



Introduction to the bash

Szu-Chi Chung

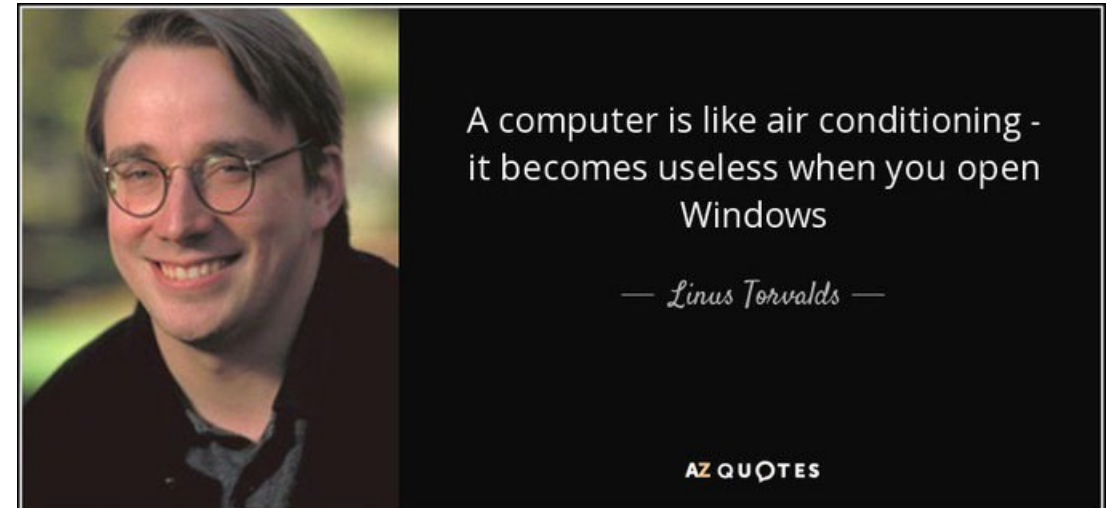
Department of Applied Mathematics, National Sun Yat-sen University

Course Outline

- ▶ Install the environment first
 - ▶ <https://learn.microsoft.com/en-us/windows/wsl/install>
 - ▶ <https://mobaxterm.mobatek.net/download.html>
- ▶ Bash commands to help you be comfortable with the command line
 - ▶ <https://github.com/RehanSaeed/Bash-Cheat-Sheet>
 - ▶ https://oit.ua.edu/wp-content/uploads/2020/12/Linux_bash_cheat_sheet-1.pdf

