



Previously On...

CIS4930

- Syllabus review
- A few word on what's Linux and open source
- More than a few words on Command Line Interfaces

Welcome to Day #2 of the “Surprise Course”



Quick Announcements

Module Section in Canvas

- “slides” posted there
- Reading assignments – to supplement slides
- Remember to take the First Week Quiz!
- Final Exam Matrix

TAs are available

- See URL for Amir (we are working on getting a room)
- Canvas message directly Dan NGuyen until specific hours are set



M01

Basic Usage

Menu for this module

CLI Essentials

- Introduction to the Linux Command Line Interface and its Bash shell.
- We will cover the basics of **navigating the file system** (more about this topic in the module dedicated to the Linux File System),
- as well as **managing processes**.

Getting Help

- "Give a man a fish and he is fed for a day, teach him to fish...".
- When it comes to Linux, "learning to fish" boils down to learning to RTFM.
- We are going to learn to use the help tools available in any Linux system.

What is interacting with Bash all about?

Basic Notions

What is Bash?

- Bourne Shell (sh) → Bourne Again Shell (bash)
- There are alternatives...
 - zsh on MacOS
 - Korn shell
 - C shell
 - TENEX csh
 - Friendly Interactive sh

Where is it used?

- Bash used in Linux / UNIX...
- On windows: Cygwin, git bash...
- On MacOS (used to be Bash, now it's Zsh)

What is the shell interpreting exactly? Commands!



“Commands” == Built-in commands

- The shell is the one interpreting them
- echo
- pwd
 - (The shell’s prompt shows the current working directory)
- type
 - `type pwd`
 - `type type`
 - `type date`
 - `type -a date`

“Commands” == Keywords: if while for ...

- Also interpreted by the shell itself

“Commands” == Functions: skip until we script

- You guessed it; also interpreted by the shell itself

“Commands” == External Commands

- date
- type date
- type -a date
- Let's go meta → bash or sh

Bash Aliases

Let us look at a few aliases

- Look at the alert alias
 - aliases == 1-line scripts
- Look at the **ls** alias
 - Commands have options
 - One letter (BSD Style)
 - One letter (standard) -a -b -c
 - Full word --help --version
 - Examples
 - ls -l
 - ls -a → Hiding files with dot
- Order does not matter & we can put them together
 - ls -a -l → ls -l -a → ls -al → ls -la
 - → order does not matter when these are **toggles**
- Bash is case sensitive
 - ls -a
 - ls -A
- Full word options
 - ls --help --version

Defining our own Aliases

- Aliases can be removed
 - `unalias ls`
 - Test it to show lack of color!
- They can be defined
 - `alias ls='ls --color=auto'`
- They go away when closing the shell
 - Close shell
 - Reopen it
 - Check for alias presence
 - → see bash config files later
- What can we (re)define aliases on?
 - → external commands
 - → builtins
 - `alias type='type -a'`

Side remark: use single quotes for now

Alias stuff='echo ' '	The single quote is interpreted as closing the first single quote	>
alias stuff=\$'echo \'	Escapes the \' to ' Alias tries to echo ' but this is an unclosed string so PS2 appears	stuff >
Alias stuff='echo ""	Concatenating a single quote in double quotes The single quote is added, same problem than above	stuff >
Alias stuff='echo' \"	Concatenating an escape single quote in double quote The alias try to do echo \'	stuff ,
Alias stuff="echo \'	Using double quotes instead works	stuff ,
Alias stuff="echo '"	Again, this would result in the alias expanding to echo ' which lacks a closing single quote	stuff >