Throw Out Shell Scripts Documentation

Nick Timkovich

Oct 15, 2019

Contents:

CHAPTER 1

Top 100 Bash Questions on Stack Overflow

Searches

- Just Bash
- Complicated

1.1 1-10

1. +4324 Get the source directory of a Bash script from within the script itself

   `__file__` is the file itself, so to get the parent:

   ```
   # fully resolved
   pathlib.Path(__file__).parent
   os.path.abspath(os.path.join(__file__, os.path.pardir))

   # may still be relative
   pathlib.Path(__file__).parent.resolve()
   os.path.normpath(os.path.join(__file__, os.path.pardir))
   ```

2. +2875 How do I tell if a regular file does not exist in Bash?

   Use `os.path.exists()` or `pathlib.Path.exists()`, or just `try` it as it's EAFP than to LBYL.

3. +2390 How to concatenate string variables in Bash

   Yes, this is not obvious in Bash. `string1 + string2`.

4. +2027 How to check if a string contains a substring in Bash

   Also tricky…in Bash. Trivial in Python: `substring in string`.

5. +1938 In the shell, what does " 2>&1 " mean?

The syntax generally isn't as obscure, and to do something like switching stdout and stderr, you don't need to master crazy heiroglyphics like `3>&1- 1>&2- 2>&3-`.

6. +1902 **'Echo newline in Bash prints literal \\n'_**

   Escaping feels inconsistent in Bash, versus in Python where there is a clear distinction between `repr()`sentations of strings and presentation of them via `print()`.

7. +1819 How to check if a program exists from a Bash script?

   Try it, or `shutil.which()`.

8. +1801 Extract filename and extension in Bash

   `os.path.splitext()`, `pathlib.PurePath.name`, `pathlib.PurePath.suffix`

9. +1692 How do I split a string on a delimiter in Bash?

   `str.split()`

10. +1561 How do I parse command line arguments in Bash?

    `argparse` or DIY from `sys.argv`.

1.2 11-20

11. +1414 How to count all the lines of code in a directory recursively?
12. +1370 How to change the output color of echo in Linux

    Here it's more about interacting with the terminal emulator, so you basically can do the same thing as Bash does. However, the escapes can be tricky a la the "how do I echo a newline" question above, while in Python it's much more clear. At the most basic:

    ```
    print("\x1B[91mTHIS IS RED\x1B[0m")
    ```

    There are also cross-platform terminal color libraries like `colorama` which can make life easier too.

13. +1364 How do I reload .bashrc without logging out and back in?
14. +1336 How can I redirect and append both stdout and stderr to a file with Bash?

    When running a subprocess, you can combine the streams if you specify `stderr=subprocess.STDOUT`.

15. **+1295 How to set a variable to the output of a command in Bash?** `[command-line] [shell]`

    Yes, this is *also* non-obvious in Bash.

16. +1245 How do I prompt for Yes/No/Cancel input in a Linux shell script?

    Read via `input()` and do whatever you want.

17. +1242 How to check if a variable is set in Bash?

    Bash doesn't

18. +1235 How do I iterate over a range of numbers defined by variables in Bash?

    `for n in range(x, y, z):`

19. +1202 Loop through an array of strings in Bash?

    `for s in strings:`

20. +1097 Looping through the content of a file in Bash

    `for line in open_file:`

1.3 21-30

21. +1061 Check existence of input argument in a Bash shell script

    argparse.

22. +1048 Difference between sh and bash

    Vaguely similar to Python 2/3; sometimes you can pretend they're the same, but you need to know the difference when it matters.

23. +1030 How to convert a string to lower case in Bash?

    `str.lower()`

24. +1020 Make a Bash alias that takes a parameter?

    A Bash alias is a canned command, you'd need to do this with a function.

25. +937 YYYY-MM-DD format date in shell script

    The `strftime` specifications are the same as they're from C, and in Python you use them with `datetime.datetime.strftime()`.

26. +920 What is the preferred Bash shebang?

    A good question for Python as well, there's `#!/usr/bin/env python`, or ... ‘‘python3‘‘...

27. +902 echo that outputs to stderr

```
import sys

print(..., file=sys.stderr)
```

28. +879 How to count lines in a document?

