
Table of Contents

Preface	xiii
1. Beginning bash	1
1.1 Decoding the Prompt	4
1.2 Showing Where You Are	5
1.3 Finding and Running Commands	6
1.4 Getting Information About Files	8
1.5 Showing All Hidden (dot) Files in the Current Directory	10
1.6 Using Shell Quoting	12
1.7 Using or Replacing Built-ins and External Commands	13
1.8 Determining If You Are Running Interactively	15
1.9 Setting bash As Your Default Shell	16
1.10 Getting bash for Linux	17
1.11 Getting bash for xBSD	20
1.12 Getting bash for Mac OS X	21
1.13 Getting bash for Unix	22
1.14 Getting bash for Windows	23
1.15 Getting bash Without Getting bash	24
1.16 Learning More About bash Documentation	25
2. Standard Output	28
2.1 Writing Output to the Terminal/Window	29
2.2 Writing Output but Preserving Spacing	30
2.3 Writing Output with More Formatting Control	31
2.4 Writing Output Without the Newline	32
2.5 Saving Output from a Command	33
2.6 Saving Output to Other Files	34

2.7	Saving Output from the ls Command	35
2.8	Sending Both Output and Error Messages to Different Files	37
2.9	Sending Both Output and Error Messages to the Same File	37
2.10	Appending Rather Than Clobbering Output	39
2.11	Using Just the Beginning or End of a File	39
2.12	Skipping a Header in a File	40
2.13	Throwing Output Away	41
2.14	Saving or Grouping Output from Several Commands	41
2.15	Connecting Two Programs by Using Output As Input	43
2.16	Saving a Copy of Output Even While Using It As Input	44
2.17	Connecting Two Programs by Using Output As Arguments	46
2.18	Using Multiple Redirects on One Line	47
2.19	Saving Output When Redirect Doesn't Seem to Work	48
2.20	Swapping STDERR and STDOUT	50
2.21	Keeping Files Safe from Accidental Overwriting	52
2.22	Clobbering a File on Purpose	53
3.	Standard Input	55
3.1	Getting Input from a File	55
3.2	Keeping Your Data with Your Script	56
3.3	Preventing Weird Behavior in a Here-Document	57
3.4	Indenting Here-Documents	59
3.5	Getting User Input	60
3.6	Getting Yes or No Input	61
3.7	Selecting from a List of Options	64
3.8	Prompting for a Password	65
4.	Executing Commands	67
4.1	Running Any Executable	67
4.2	Telling If a Command Succeeded or Not	69
4.3	Running Several Commands in Sequence	71
4.4	Running Several Commands All at Once	72
4.5	Deciding Whether a Command Succeeds	74
4.6	Using Fewer if Statements	75
4.7	Running Long Jobs Unattended	76
4.8	Displaying Error Messages When Failures Occur	77
4.9	Running Commands from a Variable	78
4.10	Running All Scripts in a Directory	79

5. Basic Scripting: Shell Variables	80
5.1 Documenting Your Script	82
5.2 Embedding Documentation in Shell Scripts	83
5.3 Promoting Script Readability	85
5.4 Separating Variable Names from Surrounding Text	86
5.5 Exporting Variables	87
5.6 Seeing All Variable Values	89
5.7 Using Parameters in a Shell Script	90
5.8 Looping Over Arguments Passed to a Script	91
5.9 Handling Parameters with Blanks	92
5.10 Handling Lists of Parameters with Blanks	94
5.11 Counting Arguments	96
5.12 Consuming Arguments	98
5.13 Getting Default Values	99
5.14 Setting Default Values	100
5.15 Using null As a Valid Default Value	101
5.16 Using More Than Just a Constant String for Default	102
5.17 Giving an Error Message for Unset Parameters	103
5.18 Changing Pieces of a String	105
5.19 Using Array Variables	106
6. Shell Logic and Arithmetic	108
6.1 Doing Arithmetic in Your Shell Script	108
6.2 Branching on Conditions	111
6.3 Testing for File Characteristics	114
6.4 Testing for More Than One Thing	117
6.5 Testing for String Characteristics	118
6.6 Testing for Equal	119
6.7 Testing with Pattern Matches	121
6.8 Testing with Regular Expressions	122
6.9 Changing Behavior with Redirections	125
6.10 Looping for a While	126
6.11 Looping with a read	128
6.12 Looping with a Count	130
6.13 Looping with Floating-Point Values	131
6.14 Branching Many Ways	132
6.15 Parsing Command-Line Arguments	134
6.16 Creating Simple Menus	137

