 图形用户界面编程(GUI)技术对 ISAPI 编程同样重要。如果已经知晓怎样在 Windows 写多线程 GUI 应用程序,那么阅读完本书后同样也能写出类似的 Web 程序。
- **CGI 编程:**ISAPI 编程模型并不是什么革命性的创新,很显然它是从 CGI 模型演化来的。如果对 CGI,特别是对 HTTP 协议有所了解,将会对理解本书很有帮助。如果

这些术语对你来说是全新的概念,不妨到书店去找一本相关的书了解一下。这种书现在书店都有。

- **面向对象编程**:本书的大多数程序采用 C++ 或 Borland Delphi 编程语言。这两种语言本身都是面向对象的,在很多情况下,我们会更多的谈到这一概念。

第 2 章“使用 IDC”和第 3 章“IDC 示例”是以上编程概念的例外。这两章不仅有经验的开发人员可以一读,而且那些在企业 Intranet 上负责提供数据访问又无太多开发经验的管理人员也可一读。

Internet 和 Intranet 开发者设计交互式 Web 站点

本书的概念主要集中于如何在 Intranet 和 Internet 上开发应用。

本书的目标

本书集中讨论在一个特殊平台上一组特别的工具集,即 Windows NT 平台上的 Microsoft IIS,以及如何使用 ISAPI 接口去同服务器交互。

我们将努力在这一平台上为开发人员提供以下三方面的关键优点:

- 对大多数常见问题的解决方案
- 解决其他 ISAPI 问题的背景
- 相关的文档

本书应作为在这一平台上使用这一接口构建应用程序和指南。它并不是一本普及的 Web 开发教程,而集中于用这一模型构建 Web 应用程序和 Web 站点。必要的背景知识在相应的章节中将会出现,但记住本书不是讨论 Web 服务器的。

本书将以现实生活中的实际应用程序为例来讨论应用 IIS 和 ISAPI 接口的许多开发技巧。目的是使本书即成为此类开发的详细手册,同时也成为解决同类问题的解决方案的快速参考书。

理想情况下,构建表单处理应用程序的开发人员读完本书相关的章节,应当感到自己能解决手头面临的问题了。对于使用日志,重定向器,检索图片等要求也是一样。

我们认为,上述是本书非常重要的特色。我们,以及许多编程者,在碰到一个特殊的问题时都希望能找到像本书一样的参考资料:我们很清楚必须在 IIS 平台上提供一个解决方案——我们想知道怎样将数据库中的图形显示出来。本书将相应的 ISAPI 知识与特定问题相关联,这对于面向方案的开发人员来说是很有吸引力的。

对类似问题提供解决方案

Web 应用编程本身在飞快地发展,同时构建这些应用程序所允许的时间却在飞速减少。客户及老板们希望更快地获得方案以避免落在竞争对手的后面——这简直就是一个可怕的循环。

一方面,开发人员想更好地了解技术以使它们能更好的应用技术。另一方面,却没有足

够的时间来准备项目所需的经验和技巧。充满了高深理论及原型结构讨论的书可以说毫无用处,开发人员根本没有时间去消化它的内容。

结果是摸着石子过河,如果开发者比较幸运的话,就会获得一个可行的应用程序,虽然设计上可能由于知识不完备而显得散乱,但至少它是可用的。如果不幸运的话,项目可能会失败,因为缺乏足够的信息作出正确决策。

本书并不打算占用你更多的时间。然而,它可能会给出典型 Web 应用中所碰到问题的解决方案。完成了本书的各种实现之后,你将能针对自己项目的特殊需要提出更合理的设计。

基于个人经验以及在 Internet 邮件列表和 Usenet 新闻组中的谈话,我们了解到 ISAPI 开发人员经常碰到类似的问题。例如,如何访问多媒体信息,如何控制对应用程序资源的访问。虽然每一应用程序都有自己的特殊技巧,但解决方案都是类似的。

