Broadview®
www.broadview.com.cn
华清远见 FARSIGHT® 嵌入式培训专家

华清远见系列图书

- 提供全书技术和案例的多媒体教学视频约850分钟
- 提供Vmware工具、Linux命令工具、编辑器工具、GCC工具、GDB工具、Shell工具、make工具、Eclipse开发工具、kdevelop开发工具以及项目管理Subversion工具等Linux常用工具教学视频约450分钟
- 提供201个常用Linux命令教学视频约580分钟

超大容量多媒体，总时长超过30小时
QQ答疑：1506551841

Linux Shell 编程从初学到精通

◎ 华清远见嵌入式培训中心 伍之昂 等编著

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

华清远见系列图书

Linux Shell 编程从初学到精通

◎ 华清远见嵌入式培训中心 伍之昂 等编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 提 要

Shell 是用户与内核进行交互操作的一种接口，是 Linux 最重要的软件之一。目前最流行的 Shell 称为 bash Shell，bash Shell 脚本编程以其简洁、高效而著称，多年来成为 Linux 程序员和系统管理员解决实际问题的利器。

本书结合大量的示例，系统、全面地介绍了 bash Shell 脚本编程的语法、命令、技巧、调试等内容，在书中还有很多练习可以引导读者思考，力求使读者掌握 Linux bash Shell 编程的所有特性。本书结构清晰、易教易学、实例丰富、可操作性强、学以致用，对易混淆和实用性强的内容进行了重点提示和讲解，并配有光盘，光盘中提供书中出现的所有脚本文件、各章的讲解 PPT，以及各章的讲解录像。

本书面向广大工程技术工作者，既可作为高等学校教师和相关专业学生的教材，又可作为各类培训班的培训教程。

图书在版编目（CIP）数据

Linux Shell 编程从初学到精通 / 华清远见嵌入式培训中心等编著. —北京：电子工业出版社，2011.3
（华清远见系列图书）

ISBN 978-7-121-12305-4

Ⅰ. ①L… Ⅱ. ①华… Ⅲ. ①Linux 操作系统－程序设计 Ⅳ. ①TP316.89

中国版本图书馆 CIP 数据核字（2010）第 224541 号

策划编辑：胡辛征
责任编辑：李利健
印　　刷：北京东光印刷厂
装　　订：河北省三河市路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1092　1/16　印张：29.25　字数：748.8 千字
印　　次：2011 年 3 月第 1 次印刷
印　　数：4000 册　定价：58.00 元（含 DVD 光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

随着 Linux 逐步成为主流的服务器操作系统，Shell 脚本编程成为一个优秀的 Linux 开发者和系统管理员必须掌握的技术之一。bash Shell 为当前大部分 Linux 版本所使用，本书旨在系统地介绍 bash 4.0 版本下的 Shell 脚本编程。

本书共分 17 章：第 1 章介绍 Shell 的概念、Shell 脚本编程的优势和结构等入门知识；第 2 章讲述 Shell 脚本编程不可或缺的 Linux 系统的基础知识；第 3 章介绍正则表达式和 grep 命令族；第 4 章阐述 sed 命令和 awk 命令两种在 Shell 编程中常用的工具；第 5 章介绍 Shell 编程在文件排序、合并和分割上的一些命令；第 6 章探讨变量和引用；第 7 章介绍退出、测试、判断及操作符；第 8 章介绍循环与结构化命令；第 9 章深入讨论变量的高级用法；第 10 章详述 I/O 重定向，包含管道、exec 命令等重要内容；第 11 章简述 UNIX/Linux 发展过程中出现的其他类型的 Shell；第 12 章介绍子 Shell、限制性 Shell 和进程等内容；第 13 章介绍函数的用法；第 14 章介绍别名、列表及数组；第 15 章罗列了无法归入其他章节的混杂主题，包含脚本编写风格、脚本优化、/dev 和/proc 文件系统等；第 16 章介绍了 Shell 脚本的调试技术；第 17 章给出六个 Shell 编程的实例，读者需要综合使用前面章节所述的 Shell 命令和编程技巧，涉及系统管理、文本处理和数据库等多个方面。

