

Scott Granneman



ESSENTIAL CODE AND COMMANDS

Second  
Edition

# Linux<sup>®</sup>

**PHRASEBOOK**



FREE SAMPLE CHAPTER



SHARE WITH OTHERS

# Linux

## PHRASEBOOK

---

SECOND EDITION

Scott Granneman

 Addison-Wesley

800 East 96th Street, Indianapolis, Indiana 46240 USA

## Linux Phrasebook, Second Edition

Copyright © 2016 by Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, 200 Old Tappan Road, Old Tappan, New Jersey 07675, or you may fax your request to (201) 236-3290.

Library of Congress Control Number: 2015955629

ISBN-13: 978-0-321-83388-4

ISBN-10: 0-321-83388-0

First printing: December 2015

### Trademarks

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

### Warning and Disclaimer

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

### Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at [corpsales@pearsoned.com](mailto:corpsales@pearsoned.com) or (800) 382-3419.

For government sales inquiries, please contact [governmentsales@pearsoned.com](mailto:governmentsales@pearsoned.com).

For questions about sales outside the U.S., please contact [international@pearsoned.com](mailto:international@pearsoned.com).

Visit us on the Web: [informit.com/aw](http://informit.com/aw)

---

<b>Acquisitions Editor</b> Mark Taber	<b>Copy Editor</b> Anne Goebel	<b>Technical Reviewer</b> Brian Tiemann	<b>Compositor</b> Mary Sudul
<b>Managing Editor</b> Sandra Schroeder	<b>Senior Indexer</b> Cheryl Lenser	<b>Editorial Assistant</b> Vanessa Evans	
<b>Senior Project Editor</b> Tonya Simpson	<b>Proofreader</b> Laura Hernandez	<b>Cover Designer</b> Chuti Prasertsith	

# Table of Contents

<b>Introduction</b>	<b>1</b>
<b>Part I: Getting Started</b>	
<b>1 Things to Know About Your Command Line</b>	<b>9</b>
Everything Is a File	9
Maximum Filename Lengths	10
Names Are Case-Sensitive	11
Special Characters to Avoid in Names	12
Wildcards and What They Mean	15
Special Files That Affect Your Command Line	20
If There's Too Much Stuff on Screen, Reset	24
Conclusion	24
<b>2 Navigating Your File System</b>	<b>27</b>
List Files and Folders	28
List the Contents of Other Folders	28
List Folder Contents Using Wildcards	30
View a List of Files in Subfolders	31
View a List of Contents in a Single Column	32
View Contents As a Comma-Separated List	33
View Hidden Files and Folders	33
Visually Display a File's Type	34
Display Contents in Color	35
List Permissions, Ownership, and More	37
Reverse the Order Contents Are Listed	43
Sort Contents by Date and Time	44
Sort Contents by Size	45
Express File Sizes in Terms of K, M, and G	46
Display the Path of Your Current Directory	47
Change to a Different Directory	49
Change to Your Home Directory	50
Change to Your Previous Directory	50
Conclusion	51

<b>3 Creation and Destruction</b>	<b>53</b>
Change a File to the Current Time	54
Change a File to Any Desired Time	55
Create a New, Empty File	57
Create a New Directory	58
Create a New Directory and Any Necessary Subdirectories	59
Copy Files	60
Copy Files Using Wildcards	62
Copy Files Verbosely	64
Stop Yourself from Copying over Important Files	65
Copy Directories	67
Copy Files As Perfect Backups in Another Directory	68
Move Files and Folders	70
Rename Files and Folders	72
Understand How Linux Stores Files	74
Create a Link Pointing to Another File or Directory	76
Delete Files	84
Remove Several Files at Once with Wildcards	85
Prevent Yourself from Deleting Key Files	86
Delete an Empty Directory	87
Remove Files and Directories That Aren't Empty	88
Deleting Troublesome Files	89
Conclusion	91
<b>4 Learning About Commands</b>	<b>93</b>
Find Out About Commands with <code>man</code>	94
Quickly Find Out What a Command Does Based on Its Name	97
Search for a Command Based on What It Does	99
Read a Command's Specific Man Page	101
Learn About Commands with <code>info</code>	103

Navigate Within <code>info</code>	104
Locate the Paths for a Command's Executable, Source Files, and Man Pages	108
Find Out Which Version of a Command Will Run	110
Discover How a Command Will Be Interpreted	111
Conclusion	113
<b>5 Building Blocks</b>	<b>115</b>
Run Several Commands Sequentially	116
Run Commands Only If the Previous Ones Succeed	118
Run a Command Only If the Previous One Fails	121
Plug the Output of a Command into Another Command	122
Understand Input/Output Streams	123
Use the Output of One Command As Input for Another	124
Redirect a Command's Output to a File	127
Prevent Overwriting Files When Using Redirection	128
Append a Command's Output to a File	129
Use a File As Input for a Command	130
Combine Input and Output Redirection	131
Send Output to a File and to <code>stdout</code> at the Same Time	133
Conclusion	135
<b>Part II: Working with Files</b>	
<b>6 Viewing (Mostly Text) Files</b>	<b>137</b>
Figure Out a File's Type	138
View Files on <code>stdout</code>	141
Concatenate Files to <code>stdout</code>	142
Concatenate Files to Another File	143
Concatenate Files and Number the Lines	144

View Text Files a Screen at a Time	145
Search Within Your Pager	148
Edit Files Viewed with a Pager	149
View the First 10 Lines of a File	150
View the First 10 Lines of Several Files	151
View the First Several Lines of a File or Files	152
View the First Several Bytes, Kilobytes, or Megabytes of a File	153
View the Last 10 Lines of a File	155
View the Last 10 Lines of Several Files	156
View the Last Several Lines of a File or Files	158
View the Constantly Updated Last Lines of a File or Files	158
Conclusion	160
<b>7 Manipulating Text Files with Filters</b>	<b>163</b>
Count the Number of Words, Lines, and Characters in a File	164
Number Lines in a File	167
Select an Entire Column of Data in a Delimited File	169
Sort the Contents of a File	172
Sort the Contents of a File Numerically	174
Remove Duplicate Lines in a File	177
Substitute Selected Characters with Others	180
Replace Repeated Characters with a Single Instance	182
Delete Matching Characters	183
Transform Text in a File	188
Print Specific Fields in a File	194
Conclusion	198
<b>8 Ownerships and Permissions</b>	<b>199</b>
Become Another User	200
Become Another User, with His Environment Variables	201
Become <code>root</code>	202

Become <code>root</code> , with Its Environment Variables	202
Change the Group Owning Files and Directories	204
Recursively Change the Group Owning a Directory	205
Change the Owner of Files and Directories	207
Change the Owner and Group of Files and Directories	208
Understand the Basics of Permissions	210
Change Permissions on Files and Directories Using Alphabetic Notation	213
Change Permissions on Files and Directories Using Numeric Permissions	215
Change Permissions Recursively	219
Set and Then Clear <code>suid</code>	221
Set and Then Clear <code>sgid</code>	225
Set and Then Clear the Sticky Bit	228
Conclusion	231
<b>9 Archiving and Compression</b>	<b>233</b>
Archive and Compress Files Using <code>zip</code>	235
Get the Best Compression Possible with <code>zip</code>	237
Archive and Compress Files of a Specified Type in Directories and Subdirectories	239
Password-Protect Compressed Zip Archives	242
Unzip Files	243
Test Files That Will Be Unzipped	244
Archive and Compress Files Using <code>gzip</code>	245
Archive and Compress Files Recursively Using <code>gzip</code>	247
Uncompress Files Compressed with <code>gzip</code>	248
Test Files That Will Be Unzipped with <code>gunzip</code>	249
Archive and Compress Files Using <code>bzip2</code>	250
Uncompress Files Compressed with <code>bzip2</code>	251
Test Files That Will Be Unzipped with <code>bunzip2</code>	252



Archive Files with <code>tar</code>	253
Archive and Compress Files with <code>tar</code> and <code>gzip</code>	255
Test Files That Will Be Untarred and Uncompressed	257
Untar and Uncompress Files	259
Conclusion	260

### Part III: Finding Files, Words, and More

<b>10 Finding Files, Directories, Words, and Phrases</b>	<b>261</b>
Search a Database of Filenames	262
Search a Database of Filenames Without Worrying About Case	264
Update the Database Used by <code>locate</code>	265
Searching Inside Text Files for Patterns	268
The Basics of Searching Inside Text Files for Patterns	269
Search Recursively for Text in Files	274
Search for Words and Highlight the Results	275
Search for Text in Files, Ignoring Case	276
Search for Whole Words in Files	277
Show Line Numbers Where Words Appear in Files	278
Search the Output of Other Commands for Specific Words	279
See Context for Words Appearing in Files	281
Show Lines Where Words Do Not Appear in Files	284
List Files Containing Searched-for Words	285
List the Number of Occurrences of Words in Files	286
Search for Words Inside Search Results	288
Conclusion	289
<b>11 The <code>find</code> Command</b>	<b>291</b>
Find Files by Name	292
Find Files by Ownership	294
Find Files by File Size	295

Find Files by File Type	298
Find Files by Time	300
Show Results If the Expressions Are True (AND)	303
Show Results If Either Expression Is True (OR)	304
Show Results If the Expression Is Not True (NOT)	307
Execute a Command on Found Files	309
Execute a Command on Found Files More Efficiently	312
Execute a Command on Found Files Containing Spaces	315
Conclusion	316

## Part IV: Your Environment

<b>12 Your Shell</b>	<b>319</b>
View Your Command-Line History	320
Run the Last Command Again	321
Run a Previous Command Using Numbers	323
Run a Previous Command Using a String	324
Search for a Previous Command and Run It	325
Display All Command Aliases	331
View a Specific Command Alias	332
Create a New Temporary Alias	332
Create a New Permanent Alias	333
Remove an Alias	334
Create a New Temporary Function	335
Create a New Permanent Function	338
Display All Functions	341
Remove a Function	342
When to Use an Alias and When to Use a Function	344
Conclusion	347
<b>13 Monitoring System Resources</b>	<b>349</b>
Discover How Long Your Computer Has Been Running	350

View All Currently Running Processes	350
View a Process Tree	353
View Processes Owned by a Particular User	355
End a Running Process	355
View a Dynamically Updated List of Running Processes	359
List Open Files	361
List a User's Open Files	363
List Users for a Particular File	364
List Processes for a Particular Program	365
Display Information About System RAM	367
Show File System Disk Usage	369
Report File Space Used by a Directory	371
Report Just the Total Space Used for a Directory	373
Conclusion	374
<b>14 Installing Software</b>	<b>375</b>
Install Software Packages (RPM)	376
Remove Software Packages (RPM)	378
Install Software Packages and Dependencies (RPM)	379
Remove Software Packages and Dependencies (RPM)	382
Upgrade Software Packages and Dependencies (RPM)	384
Find Packages Available for Download (RPM)	386
Install Software Packages (DEB)	387
Remove Software Packages (DEB)	389
Install Software Packages and Dependencies (DEB)	390
Remove Software Packages and Dependencies (DEB)	394
Upgrade Software Packages and Dependencies (DEB)	396
Find Packages Available for Download (DEB)	398

Clean Up Unneeded Installation Packages (DEB)	401
Troubleshoot Problems with APT (DEB)	402
Conclusion	404

## **Part V: Networking**

<b>15 Connectivity</b>	<b>405</b>
View the Status of Your Network Interfaces	407
Verify That a Computer Is Running and Accepting Requests	410
Trace the Route Packets Take Between Two Hosts	412
Query DNS Records	414
Configure a Network Interface	418
View the Status of Your Wireless Network Interfaces	421
Configure a Wireless Network Interface	425
Grab a New Address Using DHCP	426
Make a Network Connection Active	428
Bring a Network Connection Down	430
Display Your IP Routing Table	432
Change Your IP Routing Table	435
Troubleshooting Network Problems	439
Conclusion	443
<b>16 Working on the Network</b>	<b>445</b>
Securely Log In to Another Computer	446
Securely Log In to Another Machine Without a Password	450
Securely Transfer Files Between Machines	453
Securely Copy Files Between Hosts	456
Securely Transfer and Back Up Files	458
Download Files Non-interactively	466
Download Websites Non-interactively	472
Download Sequential Files and Internet Resources	474
Conclusion	476
<b>Index</b>	<b>477</b>

## About the Author

**Scott Granneman** is an author, educator, and small business owner. He has written seven books (about Firefox, Linux, Google Apps, and Mac OS X) and contributed to two. In addition, he was a columnist for *SecurityFocus*, one of the largest and most important security-focused sites on the Web, and *Linux Magazine* while it was in print.