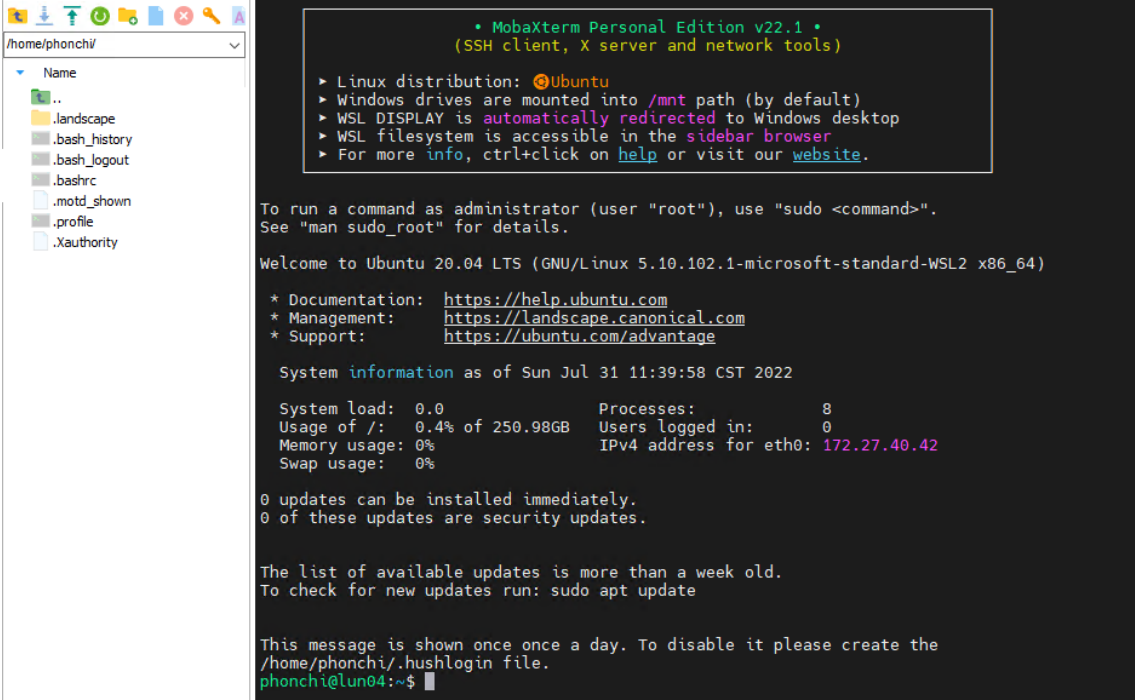
What is the shell

- ▶ Computers these days have a variety of interfaces for giving them commands
 - ▶ Fanciful graphical user interfaces, voice interfaces, and even AR/VR are everywhere
 - ▶ These are great for 80% of use-cases, but they are often fundamentally restricted in what they allow you to do — you cannot press a button that isn't there or give a voice command that hasn't been programmed. To take full advantage of the tools your computer provides, we have to go old-school and drop down to a textual interface: The Shell
- ▶ In this lecture, we will focus on the Bourne Again SHell, or “bash”
 - ▶ This is one of the most widely used shells. To open a shell prompt (where you can type commands), you first need a terminal. Your device probably shipped with one installed, or you can install it



Using the shell

- ▶ You will see a *prompt*. It tells you that you are on the machine `lun04` and that your “current working directory”, or where you currently are, is `~` (short for “home”). The `$` tells you that you are not the root user
 - ▶ At this prompt, you can type a command, which will then be interpreted by the shell
 - ▶ Tab can be used for auto-completing
 - ▶ The program will be searched under the `$PATH` variable
 - ▶ `env` will list all environment variables
 - ▶ `export` can be used to set environment variables
 - ▶ `explorer.exe` . (open `.` in mac) to open the directory



```
• MobaXterm Personal Edition v22.1 •
(SSH client, X server and network tools)

▶ Linux distribution: Ubuntu
▶ Windows drives are mounted into /mnt path (by default)
▶ WSL DISPLAY is automatically redirected to Windows desktop
▶ WSL filesystem is accessible in the sidebar browser
▶ For more info, ctrl+click on help or visit our website.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

Welcome to Ubuntu 20.04 LTS (GNU/Linux 5.10.102.1-microsoft-standard-WSL2 x86_64)

* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

System information as of Sun Jul 31 11:39:58 CST 2022

System load: 0.0 Processes: 8
Usage of /: 0.4% of 250.98GB Users logged in: 0
Memory usage: 0% IPv4 address for eth0: 172.27.40.42
Swap usage: 0%

0 updates can be installed immediately.
0 of these updates are security updates.

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

This message is shown once once a day. To disable it please create the
/home/phonchi/.hushlogin file.
phonchi@lun04:~$
```

1. Navigating Directories

- ▶ A path on the shell is a delimited list of directories, separated by / on Linux and macOS and \ on Windows. On Linux and macOS, the path / is the “root” of the file system, under which all directories and files lie, whereas on Windows there is one root for each disk partition (e.g., C:\)
 - ▶ Absolute path starts with /
 - ▶ Relative path starts with .. or .
- ▶ To see what lives in a given directory, we use the *ls* command
- ▶ Usually, running a program with the *--help* flag will print some help text that tells you what flags and options are available
 - ▶ *man* and *tldr* are also useful
 - ▶ If this is the first time you initialize the system try *sudo apt update* and *sudo apt install tldr*