本书的第一个目的就是罗列出 ISAPI 开发人员可能会经常碰到的问题。这些问题必须具有足够的代表性以体现 ISAPI 的重要特性。

提供解决其他 ISAPI 问题的背景知识

本书的第二个目标是对所提供的解决方案作出更深一步的分析。开发者必须知道怎样将一种特殊的方法运用到自己所需解决的问题中,本书各章节不但提供了解决方案的源代码,而且还详细分析了为什么要这么做以及这些方案是怎样起作用的。

阅读本书的其他部分——不管你是否指望碰到所描述的问题,都会使你更全面的了解 ISAPI 的工作机理以使你可以更充分的利用它。

提供相关的文档

由于 ISAPI 编程属于较新的技术,因此很难找到必要的参考资料以理解这一技术。而现在看得见的资料经常混乱不堪而且错误不断。

例如,过滤器通知 SF_NOTIFY_END_OF_REQUEST 是新出现的,在本书写作时它并未出现在任何 Microsoft 的出版物中。另一个例子是由 Server SupportFunction() 的文档造成的混淆,在一些文档中,Microsoft 看上去在过滤器和扩展环境中混淆了对这些函数用法的解释(它们是一个不同的函数,虽然名字相同)。

我们的目标是让你最大限度的利用 ISAPI。希望你能够走最少的弯路——作为作者在此付出的代价应已足够作为尝付这一新知识的学费了。

必须的工具

Internet 应用经常对机器的硬件软件以及其他一些工具提出越来越多的要求。打算构建这些应用程序的开发人员必须首先准备好满足要求的机器以及最新的工具。

硬件要求

在这一平台上构建 Internet 应用程序最主要的要求就是机器必须能运行 Windows NT 4.0。最少 32M 内存,多了当然会更好。如果同时还打算运行 SQL Server,最好增加更多的内存。

软件要求

本书中提供的各种方案一般需要以下软件:

- Windows NT 4.0 或更新版本:要布署 Internet 应用程序需要服务器版,开发工作可在工作站版上完成(结合 PWS)。
- 因特网信息服务器 2.0 或更新版本(或用 PWS 在工作站上完成开发)

Microsoft Visual C++ 4.2:本书中所有 C++ 代码都用这一编译器写成的,更早版本的 C++ 在 ISAPI 支持及标准模板库实现方面可能有所不同。一般更新的版本则保持向后兼容。

注意:C++ 5.0 的用户可能需要作些小的改变。特别是标准库现在的名字空间重新实现上。

因此,任何对这些库中成分的引用现在必须改为对相应的名字空间的引用。这可以简单的在每个引用前放置 `std::` 完成(如 `std::string`),或者用名字空间直接定义到现在的名字空间(如 `using namespace std;`)。

第 2 章和第 3 章主要讨论 IDC,并不需要编辑器。第 12 章和第 13 章需要 Borland Delphi 2.0 或更新版本。

目 录

第1章 Internet 信息服务器的编程选择	(1)
1.1 CGI(公共网关接口)	(1)
1.2 IDC(Internet 数据库连接器)	(2)
1.3 其他可选择的开发工具	(2)
1.3.1 OLEISAPI	(2)
1.3.2 活动服务器页面(Active Server Pages)	(3)
1.3.3 第三方解决方案	(3)
1.4 基于 ISAPI 的解决方案	(4)
1.4.1 性能	(5)
1.4.2 灵活性	(6)
1.4.3 扩展 DLL	(6)
1.4.4 过滤器	(7)
1.4.5 用 ISAPI 编程	(9)
1.5 小结	(11)
第2章 使用 IDC	(12)
2.1 什么是 IDC	(12)
2.2 何时使用 IDC	(12)
2.3 IDC 编程基础	(13)
2.3.1 IDC 文件	(14)
2.3.2 .HTX 文件	(17)
2.4 高级 IDC 编程	(22)
2.4.1 理解 IDC 处理请求的顺序	(22)
2.4.2 使用多数据库查询	(24)
2.4.3 检索计算域的值	(25)
2.4.4 使用多值变量	(25)
2.4.5 HTML 中的特殊字符处理	(27)
2.4.6 保证系统的安全	(28)
2.5 IIS 1.x 和 IIS 2.0 之间的区别	(29)
2.5.1 多数据库查询	(29)
2.5.2 适当的条件嵌套	(29)
2.6 小结	(29)
第3章 IDC 应用程序示例	(30)
3.1 应用程序目标	(30)
3.1.1 消息的组织	(30)
3.1.2 发布新消息	(30)
3.1.3 主题搜索	(30)