本书内容丰富，覆盖了 Shell 编程的大部分技术，并结合典型例子透彻地介绍了 Shell 命令、选项、结构中的重点和难点。各章最后还配有一定数量的练习题供读者学习。为了帮助读者更加直观地学习本书，我们将书中出现的所有脚本文件、各章的讲解 PPT，以及各章的讲解录像都收录到本书的配套光盘中。

本书面向广大工程技术工作者，既可作为高等学校的教师和相关专业学生的教材，又可作为各类培训班的培训教程。

本书由南京财经大学江苏省电子商务重点实验室伍之昂组织编写。在本书编写过程中，实验室主任曹杰教授在全书的体系结构、理论阐释和实例选取等方面提出了许多精辟的见解，研究生陈志杰同学精心润色了本书的文字。参加本书编写工作的还有高淑娟、李子龙、王丽娜、周毅、林小峰、刘刚、马海波、李强、吴慧、马玉刚、冯浩、唐爱琴、王明明、蒋志。在此，对他们表示诚挚的谢意。

限于笔者水平，本书一定有不少错误和不妥之处，恳请计算机专家、同行和读者批评、指正，您可以通过 E-mail 的方式与作者联系，作者邮箱是 zawu@seu.edu.cn。

编　者

目　录

Linux

第1章 Shell脚本编程概述

Shell 脚本语言是 Linux/UNIX 系统上一种重要的脚本语言，在 Linux/UNIX 领域应用极为广泛，熟练地掌握 Shell 脚本编程是一个优秀的 Linux/UNIX 开发者和系统管理员的必经之路。作为本书的开篇，本章从 Linux 和 Shell 的概念开始介绍，分析 Shell 脚本语言和高级程序设计语言的区别，总结 Shell 脚本编程的优势，再给出第一个 Shell 脚本的例子，结合例子介绍 Shell 脚本的基本元素及其执行脚本的方法。

1.1 Linux 和 Shell 概述

1.1.1 Linux 简介

Linux 是一套可免费使用和自由传播的类 UNIX 操作系统。1991 年，芬兰赫尔辛基大学学生 Linus 开发了 Linux 内核。此后，一大批程序爱好者、软件技术专家对 Linux 进行修改和完善。Linux 操作系统从诞生到现在，其开放、安全、稳定的特性得到越来越多用户的认可，又由于其低成本、自由开发以及安全可靠等优势，促使各国政府和企业纷纷对 Linux 提供强有力的支持。Linux 的应用和发展前景变得越来越广阔。

自 1991 年 10 月 5 日 Linus Torvalds 在新闻组 comp.os.minix 发表了 Linux V0.01，Linux 开启了其迅猛发展的步伐。经过近 20 年的发展，Linux 成为了一个支持多用户、多进程、多线程、实时性较好、功能强大而稳定的操作系统。它可以运行在 x86、Sun Sparc、Digital Aopha、680x0、PowerPC、MIPS、ARM 等平台上，是目前支持硬件平台最多的操作系统。由于用户操作习惯等因素的制约，Linux 在桌面领域发展不是很好，但是在其他领域都取得了巨大的进步和成功。在企业应用领域方面，Linux 得到了除微软公司之外几乎所有知名软件和硬件公司的支持，这包括 IBM、HP、Sun、Intel、AMD、Sony 等，软件公司有 CA、Veritas、BEA、Oracle、SAP、Borland 等，使得 Linux 操作系统在企业运算领域具有强大的发展潜力。

Linux 自诞生以来，像其他许多软件一样发布了很多不同的版本，最常见的有 Slackware、RedHat、Debian、S.u.s.E.等。Fedora Core（有时又称为 Fedora Linux）是众多 Linux 发行版本之一，它是一套从 Red Hat Linux 发展出来的免费 Linux 系统。Fedora 和 Redhat 这两个 Linux 的发行版本联系很密切。Redhat 自 9.0 以后，不再发布桌面版，而是把这个项目与开源社区合作，于是就有了 Fedora 发行版。Fedora 可以说是 Redhat 桌面版本的延续，只不过是与开源社区合作。

