Supercharge Your Command Line

Christopher W. Pitts

June 7, 2018

Disclaimer

The views in this presentation are entirely my own, and in no way represent or imply any sort of official or unofficial endorsement by Sandia National Laboratories or NTESS.

Follow Along?

https://gitlab.com/cwpitts/presentations

About Me

- From Albuquerque, New Mexico
- Studied computer science at Brigham Young University
- Software Systems Engineer at Sandia National Laboratories
- Ik spreek Hollands (ook Vlaams)!
- Happily married to my dear wife Karoline

Overview

Prerequisites

You don’t have to check off every item on this list, but the more you do, the more you’ll take away.

Prerequisites

You don’t have to check off every item on this list, but the more you do, the more you’ll take away.

- Experience with Linux environment

Prerequisites

You don't have to check off every item on this list, but the more you do, the more you'll take away.

- Experience with Linux environment
- Experience using command line tools

Prerequisites

You don't have to check off every item on this list, but the more you do, the more you'll take away.

- Experience with Linux environment
- Experience using command line tools
- Working knowledge of Bash

Why Bash?

Why Bash?

- Linux is everywhere

Why Bash?

- Linux is everywhere
- Command lines are fast

Why Bash?

- Linux is everywhere
- Command lines are fast
- Bash has huge market share

Hotkeys

Most of these hotkeys will work in any shell, your mileage may vary.

Hotkey	Action
ctrl-a	jump to head of line
ctrl-e	jump to end of line
ctrl-r	search history backwards
ctrl-s	search history forwards
ctrl-k	kill after cursor
ctrl-u	kill before cursor
ctrl-l	clear screen
ctrl-t	swap character at cursor left
meta*-f/ctrl-right	move cursor one word to the right
meta*-b/ctrl-left	move cursor one word to the left

* meta == alt, because these are all *Emacs* key bindings

Commands

Command	Action
!!	repeat last command
$?	exit status of last command
history	show terminal history
!*number*	repeat command *number* from history
export	set an environment variable
alias	set an alias

What is it?

The .bashrc file is a configuration file.

What is it?

The .bashrc file is a configuration file.

- But that's not how you should think of it

What is it?

The .bashrc file is a configuration file.

- But that's not how you should think of it
- It's really just a Bash script that gets run when a non-login (interactive) shell is started

What is it?

- A *login shell* is called when you *login*. Logging into your graphical desktop (GNOME, KDE, Xfce, etc.) is a login shell.
- A *interactive* or *non-login* shell is when you start a terminal program like gnome-terminal, Guake, etc.

What can I do with it?

A Bash shell will read this file at runtime, and evaluate the contents. This allows you to:

What can I do with it?

A Bash shell will read this file at runtime, and evaluate the contents. This allows you to:

- Set environment variables

What can I do with it?

A Bash shell will read this file at runtime, and evaluate the contents. This allows you to:

- Set environment variables
- Set aliases

What can I do with it?

A Bash shell will read this file at runtime, and evaluate the contents. This allows you to:

- Set environment variables
- Set aliases
- Customize your Bash prompt

Where is it?

Bash will look for this file in the home directory, you can look at yours (if you have one), with 'cat ~/.bashrc'.

Environment variables

Environment variables are variables that define things in your environment

Environment variables

Environment variables are variables that define things in your environment

- PATH - where to look for commands

Environment variables

Environment variables are variables that define things in your environment

- PATH - where to look for commands
- USERNAME - your username

Environment variables

Environment variables are variables that define things in your environment

- PATH - where to look for commands
- USERNAME - your username
- HOSTNAME - the hostname of the machine

Environment variables

Environment variables are variables that define things in your environment

- PATH - where to look for commands
- USERNAME - your username
- HOSTNAME - the hostname of the machine
- EDITOR - The program to use to open files for editing

Environment variables

To set an environment variable, use the “export” keyword.

```
export FOO=bar
```

Custom Bash prompt

There are actually four Bash prompts to customize!

Custom Bash prompt

There are actually four Bash prompts to customize!

- PS1: Interactive Bash prompt (what you usually see)

Custom Bash prompt

There are actually four Bash prompts to customize!

- PS1: Interactive Bash prompt (what you usually see)
- PS2: Continuation prompt (line-broken commands or inline function definitions)

Custom Bash prompt

There are actually four Bash prompts to customize!

- PS1: Interactive Bash prompt (what you usually see)
- PS2: Continuation prompt (line-broken commands or inline function definitions)
- PS3: Selection prompt (used with the 'select' command)

Custom Bash prompt

There are actually four Bash prompts to customize!

- PS1: Interactive Bash prompt (what you usually see)
- PS2: Continuation prompt (line-broken commands or inline function definitions)
- PS3: Selection prompt (used with the 'select' command)
- PS4: Trace output prompt (used with set -x)

Custom Bash prompt

There are actually four Bash prompts to customize!

