第一图书网, tushu007.com

<<Windows PowerShell高 >

图书基本信息

书名：<<Windows PowerShell高级编程>>

13位ISBN编号：9787302188674

10位ISBN编号：730218867X

出版时间：2008-1

出版时间：清华大学出版社

作者：（美）库马拉尔（Kumaravel,A），（美）怀特（White,J） 等著；冯权友 译

页数：333

译者：冯权友

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：http://www.tushu007.com

前言

在各种发行版的操作系统中，Windows操作系统安装使用得最广泛、最受用户欢迎。
但对系统管理工作者而言，Windows一直有个小小的缺憾，就在于没能提供像UnixShell程序那样的强大脚本支持，而使得系统管理员的工作效率不高。
PowerShell的问世彻底改变了这种局面。
PoweiShell是构建在Windows．NET平台之上的，与．NET框架紧密耦合，因此它不仅是一种脚本编程语言，还为系统管理员提供了功能丰富的编程开发环境，可以轻易操作系统中的COM组件。
目前，市场上已经出现了介绍如何使用WindowsPowerShell来提高系统管理效率的书籍，但是从API层次介绍PowerShell编程的权威之作相当稀少。
本书比SDK文档更加详尽地介绍了cmdlet、提供者（proyider）、snap-in、宿主（host）程序等方面的开发技术。
本书针对PowerShellsnap-in和宿主（host）程序开发人员而写，它从WindowsPowerShell底层API开始介绍PowerShell程序设计。
书中每个知识点都有配套的实例代码，可以从本书的合作站点上自由下载，极易验证。
本人是计算机专业的在读博士生，经常在Linux下进行脚本编程。
接触WindowsPowerShell一年多来，渐渐被它强大的功能所吸引。
在Windows操作系统中，PowerShell提供了能和Linux下的bash相媲美的强大功能。
本书翻译过程中也加入了个人的使用体会，希望能和读者在PowerShell技术方面进行探讨交流，共同进步。
翻译过程中错误之处在所难免，敬请广大读者提供反馈意见，读者可以将意见发到wkservice@vip．163．com，我们会仔细查读者发来的每一封邮件，以求进一步提高本书的质量。

内容概要

PowerShell是一种新的命令行外壳和脚本语言，用于进行系统管理和自动化。
本书的作者由微软专家团队组成，从程序员的角度对Windows PowerShell编程技术进行介绍，不仅讲解了PowerShell的核心模块和基本概念，还提供了大量利用PowerShell强大功能构建软件包的实用开发技巧，是学习PowerShell编程的最佳参考。
在学习过程中，您不仅会发现PowerShell的SDK为应用程序提供了完整的技术支持，还将体验PowerShell的伞新技术，它使所有．NET Framework对象都可以通过脚本访问，这也使得PowerShell有望成为最受欢迎的工具和对Windows管理员来说最重要的语言。
一旦升始编写命令行类的上具程序，您就会惊奇地发现PowerShell可以使您将注意力集中在核心业务逻辑上。
相信这款由微软精心打造的软件会为您的编程生活带来额外的惊喜和收获。
本书主要内容 •创建PowerShell的Snap—in •开发自定义的Cmdlet •创建自定义的提供者 •调用PowerShell执行引擎 •实现宿主用户接口 •类型扩展及格式化 本书读者对象 本书面向希望拓展Windows PowerShell功能，或者使用PowerShell技术来扩展自己的应用程序的开发人员和程序员。

作者简介

Arul Kumaravel是目前Windows：PowerShell开发小组的负责人。
从早期开始，他就参与了这个项目并领导该小组发布了第一版的Windows PowerShell产品。
现在，他正领导着下一个版本的PowerShell产品的开发。
Arul从他年轻时学习BASIC编程开始，就着迷于计算机技术。
他在Iowa大学和印度Madras工程学院都获得了计算机科学方面的硕士学位。
在微软实习期间，他为IE3浏览器编写了第一个JavaScfipt／VBScfipt调试器，同时微软公司的氛围也留给了他很好的印象，他决定为微软效力而改变数以万计人的生活。
在微软任职的11年里，他在各种各样的小组工作过，发布了很多个版本的产品，包括IE浏览器、Windows操作系统、目录管理服务器。
最近，由于对科学技术的商业运作感兴趣，Arul开始在Wharton商学院刻苦攻读M.B.A。
他的邮件地址是arulk@hotmail．com。