Fedora 是一个开放的、创新的、具有前瞻性的 Linux 操作系统和平台，无论是现在还是将来它都允许任何人自由地使用、修改和重发布。它由一个强大的社群开发，这个社群的成员以自己的不懈努力，提供并维护自由、开放源码的软件和开放的标准。Fedora 项目由 Fedora 基金会管理和控制，得到了 Red Hat, Inc.的支持。Fedora 项目的目标是与 Linux 社区一同构造一个完整的、通用的操作系统。Red Hat 工程师团队一直参与到构建 Fedora Core 的过程中，同时邀请并鼓励更多的人参与其中。通过使用这种开放的过程，他们希望可以提供一个更加贴近自由的软件和更受开源社区欢迎的操作系统。

Fedora Core 被红帽公司定位为新技术的实验场，与 Red Hat Enterprise Linux 被定位为稳定性优先不同，许多新的技术都会在 Fedora Core 中检验，如果稳定，红帽公司才会考虑加入 Red Hat Enterprise Linux 中。到目前为止，Fedora Core 已经发行了 12 个版本，最新版本为 Fedora 12。

注：本书的实验环境选择了 Fedora 11，它是在 2009 年 6 月发行的 Fedora 版本，其 Shell 是 bash Shell，版本是 4.0.16(1)-release。本书所有的例子和脚本都在 Fedora 11 系统下测试通过。

1.1.2 Shell 简介

Shell 是一种具备特殊功能的程序，它提供了用户与内核进行交互操作的一种接口。它接收用户输入的命令，并把它送入内核去执行。内核是 Linux 系统的心脏，从开机自检时就驻留在计算机的内存中，直到计算机关闭为止，而用户的应用程序存储在计算机的硬盘上，仅当需要时才被调入内存。Shell 是一种应用程序，当用户登录 Linux 系统时，Shell 就会被调入内存执行。Shell 独立于内核，它是连接内核和应用程序的桥梁，并由输入设备读取命令，再将其转为计算机可以理解的机械码，Linux 内核才能执行该命令。图 1-1 描述了 Shell 在 Linux 系统中的位置。

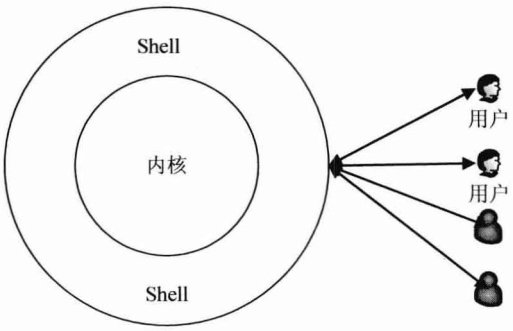

图 1-1　Shell 在 Linux 系统中的位置

用户可以通过两种方式打开 Shell，第一种是在 Linux 系统图形用户界面 GNOME 下单击“终端”打开 Shell，图 1-2 给出了 Fedora Core 11 系统下打开 Shell 的方法，“终端”菜单位于“应用程序”→“系统工具”下面。图 1-3 给出了 GNOME 下的 Shell 窗口截图，GNOME 与 Windows 操作系统风格类似，Shell 窗口打开后，会在屏幕下方的任务栏上显示出来，用户可以在命令提示符后输入 Linux 命令。

