Advanced Bash-Scripting Guide

An in-depth exploration of the art of shell scripting

Mendel Cooper

<thegrendel.abs@gmail.com>

10

10 Mar 2014

Revision History		
Revision 6.5 'TUNGSTENBERRY' release	05 Apr 2012	Revised by: mc
Revision 6.6 'YTTERBIUMBERRY' release	27 Nov 2012	Revised by: mc
Revision 10 'PUBLICDOMAIN' release	10 Mar 2014	Revised by: mc

This tutorial assumes no previous knowledge of scripting or programming, yet progresses rapidly toward an intermediate/advanced level of instruction . . . *all the while sneaking in little nuggets of UNIX® wisdom and lore*. It serves as a textbook, a manual for self-study, and as a reference and source of knowledge on shell scripting techniques. The exercises and heavily-commented examples invite active reader participation, under the premise that **`the only way to really learn scripting is to write scripts`**.

This book is suitable for classroom use as a general introduction to programming concepts.

This document is herewith granted to the Public Domain. **`No copyright!`**

Dedication

For Anita, the source of all the magic

Table of Contents

Part 1. Introduction
- 1. Shell Programming!
- 2. Starting Off With a Sha-Bang

Part 2. Basics
- 3. Special Characters
- 4. Introduction to Variables and Parameters
- 5. Quoting
- 6. Exit and Exit Status
- 7. Tests
- 8. Operations and Related Topics

Part 3. Beyond the Basics
- 9. Another Look at Variables

10. Manipulating Variables
11. Loops and Branches
12. Command Substitution
13. Arithmetic Expansion
14. Recess Time
Part 4. Commands
15. Internal Commands and Builtins
16. External Filters, Programs and Commands
17. System and Administrative Commands
Part 5. Advanced Topics
18. Regular Expressions
19. Here Documents
20. I/O Redirection
21. Subshells
22. Restricted Shells
23. Process Substitution
24. Functions
25. Aliases
26. List Constructs
27. Arrays
28. Indirect References
29. `/dev` and `/proc`
30. Network Programming
31. Of Zeros and Nulls
32. Debugging
33. Options
34. Gotchas
35. Scripting With Style
36. Miscellany
37. Bash, versions 2, 3, and 4
38. Endnotes
38.1. Author's Note
38.2. About the Author
38.3. Where to Go For Help
38.4. Tools Used to Produce This Book
38.5. Credits
38.6. Disclaimer
Bibliography
A. Contributed Scripts
B. Reference Cards
C. A Sed and Awk Micro-Primer
C.1. Sed
C.2. Awk
D. Parsing and Managing Pathnames
E. Exit Codes With Special Meanings
F. A Detailed Introduction to I/O and I/O Redirection
G. Command-Line Options
G.1. Standard Command-Line Options
G.2. Bash Command-Line Options
H. Important Files
I. Important System Directories
J. An Introduction to Programmable Completion
K. Localization

L. History Commands
M. Sample `.bashrc` and `.bash_profile` Files
N. Converting DOS Batch Files to Shell Scripts
O. Exercises
    O.1. Analyzing Scripts
    O.2. Writing Scripts
P. Revision History
Q. Download and Mirror Sites
R. To Do List
S. Copyright
T. ASCII Table
Index

List of Tables

8-1. Operator Precedence
15-1. Job identifiers
33-1. Bash options
36-1. Numbers representing colors in Escape Sequences
B-1. Special Shell Variables
B-2. TEST Operators: Binary Comparison
B-3. TEST Operators: Files
B-4. Parameter Substitution and Expansion
B-5. String Operations
B-6. Miscellaneous Constructs
C-1. Basic sed operators
C-2. Examples of sed operators
E-1. *Reserved* Exit Codes
N-1. Batch file keywords / variables / operators, and their shell equivalents
N-2. DOS commands and their UNIX equivalents
P-1. Revision History

