# Lab1-1: Ubuntu operations and Docker intro

By Peter Hung, Nick Wang, Allen Ou,Tim, Tony, Sam Liu last modified on 03/03, 2019.

The objective of the tutorial is to have your working environments: Ubuntu 16.04 system. The content is aimed for beginners who have little knowledge about basic commands in Ubuntu and Vim. We assume the users have some knowledge about programming in C++ and (a bit) Python.

*1.Any sentence behind “$” means those are commands that typed in the terminal*

*$ 代表terminal的指令*

* *laptop: do the command after $ on the laptop/PC side*
* *laptop: 代表在筆電上打指令*

## Hardware and Software Setup

The tutorial requires Ubuntu 16.04 system installed in the following:

* First option, use the provided Virtualbox ([image](https://drive.google.com/open?id=1Q8p8yYJG2gmPt9FQowKGFI_qaQe3z060), 11 GB).
* Second option, use the provided USB Virtualbox image from our Teaching Assistant
* For advanced users who wish to have native system, take a look at our Dockerfiles
* **For anyone wants to run docker on your native Ubuntu, you could take a look at** [**Native-Ubuntu-docker**](https://drive.google.com/open?id=1ey-CzgqlLGiII0q1LYrR5whvjlUzYmJXwOm5ubCEgRo)**.**