As an educator, he has taught thousands of people of all ages—from preteens to senior citizens—on a wide variety of topics, including both literature and technology. He is currently an adjunct professor at Washington University in St. Louis and at Webster University, where he teaches a variety of courses about technology, social media, the Internet, and Web development. With the shift in focus over the past few decades to Linux and other open-source technologies, he has worked to bring knowledge of these powerful new directions in software to people at all technical skill levels.

As a Principal of WebSanity, a website planning, development, and hosting firm with clients in 12 states, he manages the firm's Linux-based servers and infrastructure, researches new technologies, and works closely with other partners on the underlying WebSanity Content Management System (CMS).

# Dedication

*This book is dedicated to Linux users,  
both old and new. Welcome!*

## Acknowledgments for the First Edition (2005)

No one learns about the Linux shell in a vacuum, and I have hundreds of writers, working over decades, to thank for educating me about the awesome possibilities provided by the Linux command line. Books, websites, blogs, handouts at LUG meetings: All helped me learn about `bash` and the beauty of the Linux command line, and they continue to teach me today. If I can give back just a fraction of what I have absorbed, I'll be satisfied.

In addition to that general pool of knowledgeable individuals, I'd like to thank the people (and animals) who gave me help and support during the writing of *Linux Phrasebook*.

My agent, Laura Lewin, who has been helpful in too many ways for me to recount.

My editors at Pearson, who gave me the opportunity to write this book in the first place and have encouraged me whenever I needed prodding.

Robert Citek provided invaluable help with RPM and was always there if I had a question. That man knows Linux.

My business partner and lifelong buddy, Jans Carton, helped me focus when I needed it, and was a (mostly) willing guinea pig for many new commands and options. Looking back at that day in fifth grade when we met, who'da thunk it?

Jerry Bryan looked over everything I wrote and fixed all the little grammatical mistakes and typos I made. I promise, Jerry: One day I'll learn the difference between "may" and "might"!

My wife, Denise Lieberman, patiently listened to me babble excitedly whenever I figured out something cool, even though she had absolutely no idea what I was talking about. That's true love. Thanks, Denise!

Finally, I must point to my cute lil' Shih Tzu, Libby, who always knew exactly the right time to put her front paws on my leg and demand ear scratches and belly rubs.

## **Acknowledgments for the Second Edition (2015)**

For the second edition, many things have changed (sadly, Libby has gone to that great dog house in the sky), but one thing is the same: Lots of folks helped me during the writing of this book, and they deserve recognition and my gratitude.

Mark Taber at Pearson supported me over a *very* lengthy writing period. I can't thank you enough, Mark. You are why this book exists today.

My wife, Robin Woltman, read through every chapter and made many invaluable edits and suggestions. You did a great job, Robin!

Robert Citek looked over many of the chapters and made suggestions and technical edits that were always right on the nose. He's who I turn to when I need help!

Craig Buchek looked over a chapter or two and made some good points. As the former benevolent dictator for life of the St. Louis Linux Users Group, that was expected.

Tom Kirk loaned me a few laptops from his enormous collection that enabled me to test out things from the networking sections. Tom, you are a lifesaver, and the best hardware guy I know.

In addition to my friends above, the following people emailed to let me know about problems in the first edition.

- Joe Hancharik pointed out a problem in Chapter 1's "Wildcards and What They Mean" that was a very silly mistake on my part.
- William H. Ferguson found another silly mistake I made in Chapter 3's "Copy Files."
- Brian Greer pointed out a problem I should have noticed in Chapter 15's "Troubleshooting Network Problems."

Thank you, gentlemen! It's readers like Joe, William, and Brian that make writing a book fun. Feedback is a great thing, even if it is a bug report. If you see an error, let me know, so I can fix it for the third edition!



## We Want to Hear from You!

As the reader of this book, *you* are our most important critic and commentator. We value your opinion and want to know what we're doing right, what we could do better, what areas you'd like to see us publish in, and any other words of wisdom you're willing to pass our way.

We welcome your comments. You can email or write directly to let us know what you did or didn't like about this book—as well as what we can do to make our books better.

*Please note that we cannot help you with technical problems related to the topic of this book, and that due to the high volume of mail we receive, we might not be able to reply to every message.*

When you write, please be sure to include this book's title and author, as well as your name and phone number or email address.

E-mail: [feedback@developers-library.info](mailto:feedback@developers-library.info)

Mail: Reader Feedback  
Addison-Wesley Developer's Library  
800 East 96th Street  
Indianapolis, IN 46240 USA

## Reader Services

Visit our website and register this book at **[www.informit.com/register](http://www.informit.com/register)** for convenient access to any updates, downloads, or errata that might be available for this book.

# Introduction

Among key Linux features, the command-line shell is one of the most important. If you run a Linux server, your main interface is more than likely going to be the shell. If you're a power user running Linux on the desktop, you probably have a terminal open at all times. If you're a Linux newbie, you may think that you'll never open up the command line, but you will sometime ... and the more you use Linux, the more you're going to want to use that shell.

The shell in many ways is the key to Linux's power and elegance. You can do things with the command line that you simply can't do with whatever GUI you favor. No matter how powerful KDE or GNOME may be (or IceWM or XFCE or any of the other kajillion windowing environments out there), you will always be able to do many things faster and more efficiently with a terminal. If you want to master Linux, you need to begin by mastering the Linux command line.

The traditional method has been to use the Linux `man` pages. While `man` pages are useful, they are often not enough, for one simple reason: They lack examples. Oh, a few `man` pages here and there have a few examples, but by and large, good examples are hard to come by. This presents a real problem for users at all experience levels: It's one thing to see options listed

and explained, but it's another thing entirely to see those options used in real-world situations.

This book is all about those missing examples. I've been using Linux for two decades, and I consider myself pretty knowledgeable about this amazing, powerful operating system. I'm so addicted to the command line that I always have a terminal open. And to top it off, the Linux servers my company relies on don't have GUIs on them (just the way I like it!), so I *have* to use the terminal to work with them. But I'm always lamenting—along with my Linux-using friends, acquaintances, and LUG members—the dearth of examples found in `man` pages. When I was asked to write *Linux Phrasebook*, and told that it was to consist of hundreds of examples illustrating the most important Linux commands, I replied, “I can't wait! That's a book I'd buy in a heartbeat!”

You're holding the result in your hands: a book about the Linux commands you just have to know, with examples illustrating how to use each and every one. This is a reference book that will be useful now and for years to come, but I also hope you find it enjoyable as well, and even a little fun.

---

**NOTE:** Visit our website and register this book at [informit.com/register](http://informit.com/register) for convenient access to any updates, downloads, or errata that might be available for this book.

---

## Audience for This Book

I've written this book to be useful both to beginners—the folks that show up to meetings of our Linux Users Group seeking guidance and a helping hand as they begin the adventure of using Linux—and to experienced users who use the shell for everything from systems administration to games to programming. If you've just started using Linux, this book will help teach you about the shell and its power; if you've been using Linux for years and years, *Linux Phrasebook* will teach you some new tricks and remind you of some features you'd long ago forgotten.

There are many shells out there—`csh`, `tcsh`, `zsh`, to name but a few—but I use the default shell for virtually every Linux distro: `bash`, the Bourne Again Shell. The `bash` shell is not only ubiquitous, but also powerful and flexible. After you get comfortable with `bash`, you may choose to explore other options, but knowledge of `bash` is required in the world of Linux.

I wrote this book using Debian, which is about as generic a Linux distro as you can get, as well as one of the most widely used. Even though I used Debian, the commands I discuss should work on your distro as well. The only major difference comes when you run a command as `root`. Instead of logging in as `root`, some distros (like Ubuntu, for instance) encourage the use of the `sudo` command; in other words, instead of running `ls` or `firefox` as `root`, an Ubuntu user would run `sudo ls` or `sudo firefox`.

**NOTE:** Sharp-eyed readers may note that, in the first edition, I used Ubuntu (or, as I referred to it there, K/Ubuntu—I was trying to emphasize that I was using KDE with Ubuntu). Now I've gone more generic and I just use straight Debian, the distro upon which Ubuntu is based.

---

In order to appeal to the widest number of readers out there, I showed the commands as though you have to run them as root, without `sudo`. If you see a `#` in front of a command, that's the shell indicating that root is logged in, which means you need to be root to run that command, or utilize `sudo` if you're using Ubuntu or a similar distro.

One final thing: In order to keep this book from being even longer, I've often truncated the outputs of commands. For instance, on your Linux computer, you'd normally see the following output after entering `ls -l`:

```
-rwxr-xr-x 1 scott admins 1261 Jun 1 2012 script.sh
```

You will see that in a few places in this book in which it's appropriate, but often you'll instead find something like this:

```
-rw-r--r-- 1 scott admins script.sh
```

In that case, I cut out the stuff that wasn't important to my point, which helped keep the output to only one line instead of two. You'd be surprised how those lines can add up, which is why I made those changes when it seemed appropriate (or when my editor was about to freak out at the number of pages the book was going to have!!).

---

**TIP:** Much of the information I provide about Linux in this book can also be applied to other flavors of UNIX, such as BSD and OS X. Note that I did not say *all* or *most*—I said *much*. As long as you keep that in mind, you'll find *Linux Phrasebook* to be helpful with those operating systems as well.

---

## About the Second Edition

When Mark Taber, my editor at Pearson, first approached me about writing a second edition of *Linux Phrasebook*, I jumped at the chance. I actually use my own book as a reference a few times a month, so over the years, I've noticed errors (wincing every time) and have seen many things that I'd like to change, remove, or add.

My goal has been to make the second edition a must-buy for new readers as well as for owners of the first edition of *Linux Phrasebook*. This isn't some so-called new edition with just a few small changes. Oh no. So here's what I've done that's new or different in this book:

- I split the former Chapter 2, “The Basics,” into two chapters: Chapters 2, “Navigating Your File System,” and 3, “Creation and Destruction.” The old Chapter 2 was ridiculously long, and it crammed together different kinds of commands. The new split makes things far more manageable and sensible (although it did mean every chapter after those two had to be re-numbered).
- I removed Chapters 6, “Printing and Managing Print Jobs,” and 16, “Windows Networking,” because they just don't seem as vital as they did ten years ago. In addition, most people who need

to print or connect to and work on a Windows-based network will have GUI tools that more than adequately do those jobs. However, don't despair—you can still find those original chapters from the first edition on my website, [www.granneman.com/linux-redaction](http://www.granneman.com/linux-redaction).