    There are a few methods involving iterating over an open file and incrementing, and you can look at other Stack Overflow questions for more discussion. Notably, if this *is* a bottleneck, you can easily call out to `wc`.

29. +851 How to pipe stderr, and not stdout?

30. **+845 How to specify the private SSH-key to use when executing shell command on Git?** `[git] [shell] [ssh]`

1.4 31-40

31. +831 How to escape single quotes within single quoted strings? `[quoting] [syntax]`
32. +819 How to reload .bash_profile from the command line? `[.bash-profile] [shell]`
33. +818 Setting environment variables on OS X `[environment-variables] [macos] [path]`
34. +816 **‘Listing only directories using ls in bash: An examination‘_** `[directory] [ls]`
35. +809 Defining a variable with or without export `[linux] [shell]`
36. +805 How to compare strings in Bash `[string]`

37. +800 Passing parameters to a Bash function
38. +781 How to redirect output to a file and stdout `[file-io] [io] [linux] [stdout]`
39. +776 Pipe to/from the clipboard in Bash script `[clipboard] [linux] [macos]`
40. +760 How to iterate over arguments in a Bash script `[command-line]`

1.5 41-50

41. +759 How to trim whitespace from a Bash variable?

    `str.strip()`

42. +755 How to declare and use boolean variables in shell script?

    This is non-obvious because there isn't a boolean type in shell. There are also myriad ways to get it wrong that "look" nice if you view the question.

43. +749 How can I exclude all "permission denied" messages from "find"? `[error-handling] [file-permissions] [find]`
44. +724 How to permanently set $PATH on Linux/Unix? `[linux] [path] [unix] [zsh]`
45. +705 What are the special dollar sign shell variables?

    Python has a cleaner return semantics from calls, so it doesn't need magic variables like `$?`. That said, Python has _ which can be useful in *interactive* contexts.

46. +698 Get current directory name (without full path) in a Bash script `[shell]`
47. +682 Given two directory trees, how can I find out which files differ? `[diff] [linux] [shell] [unix]`
48. +681 Parsing JSON with Unix tools

    `json`

49. +672 Redirect all output to file `[io-redirection] [linux]`
50. +665 Propagate all arguments in a bash shell script `[command-line-arguments]`

1.6 51-60

51. +646 Add a new element to an array without specifying the index in Bash `[arrays]`
52. +640 How do I clear/delete the current line in terminal? `[terminal]`
53. +638 Count number of lines in a git repository `[git] [line-count] [shell]`
54. +633 Read a file line by line assigning the value to a variable
55. +630 How to 'grep' a continuous stream? `[grep] [linux] [shell] [tail]`
56. +628 How can I write a heredoc to a file in Bash script? `[heredoc]`
57. +620 How do I remove all .pyc files from a project?
58. +610 Add line break to 'git commit -m' from the command line `[git] [shell]`
59. +604 How to use double or single brackets, parentheses, curly braces `[syntax]`
60. +600 Redirect stderr and stdout in Bash `[pipe] [redirect] [shell]`
61. +595 In bash, how can I check if a string begins with some value? `[comparison] [string]`