6.17	Changing the Prompt on Simple Menus	138
6.18	Creating a Simple RPN Calculator	139
6.19	Creating a Command-Line Calculator	142
7.	Intermediate Shell Tools I	144
7.1	Sifting Through Files for a String	145
7.2	Getting Just the Filename from a Search	147
7.3	Getting a Simple True/False from a Search	148
7.4	Searching for Text While Ignoring Case	149
7.5	Doing a Search in a Pipeline	149
7.6	Paring Down What the Search Finds	151
7.7	Searching with More Complex Patterns	152
7.8	Searching for an SSN	153
7.9	Grepping Compressed Files	154
7.10	Keeping Some Output, Discarding the Rest	155
7.11	Keeping Only a Portion of a Line of Output	156
7.12	Reversing the Words on Each Line	157
7.13	Summing a List of Numbers	158
7.14	Counting String Values	159
7.15	Showing Data As a Quick and Easy Histogram	161
7.16	Showing a Paragraph of Text After a Found Phrase	163
8.	Intermediate Shell Tools II	165
8.1	Sorting Your Output	165
8.2	Sorting Numbers	166
8.3	Sorting IP Addresses	167
8.4	Cutting Out Parts of Your Output	170
8.5	Removing Duplicate Lines	171
8.6	Compressing Files	172
8.7	Uncompressing Files	174
8.8	Checking a tar Archive for Unique Directories	175
8.9	Translating Characters	176
8.10	Converting Uppercase to Lowercase	177
8.11	Converting DOS Files to Linux Format	178
8.12	Removing Smart Quotes	179
8.13	Counting Lines, Words, or Characters in a File	180
8.14	Rewrapping Paragraphs	181
8.15	Doing More with less	181

9. Finding Files: find, locate, slocate	184
9.1 Finding All Your MP3 Files	184
9.2 Handling Filenames Containing Odd Characters	186
9.3 Speeding Up Operations on Found Files	187
9.4 Finding Files Across Symbolic Links	188
9.5 Finding Files Irrespective of Case	188
9.6 Finding Files by Date	189
9.7 Finding Files by Type	191
9.8 Finding Files by Size	192
9.9 Finding Files by Content	192
9.10 Finding Existing Files and Content Fast	194
9.11 Finding a File Using a List of Possible Locations	195
10. Additional Features for Scripting	199
10.1 “Daemon-izing” Your Script	199
10.2 Reusing Code with Includes and Sourcing	200
10.3 Using Configuration Files in a Script	202
10.4 Defining Functions	203
10.5 Using Functions: Parameters and Return Values	205
10.6 Trapping Interrupts	207
10.7 Redefining Commands with alias	211
10.8 Avoiding Aliases, Functions	213
11. Working with Dates and Times	216
11.1 Formatting Dates for Display	217
11.2 Supplying a Default Date	218
11.3 Automating Date Ranges	220
11.4 Converting Dates and Times to Epoch Seconds	222
11.5 Converting Epoch Seconds to Dates and Times	223
11.6 Getting Yesterday or Tomorrow with Perl	224
11.7 Figuring Out Date and Time Arithmetic	225
11.8 Handling Time Zones, Daylight Saving Time, and Leap Years	227
11.9 Using date and cron to Run a Script on the Nth Day	228
12. End-User Tasks As Shell Scripts	230
12.1 Starting Simple by Printing Dashes	230
12.2 Viewing Photos in an Album	232
12.3 Loading Your MP3 Player	237
12.4 Burning a CD	242
12.5 Comparing Two Documents	244