- I added a new chapter: Chapter 7, “Manipulating Text Files with Filters.” There is a *large* amount of great new information there, and I know you will find it very useful!
- I removed sections from the old Chapters 2 (now Chapters 2 and 3), 3 (now 4), 7 (8), 8 (9), 9 (10), 10 (11), and 14 (15). Again, you will find those original sections from the first edition on my website, [www.granneman.com/linux-redaction](http://www.granneman.com/linux-redaction).
- I added new sections in Chapters 1–6 and 8–13. And I almost doubled the size of Chapter 15 by keeping the deprecated commands (since they're still on most distros) while adding all the new ones.
- I moved sections into new chapters where they made sense, which you can especially see in Chapter 8.
- I revised things in every chapter of the book. I fixed mistakes, rewrote parts that were unclear, added additional notes and tips, and improved examples by adding or revising text.
- I mentioned many more commands in passing, like `ssh-agent`, `wput`, `htop`, `dnf`, `pandoc`, `rename`, `whoami`, and `iconv`.
- I provided little tidbits of information about many different things that are good to know about, like variables, `for` loops, cron jobs, arguments, and sources. Oh, and there's more H.P. Lovecraft!

Finally, a tip: If any links are broken or don't work any longer, try entering them at the Internet Archive's Wayback Machine, which you can find at <https://archive.org>, to see if you can still find the content. Then let me know so I can fix that reference in future editions.

Thank you for reading this far, and I really hope you enjoy this new edition of *Linux Phrasebook*!

## Conventions Used in This Book

This book uses the following conventions.

- `Monospace` type is used to differentiate between code/programming-related terms and regular English, and to indicate text that should appear on your screen. For example:

The `df` command shows results in kilobytes by default, but it's usually easier to comprehend if you instead use the `-h` (or `--human-readable`) option.

It will look like this to mimic the way text looks on your screen.

- An arrow (➡) at the beginning of a line of code means that a single line of code is too long to fit on the printed page. It signifies to readers that the author meant for the continued code to appear on the same line.
- In addition to this, the following elements are used to introduce other pertinent information used in this book.



**NOTE:** A note presents interesting pieces of information related to the surrounding discussion.

---

**TIP:** A tip offers advice or teaches an easier way to do something.

---

**CAUTION:** A caution advises you about potential problems and helps you steer clear of disaster.

---

*This page intentionally left blank*

# Navigating Your File System

This chapter introduces the basic commands you'll find yourself using several times every day. Think of these as the hammer, screwdriver, and pliers that a carpenter keeps in the top of his toolbox. After you learn these commands, you can start controlling your shell and finding out all sorts of interesting things about your files, folders, data, and environment. In particular, you'll be learning about some of the metadata—the data describing your data—that Linux has to keep track of, and it may just surprise you how much there is.

---

**NOTE:** When I updated this book for its second edition, I removed the section about `mkdir -v` (which shows you what `mkdir` is doing as it does it) and `rm -v` (which does the same thing, but for `rm`). You can find the original text on my website, [www.granneman.com/linux-redactions](http://www.granneman.com/linux-redactions).

Also, I took the sections on `touch`, `mkdir`, `cp`, `mv`, `rm`, and `rmdir` and used them to create a new Chapter 3 titled “Creation and Destruction” (which of course renumbered everything after it!). Finally, the section on `su` was moved to Chapter 8, “Ownership and Permissions,” which makes a lot more sense.

---

## List Files and Folders

```
ls
```

The `ls` command is probably the one that people find themselves using the most. After all, before you can manipulate and use files in a directory (remember, *file* and *directory* are interchangeable), you first have to know what files are available. That's where `ls` comes in, as it lists the files and subdirectories found in a directory.

---

**NOTE:** The `ls` command might sound simple—just show me the files!—but there are a surprising number of permutations to this amazingly pliable command, as you'll see.

---

Typing `ls` lists the contents of the directory in which you're currently working. When you first log in to your shell, you'll find yourself in your home directory. Enter `ls`, and you might see something like the following:

```
$ ls
alias Desktop  iso  pictures program_files todo
bin  documents music podcasts src          videos
```

## List the Contents of Other Folders

```
ls [folder]
```

You don't have to be in a directory to find out what's in it. Let's say that you're in your home directory, but you want to find out what's in the `music` directory.

Simply type the `ls` command, followed by the folder whose contents you want to view, and you'll see this:

```
$ ls music
Buddy_Holly Clash Donald_Fagen new
```

In the previous example, a relative path is used, but absolute paths work just as well.

```
$ ls /home/scott/music
Buddy_Holly Clash Donald_Fagen new
```

The ability to specify relative or absolute paths can be incredibly handy when you don't feel like moving all over your file system every time you want to view a list of the contents of a directory. Not sure if you still have that video of a Bengal Tiger your brother took at the zoo? Try this (`~` represents your home directory):

```
$ ls ~/videos
airhorn_surprise.wmv
apple_knowledge_navigator.mov
b-ball-e-mail.mov
carwreck.mpg
nerdTV_1_andy_hertzfeld
nerdTV_2_max_levchin_paypal
nerdTV_3_bill_joy
tiger.wmv
Ubuntu_Talk-Mark_Shuttleworth.mpeg
```

Yes, there it is: `tiger.wmv`.

## List Folder Contents Using Wildcards

```
ls *
```

You just learned how to find a file by listing out its entire directory; but there's a faster method, especially for really long directory listings. If you know that the video of the Bengal Tiger you're looking for is in Windows Media format (Boo! Hiss!), and therefore ends with `.wmv`, you can use a wildcard to show just the files ending with that particular extension.

```
$ ls ~/videos
airhorn_surprise.wmv
apple_knowledge_navigator.mov
b-ball-e-mail.mov
carwreck.mpg
nerdtv_1_andy_hertzfeld
nerdtv_2_max_levchin_paypal
nerdtv_3_bill_joy
tiger.wmv
Ubuntu_Talk-Mark_Shuttleworth.mpeg
$ ls ~/videos/*.wmv
airhorn_surprise.wmv  tiger.wmv
```

There's another faster method, also involving wildcards: Look just for files that contain the word *tiger*.

```
$ ls ~/videos/*tiger*
tiger.wmv
```

---

**NOTE:** If your wildcard results in a match with a directory, the contents of that directory will be listed as well. If you want to omit that match and avoid showing the contents of subdirectories, add the `-d` option.

---

## View a List of Files in Subfolders

```
ls -R
```

You can also view the contents of several subdirectories with one command. Say you're at a Linux Users Group (LUG) meeting and installations are occurring around you fast and furious. "Hey," someone hollers out, "does anyone have an ISO image of the new Kubuntu that I can use?" You think you downloaded that recently, so to be sure, you run the following command (instead of `ls -R`, you could have also used `ls --recursive`):

```
$ ls -R ~/iso
/home/scott/iso:
debian-6.0.4-i386-CD-1.iso  knoppix  ubuntu

/home/scott/iso/knoppix:
KNOPPIX_V7.2.0CD-2013-06-16-EN.iso
➤ KNOPPIX_V7.4.2DVD-2014-09-28-EN.iso

/home/scott/iso/ubuntu:
kubuntu-15.04-desktop-amd64.iso
➤ ubuntu-15.04-desktop-amd64.iso
ubuntu-14.04.3-server-amd64.iso
```

There it is, in `~/iso/ubuntu: kubuntu-15.04-desktop-amd64.iso`. The `-R` option traverses the `iso` directory recursively, showing you the contents of the main `iso` directory and every subdirectory as well. Each folder is introduced with its path—relative to the directory in which you started—followed by a colon, and then the items in that folder are listed. Keep in mind that the recursive option becomes less useful when you have many items in many subdirectories, as the listing goes

on for screen after screen, making it hard to find the particular item for which you're looking. Of course, if all you want to do is verify that there are many files and folders in a directory, it's useful just to see everything stream by, but that won't happen very often.

## View a List of Contents in a Single Column

```
ls -l
```

So far, you've just been working with the default outputs of `ls`. Notice that `ls` prints the contents of the directory in alphabetical columns, with a minimum of two spaces between each column for readability. But what if you want to see the contents in a different manner?

If multiple columns aren't your thing, you can instead view the results of the `ls` command as a single column using, logically enough, `ls -l` (or `ls --format=single-column`).

```
$ ls -l ~/bin
Desktop
documents
iso
music
pictures
podcaststodo
videos
```

This listing can get out of hand if you have an enormous number of items in a directory, and more so if you use the recursive option as well, as in `ls -lR ~/`. Be prepared to press `Ctrl+C` to cancel the command



if a list is streaming down your terminal with no end in sight.

## View Contents As a Comma-Separated List

```
ls -m
```

Another option for those who can't stand columns of any form, whether it's one or many, is the `-m` option (or `--format=commas`).

```
$ ls -m ~/
alias, bin, Desktop, documents, iso, music, pictures,
▶ podcasts, program_files, src, todo, videos
```

Think of the *m* in `-m` as a mnemonic for *comma*, and it is easier to remember the option. Of course, this option is also useful if you're writing a script and need the contents of a directory in a comma-separated list, but that's a more advanced use of a valuable option.

## View Hidden Files and Folders

```
ls -a
```

Up to this point, you've been viewing the visible files in directories, but don't forget that many directories contain hidden files in them as well. Your home directory, for instance, is just bursting with hidden files and folders, all made invisible by the placement of a `.` at the beginning of their names. If you want to view these hidden elements, just use the `-a` option (or `--all`).

```
$ ls -a ~/
.          .gimp-2.2      .openoffice.org1.9.95
..         .gksu.lock     .openoffice.org1.9
.3ddesktop .glade2        .openoffice.org2
.abbrev_defs .gnome         .opera
.acrorc     .gnome2        .padminrc
.adobe      .gnome2_private pictures
.alias      .gnome_private podcasts
[List condensed due to length]
```

You should know several things about this listing. First, `ls -a` displays both hidden and unhidden items, so you see both `.gnome` and `pictures`. Second, you'll always see the `.` and `..` because `.` refers to the current directory, while `..` points to the directory above this one, the parent directory. These two hidden files exist in every single folder on your system, and you can't get rid of them. Expect to see them every time you use the `-a` option. Finally, depending on the directory, the `-a` option can reveal a great number of hidden items of which you weren't aware.

---

**TIP:** I just said that `ls -a` shows you everything beginning with a dot, including the folder references `.` and `..`; but if you don't want to see those two, use `ls -A` instead.

---

## Visually Display a File's Type

```
ls -F
```

The `ls` command doesn't tell you much about an item in a directory besides its name. By itself, it's hard to tell if an item is a file, a directory, or something else.

An easy way to solve this problem and make `ls` really informative is to use the `-F` option (or `--classify`).

```
$ ls -F ~/bin
adblock_filters.txt  fixm3u*      pix2tn.pl*
addext*             flash.xml*   pop_login*
address_book.csv    getip*      procmail/
address_book.sxc    homesize*   programs_usual*
address_book.xls    html2text.py* quickrename*
backup_to_chaucer*  list-urls.py*
```

This tells you quite a bit. An `*` or asterisk after a file means that it is executable, while a `/` or forward slash indicates a directory. If the filename lacks any sort of appendage at all, it's just a regular ol' file. Other possible endings are shown in Table 2.1.