List of Examples

2-1. *cleanup*: A script to clean up log files in /var/log
2-2. *cleanup*: An improved clean-up script
2-3. *cleanup*: An enhanced and generalized version of above scripts.
3-1. Code blocks and I/O redirection
3-2. Saving the output of a code block to a file
3-3. Running a loop in the background
3-4. Backup of all files changed in last day
4-1. Variable assignment and substitution
4-2. Plain Variable Assignment
4-3. Variable Assignment, plain and fancy
4-4. Integer or string?
4-5. Positional Parameters
4-6. *wh*, *whois* domain name lookup
4-7. Using *shift*
5-1. Echoing Weird Variables
5-2. Escaped Characters
5-3. Detecting key-presses
6-1. exit / exit status
6-2. Negating a condition using !
7-1. What is truth?
7-2. Equivalence of *test*, `/usr/bin/test`, [ ], and `/usr/bin/[`
7-3. Arithmetic Tests using (( ))

7-4. Testing for broken links
7-5. Arithmetic and string comparisons
7-6. Testing whether a string is *null*
7-7. *zmore*
8-1. Greatest common divisor
8-2. Using Arithmetic Operations
8-3. Compound Condition Tests Using && and ||
8-4. Representation of numerical constants
8-5. C-style manipulation of variables
9-1. $IFS and whitespace
9-2. Timed Input
9-3. Once more, timed input
9-4. Timed *read*
9-5. Am I root?
9-6. *arglist*: Listing arguments with $* and $@
9-7. Inconsistent `$*` and `$@` behavior
9-8. `$*` and `$@` when `$IFS` is empty
9-9. Underscore variable
9-10. Using *declare* to type variables
9-11. Generating random numbers
9-12. Picking a random card from a deck
9-13. Brownian Motion Simulation
9-14. Random between values
9-15. Rolling a single die with RANDOM
9-16. Reseeding RANDOM
9-17. Pseudorandom numbers, using awk
10-1. Inserting a blank line between paragraphs in a text file
10-2. Generating an 8-character "random" string
10-3. Converting graphic file formats, with filename change
10-4. Converting streaming audio files to *ogg*
10-5. Emulating *getopt*
10-6. Alternate ways of extracting and locating substrings
10-7. Using parameter substitution and error messages
10-8. Parameter substitution and "usage" messages
10-9. Length of a variable
10-10. Pattern matching in parameter substitution
10-11. Renaming file extensions:
10-12. Using pattern matching to parse arbitrary strings
10-13. Matching patterns at prefix or suffix of string
11-1. Simple *for* loops
11-2. *for* loop with two parameters in each [list] element
11-3. *Fileinfo:* operating on a file list contained in a variable
11-4. Operating on a parameterized file list
11-5. Operating on files with a *for* loop
11-6. Missing **`in [list]`** in a *for* loop
11-7. Generating the **`[list]`** in a *for* loop with command substitution
11-8. A *grep* replacement for binary files
11-9. Listing all users on the system

11-10. Checking all the binaries in a directory for authorship
11-11. Listing the *symbolic links* in a directory
11-12. Symbolic links in a directory, saved to a file
11-13. A C-style *for* loop
11-14. Using *efax* in batch mode
11-15. Simple *while* loop
11-16. Another *while* loop
11-17. *while* loop with multiple conditions
11-18. C-style syntax in a *while* loop
11-19. *until* loop
11-20. Nested Loop
11-21. Effects of *break* and **continue** in a loop
11-22. Breaking out of multiple loop levels
11-23. Continuing at a higher loop level
11-24. Using *continue N* in an actual task
11-25. Using *case*
11-26. Creating menus using *case*
11-27. Using *command substitution* to generate the *case* variable
11-28. Simple string matching
11-29. Checking for alphabetic input
11-30. Creating menus using *select*
11-31. Creating menus using *select* in a function
12-1. Stupid script tricks
12-2. Generating a variable from a loop
12-3. Finding anagrams
15-1. A script that spawns multiple instances of itself
15-2. *printf* in action
15-3. Variable assignment, using *read*
15-4. What happens when *read* has no variable
15-5. Multi-line input to *read*
15-6. Detecting the arrow keys
15-7. Using *read* with file redirection
15-8. Problems reading from a pipe
15-9. Changing the current working directory
15-10. Letting *let* do arithmetic.
15-11. Showing the effect of *eval*
15-12. Using *eval* to select among variables
15-13. *Echoing* the *command-line parameters*
15-14. Forcing a log-off
15-15. A version of *rot13*
15-16. Using *set* with positional parameters
15-17. Reversing the positional parameters
15-18. Reassigning the positional parameters
15-19. "Unsetting" a variable
15-20. Using *export* to pass a variable to an embedded *awk* script
15-21. Using *getopts* to read the options/arguments passed to a script
15-22. "Including" a data file
15-23. A (useless) script that sources itself
15-24. Effects of *exec*