13. Parsing and Similar Tasks	248
13.1 Parsing Arguments for Your Shell Script	248
13.2 Parsing Arguments with Your Own Error Messages	251
13.3 Parsing Some HTML	253
13.4 Parsing Output into an Array	255
13.5 Parsing Output with a Function Call	256
13.6 Parsing Text with a read Statement	257
13.7 Parsing with read into an Array	258
13.8 Getting Your Plurals Right	259
13.9 Taking It One Character at a Time	260
13.10 Cleaning Up an SVN Source Tree	261
13.11 Setting Up a Database with MySQL	262
13.12 Isolating Specific Fields in Data	264
13.13 Updating Specific Fields in Data Files	266
13.14 Trimming Whitespace	268
13.15 Compressing Whitespace	271
13.16 Processing Fixed-Length Records	273
13.17 Processing Files with No Line Breaks	275
13.18 Converting a Data File to CSV	277
13.19 Parsing a CSV Data File	278
14. Writing Secure Shell Scripts	280
14.1 Avoiding Common Security Problems	282
14.2 Avoiding Interpreter Spoofing	283
14.3 Setting a Secure \$PATH	283
14.4 Clearing All Aliases	285
14.5 Clearing the Command Hash	286
14.6 Preventing Core Dumps	287
14.7 Setting a Secure \$IFS	287
14.8 Setting a Secure umask	288
14.9 Finding World-Writable Directories in Your \$PATH	289
14.10 Adding the Current Directory to the \$PATH	291
14.11 Using Secure Temporary Files	292
14.12 Validating Input	296
14.13 Setting Permissions	298
14.14 Leaking Passwords into the Process List	299
14.15 Writing setuid or setgid Scripts	300
14.16 Restricting Guest Users	301
14.17 Using chroot Jails	303

14.18	Running As a Non-root User	305
14.19	Using sudo More Securely	305
14.20	Using Passwords in Scripts	307
14.21	Using SSH Without a Password	308
14.22	Restricting SSH Commands	316
14.23	Disconnecting Inactive Sessions	318
15.	Advanced Scripting	320
15.1	Finding bash Portably for #!	321
15.2	Setting a POSIX \$PATH	322
15.3	Developing Portable Shell Scripts	324
15.4	Testing Scripts in VMware	326
15.5	Using for Loops Portably	327
15.6	Using echo Portably	329
15.7	Splitting Output Only When Necessary	332
15.8	Viewing Output in Hex	333
15.9	Using bash Net-Redirection	334
15.10	Finding My IP Address	335
15.11	Getting Input from Another Machine	340
15.12	Redirecting Output for the Life of a Script	342
15.13	Working Around “argument list too long” Errors	343
15.14	Logging to syslog from Your Script	345
15.15	Sending Email from Your Script	345
15.16	Automating a Process Using Phases	348
16.	Configuring and Customizing bash	352
16.1	bash Startup Options	353
16.2	Customizing Your Prompt	353
16.3	Change Your \$PATH Permanently	361
16.4	Change Your \$PATH Temporarily	362
16.5	Setting Your \$CDPATH	367
16.6	Shortening or Changing Command Names	369
16.7	Adjusting Shell Behavior and Environment	371
16.8	Adjusting readline Behavior Using .inputrc	371
16.9	Keeping a Private Stash of Utilities by Adding ~/bin 373	
16.10	Using Secondary Prompts: \$PS2, \$PS3, \$PS4	374
16.11	Synchronizing Shell History Between Sessions	376
16.12	Setting Shell History Options	377