3.1.4 安全	(31)
3.2 消息库应用程序数据模型	(31)
3.2.1 消息	(32)
3.2.2 议题	(32)
3.2.3 用户	(33)
3.2.4 空白表	(33)
3.3 消息库应用程序用户界面	(34)
3.4 消息库应用程序实现要点	(35)
3.4.1 访问故障排除	(35)
3.4.2 配置系统	(35)
3.4.3 提供数据库中的初始数据	(35)
3.4.4 通过程序传播数值	(36)
3.4.5 排序消息响应	(36)
3.4.6 实现多页搜索结果	(38)
3.5 消息库应用程序的实现	(39)
3.5.1 Welcome/Login 页	(40)
3.5.2 Main Page	(41)
3.5.3 Conference Message List 页	(44)
3.5.4 View Message 页	(48)
3.5.5 Post a Follow-Up 页	(54)
3.5.6 Post a New Topic 页	(58)
3.5.7 Search Request 页	(62)
3.5.8 View Search Results 页	(64)
3.6 小结	(69)
第4章 MFC 中的基本表单处理	(70)
4.1 Web 表单	(70)
4.2 ISAPI 处理表单的优点	(71)
4.3 在 ISAPI 表单处理中使用 MFC 的优点	(72)
4.4 表单处理的一个简单的 ISAPI 扩展	(73)
4.4.1 设计表单	(73)
4.4.2 表单的 HTML	(74)
4.4.3 处理表单数据的一个 ISAPI 扩展	(76)
4.4.4 表单处理逻辑	(77)
4.5 HTML 中的表单	(80)
4.5.1 表单	(80)
4.5.2 输入域	(81)
4.5.3 选择列表	(84)
4.5.4 文本区域	(85)
4.6 MFC PARSEMAP	(85)
4.6.1 BEGIN_PARSE_MAP	(86)
4.6.2 ON_PARSE_COMMAND	(86)
4.6.3 ON_PARSE_COMMAND_PARAMS	(86)

4.6.4	DEFAULT_PARSE_COMMAND	(87)
4.6.5	END_PARSE_COMMAND	(88)
4.7	一个更高级的表单处理例子	(88)
4.7.1	表单和有效验证	(88)
4.7.2	进程	(88)
4.7.3	调用表单	(89)
4.7.4	动态产生的 HTML	(90)
4.7.5	处理用户的输入	(94)
4.7.6	动态产生 HTML 的优缺点	(98)
4.8	潜在的表单处理问题	(99)
4.8.1	ISAPI 扩展的故障排除	(99)
4.8.2	GET 命令问题	(100)
4.8.3	mfcISAPICommand 参数	(100)
4.9	何时 PARSE MAP 不工作	(101)
4.9.1	带有可变内容的表单	(101)
4.9.2	多值域	(102)
4.10	小结	(103)
第5章	从数据库中抽取多媒体图像	(104)
5.1	动态多媒体网页创建	(104)
5.1.1	在数据库中不仅仅存档图像	(105)
5.1.2	生成工具	(105)
5.2	MIME 类型	(105)
5.2.1	更多有关 MIME 类型的内容:StartContent()的调用	(106)
5.3	嵌入 ISAPI 响应调用	(107)
5.3.1	在一个 ODBC 数据库中存储 BLOB	(107)
5.4	图像抽取模块	(109)
5.4.1	使用 CRecordset 抽取图像	(112)
5.4.2	测试驱动 mmGet 扩展	(116)
5.5	线程安全(thread-safe)的图像抽取	(118)
5.6	高级话题	(122)
5.6.1	扩展性,使用 worker 线程	(122)
5.6.2	用信号灯优化线程访问	(123)
5.6.3	在注册表中存储数据库用户的名字和口令	(123)
5.6.4	增强数据库用户的权限管理	(123)
5.6.5	使用 cookie 保持状态	(123)
5.7	ISAPI 扩展的故障排除	(124)
5.8	小结	(124)
第6章	开发 E-mail 网关	(126)
6.1	SMTP 概述	(126)
6.2	从网页发送邮件的最简单方法	(128)
6.3	捕获关键信息	(129)
6.4	实现一个 SMTP Mailer DLL	(129)

