

~/bin/ there, done that!



A gentle guide to working with Bash

By: Jason Franklin

@lifecrisis on GitHub
@lifecrisis on GitLab



Roadmap

1. Introducing myself and this talk
2. Tools
3. An example
4. Learning materials and resources
5. Closing comments



Part I: Introductions

Who am I?

- BS in math from UGA
- 1 year of Apache Spark and Python
- 2.5 years of LAMP among other things
- <https://github.com/scrooloose/nerdtree>
- 21+ patches to the Vim core
- I love bash



This talk IS NOT ...

- ... about POSIX
- ... a how-to lesson
- ... a list of “bash tricks to blow your mind!”

This talk IS ...



Slides at <https://bit.ly/31pTFmB>

You have a problem.



Slides at <https://bit.ly/31pTFmB>



Part II: Tools



Tools Overview

- Git (never forget!)
- Make
- Snippets
 - <https://github.com/SirVer/ultisnips>
- Shellcheck
 - <https://www.shellcheck.net/>
- Your prompt!

Git and Make

- When you learn something cool, never forget
- Automate the installation of your scripts
- Fork this repository to get started:
 - <https://github.com/lifecrisis/scripts>
- The README.md file explains it all!

demo-snippets

demo-shellcheck

<https://raw.githubusercontent.com/git/git/master/contrib/git-jump/git-jump>

... or locate git-jump

demo of prompt

Bonus: demo of Vim!

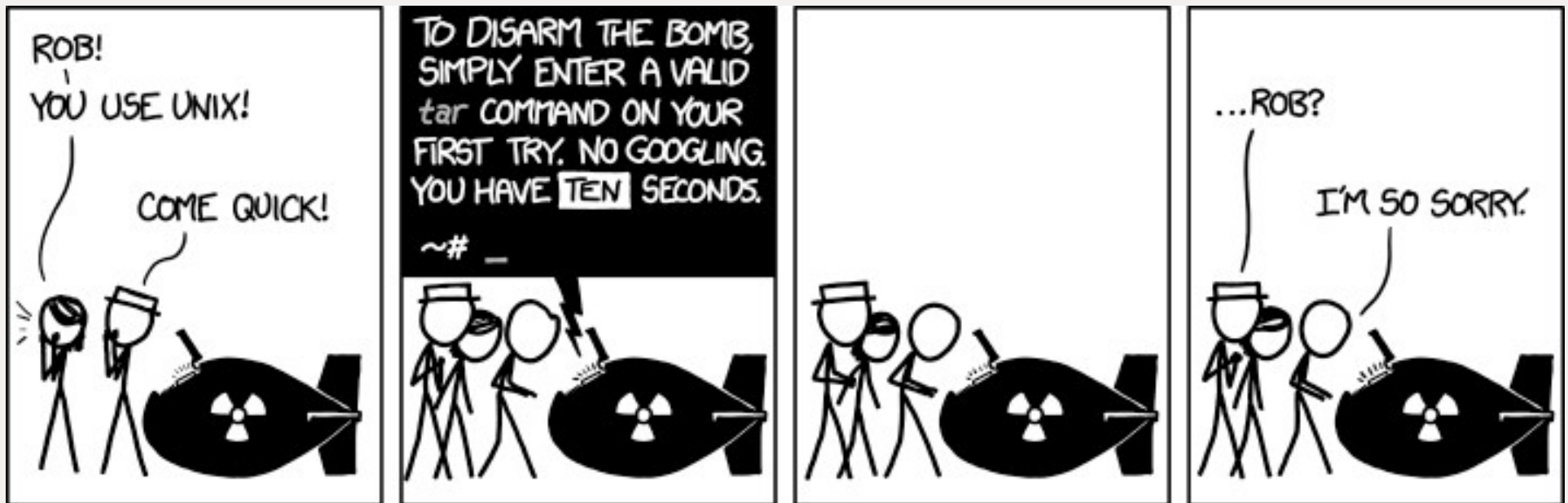


Part III: An Example

Our example: tip

Repository: <https://github.com/lifecrisis/bash-tip>

See also: <https://github.com/cheat/cheat>



demo of “tip”

Initialization

```
10 # The "TIP_DIR" environment variable tells us where tips are kept.
11 if [[ -z "${TIP_DIR+x}" ]]; then
12     echo 'tip: not configured' >&2 && exit 2
13 fi
14
15 # Remove trailing slash.
16 TIP_DIR="${TIP_DIR%/}"
17
18 # Default behavior is to print the tip.
19 EDIT=0
20
```

Usage function

```
21  usage() {  
22      cat <<EOF  
23      Usage: tip [-e] [-h] [--] COMMAND  
24      Print tips on the usage of the given COMMAND.  Use the "-e" option to edit  
25      the named tip file.  
26      EOF  
27  }  
28
```

Option processing

```
29  while getopts ':eh' opt; do
30
31      if [[ "$opt" = '?' ]]; then
32          echo "${0##*/}: invalid option -- '$OPTARG'" >&2 && exit 2
33      fi
34
35      test "$opt" = 'e' && EDIT=1
36      test "$opt" = 'h' && usage && exit
37  done
38  shift $((OPTIND - 1))
39
```

Argument processing

```
40  case $# in
41      0)
42          echo 'tip: argument required' >&2 && exit 2
43          ;;
44      1)
45          file="$TIP_DIR/$1"
46          ;;
47      *)
48          echo 'tip: too many arguments' >&2 && exit 2
49          ;;
50  esac
51
```

Edit tip

```
52  edit_tip() {  
53  
54      if [[ -f "$file" ]]; then  
55  
56          if [[ ! -r "$file" || ! -w "$file" ]]; then  
57              echo 'tip: cannot edit tip file' >&2 && exit 1  
58          fi  
59      fi  
60  
61      ${VISUAL:-vi} "$file"  
62  }  
63
```

Print tip

```
64  print_tip() {
65      local tip_lines
66
67      if [[ ! -f "$file" ]]; then
68          echo 'tip: not found' >&2 && exit 1
69      fi
70
71      if [[ ! -r "$file" ]]; then
72          echo 'tip: cannot read tip file' >&2 && exit 1
73      fi
74
75      tip_lines=$(wc -l <"$file")
76
77      if [[ $tip_lines -ge $(tput lines) ]]; then
78          less -X <"$file"
79          return
80      fi
81
82      cat "$file"
83  }
84
```

Take action!

```
85  if [[ $EDIT -eq 1 ]]; then
86      edit_tip
87      exit
88  fi
89
90  print_tip
91  exit
```




Part IV: Learning Materials

Learning Materials List

- The Unix Programming Environment (ch. 1-5)
 - https://en.wikipedia.org/wiki/The_Unix_Programming_Environment
- BashFAQ
 - <https://mywiki.woledge.org/BashFAQ>
- “man bash” and “help bash”
 - Seriously.



Part V: Closing Comments



Please...

... leave this talk/conference with the resolve to
do something for free software.

Inspiration.