16.13	Creating a Better cd Command	380
16.14	Creating and Changing into a New Directory in One Step	381
16.15	Getting to the Bottom of Things	383
16.16	Adding New Features to bash Using Loadable Built-ins	384
16.17	Improving Programmable Completion	389
16.18	Using Initialization Files Correctly	394
16.19	Creating Self-Contained, Portable RC Files	398
16.20	Getting Started with a Custom Configuration	400
17.	Housekeeping and Administrative Tasks	411
17.1	Renaming Many Files	411
17.2	Using GNU Texinfo and Info on Linux	413
17.3	Unzipping Many ZIP Files	414
17.4	Recovering Disconnected Sessions Using screen	415
17.5	Sharing a Single bash Session	417
17.6	Logging an Entire Session or Batch Job	418
17.7	Clearing the Screen When You Log Out	420
17.8	Capturing File Metadata for Recovery	421
17.9	Creating an Index of Many Files	422
17.10	Using diff and patch	422
17.11	Counting Differences in Files	426
17.12	Removing or Renaming Files Named with Special Characters	428
17.13	Prepending Data to a File	429
17.14	Editing a File in Place	432
17.15	Using sudo on a Group of Commands	434
17.16	Finding Lines in One File But Not in the Other	436
17.17	Keeping the Most Recent N Objects	439
17.18	Grepping ps Output Without Also Getting the grep Process Itself	442
17.19	Finding Out Whether a Process Is Running	443
17.20	Adding a Prefix or Suffix to Output	444
17.21	Numbering Lines	446
17.22	Writing Sequences	448
17.23	Emulating the DOS Pause Command	450
17.24	Commifying Numbers	450
18.	Working Faster by Typing Less	453
18.1	Moving Quickly Among Arbitrary Directories	453
18.2	Repeating the Last Command	455
18.3	Running Almost the Same Command	456

18.4	Substituting Across Word Boundaries	457
18.5	Reusing Arguments	458
18.6	Finishing Names for You	459
18.7	Playing It Safe	460
19.	Tips and Traps: Common Goofs for Novices	462
19.1	Forgetting to Set Execute Permissions	462
19.2	Fixing “No such file or directory” Errors	463
19.3	Forgetting That the Current Directory Is Not in the \$PATH	465
19.4	Naming Your Script Test	466
19.5	Expecting to Change Exported Variables	467
19.6	Forgetting Quotes Leads to “command not found” on Assignments	468
19.7	Forgetting That Pattern Matching Alphabetizes	470
19.8	Forgetting That Pipelines Make Subshells	470
19.9	Making Your Terminal Sane Again	473
19.10	Deleting Files Using an Empty Variable	474
19.11	Seeing Odd Behavior from printf	474
19.12	Testing bash Script Syntax	476
19.13	Debugging Scripts	477
19.14	Avoiding “command not found” When Using Functions	479
19.15	Confusing Shell Wildcards and Regular Expressions	480
A.	Reference Lists	482
	bash Invocation	482
	Prompt String Customizations	483
	ANSI Color Escape Sequences	484
	Built-in Commands and Reserved Words	485
	Built-in Shell Variables	487
	set Options	491
	shopt Options	492
	Adjusting Shell Behavior Using set, shopt, and Environment Variables	494
	Test Operators	505
	I/O Redirection	506
	echo Options and Escape Sequences	508
	printf	509
	Date and Time String Formatting with strftime	513
	Pattern-Matching Characters	514
	extglob Extended Pattern-Matching Operators	515
	tr Escape Sequences	515

Readline Init File Syntax	516
emacs Mode Commands	518
vi Control Mode Commands	520
Table of ASCII Values	522
B. Examples Included with bash	524
Startup-Files Directory Examples	524
C. Command-Line Processing	532
Command-Line Processing Steps	532
D. Revision Control	538
CVS	539
Subversion	545
RCS	550
Other	557
E. Building bash from Source	559
Obtaining bash	559
Unpacking the Archive	559
What's in the Archive	560
Who Do I Turn To?	564
Index	567