Table 2.1 Symbols and File Types

Character	Meaning
*	Executable
/	Directory
@	Symbolic link
	FIFO (AKA a <i>named pipe</i> )
=	Socket

## Display Contents in Color

```
ls --color
```

In addition to the symbols that are appended to files and folders when you use the `-F` option, you can also ask your shell to display things in color, which gives an additional way to classify items and tell them

apart. Many Linux installs come with colors already enabled for shells, but if yours does not, just use the `--color` option (I know you can't see the colors, so just pretend).

```
$ ls --color
adblock_filters.txt  fixm3u      pix2tn.pl
addext              flash.xml   pop_login
address_book.csv    getip      procmail
address_book.sxc    homesize   programs_kill
address_book.xls    html2text.py programs_usual
backup_ssh_to_chaucer list-urls.py quickrename
```

In this setup, executable files are green, folders are blue, and normal files are black (which is the default color for text in my shell). Table 2.2 gives you the full list of common color associations (but keep in mind that these colors may vary on your particular distro).

Table 2.2 Colors and File Types

Color	Meaning
Default shell text color	Regular file
Green	Executable
Blue	Directory
Magenta	Symbolic link
Yellow	FIFO
Magenta	Socket
Red	Archive (.tar, .zip, .deb, .rpm)
Magenta	Images (.jpg, .gif, .png, .tiff)
Magenta	Audio (.mp3, .ogg, .wav)

---

**TIP:** Want to see what colors are mapped to the various kinds of files on your system? Enter `dircolors --print-database`, and then read the results carefully. You can also use the `dircolors` command to change those colors as well.

---

With the combination of `--color` and `-F`, you can see at a glance what kinds of files you're working with in a directory. Now we're cookin' with gas!

```
$ ls -F --color
adblock_filters.txt      fixm3u*          pix2tn.pl*
addext*                  flash.xml*       pop_login*
address_book.csv         getip*           procmail/
address_book.sxc         homesize*        programs_kill*
address_book.xls         html2text.py*   programs_usual*
backup_ssh_to_chaucer*  list-urls.py*   quickrename*
```

## List Permissions, Ownership, and More

```
ls -l
```

You've now learned how to format the results of `ls` to tell you more about the contents of directories, but what about the actual contents themselves? How can you learn more about the files and folders, such as their size, their owners, and who can do what with them? For that information, you need to use the `-l` option (or `--format=long`).

```

$ ls -l ~/bin
total 2951
-rw-r--r--  1 scott scott  15058 2015-10-03 18:49
➔adblock_filters.txt
-rwxr-xr--  1 scott root    33 2015-04-19 09:45
➔addext
-rw-r--r--  1 scott scott  84480 2015-04-19 09:45
➔addressbook.xls
-rwxr--r--  1 scott scott   55 2015-04-19 09:45
➔batchprint_home
drwxr-xr-x  9 scott scott  1080 2015-09-22 14:42
➔bin_on_bacon
-rwxr-xr--  1 scott scott   173 2015-04-19 09:45
➔changeext
-rwxr-xr--  1 scott root   190 2015-04-19 09:45
➔convertsize
drwxr-xr-x  2 scott scott   48 2015-04-19 09:45
➔credentials
[List condensed and edited due to length]

```

The `-l` option stands for *long*, and as you can see, it provides a wealth of data about the files found in a directory. Let's move from right to left and discuss what you see.

On the farthest right is the easiest item: the name of the listed item. Want `ls` to display more about it? Then add the `-F` option to `-l`, like this: `ls -lF`. Color is easily available as well, with `ls -lF --color`.

Moving left, you next see a date and time. This is when the file was last modified, including the date (in year-month-day format) and then the time (in 24-hour military time).

Further left is a number that indicates the size of the item, in bytes. This is a bit tricky with folders—for instance, the previous readout says that `bin_on_bacon` is 1080 bytes, or just a little more than one kilobyte,

yet it contains 887KB of content inside it. The `credentials` directory, according to `ls -l`, is 48 bytes, but contains nothing inside it whatsoever! What is happening?

Remember in Chapter 1, “Things to Know About Your Command Line,” when you learned that directories are just special files that contain a list of their contents? In this case, the contents of `credentials` consists of nothing more than the `..` that all directories have in order to refer to their parent, so it’s a paltry 48 bytes; while `bin_on_bacon` contains information about more than 30 items, bringing its size up to 1080 bytes.

The next two columns to the left indicate, respectively, the file’s owner and its group. As you can see in the previous listing, almost every file is owned by the user `scott` and the group `scott`, except for `addext` and `convertsize`, which are owned by the user `scott` and the group `root`.

---

**NOTE:** Those permissions need to be changed, which you’ll learn how to do in Chapter 8 (hint: the commands are `chown` and `chgrp`).

---

The next to last column as you move left contains a number. If you’re examining a file, this number tells you how many hard links exist for that file (for more, see Chapter 3’s “Create a Link Pointing to Another File or Directory” section); if it’s a directory, it refers to the number of subdirectories it contains, including the two hidden pointers `.` (the current directory) and `..` (the parent directory), which means that even if there are no subdirectories, you will still see a `2` there.

And now you reach the final item on the left: the actual permissions for each file and directory. This might seem like some arcane code, but it's actually very understandable with just a little knowledge. There are ten items, divided (although it doesn't look that way) into four groups. The first group consists of the first character; the second group contains characters 2 through 4; the third consists of characters 5 through 7; and the fourth and final group is made up of characters 8 through 10. For instance, here's how the permissions for the `credentials` directory would be split up: `d|rwx|r-x|r-x`.

That first group tells you what kind of item it is. You've already seen that `-F` and `--color` do this in different ways, but so does `-l`. A `d` indicates that `credentials` is a directory, while a `-` in that first position indicates a file. (Even if the file is executable, `ls -l` still uses just a `-`, which means that `-F` and `--color` here give you more information.) There are, of course, other options that you might see in that first position, as detailed in Table 2.3.

**Table 2.3 Permission Characters and File Types**

Character	Meaning
-	Regular file
-	Executable
d	Directory
l	Symbolic link
s	Socket
b	Block device
c	Character device
p	Named pipe (AKA FIFO)



---

**TIP:** To view a list of files that shows at least one of almost everything listed in Table 2.3, try `ls -l /dev`.

---

The next nine characters—making up groups two, three, and four—stand for, respectively, the permissions given to the file’s owner, the file’s group, and all the other users on the system. In the case of `addext`, shown previously, its permissions are `rwxr-xr--`, which means that the owner `scott` has `rwX`, the group (in this case, also `scott`) has `r-x`, and the other users on the box have `r--`. What’s that mean?

In each case, `r` means “yes, read is allowed”; `w` means “yes, write is allowed” (with “write” meaning both changing and deleting); and `x` means “yes, execute is allowed.” A `-` means “no, do not allow this action.” If the `-` is located where an `r` would otherwise show itself, that means “no, read is not allowed.” The same holds true for both `w` and `x`.

Looking at `addext` and its permissions of `rwxr-xr--`, it’s suddenly clear that the owner (`scott`) can read, write, and execute the file; the members of the group (`root`) can read and execute the file, but not write to it; and everyone else on the machine (often called the “world”) can read the file but cannot write to it or run it as a program.

Now that you understand what permissions mean, you’ll start to notice that certain combinations seem to appear frequently. For instance, it’s common to see `rw-r--r--` for many files, which means that the owner can both read and write to the file, but both the group and world can only read the file. For programs, you’ll often see `rwxr-xr-x`, which allows everyone on the computer to read and run the program, but restricts changing the file to its owner.

Directories, however, are a bit different. The permissions of `r`, `w`, and `x` are pretty clear for a file: You can read the file, write (or change) it, or execute it. But how do you execute a directory?

Let's start with the easy one: `r`. In the case of a directory, `r` means that the user can list the contents of the directory with the `ls` command. A `w` indicates that users can add more files into the directory, rename files that are already there, or delete files that are no longer needed. That brings us to `x`, which corresponds to the capability to access a directory in order to run commands that access and use files in that directory, or to access subdirectories inside that directory.

As you can see, `-l` is incredibly powerful all by itself, but it becomes even more useful when combined with other options. You've already learned about `-a`, which shows all files in a directory, so now it should be obvious what `-la` would do (or `--format=long --all`).

```
$ la -la ~/
drwxr-xr-x    2 scott scott   200 2015-07-28 01:31
↳ alias
drwx-----    2 root  root    72 2015-09-16 19:14
↳ .aptitude
-rw-----    1 scott scott  8800 2015-10-18 19:55
↳ .bash_history
-rw-r--r--    1 scott scott    69 2015-04-20 11:00
↳ .bash_logout
-rw-r--r--    1 scott scott   428 2015-04-20 11:00
↳ .bash_profile
-rw-r--r--    1 scott scott  4954 2015-09-13 19:46
↳ .bashrc
[List condensed and edited due to length]
```

# Reverse the Order Contents Are Listed

```
ls -r
```

If you don't like the default alphabetical order that `ls` uses, you can reverse it by adding `-r` (or `--reverse`).

```
$ ls -lar ~/
-rw-r--r--  1 scott scott  4954 2015-09-13 19:46
➔.bashrc
-rw-r--r--  1 scott scott   428 2015-04-20 11:00
➔.bash_profile
-rw-r--r--  1 scott scott    69 2015-04-20 11:00
➔.bash_logout
-rw-----  1 scott scott  8800 2015-10-18 19:55
➔.bash_history
drwx-----  2 root  root    72 2015-09-16 19:14
➔.aptitude
drwxr-xr-x  2 scott scott   200 2015-07-28 01:31
➔alias
[List condensed and edited due to length]
```

---

**NOTE:** Keep in mind that this is `-r`, not `-R`. `-r` means reverse, but `-R` means recursive. Yes, that is confusing.

---

When you use `-l`, the output is sorted alphabetically based on the name of the files and folders; the addition of `-r` reverses the output, but is still based on the filename. Keep in mind that you can add `-r` virtually any time you use `ls` if you want to reverse the default output of the command and options you're inputting.

## Sort Contents by Date and Time

```
ls -t
```

Letters are great, but sometimes you need to sort a directory's contents by date and time. To do so, use `-t` (or `--sort=time`) along with `-l`; to reverse the sort, use `-tr` (or `--sort=time --reverse`) along with `-l`. Using the reverse sort can be pretty handy—the newer stuff ends up at the bottom, which is easier to see, because that's where your prompt will be when the command finishes!

```
$ ls -latr ~/
-rw-----    1 scott scott  8800 2015-10-18 19:55
└─.bash_history
drwx-----   15 scott scott  1280 2015-10-18 20:07
└─.opera
drwx-----    2 scott scott    80 2015-10-18 20:07
└─.gconfd
drwxr-xr-x    2 scott scott   432 2015-10-18 23:11
└─.qt
drwxr-xr-x   116 scott scott  5680 2015-10-18 23:11
drwx-----    3 scott scott   368 2015-10-18 23:12
└─.gnupg
drwxr-xr-x   12 scott scott  2760 2015-10-18 23:14
└─bin
drwx-----    4 scott scott   168 2015-10-19 00:13
└─.Skype
[list condensed and edited due to length]
```

All of these items except the last one were modified on the same day; the last one would have been first if you weren't using the `-r` option and thereby reversing the results.

---

**NOTE:** Notice that you're using four options at one time in the previous command: `-latr`. You could have instead used `-l -a -t -r`, but who wants to type all of those hyphens and spaces? It's quicker and easier to just combine them all into one giant option. The long version of the options (those that start with two hyphens and consist of a word or two), however, cannot be combined and have to be entered separately, as in `-la --sort=time --reverse`. Note that this technique works with many Linux commands and programs, but not all of them.