书籍目录

第1章 PowerShell简介 1.1 Windows PowerShell设计原则 1.1.1 保留用户已有的投资 1.1.2 提供一个功能强大、面向对象的外壳程序 1.1.3 扩展性是第一位的 1.1.4 剔除开发过程中的障碍 1.2 Windows PowerShell快速入门 1.3 Windows PowerShell的高层体系结构 1.3.1 宿主程序 1.3.2 Windows PowerShell引擎 1.3.3 Windows PowerShell snap-in 1.4 小结第2章 扩展Windows PowerShell 2.1 PowerShell snap-in分类 2.2 编写标准的PowerShellsnap-in 2.2.1 编写PowerShell snap-in 2.2.2 注册PowerShell snap-in 2.2.3 查看可用的PowerShellsnap-in列表 2.2.4 将PowerShell snap.in动态装载到外壳程序中 2.2.5 从外壳程序中动态删除snap-in 2.2.6 取消注册snap-in 2.2.7 注册没有实现类的PSSnapin snap-in 2.2.8 保存snap-in配置文件 2.2.9 用保存的snap-in配置文件启动PowerShell 2.2.10 使用配置文件(profile)保存snap-in配置 2.3 创建自定义的PowerShell snap-in 2.3.1 编写自定义的snap-in 2.3.2 使用自定义的snap-in 2.4 小结第3章 理解PowerShell扩展类型系统 3.1 PSObject 3.2 构造PSObject对象 3.2.1 PSObject(object) 3.2.2 PSObjecto 3.2.3 PSObject.AsPSObject(someobject) 3.3 ImmediateBaseObject属性和BaseObject属性 3.4 成员 3.4.1 PSMemberlnfoCollection 3.4.2 ReadOnlyPSMemberInfoCollection 3.4.3 基类成员、适配器成员和扩展型成员 3.5 成员分类 3.5.1 属性 3.5.2 方法 3.5.3 集合 3.6 聊eNames 3.7 查找算法 3.8 距离算法 3.9 PSObiect的固有成员和MemberSets 3.10 错误和异常 3.10.1 运行时错误 3.10.2 初始化错误 3.11 类型转换 3.11.1 PowerShell语言中的标准类型转换 3.11.2 自定义型转换 3.12 TOString方法 3.13 类型配置(聊eData) 3.13.1 常用成员 3.13.2 脚本访问 3.14 小结第4章 开发cmdlet 4.1 基本概念 4.1.1 命令行解析 4.1.2 命令发现 4.1.3 参数绑定 4.1.4 命令调用 4.2 使用参数 4.2.1 强制参数 4.2.2 位置参数 4.2.3 参数集合 4.2.4 参数值验证 4.2.5 参数转换 4.3 处理管道输入 4.4 生成管道输出 4.5 错误报告 4.5.1 ErrorRecord类 4.5.2 ErrorDetails类 4.5.3 非终结型错误和致命错误 4.6 支持ShouldProcess 4.6.1 影响确认等级 4.6.2 ShouldContinue（） 4.7 使用PowerSheU系统路径 4.8 编写cmdlet帮助文档 4.9 cmdlet开发最佳实践 4.9.1 命名约定 4.9.2 与宿主交互 4.10 小结第5章 提供程序 5.1 实现提供程序类的原因 5.2 基本概念 5.2.1 路径 5.2.2 驱动器 5.2.3 错误处理 5.2.4 功能 5.3 Hello World提供程序 5.4 内置提供程序 5.4.1 别名提供程序 5.4.2 环境提供程序 5.4.3 文件系统提供程序 5.4.4 函数提供程序 5.4.5 注册表提供程序 5.4.6 变量提供程序 5.4.7 证书提供程序 5.5 提供程序基类 5.5.1 CmdletProvier类 5.5.2 DriveCmdletProvider类 5.5.3 ItemCmdletProvider类 5.5.4 ContainerCmdletPmvider类 5.5.5 NavigationCmdletProvider类 5.6 可选的提供程序接口 5.6.1 IContentCmdletProvider接口 5.6.2 IPropertyCmdletProvider接口 5.6.3 lDynamicPropertyCmdletPmvic接口 5.6.4 ISecurityDescriptorCmdletProvi接口 5.7 CmdletProvider基类 5.7.1 CmdletProvider的方法和属性 5.7.2 DriveCmdletProvider 5.7.3 ItemCmdletProvider 5.7.4 ContainerE：mdletProvider 5.7.5 NavigationCmdletProvider 5.8 设计准则与提示 5.9 小结第6章 在应用程序中集成PowerShe引擎 6.1 运行空间和管道 6.2 入门 6.3 执行命令行 6.3.1 使用Runspaceinvoke 6.3.2 使用Runspace和Pipeline 6.4 使用管道的输出 6.4.1 Invoke()返回值 6.4.2 使用管道返回的PSObject对象 6.4.3 处理终结型错误 6.5 同步管道中的输入、输出和错误 6.5.1 将输入对象传递给管道 6.5.2 同步执行时的输出管道 6.5.3 从错误管道获取非终结型错误 6.5.4 ErrorRecord类型 6.6 操作管道的其他技巧 6.6.1 嵌套式管道 6.6.2 管道重用 6.6.3 在运行空间之间复制管道 6.7 配置运行空间 6.7.1 创建自定义配置的运行空间 6.7.2 添加和删除snap.in 6.7.3 通过控制台文件创建RunspaceConfiguration 6.7.4 通过程序集创建RunspaceConfiguration对象 6.7.5 使用SessionStateProxy设置和获取变量 6.8 异步执行管道 6.8.1 调用InvokeAsyc0 6.8.2 关闭输入管道 6.8.3 从异步管道读取输出和错误 6.8.4 监视管道的StateChanged事件 6.8.5 由PipelineStatelnfo.Reason读取终结型错误 6.8.6 停止正在执行的管道 6.9 异步运行空间操作 6.9.1 OpenAsync0方法 6.9.2 处理运行空间的StateChanged事件 6.10 编程创建管道对象 6.10.1 创建空管道对象 6.10.2 创建命令对象 6.10.3 合并命令结果 6.10.4 添加命令参数 6.10.5 向管道添加命令 6.11 使用cmdlet作为GUI程序的API层 6.11.1 高层架构 6.11.2 cmdlet与GUI成功集成的关键技术 6.11.3 提供自定义的宿主 6.12 小结第7章 宿主 7.1 宿主与Windows PowerShell引擎之间的交

