# Link Docker file Java

* <https://github.com/sunlight3d/YoutubeChannel/tree/master/Docker%20tutorials%202022/JavaDockerExample>
* https://www.youtube.com/watch?v=Bemy5gBjZrE&list=PLWBrqglnjNl3TDF6WKpAl4maE3yJ5CpYF&ab\_channel=NguyenDucHoang
* https://docs.docker.com/get-started/?utm\_campaign=2021-09-16-mailgun-email-verified&utm\_medium=email&utm\_source=mailgun

# \* Docker

Syntax cơ bản : Docker <component> <Command>

B1: Build file docker có tên ‘Dockerfile’ dấu . tìm file tên như thế ở thư mục hiện tại => Nó sẽ build ra 1 images có tên **i-firstserver:version1;**

**docker build -t i-firstserver:version1 -f Dockerfile .**

B2: Sau đó thực hiện chạy container từ image trên thì sẽ đc. Chứ ý cổng 80 nếu bạn cấu hình web ở image là cổng 80 ko thì ( spring mặc định là xxx(bên ngoài):8080(bên trong)

**docker run -it --name mywebserver -p 8888:80 i-firstserver:version1**

Tạo 1 container từ 1 image ánh xạ nó vào cổng 8888 thực tế nó chạy cổng 80

|  |  |
| --- | --- |
| Component :   * Image * Container * Network * Volume * .. | Commands :   * Ls : List * Run * Exec * Stop * Pull * prune * …. |

Basic commands image:

Docker image pull <image> :<tag> (tag có hoặc ko nếu ko thì nó sẽ là bản mới nhất)

|  |  |
| --- | --- |
| Image | Container |
| * Docker image pull <image> :<tag>   (tag có hoặc ko nếu ko thì nó sẽ là bản mới nhất) | **Docker container run [option] [--name nameYourContainer] <image:tags>** |
| Docker pull <image> : <tag> | Docker contaniner ls | Docker container ls –a  Ngọn hơn:  Docker ps | docker ps -a |
| Docker image ls | docker images | **Docker [container] start <container\_id>,**  **Docker [container] stop <container\_id>** |
| Docker image prune (xóa các image k dùng tới) | Docker container prune |
| Short – hand:   * Docker pull * Docker push | Docker container exec <ctainer\_id> <command> |
|  | Short-hand:   * Docker run * Docker run –d [--name YourContainerName] <image:tags>   (d =detach)   * Docker stop <name or id Container>, Docker start <name or id Container> * Docer exec <name or id Container> |
| Docker run –it –name <Container> <name image>  (-it là đẻ mở terminal thì phải vào terninal thì mới run đc)   * Ấn crtl + D để thoát container (nhưng mà nó sẽ kill luôn container) | Có 2 chế độ attach – detach: ấn Crtl + P +Q ( để detach container ra khỏi command line của chúng ta)   * Vào lại thì gõ Docker attach <cotainer\_id – hoặc tên> |
| Docker run –it < Container\_id > sh  Curl localhost | Thường xuyên xóa container đã đc sutdow   * Docker prune |
|  | Cat /etc/os-release (in ra phiên bản release hdh) |
| Kết hợp \ && trong Docker file để gộp nhiều lệnh vào 1 container tạm  Mỗi dòng nó sẽ build ra 1 container đấy & & thì hết & | Docker exec <container\_id> <command line>  Vd :   * Docker exec <container\_id> echo Hello-World * Docker exec <container\_id> sh –c echo path ( cái này thao tác với biến môi trường) * Docker exec <container\_id> cat /etc/os-release * **Thêm cớ -sh , - bash tức là đang kết nối từ xa với container (lúc này Crtl + D thì container sẽ ko bị exit) vì tiến trình chủ đạo pid = 1 còn cái sh là tiến trình khác** |
|  |  |