---

## Sort Contents by Size

```
ls -S
```

You can also sort by size instead of alphabetically by filename or extension, or by date and time. To sort by size, use `-s` (or `--sort=size`).

```
$ ls -laS ~/
-rw-r--r--  1 scott scott 109587 2015-10-19 11:53
➔.xsession-errors
-rw-----  1 scott scott  40122 2015-04-20 11:00
➔.nessusrc
-rw-r--r--  1 scott scott  24988 2015-04-20 11:00
➔.abbrev_defs
-rwxr--r--  1 scott scott  15465 2015-10-12 15:45
➔.vimrc
-rw-----  1 scott scott  11794 2015-10-19 10:59
➔.viminfo
-rw-----  1 scott scott   8757 2015-10-19 08:43
➔.bash_history
[List condensed and edited due to length]
```

When you sort by size, the largest items come first. To sort in reverse, with the smallest at the top, just use `-r`.

## Express File Sizes in Terms of K, M, and G

```
ls -h
```

In the previous section, the 15465 on `.vimrc`'s line means that the file is about 15KB, but it's not always convenient to mentally translate bytes into the equivalent kilobytes, megabytes, or gigabytes. Most of the time, it's more convenient to use the `-h` option (or `--human-readable`), which makes things easier to understand.

```
$ ls -laSh ~/
-rw-r--r--  1 scott scott 100K 2015-10-19 11:44
➔.xsession-errors
-rw-----  1 scott scott  40K 2015-04-20 11:00
➔.nessusrc
-rw-r--r--  1 scott scott  25K 2015-04-20 11:00
➔.abbrev_defs
-rwxr--r--  1 scott scott  16K 2015-10-12 15:45
➔.vimrc
-rw-----  1 scott scott  12K 2015-10-19 10:59
➔.viminfo
-rw-----  1 scott scott  8.6K 2015-10-19 08:43
➔.bash_history
[List condensed and edited due to length]
```

In this example, you see `k` for kilobytes; if the files were big enough, you'd see `M` for megabytes or even `G` for gigabytes. Some of you might be wondering how 40122 bytes for `.nessusrc` became 40K when you used

-h. Remember that 1024 bytes make up a kilobyte, so when you divide 40122 by 1024, you get 39.1816406 kilobytes, which `ls -h` rounds up to 40K. A megabyte is actually 1,048,576 bytes, and a gigabyte is 1,073,741,824 bytes, so a similar sort of rounding takes place with those as well. If you want exact powers of ten in your divisions (that is, what are referred to as “kibibytes,” “mebibytes,” and so on—see Chapter 6, “Viewing (Mostly Text) Files,” for more), try the `--si` option.

---

**NOTE:** In my `~/.bashrc` file, I have the following aliases defined, which have served me well for years. Use what you’ve learned in this section to extend these examples and create aliases that exactly meet your needs (for more on aliases, see Chapter 12, “Your Shell”).

```
alias l='ls -F'
alias ll='ls -lF'
alias la='ls -aF'
alias ll='ls -laFh'
alias ls='ls -F'
```

---

## Display the Path of Your Current Directory

### pwd

Of course, while you’re listing the contents of directories hither and yon, you might find yourself confused about just where you are in the file system. How can you tell in which directory you’re currently working? The answer is the `pwd` command, which stands for *print working directory*.

---

**NOTE:** The word *print* in *print working directory* means “print to the screen,” not “send to printer.”

---

The `pwd` command displays the full, absolute path of the current, or working, directory. It’s not something you’ll use all the time, but it can be incredibly handy when you get a bit discombobulated.

```
$ pwd
/home/scott/music/new
```

There is one thing you should be aware of, however, and this can really confuse people. In Chapter 3 you’re going to find out about the `ln` command (“Create a Link Pointing to Another File or Directory”), so you might want to skip ahead and read that to fully understand what I’m about to show you. Assuming you have, check out the following:

```
# ls -l
lrwxrwxrwx  scott scott  websites -> /var/www/
$ cd websites
$ pwd
/websites
$ pwd -P
/var/www
```

So there’s a soft link with a source `websites` that points to a target `/var/www`. I `cd` using the soft link and enter `pwd`. Notice what comes back: the *logical* directory of `/websites`, the *source* of the soft link, which isn’t the actual working target directory of `/var/www`. This is the default behavior of `pwd`, equivalent to entering `pwd -L` (or `--logical`).



On the other hand, if you enter `pwd -P` (or `--physical`), you instead get back the target of the soft link, which is `/var/www`.

I don't know about you, but when I use `pwd`, I usually want to get back the actual physical location (the target), not the logical location (the source). Since I actually prefer the `-P` option as the default, I like to use an alias in my `.bash_aliases` file (discussed in Chapter 12's "Create a New Permanent Alias" section) that looks like this:

```
alias pwd="pwd -P"
```

Now you understand not only how to use `pwd`, but also the gotchas you might run into when you use it.

## Change to a Different Directory

```
cd
```

It's possible to list the contents of any directory simply by specifying its path, but often you actually want to move into a new directory. That's where the `cd` command comes in, another one that is almost constantly used by shell aficionados.

The `cd` command is simple to use: Just enter `cd`, followed by the directory into which you want to move. You can use a relative path, based on where you are currently—`cd src` or `cd ../../`—or you can use an absolute path, such as `cd /tmp` or `cd /home/scott/bin`.

## Change to Your Home Directory

```
cd ~
```

You should know about a few nice shortcuts for `cd`. No matter where you are, just enter a simple `cd`, and you'll immediately be whisked back to your home directory. This is a fantastic timesaver, one that you'll use all the time. Or, if you'd like, you can use `cd ~` because the `~` is like a shortcut meaning "my home directory."

```
$ pwd
/home/scott/music
$ cd ~
$ pwd
/home/scott
```

## Change to Your Previous Directory

```
cd -
```

Another interesting possibility is `cd -`, which takes you back to your previous directory and then runs the `pwd` command for you, printing your new (or is it old?) location. It's the shell equivalent of the Back button in a GUI file navigator window. You can see it in action in this example:

```
$ pwd
/home/scott
$ cd music/new
$ pwd
/home/scott/music/new
$ cd -
/home/scott
```

Using `cd -` can be useful when you want to jump into a directory, perform some action there, and then jump back to your original directory. The additional printing to the screen of the information provided by `pwd` is just icing on the cake to make sure you're where you want to be.

## Conclusion

If this was law school, you would have learned in this chapter what misdemeanors, torts, and felonies are. If this was a hardware repair class, it would have been RAM, hard drives, and motherboards. Since this is a book about the Linux shell, you've examined the most basic commands that a Linux user needs to know to effectively use the command line: `ls`, `pwd`, and `cd`. Now that you know how to get around the file system, it's time to learn about two things that govern the universe: creation and destruction. Now not only will you be observing your environment, you'll be modifying it to your whims. Read on!

*This page intentionally left blank*

# Index

## Symbols

---

- ![##], repeating commands with numbers, 323-324
- OOQUICKSTART file, 367
- \$() (command substitution), 122-123
- \* (asterisk)
  - file extensions, risks of using in, 206
  - file types, 35
  - wildcards, 15-16
- \ (backslash) escape character, 14, 209
- ` (backtick), command substitution, 123
- ! (bang) in regular expressions, 273
- : (colon), 209
- { } (curly braces), wildcards, 15, 18-20
- . (dot)
  - in chown command, 209
  - working directory, 61
- && (double ampersand), command stacking, 118-120
- !! (double bang), repeating commands, 321-323
- >> (double greater than symbol), output redirection, 129-130
- (double hyphen) in commands, 90
- || (double pipe), command stacking, 121
- / (forward slash)
  - file types, 35
  - in filenames, 13
- ^g (Ctrl-g), searching command-line history, 325-330
- > (greater than symbol), output redirection, 127
- (hyphen) in filenames, 13
- < (less than symbol), input redirection, 130-131
- | (pipe), output redirection, 125-126
- ? (question mark), wildcards, 15-17
- "" (quotation marks)
  - filenames, 14
  - in regular expressions, 272-273
- ^r (Ctrl-r), searching command-line history, 325-330
- ^s (Ctrl-s), searching command-line history, 325-330
- ;(semicolon), command stacking, 116-118
- [] (square brackets), wildcards, 15-20, 63
- ![string], repeating commands with strings, 324-325
- ~ (tilde), home directory, 29
- \_ (underscore) in filenames, 13

## A

---

- absolute paths in search results, 294
- Access Control Lists (ACLs), 210
- accessed (timestamps), 301
- accidental deletes, preventing, 86-87
- ack command, 275
- ACLs (Access Control Lists), 210

**activating network connections, 428-430**

**alias command, 331-334**

**aliases**

- creating, 66, 332-334
- functions versus, 344-346
- for ls command
  - options, 47
- removing, 334-335
- viewing, 331-332

**alphabetic notation**

- changing permissions, 213-215
- setting suid, 223

**AND expressions, searching by, 303-304**

**appending output to files, 129-130**

**Apper, 399**

**apropos command, 99-100**

**APT, troubleshooting, 402-403**

**apt -installed list command, 396**

**apt-cache search command, 398-399**

**apt-get clean command, 401-402**

**apt-get dist-upgrade command, 398**

**apt-get -f install command, 403**

**apt-get -f remove command, 403**

**apt-get install command, 390-394**

**apt-get remove command, 394-395**

**apt-get upgrade command, 396-398**

**archiving files, 233**

- adjusting compression levels, 237-239
- bzip2 command, 250-251
- combining with compression utilities, 255-257

encryption and password-protection, 242-243

gzip command, 245-248

recursively, 239-241, 247-248

specific file types, 239-241

tar -cf command, 253-254

zip command, 235-243

**arguments, 345-346**

**ASCII encoding, 186**

**awk command, 194-198**

---

## B

**backups**

- creating, 68-69
- secure backups, 458-465

**bash shell, 319-321**

**.bash\_aliases file, 23**

adding aliases to, 333-334

adding functions to, 338-341

removing aliases from, 335

removing functions, 342-343

**.bash\_functions file, 23, 338**

**.bash\_history file, 320**

**.bash\_login file, 21**

**.bash\_logout file, 23**

**.bash\_profile file, 21**

**.bashrc file, 22**

adding aliases to, 333-334

adding functions to, 338-341

**beginning bytes of files, viewing, 153-155**

**beginning lines of files, viewing, 150-152**

**blank lines, deleting, 166**

**Bluefish, installing, 391**

**brace expansion, 16-20**

browsers, text-based, 471  
 buffers, 369  
 bunzip2 command, 251-252  
 bunzip2 -t command, 252  
 bytes, prefixes, 154  
 bzip2 archives, 251-252  
 bzip2 command, 234,  
 250-251  
 bzip2 -[0-9] command, 235  
 bzip2 -d command, 252