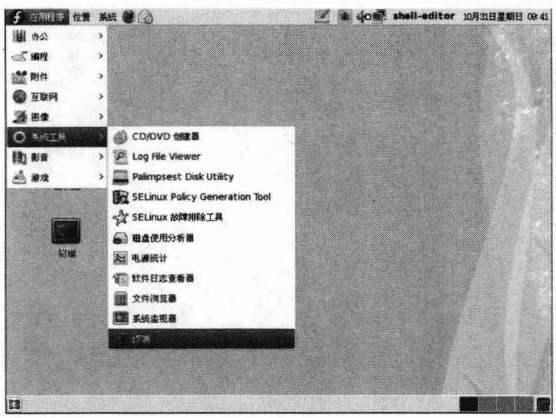

图 1-2　在 GNOME 下打开 Shell

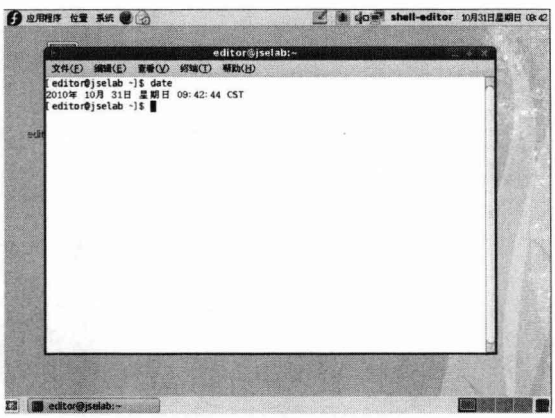

图 1-3　GNOME 的 Shell 窗口

十分熟悉 Linux 系统的用户一般不通过图形用户界面来操纵 Linux 系统，而是直接通过 Shell 登录到 Linux 系统。因此，第二种打开 Shell 的方式就是利用一些软件工具以 SSH 的方式远程登录到 Linux 系统，目前比较流行的 Shell 软件工具是 SSH Secure Shell 和 PuTTY。两种软件都是非常好用的工具，下面对这两种软件的用法作简单介绍。

（1）SSH Secure Shell 软件

该软件的风格十分简洁，单击工具栏中的“Quick Connect”按钮，即可弹出登录设置界面，输入登录主机的 IP 地址和用户名，如图 1-4 所示，单击“Connect”按钮就可远程登录 Linux 系统。图 1-5 展示了登录成功的界面，与 X Windows 的终端类似，可以在命令提示符后输入系统命令。

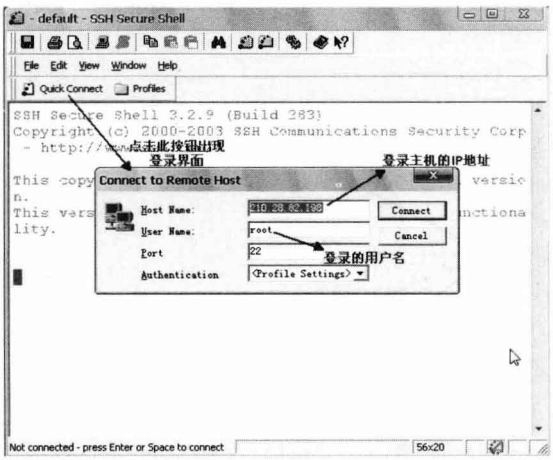

图 1-4　SSH Secure Shell 登录设置界面

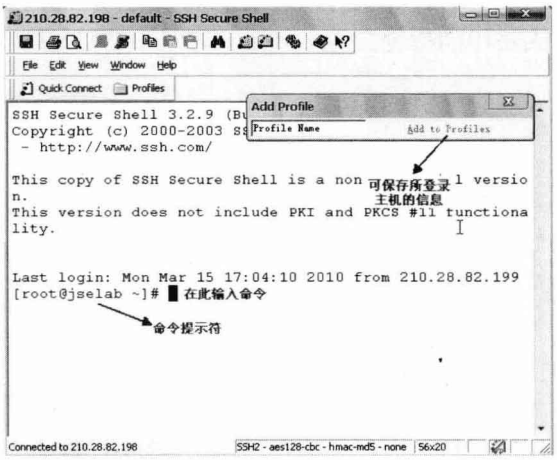

图 1-5　SSH Secure Shell 登录成功后的界面

（2）PuTTY 软件

该软件是一个非常小巧的工具，而且是绿色软件，无须安装，图 1-6 显示了 PuTTY 的登录设置界面，同样是输入登录主机的 IP 地址，再单击“Open”按钮连接，在图 1-7 所示的界面中输入登录用户名及其密码，成功登录后会出现命令提示符。