[**Lab1-1: Ubuntu operations and Docker intro**](#_jkop8kfo2r4s) **1**

[Hardware and Software Setup](#_rhtwi3gigp91) 1

[Overview](#_cz4nh1b7xbx6) 2

[Topics and Activities](#_c0rulu3euxnw) 2

[Topic/Activity 1 Ubuntu Installation](#_1nmm7g8819ns) 2

[Topic/Activity 2 Use Ubuntu and Vim](#_wlrgds6cnlye) 7

[Topic/Activity 3 Change hostname](#_sipp7qn386kk) 9

[Topic/Activity 4 Get Docker Images & Run Docker Container](#_b7phz48djoqw) 12

[Topic/Activity 5 DockerFile](#_ftmhe4cmnbmh) 15

[Topic/Activity 6 Access Workstation](#_ef32fm62ldsl) 16

## 

## 

## Overview

Estimated Time to Finish: 1 hours

After completing this tutorial you should

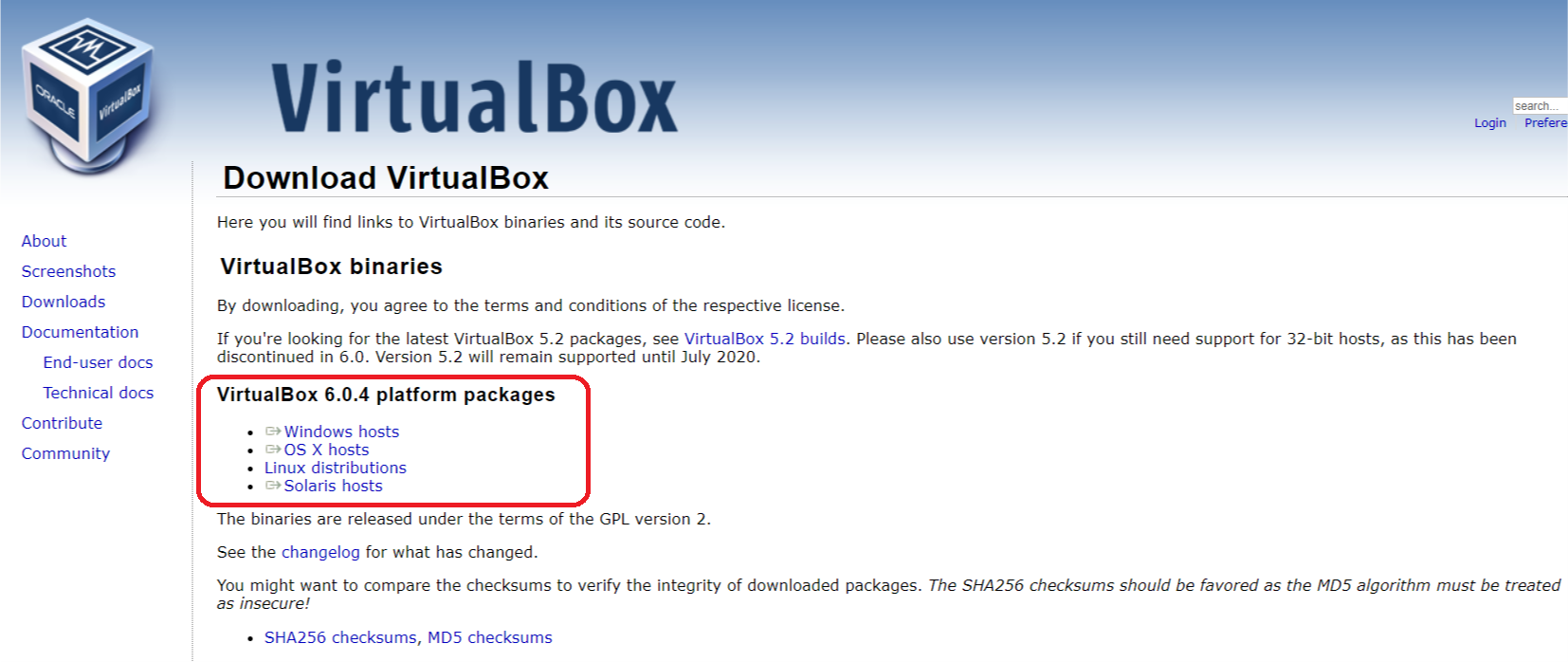
* understand how to install ubuntu system on your laptop
* be able to use vim to modify the text file
* develop the common sense of ubuntu system
* familiar with docker

## Topics and Activities

### Topic/Activity 1 Ubuntu Installation

**Download and install Virtualbox on your machine**

**MAKE SURE YOUR VIRTUALBOX IS THE LATEST VERSION**

****<https://www.virtualbox.org/wiki/Downloads>

**Start Virtualbox**

**New(新增)** -> Linux -> 64bit -> Memory 2G -> select the .vdi file.



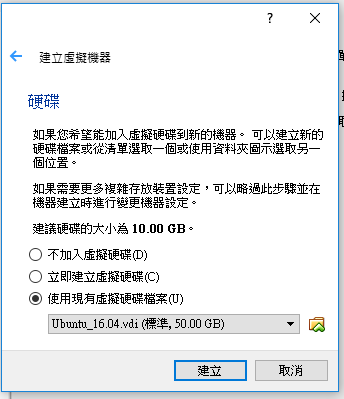
New-> **Linux -> 64bit -> Memory 2048M** -> select the .vdi file.

you can change the name of this virtual machine as you want

|  |  |
| --- | --- |
|  |  |

New-> Linux -> 64bit -> Memory 2G -> **select the .vdi file.**

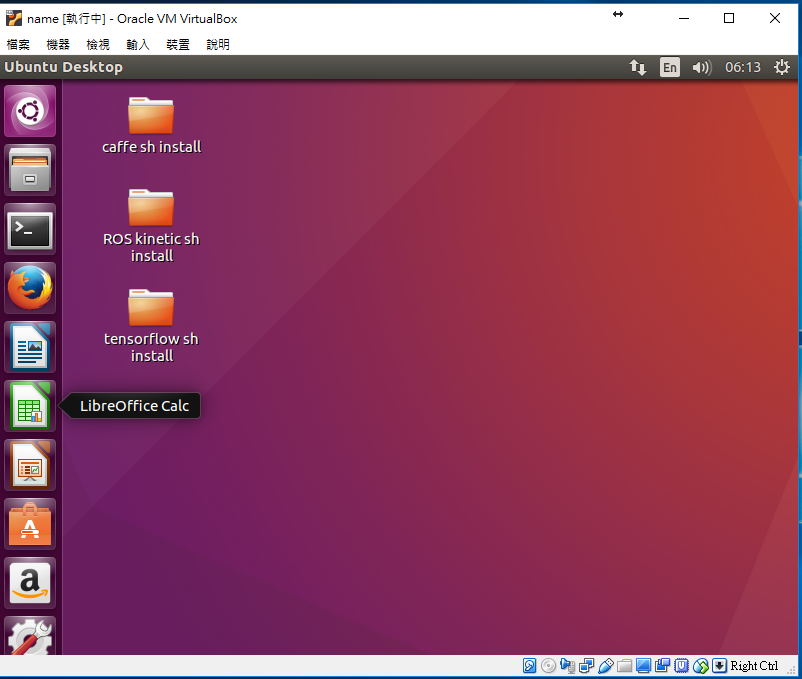
使用現有虛擬硬體檔案，選擇下載好的VirtualBox Image (.vdi檔案)