## C

---

cache, 369  
 canceling commands, 32  
 case-insensitive searching  
 for files, 264-265  
 in files, 276-277  
 case-sensitivity of filenames,  
 11-12  
 cat command, 141-144  
 cat -n command, 144-145  
 cd command, 49  
 cd - command, 50-51  
 cd ~ command, 50  
 changed (timestamps), 301  
 changing  
 directories, 49  
 text editors, 149  
 character classes, 181  
 characters  
 counting, 164-167  
 deleting, 183-188  
 replacing repeated,  
 182-183  
 substituting, 180-181  
 chgrp command, 204-205  
 chgrp -c command, 199  
 chgrp -R command, 205-206  
 chgrp -v command, 199  
 chmod command  
 changing permissions,  
 213-219

sgid, setting and clearing,  
 225-227  
 sticky bit, setting and  
 clearing, 228-231  
 suid, setting and clearing,  
 221-224  
 chmod 000 command, 218  
 chmod -R command, 219-221  
 chown command, 207-209  
 chown -R command, 208  
 clear command, 24  
 clearing screen, 24  
 color-coded output  
 for directory listings, 35-37  
 with less command,  
 147-148  
 columnar output for directory  
 listings, 32  
 columns, selecting in delimited  
 files, 169-172  
 combining  
 input and output redirec-  
 tion, 131-133  
 options, 45  
 comma-separated listing for  
 directories, 33  
 command documentation, 93  
 info command, 103-108  
 man command, 94-97  
 reading specific man  
 pages, 101-103  
 searching for, 99-100  
 type command, 111-113  
 viewing command synop-  
 ses, 97-98  
 whereis command,  
 108-109  
 which command, 110-111  
 command-line history  
 repeating last command,  
 321-325  
 searching, 325-330  
 viewing, 320-321  
 commands  
 ack, 275  
 alias, 331-334

- aliases versus functions, 344-346
- apropos, 99-100
- apt -installed list, 396
- apt-cache search, 398-399
- apt-get clean, 401-402
- apt-get dist-upgrade, 398
- apt-get -f install, 403
- apt-get -f remove, 403
- apt-get install, 390-394
- apt-get remove, 394-395
- apt-get upgrade, 396-398
- awk, 194-198
- bunzip2, 251-252
- bunzip2 -t, 252
- bzip2, 234, 250-251
- bzip2 -[0-9], 235
- bzip2 -d, 252
- canceling, 32
- cat, 141-144
- cat -n, 144-145
- cd, 49
- cd -, 50-51
- cd ~, 50
- chgrp, 204-205
- chgrp -c, 199
- chgrp -R, 205-206
- chgrp -v, 199
- chmod, 213-231
- chmod 000, 218
- chmod -R, 219-221
- chown, 207-209
- chown -R, 208
- clear, 24
- command stacking, 116-121
- command substitution, 122-123
- compress, 234
- cp, 60-62
- cp \*, 62-63
- cp -a, 68-69
- cp -i, 65-66
- cp -r, 67-68
- cp -v, 64
- cron, 465
- curl, 474-476
- cut, 169-172
- declare -f, 342
- deprecated networking commands, 405
- df, 75, 369-371
- dhclient, 426-428
- dig, 414-418
- dircolors, 37
- dpkg -i, 387-388
- dpkg -l, 389
- dpkg -r, 389
- du, 371-373
- du -s, 373
- echo, 78
- egrep, 270
- fgrep, 270
- file, 138-141
- find, 291-303, 309-316
- find +, 312-315
- find -a, 303-304
- find -fprint, 292
- find -n, 307-309
- find -o, 304-307
- free, 367-369
- function, 335-341
- grep, 126, 268-289
- grep [-ABC], 281-284
- grep -c, 286-288
- grep -color=auto, 275-276
- grep -i, 276-277
- grep -l, 285-286
- grep -n, 278
- grep -R, 274
- grep -v, 284-285
- grep -w, 277-278
- gunzip, 248-249
- gunzip -t, 249
- gzip, 234, 245-246
- gzip -[0-9], 235
- gzip -d, 249
- gzip -r, 247-248
- head, 150-151
- head -c, 153-155
- head -n, 152
- history, 177, 320-321
- host, 414-418
- htop, 361
- iconv, 186
- ifconfig, 407-410, 418-421



- ifdown, 430-431
- ifup, 428-430
- import, 117
- info, 103-108
- input redirection, 130-133
- ip addr add, 418-421
- ip addr show, 407-410
- ip link set, 418-421, 428-431
- ip route, 432-439
- iwconfig, 421-424
- kill, 355-359
- killall, 358-359
- less, 97, 125, 145-149
- ln, 76-83
- locate, 262-267, 291
- locate -i, 264-265
- locate -n, 262
- ls, 28-29
- ls \*, 30
- ls -l, 32
- ls -a, 33-34
- ls -color, 35-37
- ls -F, 34-35
- ls -h, 46-47
- ls -i, 74, 82
- ls -l, 37-42, 82
- ls -m, 33
- ls of, 361-365
- ls of -c, 365-367
- ls of -u, 363-364
- ls -r, 43
- ls -R, 31-32
- ls -S, 45-46
- ls -t, 44
- man, 94-97
- man [1-8], 101-103
- man -f, 97-98
- man -k, 99-100
- man -t, 94
- man -u, 94
- mkdir, 58-59
- mkdir -p, 59-60
- mkdir -v, 27, 53
- mlocate, 263, 267
- mtr, 414
- multitail, 157
- mv, 70-73
- netstat, 433
- nl, 167-169
- nmcli, 421-424
- options, combining, 45
- output redirection. See output redirection
- pgrep, 281
- ping, 410-412, 439-440
- ping -c, 410-412
- ps, 126, 280-281
- ps aux, 350-353
- ps axjf, 353-354
- ps U, 355
- pstree, 354
- pwd, 47-49
- rename, 310
- repeating commands, 321-325
- rm, 84-85, 89-91
- rm \*, 85
- rm -i, 86-87
- rm -rf, 88-89
- rm -v, 27, 53
- rmdir, 87-88
- route, 432-439
- rpm -e, 378
- rpm -ihv, 376-378
- rpm -qa, 378
- rpm -Uhv, 376-378
- rsync, 458-465
- scp, 456-458
- searching for previous, 325-330
- sed, 188-194
- set -o noclobber, 128-129
- sftp, 453-455
- shred, 84
- sleep, 117
- slocate, 263, 267
- sort, 172-174
- sort -h, 174-176
- sort -n, 174-176
- ssh, 446-453
- ssh-agent, 453
- stat, 301
- su, 200-202
- su -, 202-203
- su -l, 201-202

- tac, 145
- tail, 155-157
- tail -f, 158-160
- tail -n, 158
- tar, 234
- tar -cf, 253-254
- tar -pzcvf, 255-257
- tar -pzvxf, 259
- tar -zvtf, 257-258
- tee, 133-135
- telnet, 446
- time, 267
- top, 359-361
- touch, 54-58
- touch -t, 55-57
- tr, 131, 180-181
- tr -d, 183-188
- tr -s, 182-183
- traceroute, 412-414
- trash, 84
- type, 111-113, 276
- unalias, 334-335
- uniq, 177-180
- unlink, 77
- unset -f, 342-343
- unzip, 243-244
- unzip -l, 245
- unzip -t, 244-245
- updatedb, 265-267
- uptime, 350
- useradd, 207
- usermod, 207
- wc, 164-167
- wget, 466-474
- whatis, 97-98
- whereis, 108-109
- which, 110-111
- whoami, 200
- wput, 472
- xargs, 314-316
- yum install, 379-382
- yum list available, 386
- yum list installed, 384
- yum remove, 382-383
- yum search, 386
- yum update, 384-385
- zip, 234-237
- zip -[0-9], 237-239
- zip -e, 242-243
- zip -i, 239-241
- zip -P, 242-243
- zip -r, 239-241
- compress command, 234**
- compressing files, 233**
  - adjusting compression levels, 237-239
  - bzip2 command, 250-251
  - combining with archiving utilities, 255-257
  - encryption and password-protection, 242-243
  - gzip command, 245-246
  - recursively, 239-241, 247-248
  - specific file types, 239-241
  - zip command, 235-237
- compression level, adjusting, 237-239**
- concatenating files**
  - to another file, 143-144
  - with line numbers, 144-145
  - to stdout, 142-143
- configuring**
  - network connections, 418-421
  - wireless connections, 425
- connectivity. See networking**
- context of words, searching files, 281-284**
- converting files**
  - encoding formats, 186
  - to Web-ready, 183-188
- copying**
  - directories, 67-68
  - files, 60-62
    - avoiding overwrites, 65-66
    - creating backups, 68-69
    - securely, 456-458
    - verbosely, 64
    - wildcards, using, 62-63
- corrupted files, testing for, 245**
  - bzip2 archives, 252

gzip archives, 249  
tarballs, 257-258  
zip archives, 244

### counting

search terms, 286-288  
words/lines/characters,  
164-167

cp command, 60-62

cp \* command, 62-63

cp -a command, 68-69

cp -i command, 65-66

cp -r command, 67-68

cp -v command, 64

cron command, 465

Ctrl+C command, 32

Ctrl-g, 325-330

Ctrl-r, 325-330

Ctrl-s, 325-330

curl command, 474-476

cut command, 169-172

## D

database for locate command,  
updating, 265-267

date, sorting directory contents  
by, 44

datestamps, changing on files,  
55-57

### DEB packages, 375

finding, 398-399  
installing, 387-394  
listing installed, 389, 396  
removing, 389, 394-395  
removing installers,  
401-402  
troubleshooting APT,  
402-403  
upgrading with dependen-  
cies, 396-398

declare -f command, 342

default ports (SSH serv-  
ers), 455

deleting. *See* removing

delimited files, selecting col-  
umns, 169-172

### dependencies

DEB packages, 390-398  
RPM packages, 379-385  
troubleshooting APT, 403

deprecated networking com-  
mands, 405

df command, 75, 369-371

dhclient command, 426-428

DHCP (Dynamic Host Control  
Protocol) addresses, acquir-  
ing, 426-428

dig command, 414-418

Digital Signature Algorithm  
(DSA), 451

dircolors command, 37

### directories, 9

changing, 49  
contents of, listing. *See* ls  
command  
copying, 67-68  
creating, 58-60  
current directory, determin-  
ing, 47-49  
deleting, 87-89  
file space usage, monitor-  
ing, 371-373  
home directory, 29, 50  
inodes, 75  
links to, creating, 76-83  
ownership. *See* ownership  
of files  
permissions. *See* per-  
missions  
previous directory, chang-  
ing to, 50-51  
renaming, 72-73  
working directory, 61

Directory node, 106

disabling XOFF flow con-  
trol, 328

disconnecting network connec-  
tions, 430-431

- disk usage, monitoring, 369-373
- distributions, 375. *See also* DEB packages; RPM packages
- DNF (Dandified Yum), 380
- DNS (Domain Name System)
  - querying records, 414-418
  - troubleshooting, 441
- documentation of commands, 93
  - info command, 103-104
  - man command, 94-97
  - navigating in info command, 104-108
  - reading specific man pages, 101-103
  - searching for, 99-100
  - type command, 111-113
  - viewing command synopses, 97-98
  - whereis command, 108-109
  - which command, 110-111
- Domain Name System. *See* DNS
- dotfiles, 20-23
- downloading
  - files, 466-471, 474-476
  - websites non-interactively, 472-474
- dpkg -i command, 387-388
- dpkg -l command, 389
- dpkg -r command, 389
- DSA (Digital Signature Algorithm), 451
- du command, 371-373
- du -s command, 373
- duplicate lines, removing, 177-180
- Dynamic Host Control Protocol (DHCP) addresses, acquiring, 426-428

---

## E

- ECDSA (Elliptic Curve Digital Signature Algorithm), 451
- echo command, 78
- editing
  - files from pagers, 149
  - .inputrc file, 329-330
  - known\_hosts file, 339
- egrep command, 270
- Eiciel, 210
- empty directories, deleting, 87-88
- empty files, creating, 57-58
- encoding formats, converting, 186
- encrypting zip archives, 242-243
- encryption algorithms, 451
- ending lines of files, viewing, 155-160
- ending processes, 355-359
- environment variables, 201-203
- executable programs, finding, 108-109
- executing commands on search results, 309-316
- extensions, determining file type, 138-141