## Port mapping ( tức là ánh xạ port vì mạc định container http là port 80 chúng ta cần ánh xạ xang port client khác)

* Lệnh run container với port client và container nhưu sau:
* **Docker run –p <target\_port> : <container\_port>….**
* **Vd docker run –p 80:80**

## Log trong Docker

* Docker logs –tail –f <container\_id>

## Volume – bind mount

Vì image là bất biến( immulate) và ko trạng thái

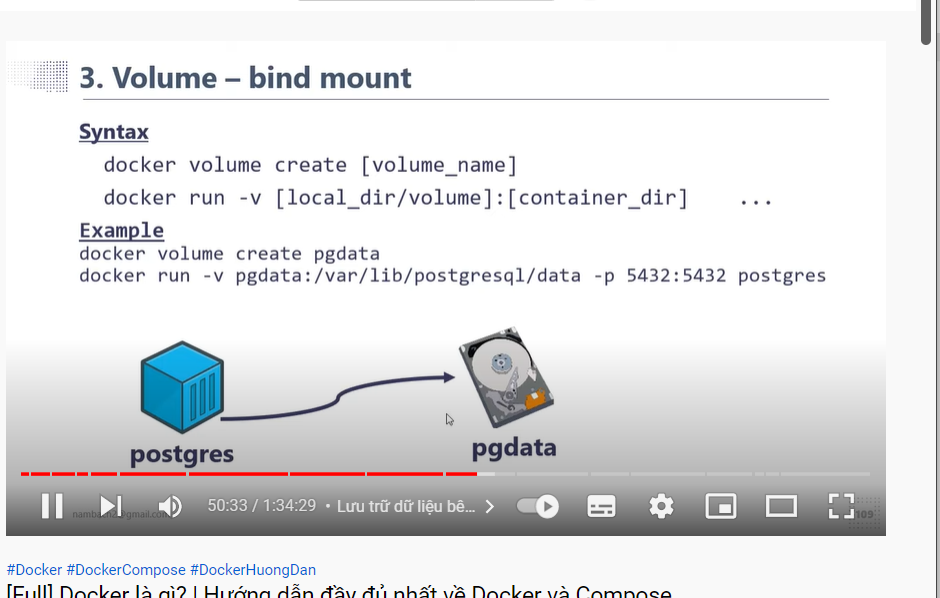
Volume đc sinh ra để giải quyết vấn đề này tóm lại volume nhưu là 1 cái usb để lưu thông tin từ containe này sang container khác. Hoặc giữ nguyên dữ liệu cho an toàn khi container bị xóa.

Syntax:

* Docker volume create [volume\_name]
* Docker run –v [local\_dir/volume] :[container\_dir]

Exemple

* Docker volume create pgdata
* Docker run –v pgdata:/var/lib/postgresql/data –p 5432:5432 postgres



* **Cách để giữ 1 container thì ko dùng volume mà nên tọa 1 image từ 1 container**

## Dockerfile

Syntax

* Để chạy 1 file Dockerfile

Vd:

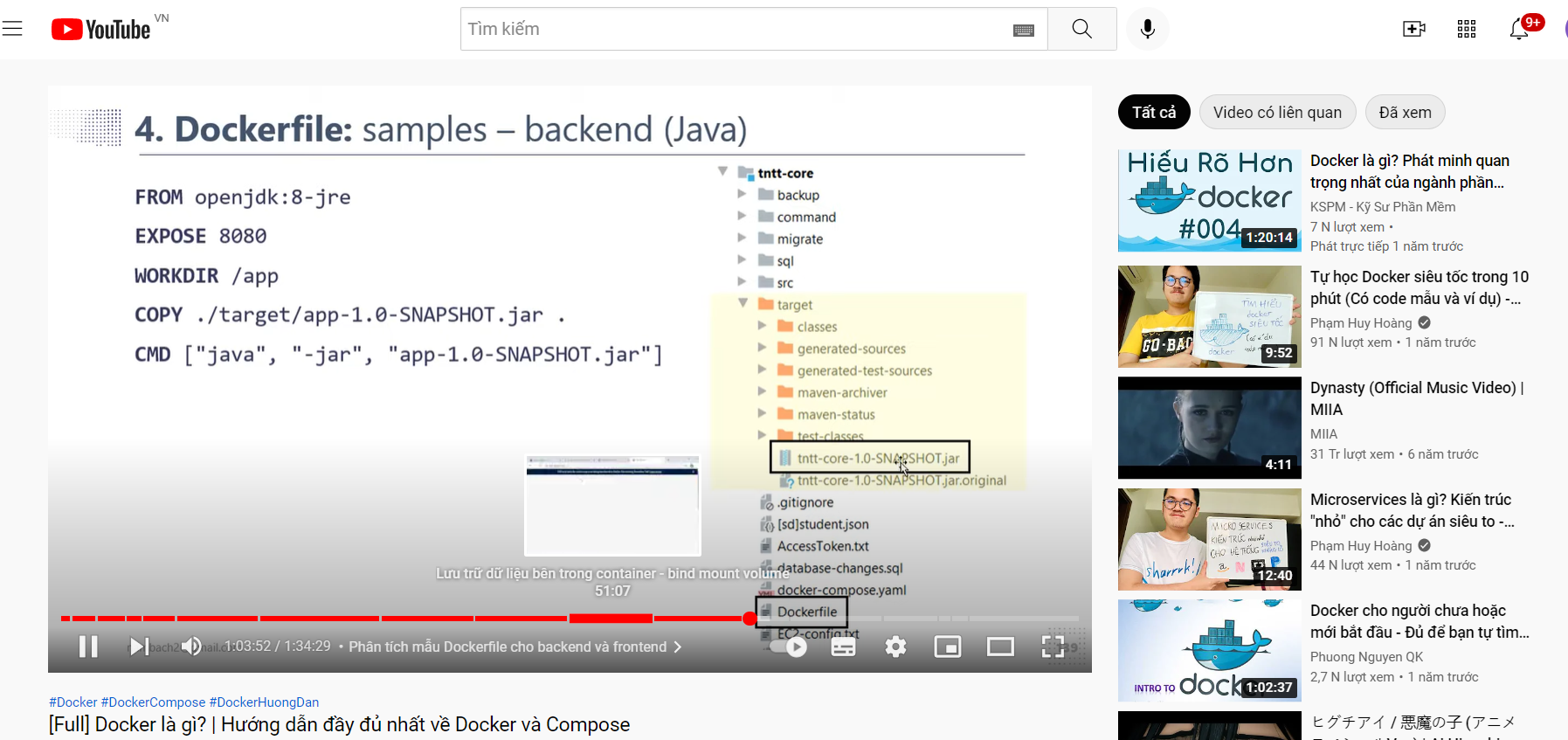
* + - * docker build -t nameimage:version --force-rm -f Dockerfile .

Docker build –t <image\_name>:<tag> . ( . là 1 tham số quan trong build context bao gồm cả dockerfile này tóm lại . để tìm file docker)

