



DOCKER

For Beginner

By Chatri Ngambenchawong

Who am I



Name: Chatri Ngambenchawong (ping)

Role : Dev (dotnet)

System Analyst

DevOps

Blogger

More [About Me | naiwaen@DebuggingSoft](#)

Find me on:



Agenda

Prerequisite: Install Docker Desktop / Account Docker Hub

Day1

- Why Container ?
- What is a Container ?
- Docker Architecture and Components
- Magic of Container, Why Linux ?
- Basic Docker Command
- Container - Resource Monitoring & Limit
- Container - Networking
- Container - Store Data

Day2

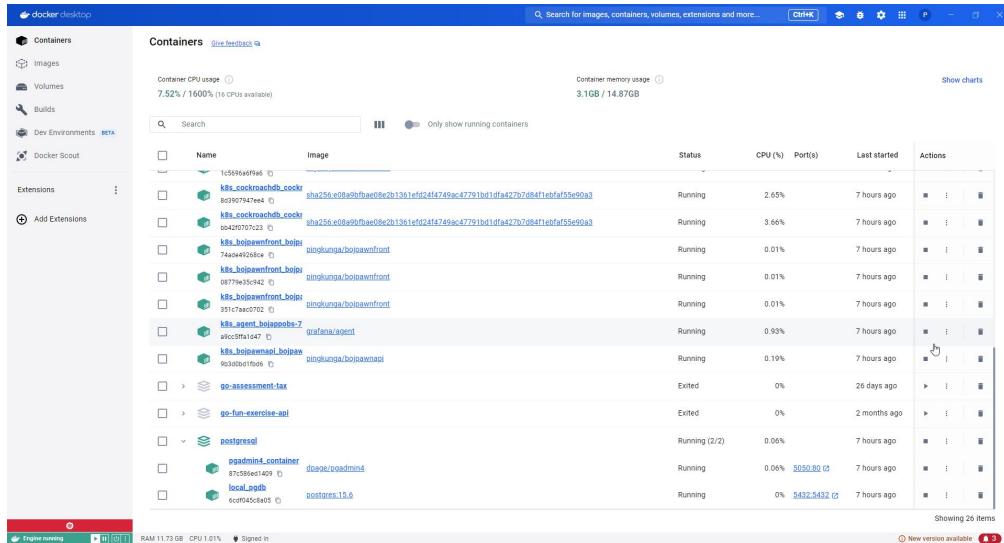
- Building Container (Dockerfile)
- Container Best Practice
- Docker Compose
- Portainer
- Wrap Up

Resource [pingkunga/train-docker-at-BRSU \(github.com\)](https://pingkunga/train-docker-at-BRSU)

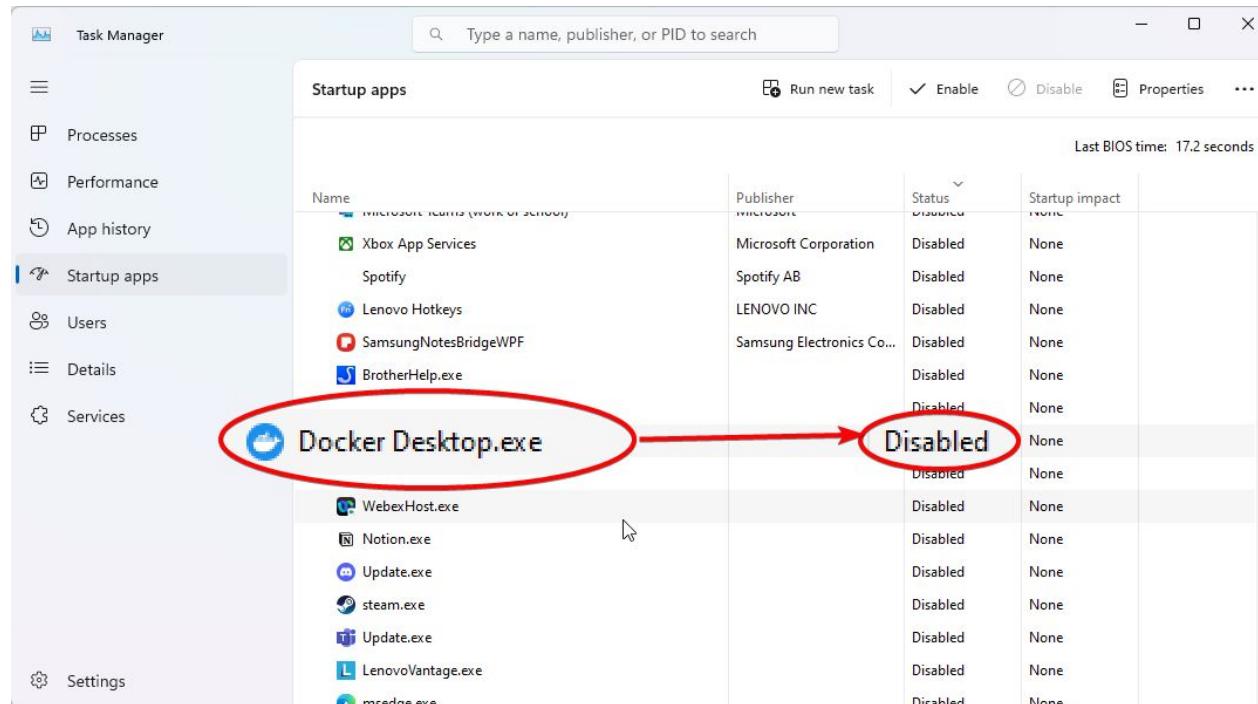


Prerequisite

- docker hub account
- docker
 - [Docker Desktop \(Windows\)](#)
 - [Install Docker Engine on Ubuntu](#)
 - | [Docker Docs](#)
- terminal (Optional)
 - [windows terminal](#)
 - command prompt