6.4.1 SMTP Mailer API	(131)
6.4.2 实现 SMTP Mailer	(133)
6.5 SMTP Mailer 的测试	(140)
6.6 采用 SMTP Mailer 的 ISAPI 扩展	(141)
6.7 LiteMail SMTP 网关 ISAPI 扩展	(143)
6.8 小结	(149)
第7章 理解 ISAPI 过滤器	(150)
7.1 过滤器定义	(150)
7.2 ISAPI 过滤器和扩展	(150)
7.2.1 ISAPI 过滤器应用程序	(150)
7.3 ISAPI 过滤器的寿命周期	(151)
7.3.1 装载和卸载过滤器	(151)
7.3.2 配置过滤器	(152)
7.4 ISAPI 过滤器 DLL 入口点	(154)
7.4.1 GetFilterVersion()函数	(154)
7.4.2 HttpFilterProc()函数	(155)
7.5 线程	(163)
7.5.1 资源管理	(163)
7.5.2 环境管理	(163)
7.6 内存管理	(164)
7.6.1 使用 AllocMem()作内存管理	(164)
7.6.2 使用通知作内存管理	(165)
7.7 用 C 建立一个过滤器	(166)
7.7.1 实现一个简单的过滤器	(166)
7.7.2 运行过滤器	(171)
7.7.3 理解过滤器输出	(172)
7.8 MFC 过滤器支持	(174)
7.8.1 CHttpFilter	(174)
7.8.2 一个基于 MFC 的 GetFilterVersion()函数	(175)
7.8.3 CHttpFilterContext 类	(178)
7.9 小结	(178)
第8章 用过滤器增强 IIS 日志	(179)
8.1 IIS 日志	(179)
8.1.1 记录文本文件	(179)
8.1.2 记录到数据库	(181)
8.1.3 确定在何处以及如何存储数据	(182)
8.2 过滤日志事件	(182)
8.2.1 过滤日志事件的理由	(182)
8.2.2 增加新域给日志	(183)
8.2.3 更改已有日志域	(184)
8.3 一个简单的日志过滤器	(184)
8.3.1 设计 ISAPI 日志过滤器	(185)

8.3.2 实现 CExtraDataFilter::GetFilterVersion()	(185)
8.3.3 用 Onlog()函数处理日志通知	(186)
8.4 有效地使用日志过滤器	(187)
8.4.1 避免耗时的操作	(188)
8.4.2 使用离线选择	(188)
8.5 小结	(189)
第9章 基于客户浏览器重定向	(190)
9.1 为什么以用户浏览器为基础进行重定向	(190)
9.1.1 优化站点还是使站点更具普遍性	(190)
9.1.2 自动选页的手工执行	(191)
9.2 用过滤器重定向	(192)
9.3 用扩展重定向	(192)
9.4 描述浏览器功能	(193)
9.4.1 识别远程浏览器	(193)
9.4.2 建立一个浏览器功能机制	(194)
9.5 实现一个重定向扩展	(205)
9.5.1 要求	(205)
9.5.2 声明扩展类	(205)
9.5.3 定义 PARSEMAP	(206)
9.5.4 初始化重定向扩展	(206)
9.5.5 处理 IfSupported 命令	(208)
9.6 实现重定向过滤器	(210)
9.6.1 要求	(210)
9.6.2 为重定向过滤器选择通知方法	(211)
9.6.3 设计发送	(211)
9.6.4 声明过滤器类	(211)
9.6.5 实现 GetFilterVersion()函数	(212)
9.6.6 实现 OnPreProcHeaders()函数	(213)
9.7 小结	(214)
第10章 用过滤器验证	(215)
10.1 验证与安全性	(215)
10.2 在 HTTP1.1 中验证	(216)
10.2.1 Challenge/Response 进程	(216)
10.2.2 验证方案	(218)
10.3 验证和 Windows NT 安全	(220)
10.3.1 匿名用户	(221)
10.3.2 安全环境和 IIS	(221)
10.3.3 NT 访问权限和验证	(221)
10.4 验证过滤器	(222)
10.4.1 验证过滤器做什么	(222)
10.4.2 为什么使用验证过滤器	(222)
10.5 设计和实现验证过滤器	(223)