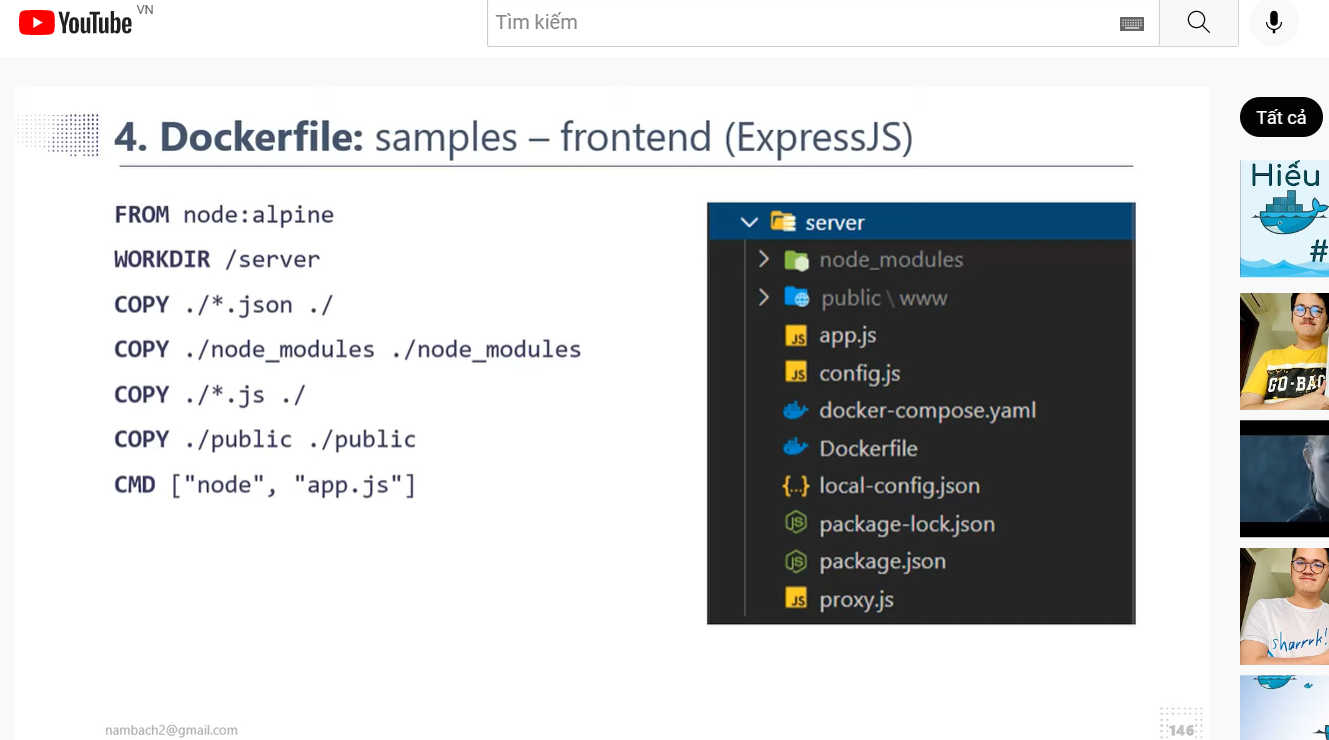
* Cấu trúc trúc file dockerfiel thì ở dưới
* Viêt trên 1 dòng duy nhất

|  |
| --- |
| * FROM <image> tạo ra 1 image mới từ image cũ * RUN <command> * WORKDIR <directory> thư mục mặc định của containe * COPY copy từ local vào image * ADD <src/URL> <dest> * EXPOSE <port> * CMD [“command”,”argument1”,”argument2”,…] |

Java



FE gì cũng thế build ra html



* Docker ignore

## Tải Images lên Registry (Docker Hub)

Login into the registry with

* **docker login**

Push image:

* **docker push <image> : <tag>**

Image cache

* Docker caches layes that remain unchanged only push change file
* To clear cache, user no cache –no-
* Tóm lại cái nào nặng thì tạo 1 cantainer riêng để tận dụng image cache

<https://xuanthulab.net/gioi-thieu-ve-docker-lam-quen-voi-docker-tao-container.html>

## B1 Cài Docker cái này lên web phía trên web xuân thu mà tìm kiếm

* Ubuntu

Chạy các lệnh để cài đặt:

* sudo apt update
* sudo apt install apt-transport-https ca-certificates curl software-properties-common
* curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add
* sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"
* sudo apt update
* apt-cache policy docker-ce
* sudo apt install docker-ce
* sudo systemctl status docker

Sau khi cài đặt, bạn có thể cho user hiện tại thuộc group docker, để khi gõ lệnh không cần xin quyền sudo

* sudo usermod -aG docker $USER

Logout sau đó login lại để có hiệu lực.