15-25. A script that *exec's* itself
15-26. Waiting for a process to finish before proceeding
15-27. A script that kills itself
16-1. Using *ls* to create a table of contents for burning a CDR disk
16-2. Hello or Good-bye
16-3. *Badname*, eliminate file names in current directory containing bad characters and whitespace.
16-4. Deleting a file by its *inode* number
16-5. Logfile: Using *xargs* to monitor system log
16-6. Copying files in current directory to another
16-7. Killing processes by name
16-8. Word frequency analysis using *xargs*
16-9. Using *expr*
16-10. Using *date*
16-11. *Date* calculations
16-12. Word Frequency Analysis
16-13. Which files are scripts?
16-14. Generating 10-digit random numbers
16-15. Using *tail* to monitor the system log
16-16. Printing out the *From* lines in stored e-mail messages
16-17. Emulating *grep* in a script
16-18. Crossword puzzle solver
16-19. Looking up definitions in Webster's 1913 Dictionary
16-20. Checking words in a list for validity
16-21. *toupper*: Transforms a file to all uppercase.
16-22. *lowercase*: Changes all filenames in working directory to lowercase.
16-23. *du*: DOS to UNIX text file conversion.
16-24. *rot13*: ultra-weak encryption.
16-25. Generating "Crypto-Quote" Puzzles
16-26. Formatted file listing.
16-27. Using *column* to format a directory listing
16-28. *nl*: A self-numbering script.
16-29. *manview*: Viewing formatted manpages
16-30. Using *cpio* to move a directory tree
16-31. Unpacking an *rpm* archive
16-32. Stripping comments from C program files
16-33. Exploring `/usr/X11R6/bin`
16-34. An "improved" *strings* command
16-35. Using *cmp* to compare two files within a script.
16-36. *basename* and *dirname*
16-37. A script that copies itself in sections
16-38. Checking file integrity
16-39. Uudecoding encoded files
16-40. Finding out where to report a spammer
16-41. Analyzing a spam domain
16-42. Getting a stock quote
16-43. Updating FC4
16-44. Using *ssh*
16-45. A script that mails itself
16-46. Generating prime numbers
16-47. Monthly Payment on a Mortgage
16-48. Base Conversion
16-49. Invoking *bc* using a *here document*
16-50. Calculating PI
16-51. Converting a decimal number to hexadecimal
16-52. Factoring