press the “activate” to activate your virtual machine



The password is **bocelli2015**

****

**Network Setup**

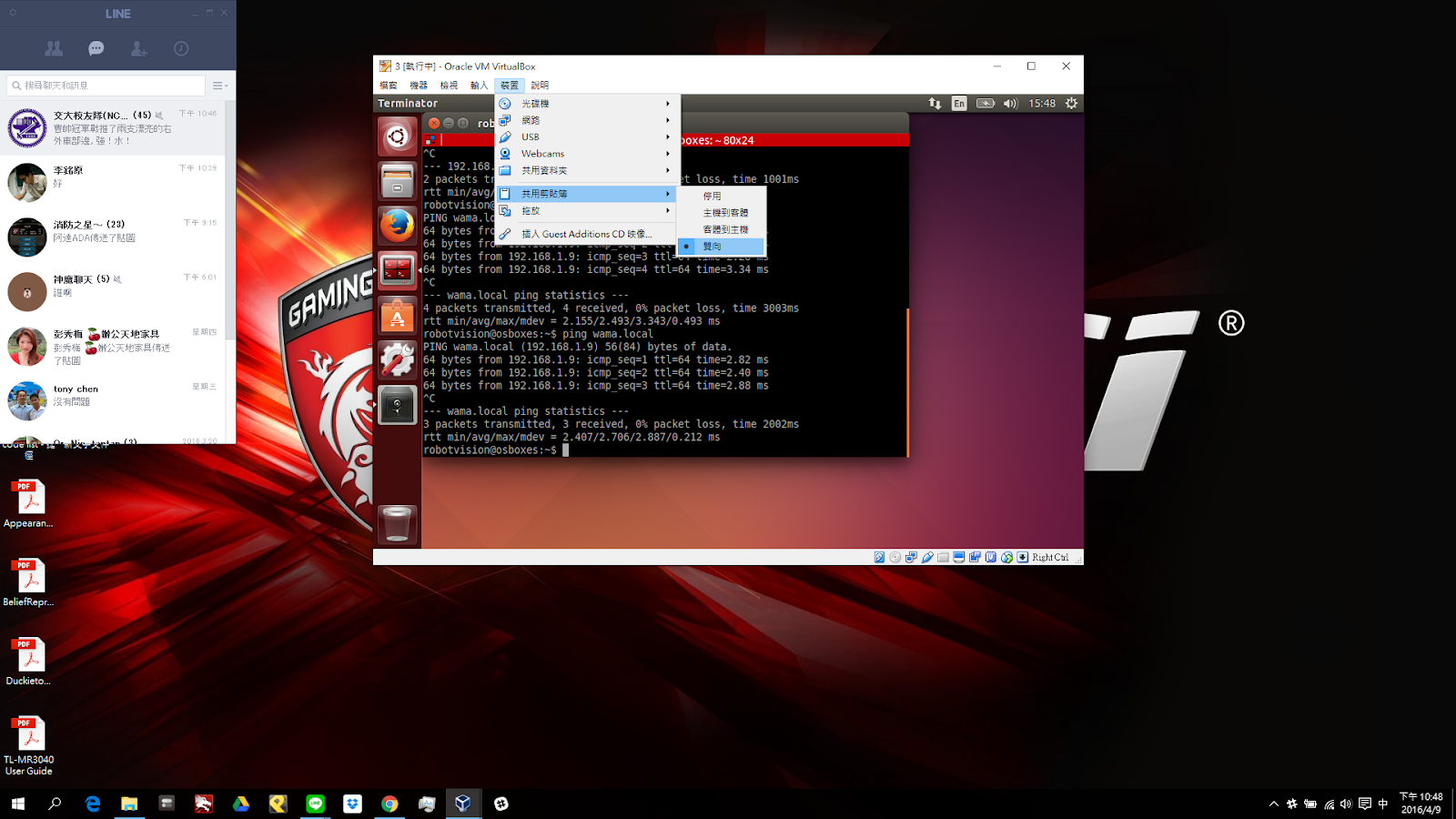
open the setting terminal from 裝置 -> 網路 -> 網路設定

