What the shell?
Command-line tips and tricks

Ricky Elrod
Youngstown State University
November 12, 2014

An outline of this talk

- Basic terminology: Shells
- What do many command-line applications have in common?
- Handy keyboard shortcuts
- Hidden Bash features
- Useful command-line applications
- Link to slides!

Shells

- A *shell* is an interface that provides access to the operating system's services.
- Shells can be either command-line or graphical.
  - We will stick to command-line in this talk.
- The goal is to achieve the best possible workflow for intended tasks.

A brief history of Bash

- Stands for “Bourne-again shell”.
- Written to replace another shell (the “Bourne” shell) for various (incl. licensing) reasons.
- *Brian Fox* started coding it in January 1988 as an employee of the Free Software Foundation.
- Now it (or a compatible alternative) ships as the default shell in a lot of Linux distributions.

Why I use it

and why I think you should master it

- Simply put, it's pretty much everywhere.
- There is a lot of tooling out there for it.
- Most (but not all) of what you learn about it applies to other shells too.

A quick look at other shells: Fish

- On-the-fly syntax highlighting of shell code.
- Really cool tab-completion options.
- But lacks support a lot of the tricks I’m going to mention.
- Not installed on most distributions by default.

A quick look at other shells: Zsh

- Also has really helpful tab-completion options.
- Has an insane number of configuration options.
- But isn’t 100% bash compatible.
- Also not installed by default on most distributions.

Back to Bash

- Bash makes use of a library called Readline.
- The next portion of this talk applies to most applications which also use Readline.

What, then, is Readline?

- Readline offers interactive command-line apps easy line-editing and history capabilities.
- It allows you to move the text cursor, traverse command history, and much more.
- It is cross-platform, so applications on various systems can have the same behavior.

Readline advantages

- Readline is used by many interactive command-line applications
- therefore, learning about it for one app makes you more proficient in other apps.
- It handles keyboard input in the same way in most of them.
- The default keyboard shortcut bindings come from the Emacs text editor.
- Other things use these bindings too! Mac OS X's Cocoa Text System.

Why master Readline? What is the point?

- Unified keybindings across all applications that use it.
- This means less to memorize. (But learning them in the first place can be challenging)
- Leave the home-row way less. Stay away from those silly arrow keys!

Enter: Keyboard Hell

Notation

`C` = *Control*

`M` = *Meta* (aka: *Alt*)

Enter: Keyboard Hell

Basic (default) Readline Keybindings

- `C-a`: Jump to start of line
- `C-e`: Jump to end of line
- `C-f`: Move forward one character
- `C-b`: Move back one character
- `C-d`: Delete forward one character
- `C-j`: (Alias for `enter`.)
- `C-p`: Previous command
- `C-n`: Next command
- `M-f`: Move forward one **word**
- `M-b`: Move back one **word**
- `M-d`: Delete forward one **word**

Keyboard Hell (Cont.)

Less-basic (default) Readline Keybindings

- `C-r`: Reverse search through history
- `C-w`: Delete current word until closest space
- `M-backspace`: Delete current word until closest non-alphanumeric character
- `C-u`: Delete (cut) to beginning of line
- `C-k`: Delete (cut) to end of line
- `M-#`: Comment out current line and show a new input line
- `C-t`: Transpose two characters
- `M-t`: Transpose two words.

Keyboard Hell (Cont.)

Less-basic (default) Readline Keybindings

- `M-u`: Uppercase word at which the cursor is at the start
- `M-l`: Lowercase word at which the cursor is at the start
- `M-c`: Capitalize word at which the cursor is at the start
- `C-/`: Undo
- `C-x C-x`: Jump between two points in the current command

Hidden Bash features

The Kill Ring

- Not as scary as it sounds! Does not kill anyone!
- Bash (most Readline-using apps) have a clipboard built in.
- Not your terminal emulator's clipboard. This works at a text console too.
- The following bindings from above (and some others) actually store things into the "kill ring":
  - `C-w`: Delete current word until closest space
  - `C-u`: Delete (cut) to beginning of line
  - `C-k`: Delete (cut) to end of line

Hidden Bash features

The Kill Ring

- You can recall from the kill ring using `C-y` (“yank”).
- Scroll through recall options with `M-y`.

Hidden Bash features

Useful Variables

- You probably know some of these, so they might not be *that* hidden.
- `$!`: Process ID of last command
- `$?`: Exit/return code of last command/function call
- `$_`: Last parameter of last command

Hidden Bash features

Substitutions

- `!:1` - First argument of previous command
- `!:2` - Second argument of previous command (...and so on)
- `!:*` - All arguments of last command (but excludes the command itself)
- `!!` - Last command and all arguments to it (useful for `sudo !!`).
- `!$` - Last parameter of last command
- `^foo^bar` - Replace `foo` with `bar` from previous command and run it
- `!foo` - Run the last command starting with `foo`. (This can be dangerous.)

Hidden Bash features

Misc.

- `C-M-e`: Expand all variables and substitutions on current input line, without running command
- `C-x C-e`: Open the current command in whatever editor `$EDITOR` is set to. Runs when editor closed.

Useful Command-line Applications

In no particular order

- **coreutils**: 102 programs included on pretty much every Linux/BSD box you will ever touch. Learn them and learn them well.
  - Simple things like `ls`, `cat`, `stat`
  - Hashing commands like `sha512sum`, `sha256sum`, `md5sum`, etc
  - String manipulation (`cut`, `wc`, `fmt`, `tr`, `truncate`, `uniq`)
  - System commands (`whoami`, `users`, `touch`, `timeout`, `uname`, `shred`)
  - Much more

Useful Command-line Applications

In no particular order

- `hub`: Command-line wrapper for Git specifically for interacting with GitHub
  - Fork repositories right from the command-line
  - Send pull requests with one command
  - Clone with shorthand notation (`git clone username/repo`)

Useful Command-line Applications

In no particular order

- `htop`: Like `top` (shows running processes, CPU/RAM usage, etc.) but colorful and pretty.
- `shellcheck`: Static analyzer for shell scripts.
- `howdoi`: Searches stackoverflow for code snippets from the commandline
- `autojump`: Jump between directories based on patterns
- `httpie`: An awesome HTTP client for the command-line
- `jq`: Traverse JSON structures from the commandline

These slides

Available on http://elrod.me/talk-archive.html (or http://da.gd/ta) Released under CC-BY.