互 7.2 cmdlet和宿主的交互 7.3 PSHost类 7.3.1 Instanceld 7.3.2 Name 7.3.3 Version 7.3.4 CurrentCulture 7.3.5 CurrentVICulture 7.3.6 PrivateData 7.3.7 EnterNestedPrompt 7.3.8 ExitNestedPrompt 7.3.9 应用程序通知方法 7.3.10 SetShouldExit 7.4 PSHostUserInterface类 7.4.1 WriteDebugLine 7.4.2 WriteVerboseLine 7.4.3 WriteWamingLine 7.4.4 WriteProgress 7.4.5 WriteErrorLine 7.4.6 Write方法 7.4.7 Prompt方法 7.4.8 PromptForCredential 7.4.9 Read方法 7.5 PSHostRawUserInterface类 7.6 小结第8章 格式与输出 8.1 四种视图类型 8.1.1 Table视图：format-table 8.1.2 List视图：format-1ist 8.1.3 custom视图：format-custom 8.1.4 Wide视图：format-wide ” 8.2 不使用*format-pslxml配置文件进行格式化 8.3 格式配置文件示例 8.4 加载格式文件 8.4.1 update.formatdata 8.4.2 snap—in 8.4.3 RunspaceConfigttration类的API接口 8.5 格式配置文件详解 8.5.1 VieW 8.5.2 Name 8.5.3 ViewSelectedBy 8.5.4 GroupBy 8.6 TableControl 8.6.1 TableHeader 8.6.2 TableRowEntries 8.7 ListContr01 8.8 WideControl 8.9 CustomControl 8.10 其他配置条目 8.10.1 Wrap 8.10.2 AutoSize 8.11 使用场合 8.11.1 格式化字符串 8.11.2 反序列化对象的格式问题 8.11.3 类继承问题 8.11.4 选择集 8.11.5 颜色 8.12 小结附录A cmdIet动词命名准则附录B CITIdIet参数命名准则附录C 元数据附录D 提供程序基类与重载／接口附录E 用于提供程序交互的核心cmdIet