---

## F

- fgrep command, 270
- fields, printing specific, 194-198
- file attributes, abbreviations, 211
- file command, 138-141
- file size
  - human-readable, 46-47
  - listing, 37-42
- file system disk usage, viewing, 369-371
- filename expansion, 16

**filenames**

- case-sensitivity, 11-12
- extensions, determining
  - file type, 138-141
- length of, 10-11
- listing as search results, 285-286
- searching by, 292-294
- with spaces, executing
  - commands on, 315-316
- special characters, 12-14, 89-91

**files, 9-10**

- appending output to, 129-130
- archiving. See archiving files
- backing up securely, 458-465
- combining input and output redirection, 131-133
- as command input, 130-131
- compressing. See compressing files
- concatenating. See concatenating files
- converting. See converting files
- copying. See copying, files
- counting words/lines/characters, 164-167
- creating links to, 76-83
- deleting, 84-91
- delimited files, selecting columns, 169-172
- determining type, 138-141
- downloading, 466-471, 474-476
- editing from pagers, 149
- empty files, creating, 57-58
- file type, displaying, 34-35
- filters, 163-164
- hidden files, viewing, 33-34
- inodes, 74-75

- listing. See ls command;
  - ls of command
- moving, 70-72
- naming, 10-14
- numbering lines, 167-169
- output redirection to, 127
- overwriting, avoiding, 65-66, 128-129
- ownership. See ownership of files
- permissions. See permissions
- printing specific fields, 194-198
- removing duplicate lines, 177-180
- renaming, 72-73
- searching for. See searching, for files
- searching in. See searching, in files
- sorting contents, 172-176
- splitting output streams, 133-135
- storing, 74-75
- time, changing on, 54-57
- transferring securely, 453-455
- transforming text, 188-194
- unarchiving, testing for corruption, 257-258
- uncompressing. See uncompressing files
- undeleting, 84
- uploading non-interactively, 472
- viewing. See viewing, files

**filters, 163-164**

- counting words/lines/characters, 164-167
- deleting characters, 183-188
- numbering lines in files, 167-169
- printing specific fields, 194-198
- removing duplicate lines, 177-180

- replacing repeated characters, 182-183
  - selecting columns in delimited files, 169-172
  - sorting file contents, 172-176
  - substituting characters, 180-181
  - transforming text, 188-194
  - find command**
    - executing commands on results, 309-312, 315-316
    - locate command versus, 291
    - searching by file size, 295-297
    - searching by file type, 298-299
    - searching by name, 292-294
    - searching by ownership, 294-295
    - searching by time, 300-303
  - find + command, 312-315**
  - find -a command, 303-304**
  - find -fprint command, 292**
  - find -n command, 307-309**
  - find -o command, 304-307**
  - finding**
    - command paths, 108-109
    - command versions, 110-111
    - DEB packages, 398-399
    - files. *See* searching for files
    - GID (Group ID), 205
    - graphical package managers, 387
    - RPM packages, 386
  - folders. *See* directories**
  - Forta, Ben, 98**
  - free command, 367-369**
  - function command, 335-341**
  - functions**
    - aliases versus, 344-346
    - creating, 335-341
    - removing, 342-343
    - viewing all, 341-342
    - Web-ready files function, 183-188
- 
- ## G
- GID (Group ID), 204-205**
  - GitHub, viewing dotfiles, 21**
  - GNOME PackageKit, 400**
  - GNOME Software, 400**
  - graphical package managers, 387, 399-401**
  - Greer, Brian, 441**
  - grep command, 126**
    - regular expressions, explained, 269-273
    - searching command output, 279-281
    - searching in files, 268-269
    - searching within results, 288-289
    - versions of, 269
  - grep [-ABC] command, 281-284**
  - grep -c command, 286-288**
  - grep -color=auto command, 275-276**
  - grep -i command, 276-277**
  - grep -l command, 285-286**
  - grep -n command, 278**
  - grep -R command, 274**
  - grep -v command, 284-285**
  - grep -w command, 277-278**
  - Group ID (GID), 204-205**
  - group ownership**
    - changing, 204-209
    - searching by, 294-295
  - gunzip command, 248-249**
  - gunzip -t command, 249**
  - gzip archives, 248-249**
  - gzip command, 234, 245-246**
  - gzip -[0-9] command, 235**

`gzip -d` command, 249  
`gzip -r` command, 247-248

---

## H

hard links, 76-78, 81-83  
`head` command, 150-151  
`head -c` command, 153-155  
`head -n` command, 152  
 hidden files, viewing, 33-34  
 hidden startup scripts. See `dotfiles`  
 highlighting search results, 275-276  
 history. See `command-line history`  
`history` command, 177, 320-321  
 home directory, 29, 50  
`host` command, 414-418  
`$HOSTNAME` environment variable, 171  
`htop` command, 361  
 human-readable file sizes, 46-47

---

## I

`iconv` command, 186  
`ifconfig` command, 407-410, 418-421  
`ifdown` command, 430-431  
`ifup` command, 428-430  
`import` command, 117  
`info` command, 103-108  
 inodes, 74-75, 82  
 input/output streams, 123-124
 

- combining input and output redirection, 131-133
- input redirection, 130-131
- output redirection. See `output redirection`

`.inputrc` file, 23, 329-330  
 installers, removing, 401-402

installing
 

- Bluefish, 391
- kernels, 378
- Shotwell, 380-382
- Skype, 387
- software. See `package management`

 interactive shells, 22  
`ip addr add` command, 418-421  
`ip addr show` command, 407-410  
`ip link set` command, 418-421, 428-431  
`ip route` command, 432-439  
 IP routing table
 

- changing, 435-439
- displaying, 432-435

 IPv4 addressing, 406, 433  
 IPv6 addressing, 406  
`iwconfig` command, 421-424

---

## K

kernels, installing, 378  
 key commands for `less` command, 146  
`kill` command, 355-359  
`killall` command, 358-359  
`known_hosts` file (SSH), 339, 448-449

---

## L

length of filenames, 10-11  
`less` command, 125
 

- color control characters, 147-148
- editing files from, 149
- with `man` command, 97
- searching in, 148
- viewing files per screen, 145-147

 line numbers
 

- adding to files, 144-145, 167-169

of search results, showing, 278

### lines

blank lines, deleting, 166  
counting, 164-167  
removing duplicates, 177-180

### links

creating, 76-83  
hard links, 76-78, 81-83  
removing, 77  
soft links, 79-83  
viewing, 82

**Linux distributions, 375. See also DEB packages; RPM packages**

**LinuxMM, 369**

**listing. See also viewing**

filenames as search results, 285-286  
files in zip archives, 245.  
See also ls command;  
ls of command  
installed DEB packages, 389, 396  
installed RPM packages, 378, 384  
open files, 361-363  
program processes, 365-367  
users of files, 364-365  
users' open files, 363-364

**ln command, 76-83**

**locate command, 262-267, 291**

**locate -i command, 264-265**

**locate -n command, 262**

**log files, viewing updated ending lines, 158-160**

**logging in remotely, 446-453**

**login shells, 21**

**loopback addresses, 408**

**ls command, 28-29**

**ls \* command, 30**

**ls -l command, 32**

**ls -a command, 33-34**

**ls -color command, 35-37**

**ls -F command, 34-35**

**ls -h command, 46-47**

**ls -i command, 74, 82**

**ls -l command, 37-42, 82**

**ls -m command, 33**

**ls -r command, 43**

**ls -R command, 31-32**

**ls -S command, 45-46**

**ls -t command, 44**

**ls of command, 361-365**

**ls of -c command, 365-367**

**ls of -u command, 363-364**

## M

---

**man command, 94-97**

**man [1-8] command, 101-103**

**man -f command, 97-98**

**man -k command, 99-100**

**man -t command, 94**

**man -u command, 94**

**man pages**

finding, 108-109

navigating, 96

reading specific, 101-103

searching, 96

sections in, 96

**Markdown files, 140**

**memory usage, monitoring, 367-369**

**mkdir command, 58-59**

**mkdir -p command, 59-60**

**mkdir -v command, 27, 53**

**mlocate command, 263, 267**

**modified (timestamps), 301**

**monitoring. See system monitoring**

**moving**

files, 70-72

soft links, 73

**mtr command, 414**



**multi-line commands, repeating, 323**

**multiple files**

- viewing ending lines, 156-157
- viewing beginning lines, 151

**multitail command, 157**

**Muon Discover, 400**

**mv command, 70-73**

---

## N

**names. See filenames**

**naming files**

- case-sensitivity of filenames, 11-12
- length of filenames, 10-11
- special characters in filenames, 12-14

**navigating**

- in info command, 104-108
- man pages, 96

**negative matches, searching files for, 284-285**

**netstat command, 433**

**networking, 405**

- acquiring DHCP addresses, 426-428
- activating connections, 428-430
- changing IP routing table, 435-439
- configuring connections, 418-421
- deprecated commands, 405
- disconnecting connections, 430-431
- displaying IP routing table, 432-435
- IPv6 addressing, 406
- loopback addresses, 408
- non-interactive file downloads, 466-471
- non-interactive website downloads, 472-474

querying DNS records, 414-418

security. *See* security sequential file downloads, 474-476

tracing packet routes, 412-414

troubleshooting, 439-443

verifying connection status, 410-412

viewing connection status, 407-410

wireless connections, 421-425

**nl command, 167-169**

**nmcli command, 421-424**

**noclobber option, 128-129**

**nodes, 103-108**

**non-interactive file downloads, 466-471**

**non-interactive shells, 22**

**non-interactive website downloads, 472-474**

**nonlogin shells, 22**

**NOT expressions, searching by, 307-309**

**NR variable (awk), 197**

**numbered lines**

- adding to files, 144-145, 167-169
- in search results, 278

**numerically sorting file contents, 174-176**

**numeric notation**

- changing permissions, 215-219
- repeating commands, 323-324
- setting suid, 223

---

## O

**octal notation. See numeric notation**

**Octopi, 400**

- open files, listing, 361-364
- options, combining, 45
- OR expressions, searching by, 304-307
- output of commands, searching, 279-281
- output redirection
  - combining with input redirection, 131-133
  - concatenating files, 143-144
  - double greater than symbol (>>), 129-130
  - greater than symbol (>), 127
  - pipe (|), 124-126
  - preventing overwrites, 128-129
  - tee command, 133-135
- overwriting files
  - avoiding, 65-66
  - preventing, 128-129
- ownership of files, 199
  - changing, 207-209
  - group owners, changing, 204-206
  - listing, 37-42
  - searching by, 294-295

## P

---

### package management, 375-376

- Debian distributions. See DEB packages
- graphical package managers, 399-401
- RPM-based distributions. See RPM packages
- packets, tracing routes, 412-414
- paggers, 97, 145
  - editing files from, 149
  - searching in, 148
- pandoc, 394
- parameters, arguments versus, 345

- Parent Process ID (PPID) number, 354
- password-protecting zip archives, 242-243
- passwords, remote logins without, 450-453
- paths
  - for commands, finding, 108-109
  - current directory path, displaying, 47-49
  - in search results, 293
- pattern-searching. See regular expressions; searching, in files
- permanent aliases, creating, 333-334
- permanent functions, creating, 338-341
- permissions, 199, 210-212
  - alphabetic notation, changing with, 213-215
  - changing recursively, 219-221
  - file attributes, abbreviations for, 211
  - listing, 37-42
  - numeric notation, changing with, 215-219
  - root users, 212
  - sgid, setting and clearing, 225-227
  - sticky bit, setting and clearing, 228-231
  - suid, setting and clearing, 221-224
- pgrep command, 281
- PID, with tail command, 160
- ping command, 410-412, 439-440
- ping -c command, 410-412
- positional parameters, 337, 345
- PPID (Parent Process ID) number, 354
- prefixes for bytes, 154

previous directory, changing to, 50-51

printing specific fields, 194-198

process trees, viewing, 353-354

#### processes

- ending, 355-359
- process trees, viewing, 353-354
- program processes, listing, 365-367
- running processes, viewing, 350-353, 359-361
- user-owned processes, viewing, 355