62. +592 Extract substring in Bash [shell] [string] [substring]
63. +588 How to do a recursive find/replace of a string with awk or sed? [awk] [replace] [sed]
64. +584 Check number of arguments passed to a Bash script [parameter-passing]
65. +579 Replace one substring for another string in shell script [shell]
66. +579 Reliable way for a Bash script to get the full path to itself [path]
67. +577 How to call shell script from another shell script? [shell]
68. +559 Check if pull needed in Git [git] [shell]
69. +551 In a Bash script, how can I exit the entire script if a certain condition occurs? [exit] [exit-code] [scripting]
70. +545 Find and kill a process in one line using bash and regex [awk] [regex] [terminal]
71. +542 How do I pause my shell script for a second before continuing? [shell] [terminal] [unix]
72. +538 Syntax for a single-line Bash infinite while loop [loops] [while-loop]
73. +537 How to determine the current shell I'm working on? [csh] [shell] [tcsh] [unix]
74. +536 When do we need curly braces around shell variables? [curly-braces] [shell] [syntax]
75. +536 Why is whitespace sometimes needed around metacharacters? [shell] [syntax] [syntax-error]
76. +522 How do I compare two string variables in an 'if' statement in Bash? [if-statement] [scripting]
77. +519 How to do a logical OR operation in Shell Scripting [if-statement] [sh] [unix]
78. +517 What does set -e mean in a bash script? [linux] [sh] [shell]
79. +514 sudo echo "something" >> /etc/privilegedFile doesn't work [permissions] [scripting] [shell] [sudo]
80. +513 Assigning default values to shell variables with a single command in bash [shell]
81. +511 Automatic exit from bash shell script on error [exit] [shell]
82. +507 Split string into an array in Bash [arrays] [split]
83. +503 How do I know the script file name in a Bash script? [linux] [scripting] [shell]
84. +503 How do I prompt a user for confirmation in bash script?
85. +499 Get current time in seconds since the Epoch on Linux, Bash [datetime] [linux]
86. +497 Capturing multiple line output into a Bash variable [variables]
87. +495 How do I test if a variable is a number in Bash? [linux] [shell]
88. +490 How to kill all processes with a given partial name? [linux] [posix]
89. +485 How to run a shell script on a Unix console or Mac terminal? [linux] [macos] [shell] [unix]
90. +482 How does "cat << EOF" work in bash? [heredoc] [linux] [scripting]
91. +480 How do I get cURL to not show the progress bar? [curl] [linux] [scripting] [unix]
92. +470 How to pass all arguments passed to my bash script to a function of mine? [function] [parameter-passing]
93. +468 Bash tool to get nth line from a file [awk] [sed] [shell] [unix]
94. +464 How to wait in bash for several subprocesses to finish and return exit code !=0 when any subprocess ends with code !=0? [process] [wait]

95. +464 What's a concise way to check that environment variables are set in a Unix shell script? `[shell]` `[unix]`
96. +463 How do I write stderr to a file while using "tee" with a pipe? `[linux]` `[unix]`
97. +460 How can I add numbers in a bash script `[mathematical-expressions]`
98. +459 How can I remove the first line of a text file using bash/sed script? `[scripting]` `[sed]`
99. +457 Find and Replace Inside a Text File from a Bash Command `[ironpython]` `[replace]` `[scripting]` `[ssh]`
100. +455 How to define hash tables in Bash?

    `[associative-array]` `[dictionary]` `[hashtable]`

CHAPTER 2

Packages

You can accomplish a lot with standard Python, and in the context where I'm advocating for its use here, i.e. instead of Bash, I'm largely trying to illustrate how much you can do without any dependencies when it's just there to be used on many distributions (Ubuntu, CentOS. . . sorta).

However, there are packages that can supercharge the powers so you can do much, much more. Here are a few I've used in the past.

2.1 Argument Parsing: Click

2.2 Argument Parsing: docopt

2.3 Remote Execution: Fabric3

Python 3 (and 2) compatible fork of Fabric.

2.4 Remote Execution: Fabric 2.x

Large rewrite of Fabric (1.x)

2.5 Jobs: Invoke

2.6 Flair: Colorama

CHAPTER 3

The Competitors

3.1 Bash: The Language

One base type: character strings. Arrays available, but as Unix commands are all about chaining them together with pipes, newline- or null-delimited "arrays" are more common as those can be interpreted by other commands.

Shell strengths: process forking, stream management Shell weaknesses: types, logic, syntax

If you need to pipe…

Where needed, you can stitch together multiple processes…

3.2 PowerShell

Has types, interactive help, all-in-all pretty nice. Only available on Windows.

CHAPTER 4

Primary Uses

- **Wholesale replacement of scripts. The scripts may be direct shell scripts or Makefiles. Common script usage:**
  - Tie together multiple executables
  - Perform a bunch of file/path manipulations
- **Wrapping a confusing/awful interface to another program**
  - `os.exec*` functions if you want to just do something at the start
  - The program may also be ordinarily fine, but say you wanted to run it several times with different parameters, or save a set of parameters as a default.

The standard library packages in Python that can replicate much of the basic functionality that shell scripts generally provide are the `os`, `pathlib`, `shutil`, and `subprocess` packages.

CHAPTER 5

Indices and tables

- genindex
- modindex
- search