16-53. Calculating the hypotenuse of a triangle
16-54. Using *seq* to generate loop arguments
16-55. Letter Count"
16-56. Using *getopt* to parse command-line options
16-57. A script that copies itself
16-58. Exercising *dd*
16-59. Capturing Keystrokes
16-60. Preparing a bootable SD card for the *Raspberry Pi*
16-61. Securely deleting a file
16-62. Filename generator
16-63. Converting meters to miles
16-64. Using *m4*
17-1. Setting a new password
17-2. Setting an *erase* character
17-3. *secret password*: Turning off terminal echoing
17-4. Keypress detection
17-5. Checking a remote server for *identd*
17-6. *pidof* helps kill a process
17-7. Checking a CD image
17-8. Creating a filesystem in a file
17-9. Adding a new hard drive
17-10. Using *umask* to hide an output file from prying eyes
17-11. *Backlight*: changes the brightness of the (laptop) screen backlight
17-12. *killall*, from `/etc/rc.d/init.d`
19-1. *broadcast*: Sends message to everyone logged in
19-2. *dummyfile*: Creates a 2-line dummy file
19-3. Multi-line message using *cat*
19-4. Multi-line message, with tabs suppressed
19-5. Here document with replaceable parameters
19-6. Upload a file pair to *Sunsite* incoming directory
19-7. Parameter substitution turned off
19-8. A script that generates another script
19-9. Here documents and functions
19-10. "Anonymous" Here Document
19-11. Commenting out a block of code
19-12. A self-documenting script
19-13. Prepending a line to a file
19-14. Parsing a mailbox
20-1. Redirecting `stdin` using *exec*
20-2. Redirecting `stdout` using *exec*
20-3. Redirecting both `stdin` and `stdout` in the same script with *exec*
20-4. Avoiding a subshell
20-5. Redirected *while* loop
20-6. Alternate form of redirected *while* loop
20-7. Redirected *until* loop
20-8. Redirected *for* loop
20-9. Redirected *for* loop (both `stdin` and `stdout` redirected)
20-10. Redirected *if/then* test
20-11. Data file *names.data* for above examples
20-12. Logging events
21-1. Variable scope in a subshell
21-2. List User Profiles

21-3. Running parallel processes in subshells
22-1. Running a script in restricted mode
23-1. Code block redirection without forking
23-2. Redirecting the output of *process substitution* into a loop.
24-1. Simple functions
24-2. Function Taking Parameters
24-3. Functions and command-line args passed to the script
24-4. Passing an indirect reference to a function
24-5. Dereferencing a parameter passed to a function
24-6. Again, dereferencing a parameter passed to a function
24-7. Maximum of two numbers
24-8. Converting numbers to Roman numerals
24-9. Testing large return values in a function
24-10. Comparing two large integers
24-11. Real name from username
24-12. Local variable visibility
24-13. Demonstration of a simple recursive function
24-14. Another simple demonstration
24-15. Recursion, using a local variable
24-16. *The Fibonacci Sequence*
24-17. *The Towers of Hanoi*
25-1. Aliases within a script
25-2. *unalias*: Setting and unsetting an alias
26-1. Using an *and list* to test for command-line arguments
26-2. Another command-line arg test using an *and list*
26-3. Using *or lists* in combination with an *and list*
27-1. Simple array usage
27-2. Formatting a poem
27-3. Various array operations
27-4. String operations on arrays
27-5. Loading the contents of a script into an array
27-6. Some special properties of arrays
27-7. Of empty arrays and empty elements
27-8. Initializing arrays
27-9. Copying and concatenating arrays
27-10. More on concatenating arrays
27-11. The Bubble Sort
27-12. Embedded arrays and indirect references
27-13. The Sieve of Eratosthenes
27-14. The Sieve of Eratosthenes, Optimized
27-15. Emulating a push-down stack
27-16. Complex array application: *Exploring a weird mathematical series*
27-17. Simulating a two-dimensional array, then tilting it
28-1. Indirect Variable References
28-2. Passing an indirect reference to *awk*
29-1. Using `/dev/tcp` for troubleshooting
29-2. Playing music
29-3. Finding the process associated with a PID
29-4. On-line connect status
30-1. Print the server environment
30-2. IP addresses
31-1. Hiding the cookie jar
31-2. Setting up a swapfile using `/dev/zero`
31-3. Creating a ramdisk