.profile file, 21

program processes, listing, 365-367

ps aux command, 350-353

ps axjf command, 353-354

ps command, 126, 280-281

ps U command, 355

pstree command, 354

pwd command, 47-49

---

## Q

querying DNS records, 414-418

---

## R

RAM usage, monitoring, 367-369

ranges, wildcards, 19

reading specific man pages, 101-103

Readline library, 23

recovering deleted files, 84

#### recursive actions

- archiving/compressing files, 239-241, 247-248
- changing file ownership, 208

- changing group ownership, 205-206

- changing permissions, 219-221

- searching in files, 274

redirection. *See* input/output streams, input redirection; output redirection

#### regular expressions

- in grep command, 269-273
- quotation marks in, 272-273
- resources for information, 98
- with sed command, 191-194
- special characters in, 271
- with whatis command, 98
- wildcards versus, 271

relative paths in search results, 293

remote logins, 446-453

#### removing

- aliases, 334-335
- blank lines, 166
- characters, 183-188
- DEB packages, 389, 394-395, 401-402
- directories, 87-89
- duplicate lines, 177-180
- files, 84-91
- functions, 342-343
- links, 77
- RPM packages, 378, 382-383
- Shotwell, 382-383

rename command, 310

renaming files/directories, 72-73

repeated characters, replacing, 182-183

repeating commands, 321-325

replacing repeated characters, 182-183

repositories, 392

reverse DNS lookups, 415

reverse-order output for directory contents, 43

**rm** command, 84-85, 89-91

**rm \*** command, 85

**rm -i** command, 86-87

**rm -rf** command, 88-89

**rm -v** command, 27, 53

**rmdir** command, 87-88

**root** user

cp command, 66

permissions, 212

switching to, 202-203

**route** command, 432-439

**routing** table

changing, 435-439

displaying, 432-435

**rpm -e** command, 378

**rpm -ihv** command, 376-378

**rpm -qa** command, 378

**rpm -Uhv** command, 376-378

**RPM** packages, 375

APT and, 391

finding, 386

installing, 376-382

listing installed, 378, 384

removing, 378, 382-383

upgrading with dependencies, 384-385

**RSA** encryption, 451

**rsync** command, 458-465

**running** processes

ending, 355-359

viewing, 350-353, 359-361

## S

---

*Sams Teach Yourself Regular Expressions in 10 Minutes* (Forta), 98, 269

**scp** command, 456-458

screen, clearing, 24. *See also* **stdout**

screenshots, 117

**searching**

for command documentation, 99-100

command output, 279-281

command-line history, 325-330

for files

with AND expressions, 303-304

case-insensitive searching, 264-265

executing commands on results, 309-316

locate command, 262-264

by name, 292-294

with NOT expressions, 307-309

with OR expressions, 304-307

by ownership, 294-295

by size, 295-297

by time, 300-303

by type, 298-299

updating locate database, 265-267

in files

case-insensitive searching, 276-277

grep command, 268-269

highlighting results, 275-276

line numbers of results, showing, 278

listing filenames as search results, 285-286

for negative matches, 284-285

recursively, 274

regular expressions, explained, 269-273

whole-word searching, 277-278

within search results, 288-289

word counts, 286-288

words in context, 281-284

- info pages, 107
- man pages, 96
- in pagers, 148
- secondary prompts, 337**
- section numbers of man pages, 101-103**
- security**
  - backing up files, 458-465
  - .bash\_history file, 320
  - copying files, 456-458
  - file transfers, 453-455
  - passwordless remote log-  
ins, 450-453
  - remote logins, 446-449
- sed command, 188-194**
- selecting columns in delimited files, 169-172**
- sequential commands, 116-121**
- sequential file downloads, 474-476**
- set -o noclobber command, 128-129**
- sftp command, 453-455**
- sgid, 211, 225-227**
- shared object (.so) files, 367**
- shells, 21-22, 319-321**
- Shotwell, 380-383**
- shred command, 84**
- size**
  - searching by, 295-297
  - sorting directory contents  
by, 45-46
- Skype, installing, 387**
- sleep command, 117**
- slocate command, 263, 267**
- Smart, 400**
- .so (shared object) files, 367**
- soft links, 79-81**
  - hard links versus, 81-83
  - moving, 73
- software installation. See package management**
- sort command, 172-174**
- sort -h command, 174-176**
- sort -n command, 174-176**
- sorting**
  - directory contents, 44-46
  - file contents, 172-176
- source files, finding, 108-109**
- spaces in filenames, 13, 315-316**
- special characters**
  - in filenames, 12-14, 89-91
  - in regular expressions, 271
- specific file types, archiving/  
compressing, 239-241**
- splitting output streams, 133-135**
- ssh command**
  - passwordless remote log-  
ins, 450-453
  - secure remote logins,  
446-449
- SSH keys, editing known\_hosts  
file, 339**
- SSH servers, default port, 455**
- ssh-agent command, 453**
- startup scripts. See dotfiles**
- stat command, 301**
- STAT letters, ps com-  
mand, 352**
- stderr, 123-124. See also  
input/output streams**
- stdin, 123-124. See also  
input/output streams**
- stdout, 123-124. See also  
input/output streams**
  - concatenating files to,  
142-143
  - splitting output streams,  
133-135
- sticky bit, 211, 228-231**
- storing files, 74-75**
- stream editor. See sed  
command**
- streams. See input/output  
streams**

strings, repeating commands, 324-325

su command  
becoming root, 202  
switching users, 200

su - command, 202-203

su -l command, 201-202

subdirectories  
creating with directories, 59-60  
listing contents, 31-32

subnodes, 103-108

subshells, 21

substituting characters, 180-181

successful commands, command stacking, 118-120

suid, 211, 221-224

switching users, 200-203

symbolic links. *See* soft links

symlinks. *See* soft links

Synaptic, 400

system monitoring, 349  
directory file space usage, 371-373  
file system disk usage, 369-371  
files, listing users, 364-365  
open files, listing, 361-364  
processes. *See* processes  
RAM usage, monitoring, 367-369  
runtime of system, 350

## T

---

tac command, 145

tail command, 155-157

tail -f command, 158-160

tail -n command, 158

tar command, 234

tar -cf command, 253-254

tar -pzcvf command, 255-257

tar -pvxf command, 259

tar -zvtf command, 257-258

tarballs, 234, 253  
testing for file corruption, 257-258  
uncompressing, 259

Taylor's Law of Programming, 198

tee command, 133-135

telnet command, 446

temporary aliases, creating, 332-333

temporary functions, creating, 335-338

testing for corrupted files, 245  
bzip2 archives, 252  
gzip archives, 249  
tarballs, 257-258  
zip archives, 244

text, transforming, 188-194

text-based Web browsers, 471

text editors  
changing, 149  
pipe (|) and, 126

text files. *See* files

time  
changing on files, 54-57  
sorting directory contents by, 44

time command, 267

timestamps, searching by, 300-303

top command, 359-361

touch command, 54-58

touch -t command, 55-57

tr command, 131, 180-181

tr -d command, 183-188

tr -s command, 182-183

traceroute command, 412-414

transferring file securely, 453-455

transforming text, 188-194

trash command, 84

troubleshooting  
 APT, 402-403  
 Ctrl-s functionality, 328  
 networking, 439-443  
 type command, 111-113, 276  
 type of files, searching by,  
 298-299

## U

---

Ubuntu Software Center, 400  
 UID (User ID), 204  
 unalias command, 334-335  
 unarchiving files, 257-258  
 uncompressing files  
 bunzip2 command,  
 251-252  
 gzip command, 248-249  
 listing files, 245  
 tar -pzxvf command, 259  
 testing for corruption,  
 244-245, 249, 252  
 unzip command, 243-244  
 undeleting files, 84  
 uniq command, 177-180  
 unlink command, 77  
 unset -f command, 342-343  
 unsuccessful commands, com-  
 mand stacking, 121  
 unzip command, 243-244  
 unzip -l command, 245  
 unzip -t command, 244-245  
 updated ending lines of files,  
 viewing, 158-160  
 updatedb command, 265-267  
 updating locate database,  
 265-267  
 upgrading  
 DEB packages with depen-  
 dencies, 396-398  
 RPM packages with depen-  
 dencies, 384-385  
 uploading files non-interactively, 472

uptime command, 350  
 User ID. *See* UID  
 user-owned processes, view-  
 ing, 355  
 useradd command, 207  
 usermod command, 207  
 users  
 abbreviations, 210  
 of files, listing, 364-365  
 listing open files, 363-364  
 ownership of files, search-  
 ing by, 294-295  
 switching, 200-203  
 UTF-8 encoding, 186

## V

---

-v (verbose) option, 65  
 verbosely copying files, 64  
 verifying network connection  
 status, 410-412  
 versions of commands, find-  
 ing, 110-111  
 vi, 150  
 viewing. *See also* listing  
 aliases, 331-332  
 command-line history,  
 320-321  
 command synopses,  
 97-98  
 dotfiles at GitHub, 21  
 file system disk usage,  
 369-371  
 files, 137  
 beginning bytes,  
 153-155  
 beginning lines, 150-152  
 cat command, 141-142  
 ending lines, 155-158  
 per screen, 145-147  
 updated ending lines,  
 158-160  
 functions, 341-342  
 hidden files, 33-34  
 inode numbers, 74, 82  
 IP routing table, 432-435

- links, 82
- network connection status, 407-410
- process trees, 353-354
- RAM usage, 367-369
- running processes, 350-353, 359-361
- user-owned processes, 355
- wireless connection status, 421-424

**vim**, 150

---

## W

**Wall, Larry**, 310

**wc command**, 164-167

**Web browsers, text-based**, 471

**Web-ready files**, 183-188

**websites, downloading non-interactively**, 472-474

**wget command**, 466-474

**whatis command**, 97-98

**whereis command**, 108-109

**which command**, 110-111

**whoami command**, 200

**whole-word searching in files**, 277-278

**Wieers, Dag**, 391

**wildcards, 15-20. See also regular expressions**

- copying files, 62-63
- deleting files, 85, 89
- directory contents, listing, 30
- regular expressions versus, 271
- searching for command documentation, 100
- searching man database, 98

**wireless connections**, 421-425

**word counts**, 164-167, 286-288

**words in context, searching files**, 281-284

**working directory**, 61

**wput command**, 472

---

## X

**xargs command**, 314-316

**XOFF flow control, disabling**, 328

---

## Y

**yum install command**, 379-382

**yum list available command**, 386

**yum list installed command**, 384

**yum remove command**, 382-383

**yum search command**, 386

**yum update command**, 384-385

**Yumex**, 400

---

## Z

**Zero Install**, 401

**zip archives**

- listing files in, 245
- testing for file corruption, 244-245
- uncompressing, 243-244

**zip command**, 234-237

**zip -[0-9] command**, 237-239

**zip -e command**, 242-243

**zip -i command**, 239-241

**zip -P command**, 242-243

**zip -r command**, 239-241

**zombie processes**, 352, 358