10.5.1 数据存储要求	(223)
10.5.2 执行效率要求	(223)
10.5.3 设计过滤器	(224)
10.5.4 实现过滤器	(224)
10.6 小结	(233)
第11章 高级表单处理	(234)
11.1 MFC PARSEMAP 的局限性	(234)
11.2 MFC ISAPI 支持的优点	(234)
11.3 设计一种机制来支持灵活表单内容	(235)
11.3.1 需要的功能	(235)
11.3.2 环境特征	(235)
11.4 使用未提供文档的 MFC 来扩展或增强 MFC	(237)
11.4.1 什么是未被提供文档的 MFC	(238)
11.4.2 未被提供文档的 MFC 代码的风险和回报	(238)
11.4.3 使用未被提供文档的 MFC 的基本原理	(238)
11.5 实现处理可变表单内容的增强	(239)
11.5.1 扩展 CHttpServer	(239)
11.5.2 用 CHttpFieldTable 类管理数据	(245)
11.6 在一个 ISAPI 扩展中使用 CHttpFieldServer 类	(257)
11.6.1 转换一个 MFC ISAPI 扩展来使用 FieldTable 类	(257)
11.6.2 一个域表扩展样本	(257)
11.7 小结	(260)
第12章 使用 Borland Delphi 2.0 的 ISAPI 编程	(261)
12.1 使用 Delphi 的好处	(261)
12.2 ISAPI 中 Delphi 的局限性	(261)
12.2.1 在 ISAPI 程序中没有用户接口	(262)
12.2.2 数据库问题	(262)
12.2.3 缺少开发 ISAPI 应用的应用程序框架	(262)
12.3 Delphi 与 C++ ISAPI 编程	(262)
12.3.1 数据类型对应	(263)
12.3.2 MFC 或其他框架不可用	(263)
12.4 用 Delphi 写一个 ISAPI 应用程序	(263)
12.4.1 ISAPIExt 单元	(264)
12.4.2 在 DLL 中提供 ISAPI 应用程序入口点	(266)
12.4.3 使用 ECB 与服务器交互	(269)
12.4.4 检索头部信息和服务器变量	(269)
12.4.5 解释请求	(270)
12.4.6 发送数据给用户	(271)
12.4.7 保持状态信息,使用 Cookies	(273)
12.5 编写一个 ISAPI 过滤器	(275)
12.6 小结	(275)

第13章 使用 Delphi ISAPI 应用程序框架	(276)
13.1 设计目标	(276)
13.2 使用 ISAPI 应用程序框架	(277)
13.2.1 为你的扩展建立一个新项目程序	(277)
13.2.2 用 ISAPICOMMAND 域准备 HTML 表单	(277)
13.2.3 编写并注册一个类来处理请求	(278)
13.2.4 HTTPServerContext 对象	(280)
13.2.5 检索表单的域值	(281)
13.2.6 检索头文件信息和服务器变量	(282)
13.2.7 发送头文件给用户	(282)
13.2.8 获取对参数和 Cookies 的直接访问	(283)
13.2.9 写数据给用户	(285)
13.2.10 处理异常	(285)
13.3 建立一个简单的请求处理程序	(286)
13.3.1 温度转换请求	(286)
13.3.2 温度转换请求处理程序	(287)
13.4 高级专题	(291)
13.4.1 HTTPUtil 工具类	(291)
13.4.2 读写 Cookies	(293)
13.4.3 写二进制数据给用户	(294)
13.4.4 直接访问特殊函数的 ECB	(294)
13.5 Delphi ISAPI 应用程序框架如何工作	(294)
13.5.1 作为函数符的请求处理程序	(295)
13.5.2 类注册和对象工厂	(297)
13.5.3 检索输入数据	(301)
13.5.4 保存和发送头部	(314)
13.5.5 异常结构	(319)
13.6 小结	(323)
第14章 调试 ISAPI 过滤器	(324)
14.1 重入	(324)
14.2 代码的关键部分(Critical Section)	(325)
14.3 内存泄漏	(326)
14.4 在 Windows NT 下调试	(326)
14.4.1 在调试器中浏览代码	(327)
14.4.2 建立调试	(327)
14.4.3 在调试器中装入 Web 服务器	(327)
14.4.4 用 OutputDebugString()函数调试	(327)
14.4.5 用 MessageBox()函数调试	(328)
14.4.6 更方便的调试方法提示	(328)
14.4.7 用 ISMOKE.EXE 调试	(329)
14.4.8 用 CGIWRAP 调试	(329)
14.5 使之完善	(329)