Change the first thing to bridged adapter (橋接介面卡)



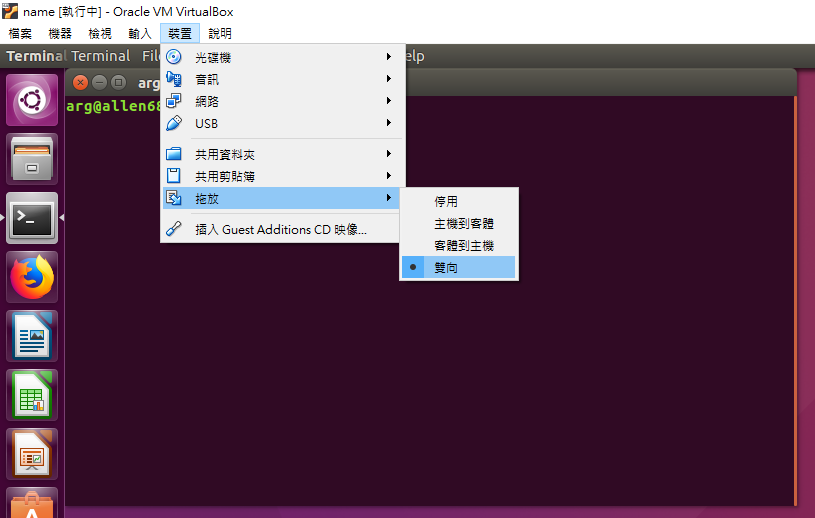
**Scrapbook Setup**

共用剪貼簿



**Drag and Drop**

托放



### Topic/Activity 2 Use Ubuntu and Vim

You should know the basic commands in Ubuntu.

|  |  |
| --- | --- |
| **sudo** [COMMAND] | act as root before other commands (**最高權限進行指令**) |
| **cd** [directory] | enter a folder (**進入資料夾**) |
| **ls** | list all files and folders in current directory(**列出該目錄所有檔案、資料夾**) |
| **mkdir** [dir name] | create a folder (**建立新資料夾**) |
| **cp** [file] [new file] | copy files **(複製檔案)** |
| **rm** [file] | delete files **(刪除檔案)** |
| **apt-get update** | update packages **(取得遠端更新伺服器的套件檔案清單)** |
| **apt-get instal**l [package name] | install packages **(安裝套件)** |
| **git clone** [url of remote repository] | Git command line utility which is used to target an existing repository and create a clone, or copy of the target repository. **(透過Git系統從遠端repo複製一份repo到本地端)** |
| [COMMAND1] **&&** [COMMAND2] | **(做完COMMAND1 後 做COMMAND2)** |

Exercise: Create a folder and use vim to edit a file.

