

第 1 章 背景介绍

自从 2006 年第一版 Windows PowerShell 面世以来，我们就一直在致力于对该技术进行教学推广。那时候，PowerShell 的大部分使用者都是长期使用 VBScript 的用户，而且他们也非常期待能通过对 VBScript 的熟悉来学习 PowerShell。于是，开展培训以及编写 PowerShell 书籍的作者都采用了一种和其他编程语言教学一样的方式来教学 PowerShell。

但是从 2009 年开始发生了一些改变。越来越多没有 VBScript 经验的人开始学习 PowerShell 这门语言。因为之前我们主要关注于脚本的编写，所以对 PowerShell 的教学不再那么卓有成效。也就是在那个时候，我们意识到 PowerShell 并不仅仅是一门脚本语言，其实是一种运行命令行工具的命令行 Shell。和其他优秀的 Shell 一样，虽然 PowerShell 可以通过脚本实现很复杂的功能，但脚本仅是使用 PowerShell 的一种方式，因此学习 PowerShell 并不一定需要从脚本开始。之后，我们在每年的技术演讲会议上逐渐改变了我们的教学方式，同时也将这些教学方式的变化体现在我们的教学课程中。最后，我们出版了这本书，这也是我们想出的针对非编程背景的人员教学 PowerShell 的最好方式。但是在开始学习之前，我们需要了解一下背景。

1.1 为什么要重视 PowerShell

从 Batch、KiXtart、VBScript 到现在，可以看到 Windows PowerShell 并不是微软（或者其他公司）首次为 Windows 管理员提供自动化管理的工具。我们认为，有必要让你们了解为什么需要关注 PowerShell 这个工具。因为当你们这样做的时候，会发现花费一定的时间去学习 PowerShell 是值得的。想象一下，在没有使用 PowerShell 之前我们的工作是怎样的，在使用该工具后又有哪些变化。

没有 PowerShell

Windows 操作系统管理员总是喜欢通过单击用户图形化界面去完成他们的工作。

GUI（用户图形化界面）是 Windows 操作系统的一个最大的特点——毕竟这个操作系统并不是“文字模式”。因为 GUI 总是让我们很轻易找到我们能做的一切，所以它是那么强大。笔者仍然还记得第一次展开活动目录下的用户和计算机的场景。通过单击各种按钮，阅读工具栏提示信息，选择下拉菜单，右键单击某些图标，来查看用户与计算机中的各项功能。GUI 是使得我们能够更容易学习的一种工具。但是不幸的是，GUI 并不能带来任何效率提升上的回报。如你花费 5 分钟在活动目录中创建一个新的用户（合理的设想下，需要填写大量的信息），之后再新建用户时，也不会更快。那么新建 100 个新用户就会花费 500 分钟来完成——没有其他任何办法使得我们输入信息以及单击操作更快，从而加快该过程。

微软之前也尝试去解决该问题，VBScript 可能算是其中最成功的一次尝试。如果你需要花费一小时去编写一条 VBScript 语句来将 CSV 文件中的新用户导入到活动目录中，但是以后你可能只需要花费几秒钟就可以完成同样的工作。VBScript 的问题在于微软没有全心全意地对其提供支持，微软需要确保各种对象都可以通过 VBScript 访问、调用，而如果开发人员因为时间的原因或者是忘记这块知识，那么你就只能卡在那儿了。例如，想通过 VBScript 修改网卡 IP，没问题。但是，想检查网络连接的速度，那就不行了，因为没人记得可以把这个功能设置为 VBScript 可访问的形式。这也算是一种遗憾。Jeffery Snover, Windows PowerShell 的架构师称之为“最后一英里”。你可以通过 VBScript（或者其他类似的技术）来做很多事情，但是在某些时刻总会让人失望，从来不会让我们顺利通过“最后一英里”完成之后的工作。

Windows PowerShell 正是微软公司试图改善这一缺陷的尝试，让你顺利通过“最后一英里”，进而完成工作。

拥有 PowerShell

微软对 Windows PowerShell 的定位是我们可以通过该 Shell 完全管理 Windows 系统中的功能。微软仍在继续开发 GUI 的控制台，但是底层执行的仍然是 PowerShell 命令。通过这种方式，微软保证我们可以在该 Shell 中完成 Windows 系统中任意的工作。如果需要自动化一个重复性的任务或者完成在 GUI 中不支持的工作，那么你可以使用该 Shell 来达成所愿。

很多微软的产品都已经采用了这种开发方法，如 Exchange Server 2007 和 2010、Sharepoint Server 2010、大部分 System Center 产品以及 Windows 系统中大量的组件。接下来，越来越多的产品和 Windows 系统中组件会采用这个 Shell。Windows Server 2012（首次采用 PowerShell V3）甚至可以完全通过 PowerShell 或者使用基于 PowerShell 的 GUI 工具来进行管理。这也就是为什么我们要重视 PowerShell。在接下来的几年，PowerShell 会成为越来越多的管理功能的底层实现。