图 1-6　PuTTY 登录设置界面

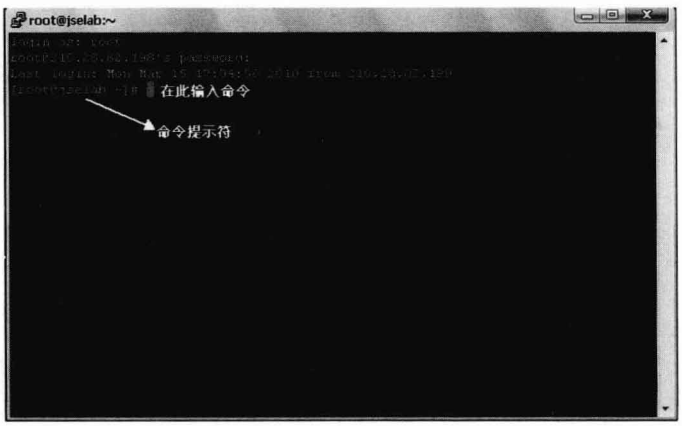

图 1-7　PuTTY 登录成功后的界面

注：本节首先简单介绍了 Shell 的概念，对 Shell 概念的介绍并不完整，因为要介绍清楚 Shell，必然要涉及很多操作系统方面的知识，而本书的主旨是 Shell 编程；然后，给出三种启动 Shell 的方法，这是进行 Shell 编程的基础，我们通常不建议在 GNOME 下进行 Shell 编程，推荐读者使用 SSH Secure Shell 或 PuTTY 远程连接 Linux 主机后进行编程。

1.2 Shell 脚本编程的优势

脚本语言（Script Language）是相对于编译型语言而言的，它是为了缩短编译型语言编写—编译—链接—运行（Edit-Compile-Link-Run）过程而创建的计算机编程语言。由于脚本语言常常运行于底层，所处理的是字节、整数、浮点数或其他机器层的对象，因而，脚本语言是低级程序设计语言。如 C\C++、Ada、Java、C#等都属于编译型语言，也可称为高级程序设计语言，这类语言所编写的程序需要经过编译，将源代码转化为目标代码才能运行。而脚本语言往往是解释运行而非编译，即由解释器（Interpreter）读入脚本程序代码，将其转换成内部的形式执行，而解释器本身则是编译型程序。

脚本语言的好处是简单、易学、易用，适合处理文件和目录之类的对象，以简单的方式快速完成某些复杂的事情通常是创建脚本语言的重要原则，脚本语言的特性可以总结为以下几个方面：

- 语法和结构通常比较简单。
- 学习和使用通常比较简单。
- 通常以容易修改程序的“解释”作为运行方式，而不需要“编译”。
- 程序的开发产能优于运行效能。

脚本语言的灵活性是以执行效率为代价的，脚本语言的执行效率通常不如编译型语言。当然，脚本语言一般不适用于大型的项目、计算复杂的工程或有高级需求的应用软件，它适用于系统管理、文本处理等方面完成特定功能的常用的小工具或小程序。

Shell 脚本语言是 Linux/UNIX 系统上一种重要的脚本语言，在 Linux/UNIX 领域应用极为广泛，熟练掌握 Shell 脚本语言是一个优秀的 Linux/UNIX 开发者和系统管理员的必经之路。利用 Shell 脚本语言可以简洁地实现复杂的操作，而且 Shell 脚本程序往往可以在不同版本的 Linux/UNIX 系统上通用。

尽管 Shell 脚本语言延续了脚本语言易学的特性，易学体现在 Shell 脚本语言门槛较低，易于上手，读者可以毫不费力就学会编写一个简单的 Shell 脚本程序，并且很容易学会执行它。但是，要深入透彻地学会 Shell 脚本语言是有难度的，因为 Shell 脚本语言涉及几乎所有 Linux 命令的灵活使用，而且 Linux 系统下的小工具（如 awk、sed）也较多，它们常常出现在 Shell 脚本之中。另外，Shell 脚本语言还提供了类似于高级程序设计语言的语法结构，如分支判断语句、变量和函数、循环结构、数组、算术和逻辑运算等。

综上所述，Shell 脚本语言是 Linux/UNIX 系统上应用广泛的实用程序设计语言，它是“易学难精”的，真正学会 Shell 脚本编程，需要读者清晰地掌握 Linux 重要命令的语法，理解 Linux 命令重要选项的作业和区别，还需要掌握 Shell 脚本语言的语法结构以及一些常用的小