練習: 創造一個新資料夾，並用vim去修改檔案

**laptop$ ls**

****

**laptop$ mkdir csp2017**

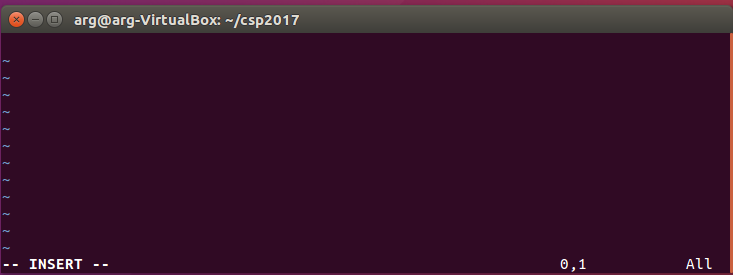
**laptop$ ls**

****

**laptop$ cd ~/csp2017**

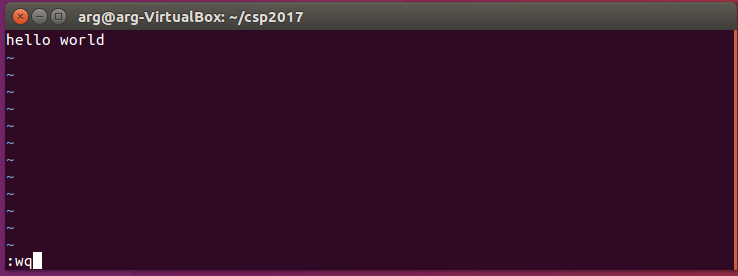
**laptop$ vim test**

**press “i” to enter “Insert mode**

****

**please type “hello world” as shown below**

**(在檔案裡打一行”hello world”)**

****

**press “esc” and type :wq and press “enter”**

**w: write**

**q: quit**

**laptop$ ls**

****

**laptop$ cp test test2**

**laptop$ ls**

****

**laptop$ rm test**

**laptop$ ls**

****

### Topic/Activity 3 Change hostname

Due to our fleet setting, we wish that all laptops have unique hostname. We suggest to use:

* [Name] for your laptop, ex: allen

**在更改筆電的hostname時，請不要使用特殊字元 (e.g. %^#@!\_,;:” )，只用數字以及小寫英文字母**

Open a terminal (press this terminal app shown below or type [**ctrl + alt + T**]):

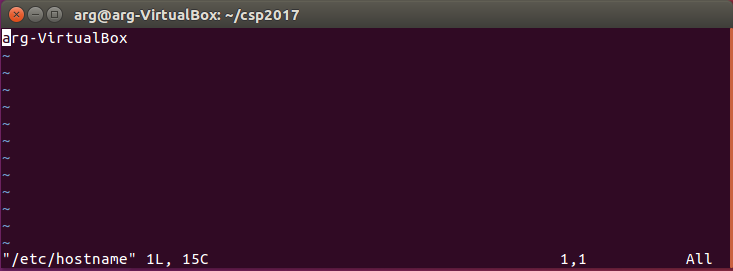


type in the command below:

記得打sudo, 沒有sudo會無法更改此檔案

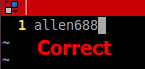
**laptop$ sudo vim /etc/hostname**

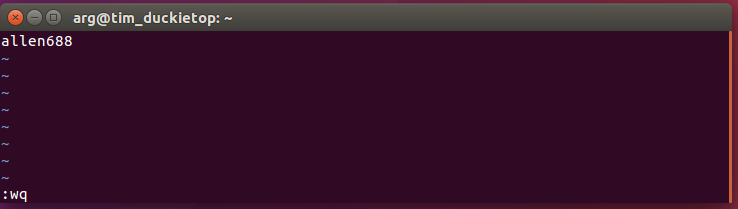
[sudo] password for arg: bocelli2015



若不小心忘了打sudo卻已經更改檔案，請打”**:q!**”強制退出後再加上sudo進去一次

Change "hostname" to your own hostname. For example: allen





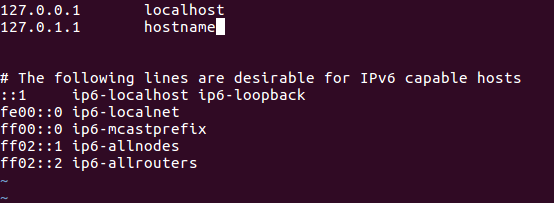
For information about vim & how to use it,

check<http://linux.vbird.org/linux_basic/0310vi.php>

After exiting vim editing and go back to terminal, type in the command below:

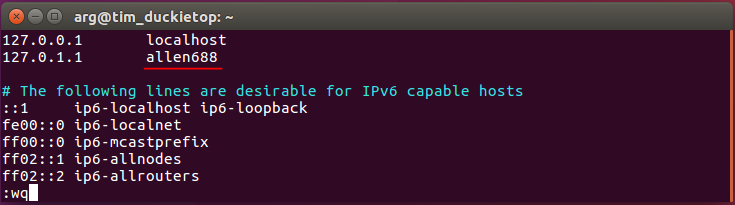
記得打sudo, 若沒有sudo會無法更改此檔案

**laptop$ sudo vim /etc/hosts**

****

change "hostname" to the one you typed in the /etc/hostname file, don’t forget to save it (:wq) after modification.

將hostname改的跟剛剛在/etc/hostname一樣的名字，並儲存

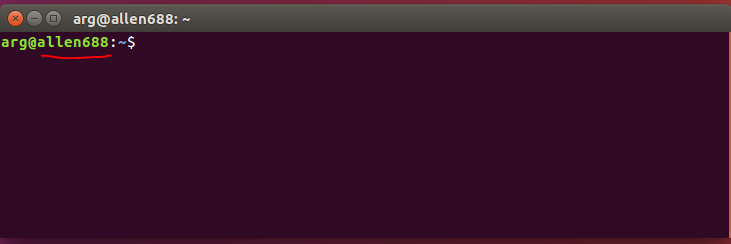


Then reboot (重開機)

**laptop $ sudo reboot**

Double check if hostname is changed.

確認hostname改變了



Check point (檢查點):

you should have a virtual machine called arg@**[HostName]**

確認你的虛擬裝置, hostname叫做 arg@**[HostName]**

****

Reference

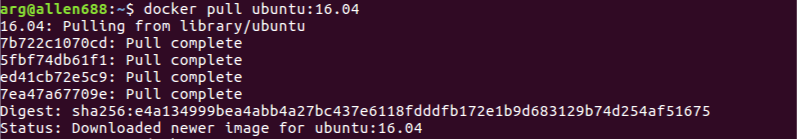
Duckiebook:<http://book.duckietown.org/master/duckiebook/setup_laptop.html#sec:setup-laptop> (8.1 - 8.3)

### 

### Topic/Activity 4 Get Docker Images & Run Docker Container

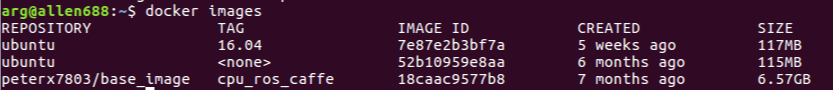
**For anyone wants to run docker on your native Ubuntu, you could take a look at** [**Native-Ubuntu-docker**](https://drive.google.com/open?id=1ey-CzgqlLGiII0q1LYrR5whvjlUzYmJXwOm5ubCEgRo)**.**

**Laptop $ docker pull ubuntu:16.04**



“docker pull” command pull existed docker image from dockerhub.

**Laptop $ docker images**



“docker images” command show all the docker image info that you have download.

**Laptop $ cd ~/ && docker save -o ubuntu\_1604.tar ubuntu:16.04**

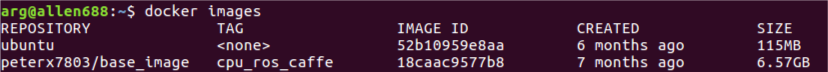
“docker save” command could export your docker image into a compressed file, and easy to save in any usb flash drive.

-o parameter: Output filename

Delete docker image from your disk

**Laptop $ docker rmi ubuntu:16.04**

**Laptop $ docker images**

****

**Laptop $ cd ~/ && docker load -i ubuntu\_1604.tar**



“docker load” command import your compressed tar image.