此时，我们仔细想想：如果你正在管理一个拥有很多 IT 工程师的团队，你希望谁的职级更高，希望谁能拿更多的薪水，是每次都要花费几分钟使用 GUI 来完成一个任务的人，还是一个可以通过脚本花费几秒钟自动化完成的人？无论你是来自哪个领域的 IT

从业人员，我们都知道应该如何选择。询问一个思科的管理员、AS/400 的操作员或者 Unix 管理员，他们都会回答“我更希望选择可以借助命令行更有效率地完成工作的人员”。以后的 Windows 系统工程师可以简单分为两类，一部分会使用 PowerShell，另一部分则不会。正如 Don 在微软 2010TechEd 会议上著名的言论：我们的选择是“学习 PowerShell”，还是“来包炸薯条”？

我们很欣慰，你已经决定来学习 PowerShell。

1.2 本书适用读者

这本书并不是适合所有人。实际上，微软 PowerShell 团队已经定义了三类适用 PowerShell 的人群：

- 主要使用命令行以及采用第三方开发的工具的管理员；
- 能将命令行和工具集成为一个更复杂的工具（之后那些缺乏经验的成员可以立即使用这个工具来完成相关工作）的管理员；
- 开发可重复使用的工具或者程序的管理员或者开发人员。

本书主要是针对第一类人编写的。包括开发者在内的所有人去理解该 Shell 如何执行命令是非常有必要的。毕竟，如果你正准备去开发一个工具或者编写一些命令，那么你应该知道这个 Shell 是如何运行的，这样可以保证开发出来的工具或者命令能像在 Shell 中运行得那么顺畅。

如果你对如何利用命令行来完成复杂的命令感兴趣，比如新建一个用户，在学习完本书后，你可以看到如何实现该功能。甚至你可以编写自己的脚本，并且该脚本可以让其他管理员任意使用。但是本书并不会很深入地讲解 PowerShell 的每项功能。我们的宗旨是让你能够使用该 Shell，并能立即应用到生产环境。

我们也会使用多种方法来演示如何将 PowerShell 关联到其他的管理工具。在后续章节中，我们会以 WMI (Windows Management Instrumentation) 以及常用的命令作为示例。大体上，我们仅会介绍 PowerShell 可以与哪些技术进行关联，并且讲解它们之间是如何进行关联的。其实，这些主题甚至都可以单独出书介绍（我们会在本书适当的地方给出对应的建议）。在本书中，我们仅仅介绍跟 PowerShell 相关的部分。如果你对更深入地学习这部分技术感兴趣，我们将会提供针对后续学习的建议。

1.3 如何使用本书

本书的理念是每天完成一章的学习。我们不需要在用餐时间阅读本书，因为我们只需要接近 40 分钟就可以完成对一章的阅读，之后再花 20 分钟去享用剩余的三明治以及进行对应的练习。

主要章节

第 2 章至第 25 章为本书的主要内容，算下来差不多只要花费 24 顿午餐的时间来完成阅读。这也就意味着你可以在一个月内完成对本书主要章节的阅读。你需要尽可能地严格遵守制订的学习计划，不需要在既定的时间里去阅读其他章节。更为重要的是，我们需要花费一定的时间去完成每个章节之后的练习题目，用以巩固我们的学习成果。当然，并不是每个章节都需要花费完整的一小时，所以有时你在上班之前有更多的时间进行练习（或者吃午餐）。

动手实验

在主要章节的结尾都布置了需要完成的实验题目。我们会给你对应的说明，甚至可能是一两个提示，但是在本书中并不会直接给出答案。这些动手实验的答案，我们会放在 MoreLunches.com 上，但是建议你在查看这些答案之前尽力独立完成这部分实验。

补充资料

MoreLunches.com 网站中也包含了其他一些学习资料，如额外的章节、配套视频等。实际上，每个章节至少都有一段配套视频，其中包含本章节讲解的主要内容。每段视频大概只有五分钟时间。当阅读完某章节后，你可以通过该视频回顾对应章节学习的内容。同时，你可以看到本书第一版中的视频汇总。这些视频也适用于第三版的 PowerShell，并且它们都是免费的。

进一步学习

本书的某些章节仅会简单介绍一些比较酷炫的技术，在对应章节的结尾部分会给出深入学习这部分技术的建议。我们会列出额外的学习资源，包含一些免费的资料。如果你有兴趣，可以借助这些资料进行深入学习。