14.5.1	ASSERT 语句	(330)
14.5.2	运行时调试库	(331)
14.5.3	提供错误消息	(332)
14.5.4	强化测试	(332)
14.5.5	用 Hammer 强化测试	(333)
14.6	小结	(340)
附录A	ISAPI 函数、结构和常量	(341)
A.1	扩展	(341)
A.1.1	DLL 入口	(341)
A.1.2	回调	(343)
A.2	过滤器	(346)
A.2.1	DLL 入口点	(346)
A.2.2	回调	(352)
附录B	建立一个系统数据源名	(357)
B.1	增加一个系统数据源名	(357)
B.2	配置一个访问系统数据源名	(358)
B.3	配置 SQL 服务器的系统 DSN	(369)
附录C	HTTP 状态码	(360)
附录D	服务器变量	(362)
附录E	常用的头部	(363)
附录F	光盘上有什么?	(365)
F.1	源代码	(365)
F.2	开发工具	(365)
F.3	文献	(365)
F.4	ISAPI 组件	(366)
F.5	网络链接	(366)

第 1 章 Internet 信息服务器的编程选择

在 Web 服务器这个竞争激烈的领域里,微软的 Internet 信息服务器——IIS(Internet Information Server)是当今 Windows NT 操作平台上执行效率最佳的 Web 服务器之一,甚至有些广告说它是执行效率最佳的。由于 Windows NT 在当今商业上的广泛应用,再加上 IIS 的价格优势(免费),确实值得花些时间在你的网址上探索这种服务器的使用。

IIS 提供所有传统的内容传递方式,它对静态网页有着良好的支持。另外,动态网址能通过 CGI(公共网关接口)实现,并且添加两三个新的因素,未来的 IIS 版本将会日臻完善。

对只打算基于数据库内容生成简单的动态页面的应用程序,IIS 提供了 IDC(Internet Database Connector)。利用它几乎不需要多少编程专业知识就可以快速创建一个应用程序。

对更加复杂的应用程序,IIS 提供了功能更强的应用框架。应用 ISAPI(Internet Server Application Program Interface)能够创造极高性能的应用程序。用 ISAPI 编程可能比用 IDC 或 CGI 及 Perl 花费更多的时间,但是它的优点是极其巨大的。书写良好的 ISAPI 的扩展其性能可能超过类似的 CGI 应用程序一个数量级。此外 ISAPI 灵活性质一些事情变得简单,它可以完成某些其他方法不能完成的工作。

在 IIS 中编写动态 Web 应用程序也有其他的方法。许多第三方的工具使你能在 Visual Basic 下编程。微软的 ActiveX 及 OLE 技术同样也是作为功能强大的第三方编程工具应用在 Web 服务器上的。特别是第三版的 IIS 支持活动服务器页面(Active Server Page),在这些页面里,你可以把在网络服务器上运行的 ActiveX 对象之间的交互作用改写。