-i parameter: Read from tar archive file, instead of STDIN

Run Docker Container

Docker can access the external device like usb device, with giving the correct parameters, we could directly using them inside of docker.

First we start a docker container that we already have by normal user model.

**Laptop $ docker run -it --rm --name ubuntu ubuntu:16.04**

-it parameter: Open a pseudo-TTY connected to the container’s stdin; creating an interactive bash shell in the container.

--rm parameter: Remove the container when it stopped.

--name parameter: Name of the opening container, this could help you command it easily.

**Container $ ls /dev**



you could see there are few devices could be accessed by a normal user container.

**Container $ exit**

**Laptop $ docker run -it --rm \**

**--name ubuntu \  
-v /home/arg:/hosthome \  
--privileged \**

**-p 8080:8080 \**

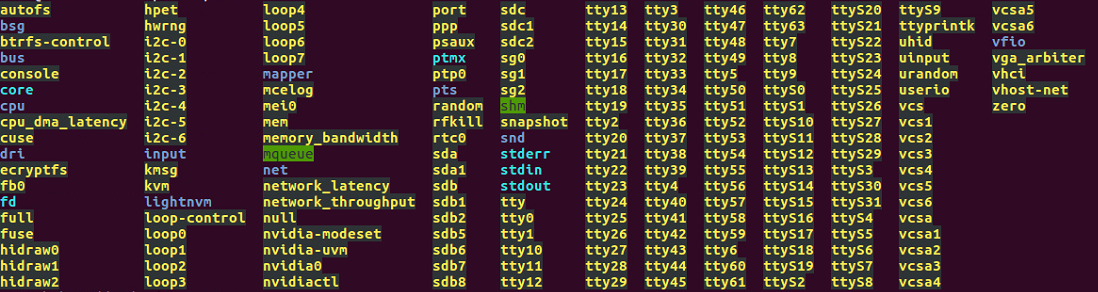
**ubuntu:16.04**

-v parameter : volume

--privileged parameter: privileged mode

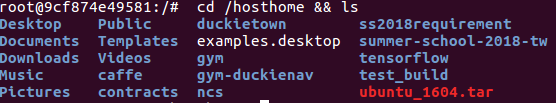
-p parameter: port mapping

**Container $ ls /dev**



You could see a lot of devices could be accessed by privileged mode.

**Container $ cd /hosthome && ls**



Access files located at host by mounted volume between host and container

Open another terminal

**Laptop $ docker port ubuntu**



Container port 8080 is mapping to host port 8080.

**Container $ exit**

### 

### Topic/Activity 5 DockerFile

First download [test\_build.tar](https://drive.google.com/file/d/13gPjYG5RbFv1F6lADw25K6JAdKFTJvtF/view?usp=sharing) and untar it and move the folder to ~/

We could choose to build our image by starting from scratch. But normally building our own image from some official image is more convenient to set up our develop environment.

Dockerfile is the key part to build our own image.

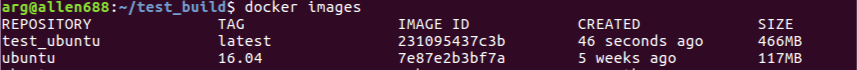
**Laptop $ cd ~/test\_build**

**Laptop $ vim Dockerfile**

|  |
| --- |
| FROM: Creates a layer from the Docker image.  RUN: Execute any commands in a new layer on top of the current image and commit the results.  COPY: Adds files from your Docker client’s current directory.  EXPOSE: Indicates the ports on which a container listens for connections.  CMD: Specifies what command to run within the container by default. |

**Laptop $ docker build -t test\_ubuntu -f Dockerfile .**

**Laptop $ docker images**



Now you have a new images with new packages installed. Also this image will execute default command if you not give the specific command.

**Laptop $ docker run -it --rm \**

**--name ubuntu \**

**test\_ubuntu**