Ngoài ra khi sử dụng đến thành phần docker-compose thì bạn cài thêm

* sudo apt install docker-compose

Nếu có nhu cầu sử dụng Docker Machine trên Ubuntu, (công cụ tạo - quản lý các máy ảo chạy Docker Engine, các máy ảo này tạo bởi VirtualBox, bạn sử dụng Docker-machine để thực hành các ví dụ kết nối nhiều máy chạy Docker Engine khác nhau tạo thành cụm Server), thì cài thêm Docker Machine. Phiên bản mới nhất lấy tại [Docker Machine](https://github.com/docker/machine/releases)

Ví dụ cài bản v0.16.1, gõ lệnh sau:

* curl -L https://github.com/docker/machine/releases/download/v0.16.1/docker-machine-`uname -s`-`uname -m` >/tmp/docker-machine &&
* chmod +x /tmp/docker-machine &&
* sudo cp /tmp/docker-machine /usr/local/bin/docker-machine

Tất nhiên là Docker Machine cần VirtualBox để làm việc, nếu chưa có thì cài thêm

* sudo apt install virtualbox

## B2 Tải image trên web chính docker link mình để ở phía này nhé

Cần cài hệ điều hành Ubuntu Docker-Image để có 1 môi trường ubuntu nhé.

Các image có sẵn của docker <https://hub.docker.com/search?q=&type=image>

# \*\* Tóm tắt mấy lệnh cơ bản như sau có thể xem chi tiết ở phía dưới

*#kiểm tra phiên bản*

**docker --version**

*#liệt kê các image*

**docker images -a**

*#xóa một image (phải không container nào đang dùng)*

**docker images rm imageid**

*#tải về một image (imagename) từ hub.docker.com*

**docker pull imagename**

*#liệt kê các container*

**docker container ls -a**

*#xóa container*

**docker container rm containerid**

*#tạo mới một container*

**docker run -it imageid**

Tạo / chạy một container đặt tên là mycentos của centos

docker run -it --name mycentos centos

*#thoát termial vẫn giữ container đang chạy*

**CTRL +P, CTRL + Q, CTRL + Q +P**

*#Vào termial container đang chạy*

**docker container attach containerid**

*#Chạy container đang dừng*

**docker container start -i containerid**

*#Chạy một lệnh trên container đang chạy*

**docker exec -it containerid command**

#Xóa image có liên kết với các images con

**docker rmi b721d1cdaac7 -f**

Nếu chạy lệnh ntn mà nó vẫn báo:Response : Error response from daemon: conflict: unable to delete b721d1cdaac7 (cannot be forced) - image has dependent child images

→ Cần chạy lệnh này để xóa cả nó và các images con của nó (id image b721..):

docker image rm $(docker images --filter since=b721d1cdaac7 -q) -f

# \*\*\* Các lệnh cơ bản khác:

### Kiểm tra docker version

* docker –version
* docker info (Cái này thông tin chi tiết hơn về Dockẻ đang dùng)

### ****Tải 1 image trên webHub Docker****

* docker pull nameimage:tag

#Hoặc tải về bản cuối

* docker pull nameimage

Bây giờ hệ thống không có hệ điều hành Ubuntu, muốn tải nó về - vì trên hub.docker.com nó có tên ubuntu, ta sẽ tải về bản mới nhất bằng lệnh:

* docker pull ubuntu

Các image có sẵn của docker <https://hub.docker.com/search?q=&type=image>

### ****Tạo và chạy container****

Kiểm tra có các container nào đang chạy:

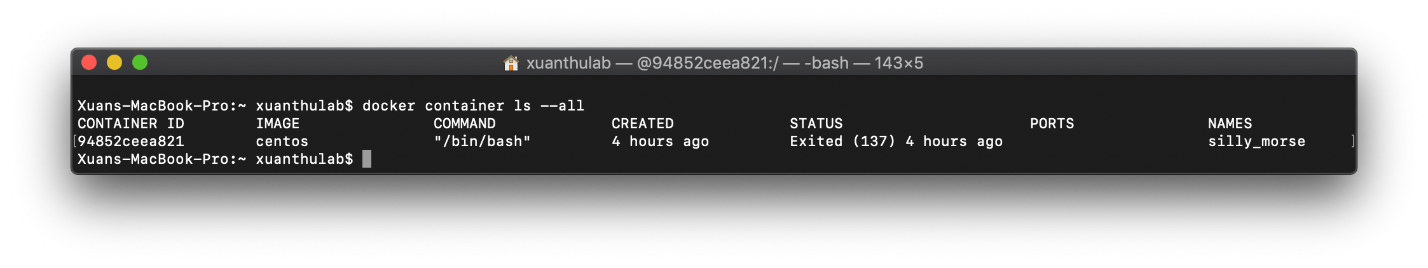
* docker ps

Liệt kê tất cả các container:

* docker container ls --all

Để tạo và chạy một container theo cú pháp:

* docker run [OPTIONS] IMAGE [COMMAND] [ARG...]



Các container bạn đã tạo, liệt kê ra hãy chú ý mấy thứ

* CONTAINER ID một con số (mã hash) gán cho container, bạn dùng mã này để quản lý container này, như xóa bỏ, khởi động, dừng lại ...
* IMAGE cho biết container sinh ra từ image nào.
* COMMAND cho biết lệnh, ứng dụng chạy khi container chạy (/bin/bash là terminate)
* STATUS cho biết trạng thái, (exit - đang dừng)

Trở lại phần trên, đang có image hệ điều hành Ubuntu, giờ muốn chạy nó (tạo một container từ nó), hãy gõ lệnh sau:

* docker run -it ubuntu

Bạn chú ý có tham số -it để khi container chạy, bạn có terminate làm việc ngay với Ubuntu. Tham số này có nghĩa là

* -t nó có nghĩa là console, cho phép kết nối với terminal để tương tác
* -i có nghĩa duy trì mở stdin để nhập lệnh.

Sau lệnh này, bạn đang chạy Ubuntu, bạn có thể gõ vài lệnh Ubuntu kiểm tra xem. Ví dụ kiểm tra phiên bản Ubuntu, xem hình dưới với lệnh cat /ect/\*release. Như vậy bạn đã tạo một container

### ****Một vài tham số khác:****

Tạo và chạy container, container tự xóa khi kết thúc thì thêm vào tham số --rm

* docker run -it --rm ubuntu

Tạo và chạy container - khi container chạy thi hành ngay một lệnh nào đó, ví dụ ls –la

* docker run -it --rm debian ls -la

Tạo và chạy container - ánh xạ một thự mục máy host vào một thư mục container, chia sẻ dữ liệu

* docker run -it --rm -v path-in-host:path-in-container debian

### ****Vào container đang đang chạy****

Kiểm tra xem bằng lệnh docker ps, nếu có một container với ID là containerid đang chạy, để quay quay trở lại terminal của nó dùng lệnh:

* docker container attach containerid

Chạy một container đang dừng

* docker container start -i containerid

Nếu cần xóa bỏ hẳn một container thì dùng lệnh

* docker container rm containerid

## ****Khái niêm cơ bản image và container trong Docker****

image là một gói phần mềm trong đó chứa những thứ cần như thư viện, các file cấu hình, biến môi trường để chạy mội ứng dụng nào đó. Nó giống như cái USB chứa bộ cài đặt hệ điều hành Windows!

Khi một phiên bản của image chạy, phiên bản chạy đó gọi là container - (vậy muốn có container phải có image). Bất ký lúc nào bạn cũng có thể kiểm tra xem có bao nhiêu container đang chạy và nó sinh ra từ image nào.