补充说明

在学习 PowerShell 的时候，有些时候我们可能会钻入死胡同去研究为什么会这样或那样运行。如果这样学习，我们就不会学到很多实用的技能，但是我们可以对这个 Shell 到底是什么及其工作原理有更深入的了解。我们在“补充说明”章节中会提供这方面的信息。这些信息只需要花费几分钟就可以读完。如果你是那种喜欢钻研原理部分的人，这部分信息也可以提供一些有用的材料。如果你觉得这个小节会使得你分心而不能很好地完成实践学习，那么你可以在首次阅读时忽略这个小节。当然，如果你掌握了所有章节部分的主要内容，建议再返回阅读这部分。

1.4 搭建自己的实验环境

在本书的学习过程中，你会进行大量的 PowerShell 的动手实验，那么你必须构建一个属于你自己的实验环境（请记住，不要在生产环境中进行测试）。

你需要在带有 PowerShell 的 Windows 中运行本书中大部分示例以及完成每章节的动

手实验。环境可以是 Windows Vista, Windows 7, Windows Server 2008, Windows Server 2008 R2, Windows 8 或者是 Windows Server 2012。但是需要注意的是, 某些版本(如简易版)的操作系统中可能不存在 PowerShell。如果你对 PowerShell 学习抱有很大的兴趣, 那么你必须找到一个带有 PowerShell 的 Windows 系统。同时, 有些动手实验是基于 Windows 8 或者 Windows Server 2012 中 PowerShell 的新特性才能完成的。在每个动手实验开始时, 我们都会特别说明你需要在什么操作系统中去完成这部分实践。我们建议, 使用 Windows 8 或者 Windows Server 2012 去学习 PowerShell, 甚至你可以使用虚拟机。

在本书中, 我们都是以 64 位(X64)操作系统为环境进行学习的。我们知道有两个版本: Windows PowerShell 以及特定版本的图形化 Windows PowerShell ISE。在开始菜单(Windows 8 中是叫“开始”界面), 这两个组件的 64 位版本显示为“Windows PowerShell”和“Windows PowerShell ISE”。32 位版本的在快捷方式中会显示“X86”字样。在使用 X86 版本 PowerShell 时, 在窗口栏中也会看到 X86 字样。如果操作系统本身就是 32 位的, 那么你能只能安装 32 位的 PowerShell, 并且不会显示 X86 字样。

本书中的示例基于 64 位版本的 PowerShell 和对应的 ISE。如果你并不是使用的 64 位环境, 那么有些时候运行示例时可能和我们得出的结果不一致, 甚至某些动手实验部分根本无法正常进行。32 位版本的 PowerShell 主要是针对向后兼容性。例如, 一些 Shell 扩展程序只存在于 32 位 PowerShell 中, 并且也只能导入到 32(或者 X86)的 Shell 中。除非你确实需要使用这部分扩展程序, 否则我们建议你在 64 位操作系统上使用 64 位的 PowerShell。微软后续主要的精力会放在 64 位 PowerShell 上; 如果你现在因为使用的 32 位操作系统而无法进行下去, 那么很遗憾, 以后仍然会无法继续进行。

提示: 我们完全可以在一个独立操作系统的 PowerShell 环境中完成本书的所有学习。但是如果使用同一个域的两台或者三台计算机的 PowerShell 环境联合起来进行测试, 那么某些动手实验可能会变得更有趣。在本书中, 我们在 CloudShare.com 上创建多个虚拟机来解决该问题。如果你对这种场景感兴趣, 你可以了解一下这个服务或者其他类似的一些服务。但是需要注意, CloudShare.com 并不是在所有国家都可以访问。

1.5 安装 Windows PowerShell

从 Windows Server 2008、Windows Server 2008 R2、Windows 7 操作系统开始, 我们已经可以使用第三版的 Windows PowerShell。Windows Vista 操作系统无法支持第三版, 但是可以使用第二版 PowerShell。最近发布的几个操作系统中已经预装了 Windows PowerShell。如果采用老版本的操作系统, 那么必须手动去安装 PowerShell。当然, 新版本的操作系统可能会采用更新版本的 PowerShell, 当然这没什么坏处。

提示: 你可以采用如下方法来检查安装的 PowerShell 版本: 进入 PowerShell 控制台, 输入 `$PSVersionTable`, 然后按回车键。如果返回错误或者输出结果并未显示为“PSVersion

3.0”，那么你安装的版本就不是第三版 PowerShell。

第三版 PowerShell 可以与第二版 PowerShell 安装于一台机器上，也就意味着不会损坏那些依赖于第二版 PowerShell 的程序。另外，我们没有必要安装第一版 PowerShell，安装第三版后会自动覆盖它。最近发布的微软软件都不会依赖于第一版 PowerShell。