1.1 CGI(公共网关接口)

使用 CGI 编程在 Web 应用程序开发这一新兴领域中算得上是老前辈了。与其他方法相比,其性能很差,需要 HTTP 协议的大量知识,而且 CGI 应用程序不如 ISAPI 灵活。

术语 CGI 实际上指的是一种方法结构,而不是一种特殊的工具或编程语言。尽管 Perl 是迄今为止最流行的编程选择,但事实上,CGI 应用程序几乎可用任何一种语言来编写。公共网关接口还有其他几种变体,其中之一为 WinCGI,据了解它在提高性能方面比一般 CGI 要好。

在 CGI 方法中,Web 服务器通过执行一个外部程序来处理每一个应用要求。服务器分析应用请求的内容并把这个信息传给 CGI 程序。CGI 程序可以对输入信息进行必要的处理。给客户机回送数据通过标准输出设备进行,网络服务器会接收输出并把它传给客户机。

CGI 最突出的缺点是操作太慢。对接到的每一个请求,CGI 方法要求服务器执行一个外部程序。这涉及到在操作系统中创建一个新的进程,从磁盘上装载可执行映像,并在完成时再把它全部清除。另外,在每一次调用时(如数据库连接资源)必须重新建立。它们既不能被缓存也不能重用,因为在两个调用之间进程不能保持。

在许多情况下这可能并不会造成麻烦。如果应用请求不是很多,以致于性能问题不必担

忧,为了在最短的时间内完成应用开发,CGI 可能是一个较好的选择。

1.2 IDC (Internet 数据库连接器)

通过使用 Internet 数据库连接器 (IDC, Internet DataBase Connector) 可以发布你的数据库,允许用户访问数据。这仅需要开发者编写最少量的代码。你只需提供 SQL 查询 (IDC 将在数据库上执行它) 和模板 (描述数据显示和收集的方式),而不用进行复杂的程序设计。

实现一个 IDC 应用程序不需要任何诸如 C++, DLL 或类似的编程概念知识,而这些在 ISAPI 中则是无处不在,在 CGI 中要求的程度稍低。但它的确需要 HTML 及 SQL 的知识。这种简单性是以牺牲应用程序的稳健及灵活性为代价的。通过应用程序接口你本可以实现大量的控制以及验证数据是否有效。

对大多数的应用来讲采用折衷是合适的。例如,一个只有本部门的成员才使用的简单的应用程序不需要特别的精巧,你可以告诉用户他们所输入的数据可能会有什么样的影响,从而省去在应用程序中验证数据有效性的考虑。

因此,对这一类应用程序 IDC 可以做得很好。对更复杂或更高的需求,需要使用其他的工具。

1.3 其他可选择的开发工具

在开发 Web 应用程序方面决定采用哪种方法需要考虑大量的因素。这些因素包括:

- **应用程序的性能:** 应用程序对用户请求的反应有多快,它能支持多少用户?
- **实现应用程序的难度:** 同技术联系在一起的学习曲线是什么,此方法需要编写多少代码?
- **稳健性:** 应用能否经受特殊的环境,包括无意的错误使用和有意的黑客攻击?
- **组件的可重用性:** 投资于应用程序的工作能否移植到未来其他的应用程序中,已有的代码能否在这一应用程序中使用?

所有上述因素为折衷考虑提供了机会。许多开发者包括微软已经为开发 Web 应用程序提供了许多开发方法,它们各有所长。ISAPI 除了实现较为复杂外,在其他方面表现相当突出。CGI 容易实现,但性能很差。为 IDC 所写的代码功能不是很强,而且不能移植到其他的平台上。

除了用 CGI, IDC, ISAPI 以外,下面几种可供选择的工具也可用于编写交互式的 Web 应用程序。

1.3.1 OLEISAPI

OLEISAPI 是 IIS 和 OLE 服务器之间的一种接口。它能让你通过使用 Visual Basic 一类的工具,开发出作为 OLE 服务器的 Web 应用程序。

OLEISAPI 最初是由微软公司作为一个 ISAPI 和 OLE 交互作用的例子提供的。作为一个分解样本程序,把它用于关键任务系统是不合适的。而且其原始代码中存在一定的缺陷,