- PS1: Interactive Bash prompt (what you usually see)
- PS2: Continuation prompt (line-broken commands or inline function definitions)
- PS3: Selection prompt (used with the 'select' command)
- PS4: Trace output prompt (used with set -x)

PS1 is probably the most customized.

Coloring the prompt

There are 256 colors available for the background, and 256 available for the foreground (text color).
To set the background use: \e[48;5;<color>m
For the foreground use: \e[38;5;<color>m

Making your prompt more informative

Along with colorizing your prompt, you can also get more information out of it. Here are some escape sequences for inserting information into your prompt:

Sequence	Result
\u	username
\h	hostname
\H	fully qualified domain name
\w	current working directory
\W	basename of current working directory
\!	history number of command
\t	24-hour time
\T	12-hour time

Special characters

Bash can output special (non-ASCII) characters to the screen for extra fun prompts! Some examples:

Character	Encoding
Lambda	\u03BB
Skull and crossbones	\u2620
Lightning bolt	\u2b4d
Hot Spring/Java logo	\u2668
Check mark	\u2714
'X' mark	\u2718

Functions

Bash has functions, just like other programming languages.

Functions

Bash has functions, just like other programming languages.

- You can use these functions to automate certain tasks

Functions

Bash has functions, just like other programming languages.

- You can use these functions to automate certain tasks
- You can create entirely new commands

Functions

Bash has functions, just like other programming languages.

- You can use these functions to automate certain tasks
- You can create entirely new commands
- You can modify existing commands

Functions

Automating tasks

```
# Open notes in Emacs for editing
function class()
{
    emacs "${HOME}/Documents/school/winter-2017/${1,,}/notes.org" & > /dev/null
}

# LaTeX templating function
function latex-template()
{
    # Require arguments
    if [ $# -ne 1 ]
    then
        printf "error: requires filename\n"
        return
    fi

    # The argument is the document name
    docname=${1}

    # Copy the template documents over
    printf "Copying templates..."
    cp "${HOME}/.templates/latex/docname.tex" "${docname}.tex"
    cp "${HOME}/.templates/latex/makefile" makefile

    # Configure the makefile
    printf "Configuring makefile..."
    sed -i -e "s/docname/${docname}/g" makefile

    printf "Done\n"
}
```

Functions

Creating new commands

```
# Creating diskspace command
function diskspace
{
    # Get argument for directory
    checkdir="${1}"
    if [[ -z ${checkdir} ]]
    then
        # Default to current directory
        checkdir="$(pwd)"
    fi

    # Go to directory
    builtin cd "${checkdir}"

    # Get temporary log file
    tmpfile=$(mktemp)
    printf "Creating temporary log in %s\n" "${tmpfile}"

    # Check diskspace
    printf "Checking diskspace in %s...\n" "${checkdir}"
    du -S -h --max-depth=1 | sort -n -r > ${tmpfile}
    less ${tmpfile}

    # Return to previous directory
    cd -
}
```

Functions

Modifying existing functionality

```
# Changing behavior of cd
function cd
{
    # This will pass all the arguments on the
    # command line (the "$@") to the 'cd' builtin,
    # and then execute an 'ls' on the current directory
    builtin cd "$@" && ls
}
```

Functions

You can add special behavior for “cd” when overloading it in this fashion. Some examples:

- Only do an “ls” in the folder is there are less than *n* files
- Activate/deactivate aliases based on the presence/absence of certain files

Aliases

Aliases can be used to:

Aliases

Aliases can be used to:

- Shorten command names

Aliases

Aliases can be used to:

- Shorten command names
- Add default arguments

Aliases

Aliases can be used to:

- Shorten command names
- Add default arguments
- Change command names

Aliases

Aliases can be used to:

- Shorten command names
- Add default arguments
- Change command names

Check your current aliases by typing 'alias' at the prompt.

Aliases

Shortening command names

```
alias dcm='docker-compose'
alias nextcloud='/home/chris/.usr/local/nextcloud'
alias xonotic='${HOME}/.usr/bin/xonotic'
```

Aliases

Adding default arguments

```
alias mv='mv -v'
alias cp='cp -v'
alias rm='rm -v'
alias l='ls -CF'
alias la='ls -A'
alias ll='ls -alF'
alias ls='ls --color=auto'
```

Aliases

Change command names

```
alias goodbye-ram='google-chrome-stable'
```

References

Bash

`https://misc.flogisoft.com/bash/tip_colors_and_formatting`

`https://en.wikipedia.org/wiki/ANSI_escape_code#Colors`

Linux

`http://tldp.org`

`https://linux.org`

LaTeX

`http://latex.org`

`https://latex-project.org`

Slides And Code

https://gitlab.com/cwpitts/presentations