工具。

1.3 第一个 Shell 脚本例子

1.3.1 Shell 脚本的基本元素

使用 Shell 脚本的最初动机可能在于省去手动输入命令的麻烦，Shell 脚本将一系列的 Linux 命令放在一个文件中，这样，我们就不必每次都手动输入同样的命令。比如：当一个用户登录系统后，他每次都是先执行.bash_profile 文件配置命令行环境，再查看当前有哪些用户在登录，那么，这个用户将会依次输入以下命令完成上述操作：

```
. .bash_profile
date
who
```

显然，用户每次输入重复的命令显得比较麻烦，我们能否把多个命令写入一个文件，然后通过执行这个文件来执行这些命令呢？这就是最原始的 Shell 脚本。上述的三条命令可以写入下面的文件中，该文件取名为 whologged.sh：

```
#!/bin/bash

cd                              #切换到用户根目录，因为.bash_profile 在根目录下
. .bash_profile                 #配置用户的命令行环境
date                            #显示日期命令
who                             #显示当前的登录用户
```

whologged.sh 文件就是一个典型的 Shell 脚本。需要注意的是，whologged.sh 文件的第 1 行是“#!/bin/bash”，“#!”符号称为“Sha-bang”符号，是 Shell 脚本的起始符号，“#!”符号是指定一个文件类型的特殊标记，它告诉 Linux 系统这个文件的执行需要指定一个解释器。“#!”符号之后是一个路径名，这个路径名指明了解释器在系统中的位置，对于一般的 Shell 脚本而言，解释器是 bash，也可以是 sh，即用下面的两种方式作为脚本的第 1 行都是正确的：

```
#!/bin/bash
#!/bin/sh
```

当然，Linux 还存在其他的一些解释器，如 sed 和 awk 等，指定这些解释器就需要对第 1 行做相应的改动。本书第 4 章将会介绍 sed 和 awk，在此，请读者记住大部分脚本都是用 bash 解释器的，但是，其他解释器依然是存在的。

“#!/bin/bash”行之后，whologged.sh 文件按顺序写入需要执行的三条命令，每条命令后面有一段以“#”符号起始的中文，“#”符号是注释符，它后面直到本行结束的所有内容是注释，脚本执行时是不执行注释的，“#”符号类似于 C++和 Java 语言中的“//”符号，脚本注释可以是整行，也可以在某行的后面：

```
command    #在行后面的注释
#整行的注释
```

注释能增加 Shell 脚本的可读性，便于人们理解该脚本。因此，读者在编写脚本时，应养成勤加注释的好习惯。

从 whologged.sh 脚本也可以看出，命令（command）是 Shell 脚本的最基本元素，命令通常由命令名称、选项和参数三部分组成，三部分之间用空格键或 Tab 键分隔。我们以下面的例 1-1 来说明命令的三个组成部分。

```
#例 1-1：介绍 Linux 命令的组成部分
[root@jselab ~]# ls -l /etc/sh*                                    #一条简单的 Linux 命令
-r--------. 1 root root 1181 2009-09-04 /etc/shadow
-r--------. 1 root root 1181 2009-09-04 /etc/shadow-
-rw-r--r--. 1 root root   32 2009-04-10 /etc/shells
[root@jselab ~]#
```

例 1-1 中的命令用于列出/etc 目录下以“sh”开头文件的详细信息，这一条简单的 Linux 命令就由三个部分组成，“ls”是命令名称，“-l”是选项，“/etc/sh*”是参数。命令名称在命令中是不可或缺的，而选项和参数则可以不出现。选项的开头符号是一个减号（–），后面跟一个或多个字母，选项是对命令的补充说明，读者在学习 Linux 命令时需要在辨析和理解选项上花力气，读者在后续章节的学习中一定能够体会到选项的重要性。参数可以理解为命令的作用对象，“/etc/sh*”参数中“*”符号称为通配符，通配符经常在命令参数中出现。我们将在第 3 章详细讨论通配符的用法。

Linux 系统有成千上万条命令，我们很难全部掌握这些命令，而且即便是经常使用的命令，我们往往也会忘记这些命令的选项和用法，此时可利用 Linux 系统提供的命令的手册页（manual page），我们可以用如下格式的命令打开某命令的手册页：

```
man [命令名称]
```

man 命令可以打开其他 Linux 命令的手册页，手册页上能详细地显示命令名称、基本格式、对选项的详细描述等信息，man 是 Linux 程序员和管理员用于查询命令用法的常用手段。