Navigating Directories

- ▶ *ls -la* will give you more information about each file or directory present
 - ▶ First, the *d* at the beginning of the line tells us that it is a directory
 - ▶ Then follow three groups of three characters (rwx). These indicate what permissions the owner of the file, the owning group (phonchi), and everyone else respectively has on the relevant item. A - indicates that the given principal does not have the given permission
 - ▶ You can check the groups using *groups*
 - ▶ You can change the permission using *chmod*
- ▶ *cd* can change the directory
 - ▶ *pwd* prints the current directory

```
phonchi@lun04:~$ ls -la
total 32
drwxr-xr-x 3 phonchi phonchi 4096 Jul 31 11:39 .
drwxr-xr-x 3 root    root    4096 Jul 30 17:36 ..
-rw----- 1 phonchi phonchi   51 Jul 31 11:39 .Xauthority
-rw----- 1 phonchi phonchi    3 Jul 30 19:13 .bash_history
-rw-r--r-- 1 phonchi phonchi  220 Jul 30 17:36 .bash_logout
-rw-r--r-- 1 phonchi phonchi 3771 Jul 30 17:36 .bashrc
drwxr-xr-x 2 phonchi phonchi 4096 Jul 30 17:37 .landscape
-rw-r--r-- 1 phonchi phonchi    0 Jul 31 11:39 .motd shown
-rw-r--r-- 1 phonchi phonchi  807 Jul 30 17:36 .profile
```

2. File operations

- ▶ *mkdir* can be used to make new directories while *mkdir -p* can be used to create nested directories
- ▶ *touch* can be used to create files, while *cat/head/tail* can be used to print the content of a file
- ▶ *rm* can be used to remove files and *rm -rf* can be used to remove directories recursively
- ▶ *cp* can be used to copy files while *mv* can be used for moving files or renaming files
 - ▶ *cp -r* can be used to copy directories
- ▶ *ln -s* can be used to create a symbolic link

3. Connecting programs

- ▶ In the shell, programs have two primary “streams” associated with them: their input stream and their output stream
 - ▶ Normally, a program’s input and output are both your terminals. That is, your keyboard as input and your screen as output. However, we can also rewire those streams!
 - ▶ The simplest form of redirection is `> file` (Overwrite)
- ▶ You can also use `>>` to append to a file. Where this kind of input/output redirection really shines is in the use of pipes. The `|` operator lets you “chain” programs such that the output of one is the input of another
- ▶ The `xargs` command will execute a command using STDIN as arguments. For example, `ls | xargs rm` will delete the files in the current directory

4. Finding files/code

- ▶ All UNIX-like systems come packaged with *find*, a great shell tool to find files. *find* will recursively search for files matching some criteria
 - ▶ *locate* is another useful tool, but you need to install it with `apt` on Ubuntu
- ▶ Finding files by name is useful, but quite often, you want to search based on file content
 - ▶ A common scenario is wanting to search for all files that contain some pattern, along with where in those files said pattern occurs. To achieve this, most UNIX-like systems provide *grep*
 - ▶ `grep "this" file.txt`
 - ▶ `grep "this" . -r`

Finding shell commands

- ▶ You may want to find specific commands you typed at some point. Typing the up arrow will give you back your last command, and if you keep pressing it, you will slowly go through your shell history
 - ▶ The `history` command will let you access your shell history
 - ▶ In most shells, you can make use of `Ctrl+R` to perform the backward search
- ▶ `Ctrl+A` can move the cursor to the front while `Ctrl+E` can move the cursor to the end

5. Job Control

- ▶ *top/htop* will list all process
- ▶ When typing *Ctrl – C* this prompts the shell to deliver a SIGINT signal to the process and many programs can be stopped
- ▶ *&* suffix in command will run the command in the background
- ▶ *jobs* will list all background jobs
- ▶ *ps -ef* will list all processes
- ▶ A more generic signal for asking a process to exit gracefully is to use the *kill* command, with the syntax *kill – 9 < PID >*
- ▶ *tmux* can be used as multiplexers

6. Dotfiles

- ▶ Many programs are configured using plain-text files known as dotfiles (because the file names begin with a .)
 - ▶ Shells are one example of programs configured with such files. On startup, your shell will read many files to load its configuration
 - ▶ For bash, editing your `~/.bashrc`
 - ▶ <https://github.com/Bash-it/bash-it>
 - ▶ For git, editing your `~/.gitconfig`
- ▶ Use *source* to activate the dotfiles

Try the GUI Programs

- ▶ xclock
- ▶ xeyes
- ▶ xcalc
- ▶ gedit