如果你使用的是老版本的 PowerShell，则需访问 <http://download.microsoft.com>，然后在搜索框中键入 PowerShell 3，之后根据你的操作系统选择到对应版本的 PowerShell，然后进行安装。你需要找的是 Windows Management Framework 程序包，PowerShell 是集成在这个包中进行发布的。再次申明，你需要选择到正确的版本，X86 代表 32 位的安装包，X64 代表 64 位的安装包。在网站上无法找到最近发布的 Windows 操作系统，那是因为 PowerShell 已经被预装到这些系统中了。

提示：PowerShell 最低要求 .Net Framework V4，当然如果能使用更新版本的 Framework 就更好了。我们建议同时最少也要安装 .Net Framework 3.5 SP1 以及 .Net Framework 4.5 版本，这样可以使 PowerShell 更多的功能。

安装 PowerShell 的同时也会安装一些配套程序，其中包含 Windows 远程管理服务 (WinRM)，在本书后续章节中会讲到这部分。PowerShell 采用类似 Hotfix 的方式进行安装，也就意味着安装后，也可以单独卸载。当然，一般来讲，你肯定不会希望去卸载它。PowerShell 现在已经正式成为 Windows 操作系统核心组件的一部分，因此对 PowerShell 的更新和其他 Windows 组件一样，以 Windows 的 hotfix 或者 SP 形式进行发布。

PowerShell 包含两部分：基于文本的标准控制台 (PowerShell.exe) 和集成了命令行环境的图形化界面 (ISE; PowerShell_ISE.exe)。我们大部分时间都会使用基于文本的控制台。当然，如果你更喜欢 ISE，你也可以使用。

注意：PowerShell ISE 组件并没有预装到 Server 版操作系统中。如果你需要使用，那么你需要进入 Windows 的功能（使用“服务器管理器”），然后手动添加 ISE 功能（你也可以打开 PowerShell 的控制台，再执行 `Add-WindowsFeature PowerShell -ise`）。在未包含完整 GUI 模式的操作系统（如 Server Core 版本的系统）对应的安装程序中并没有包含 ISE 的安装程序。

在你继续学习 PowerShell 之前，建议花几分钟去设置 Shell 的显示界面。如果你使用基于文本的控制台，那么强烈建议你修改显示的字体为 Lucida（固定宽度），不要使用默认的字体。假如使用默认字体，我们会很难去区分 PowerShell 使用的一些特殊字符。可以参照下面的步骤来修改显示字体。

(1) 右键单击控制台界面上侧边框（PowerShell 字符位于控制台界面的左上方），选择目录中的属性。

(2) 在弹出的会话框中，可以在几个标签页中修改字体、窗口颜色、窗口大小和位置等。

提示：强烈建议窗口大小和屏幕缓冲器使用相同的宽度。

另外，需要注意的是，当应用对默认控制台的修改之后，后续所有新开的窗口都会使用变更之后的设置。

1.6 在线资源

在前文中，我们已经多次提及 MoreLunches.com 网站，希望你有时间就去访问一下。这个网站上包含如下补充资料：

- 每个章节的配套视频；
- 每章结尾动手实验的答案；
- 可供下载的代码清单（所以没有必要自己输入）；
- 额外的章节以及文章；
- 我们 Windows PowerShell 博客的链接地址，其中包含更多的示例以及文章；
- Don 的 Windows PowerShell FAQ 链接地址；
- 链接到其他论坛板块的地址。在这些板块中，大家可以提问或者提交关于本书的一些反馈。

我们热衷于帮助像你这样的人来学习 PowerShell，我们也在尽力提供各种不同的学习资源。同时，我们也很欢迎你给我们进行反馈，这样会促使我们找寻可以加入到我们网站的新资源，以及提升本书后续版本的质量。你可以通过 MoreLunches.com 网站联系到我们，也可以通过 <http://ITPro.ConcentratedTech.com> 或者在 Twitter 上 @concentrateddon 找到 Don。

另外，也可以通过 <http://jdhitsolutions.com/blog> 博客或者在 Twitter 上 @jeffhicks 找到 Jeffery。

1.7 赶紧使用 PowerShell 吧

“可以立即使用”是我们编写本书的一个主要目标。我们在每一章中尽可能仅关注某一部分的知识，并且你在学习之后，可以立即在生产环境中使用。这就意味着，在开始的时候，我们可能会避开一些细节的讨论，但是在必要时，我们承诺后续会回到这些问题并给出详细说明。在很多情形下，我们必须在首先给出 20 页的理论或者直接讲解并完成某些部分的学习（暂不解释分析其中的细微差别或者详细情况）中做出选择。当需要做出这类选择时，我们总是选择第二个，以便使得你可以立即使用起来。但是之前的那些细节，我们会在另外一个时间去进行分析讲解（针对那些不会影响本书内容的小细节，我们可能会在 MoreLunches.com 的在线文章中去进行讲解）。

好了，背景知识大概就介绍到这里。下面就开始第 2 章课程的学习。