分号（;）可以用来隔开同一行内的多条命令，Shell 会依次执行用分号隔开的多条命令，例 1-2 演示了分号的用法。

```
#例 1-2：分号的用法
[root@jselab ~]# ls -l /etc/sh*;date;who                           #同一行内有三条命令，用分号隔开
-r--------. 1 root root 1181 2009-09-04 /etc/shadow                #ls 命令的结果
-r--------. 1 root root 1181 2009-09-04 /etc/shadow-
-rw-r--r--. 1 root root   32 2009-04-10 /etc/shells
2010 年 03 月 23 日 星期二 12:33:25 CST                             #date 命令的结果
root     pts/0        2010-03-23 11:42 (210.28.82.132)             #who 命令的结果
root     pts/1        2010-03-23 11:43 (210.28.82.199)
[root@jselab ~]#
```

例 1-2 的同一行内有三条命令，用分号隔开，依次显示了这三条命令的执行结果。

本节给出了第一个 Shell 脚本，Shell 脚本以“#!”符号开头，该符号后面跟了解释器的路径；命令是 Shell 脚本的最基本元素，它定义了 Shell 脚本的动作，包含命令名称、选项和参数三个部分；Shell 脚本中也可以添加注释，以注释符“#”引出。除此之外，Shell 脚本还可以包含变量、各种控制结构，以及算术和逻辑运算符，这些内容将在后续章节中展开。

1.3.2 执行 Shell 脚本

在编写好一个 Shell 脚本后，如何执行这个 Shell 脚本呢？这与 Linux 系统对文件的管理有关，Linux 系统管理文件结合考虑了用户权限和文件权限，本书第 2 章将详细介绍 Linux 系统用户管理和文件管理的基础知识，简言之，要执行一个 Shell 脚本，只需要使当前用户

具备执行该脚本文件的权限。一般来说，当我们用文本编辑器创建一个 Shell 脚本文件时，该文件是没有可执行权限的，即 x 权限。因此，我们需要先赋给 Shell 脚本可执行权限，再去执行它。下面的例 1-3 给出了 whologged.sh 脚本的执行过程。

```
#例 1-3：执行 whologged.sh 脚本
[root@jselab shell-book]# chmod u+x whologged.sh          #为 whologged.sh 脚本赋可执行权限
[root@jselab shell-book]# ls -l whologged.sh              #查看 whologged.sh 的权限
-rwxr--r--. 1 root root 155 03-23 13:14 whologged.sh      #具备 x 权限了
[root@jselab shell-book]# ./whologged.sh                  #执行 whologged.sh 脚本
2010 年 03 月 23 日 星期二 13:14:35 CST                    #whologged.sh 脚本的执行结果
root     pts/0        2010-03-23 11:42 (210.28.82.132)
root     pts/1        2010-03-23 11:43 (210.28.82.199)
[root@jselab shell-book]#
```

例 1-3 首先利用 chmod 命令为 whologged.sh 脚本赋可执行权限，chmod 命令将在第 2 章详细介绍，查看 whologged.sh 的权限时发现，whologged.sh 文件确实具备了 x 权限，即 whologged.sh 文件具备了可执行权限。最后，使用./[脚本名]格式的命令执行 whologged.sh 脚本，Shell 打印出 whologged.sh 脚本的执行结果。

缺乏 Linux 文件管理基础知识的读者可能并不理解 Shell 脚本执行的原因，这类读者可以通过阅读本书第 2 章掌握 Linux 用户管理、文件管理和文本编辑器，掌握第 2 章内容之后，一定能轻松地掌握 Shell 脚本的执行。

1.4 本章小结

本章首先简明扼要地介绍了 Linux 操作系统的起源与发展、Shell 的概念和作用、Shell 脚本编程的优势等理论性内容，试图让读者理解 Linux 和 Shell 是什么、为什么需要学习 Shell 脚本编程。然后，本章给出第一个 Shell 脚本的例子，从该例子说明 Shell 脚本的基本元素，以及执行脚本的方法。