32-1. A buggy script
32-2. Missing keyword
32-3. *test24*: another buggy script
32-4. Testing a condition with an *assert*
32-5. Trapping at exit
32-6. Cleaning up after **Control-C**
32-7. A Simple Implementation of a Progress Bar
32-8. Tracing a variable
32-9. Running multiple processes (on an SMP box)
34-1. Numerical and string comparison are not equivalent
34-2. Subshell Pitfalls
34-3. Piping the output of *echo* to a *read*
36-1. *shell wrapper*
36-2. A slightly more complex *shell wrapper*
36-3. A generic *shell wrapper* that writes to a logfile
36-4. A *shell wrapper* around an awk script
36-5. A *shell wrapper* around another awk script
36-6. Perl embedded in a *Bash* script
36-7. Bash and Perl scripts combined
36-8. Python embedded in a *Bash* script
36-9. A script that speaks
36-10. A (useless) script that recursively calls itself
36-11. A (useful) script that recursively calls itself
36-12. Another (useful) script that recursively calls itself
36-13. A "colorized" address database
36-14. Drawing a box
36-15. Echoing colored text
36-16. A "horserace" game
36-17. A Progress Bar
36-18. Return value trickery
36-19. Even more return value trickery
36-20. Passing and returning arrays
36-21. Fun with anagrams
36-22. Widgets invoked from a shell script
36-23. Test Suite
37-1. String expansion
37-2. Indirect variable references - the new way
37-3. Simple database application, using indirect variable referencing
37-4. Using arrays and other miscellaneous trickery to deal four random hands from a deck of cards
37-5. A simple address database
37-6. A somewhat more elaborate address database
37-7. Testing characters
37-8. Reading N characters
37-9. Using a *here document* to set a variable
37-10. Piping input to a read
37-11. Negative array indices
37-12. Negative parameter in string-extraction construct
A-1. *mailformat*: Formatting an e-mail message
A-2. *rn*: A simple-minded file renaming utility
A-3. *blank-rename*: Renames filenames containing blanks
A-4. *encryptedpw*: Uploading to an ftp site, using a locally encrypted password
A-5. *copy-cd*: Copying a data CD
A-6. Collatz series

A-7. *days-between*: Days between two dates
A-8. Making a *dictionary*
A-9. Soundex conversion
A-10. *Game of Life*
A-11. Data file for *Game of Life*
A-12. *behead*: Removing mail and news message headers
A-13. *password*: Generating random 8-character passwords
A-14. *fifo*: Making daily backups, using named pipes
A-15. Generating prime numbers using the modulo operator
A-16. *tree*: Displaying a directory tree
A-17. *tree2*: Alternate directory tree script
A-18. *string functions*: C-style string functions
A-19. Directory information
A-20. Library of hash functions
A-21. Colorizing text using hash functions
A-22. More on hash functions
A-23. Mounting USB keychain storage devices
A-24. Converting to HTML
A-25. Preserving weblogs
A-26. Protecting literal strings
A-27. Unprotecting literal strings
A-28. Spammer Identification
A-29. Spammer Hunt
A-30. Making *wget* easier to use
A-31. A *podcasting* script
A-32. Nightly backup to a firewire HD
A-33. An expanded *cd* command
A-34. A soundcard setup script
A-35. Locating split paragraphs in a text file
A-36. Insertion sort
A-37. Standard Deviation
A-38. A *pad* file generator for shareware authors
A-39. A *man page* editor
A-40. Petals Around the Rose
A-41. Quacky: a Perquackey-type word game
A-42. Nim
A-43. A command-line stopwatch
A-44. An all-purpose shell scripting homework assignment solution
A-45. The Knight's Tour
A-46. Magic Squares
A-47. Fifteen Puzzle
A-48. *The Towers of Hanoi, graphic version*
A-49. *The Towers of Hanoi, alternate graphic version*
A-50. An alternate version of the getopt-simple.sh script
A-51. The version of the *UseGetOpt.sh* example used in the Tab Expansion appendix
A-52. Cycling through all the possible color backgrounds
A-53. Morse Code Practice
A-54. Base64 encoding/decoding
A-55. Inserting text in a file using *sed*
A-56. The Gronsfeld Cipher
A-57. Bingo Number Generator
A-58. Basics Reviewed
A-59. Testing execution times of various commands
A-60. Associative arrays vs. conventional arrays (execution times)
C-1. Counting Letter Occurrences
J-1. Completion script for *UseGetOpt.sh*
M-1. Sample `.bashrc` file
M-2. `.bash_profile` file
N-1. VIEWDATA.BAT: DOS Batch File
N-2. *viewdata.sh*: Shell Script Conversion of VIEWDATA.BAT
T-1. A script that generates an ASCII table
T-2. Another ASCII table script
T-3. A third ASCII table script, using *awk*