章节摘录

第1章 PowerShell简介Windows PowerShell是.NET平台之上基于对象的命令行外壳程序和脚本语言。
PowerShell为Windows平台上的IT事务管理工作提供了更高级的控制和自动化支持，更有利于提高IT专业人士和开发人员的工作效率。
市场上向IT专业人士介绍Windows PowerShell的书已经屡见不鲜，但从cmdlet、提供程序(Provider)类和宿主(Host)类开发方面来介绍PowerShell开发技术的书却寥寥无几。
本书从使用Windows PowerShell软件包的过程入手，向读者介绍其中的基本概念、常用组件以及开发技术，试图弥补前面提到的图书市场空白。
对于那些意图扩展Windows PowerShell的功能或者使用PowerShell扩展自己程序功能的开发人员来说，本书是最佳的选择。
一般来说，在书写命令行工具时，程序员需要书写代码来完成参数解析、赋值绑定工作。
此外，程序员还要书写代码，用来格式化命令输出信息。
Windows PowerShell为程序员提供了一个自带解释器的运行时引擎，简化了参数解析、赋值绑定等繁琐的工作。
当输出对象需要显示时，PowerShell也为程序开发人员提供了可定制的格式化功能。
使用Windows PowerShell完成那些开发命令行工具时的常规事务，开发人员可以集中精力关注程序的业务逻辑问题，从而摆脱那些琐碎的小问题。
1.1 Windows PowerShell设计原则多年以来，广大用户对微软Windows操作系统上的系统管理工作提了许多意见，Windows Powershell就是对这些用户反馈信息的响应。
最初，许多用户常常询问为什么像一些传统Unix系统上的外壳程序没有被授权并包含在Windows操作系统中。
对于这个问题，我们认为，只有开发一种独立于那些传统外壳程序的全新外壳程序才能解决这个问题。
这个想法进而分解为四条指导原则，这四个原则构成了设计PowerShell的指导思想。
1.1.1 保留用户已有的投资一项新技术发布后，必须经历一定的时间才能被广泛采用。
此外，客户很可能在原有技术上已经投资了很多，要他们抛弃已有的投资是不现实的。
因此，PowerShell在本质上与现有的Windows管理技术完全兼容，保留了用户已有的投资。
事实上，在PowerShell环境中，操作系统原有的命令和脚本可以直接运行。
PowerShell除了和.NET紧密结合外，它和COM、WMI和ADSI技术也几乎是无缝集成的。
PowerShell提供了一个统一的操作环境，用户可以处理前面提到的各种对象，这是PowerShell的最大特色。
在本章后面的PowerShell快速入门部分，您可以看到本设计原则和其他设计原则相关的代码演示。
1.1.2 提供一个功能强大、面向对象的外壳程序CMD.exe和其他的外壳程序都是基于文本的，也就是说在这些外壳程序中，命令接收文本输入，产生文本输出。
在内部处理时，这些命令把文本转化为其他对象，但仍以文本方式进行输出。
在传统外壳程序中，当许多简单的命令通过管道连接起来时，命令之间需要进行许多的文本处理工作才能产生需要的输出。
这方面的工具比如SED、AWK和Perl，由于它们出色的文本处理能力，很受命令行脚本程序员的青睐。

编辑推荐

《Windows PowerShell高级编程》由清华大学出版社出版。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:http://www.tushu007.com