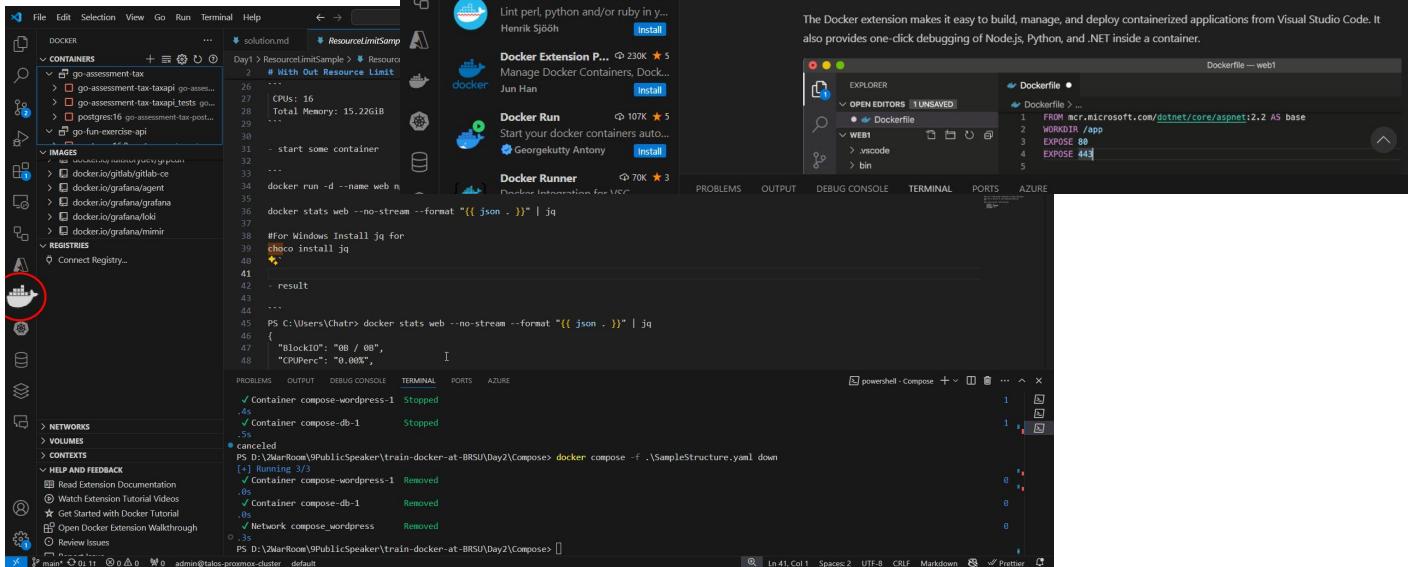


Prerequisite



Prerequisite

- Text Editor VS Code
 - [Docker extension for Visual Studio Code](#)
- Git (Optional)



Prerequisite

- How to Install Chocolatey

The screenshot shows the 'How to Install Chocolatey' page on the chocolatey.org website. At the top, there are two bullet points: 'PowerShell v2+' and '.NET Framework 4.8'. Below this, a section titled '1. Choose How to Install Chocolatey:' has a 'Individual' tab selected, along with tabs for 'Generic', 'Ansible', 'CHEF', 'PS DSC', and 'puppet'. A note below says 'Install Chocolatey for Individual Use:'. It lists two steps: 'First, ensure that you are using an [administrative shell](#) - you can also install as a non-admin, check out [Non-Administrative Installation](#).', and 'Install with powershell.exe'. A 'NOTE' box contains a warning about inspecting scripts from the internet. It then provides instructions for bypassing PowerShell execution policy restrictions, including running `Get-ExecutionPolicy` and `Set-ExecutionPolicy Bypass -Scope Process`. It also shows a command-line example of how to run the command. Finally, it lists five steps for installation: 1. Copy the command, 2. Paste it into a shell, 3. Press Enter, 4. Wait for completion, and 5. Check for errors. At the bottom, it says 'Chocolatey Licensed Install:' and includes a cookie consent banner.

• PowerShell v2+ (minimum is v3 for install from this website due to [TLS 1.2 requirement](#))
• .NET Framework 4.8 (the installation will attempt to install .NET 4.8 if you do not have it installed)

1. Choose How to Install Chocolatey:

Generic Individual Ansible CHEF PS DSC puppet

Install Chocolatey for Individual Use:

1. First, ensure that you are using an [administrative shell](#) - you can also install as a non-admin, check out [Non-Administrative Installation](#).

2. Install with powershell.exe

NOTE

Please inspect <https://community.chocolatey.org/install.ps1> prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of [any](#) script from the Internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine. We take security very seriously. [Learn more about our security protocols](#).

With PowerShell, you must ensure `Get-ExecutionPolicy` is not Restricted. We suggest using `Bypass` to bypass the policy to get things installed or `AllSigned` for quite a bit more security.

- Run `Get-ExecutionPolicy`. If it returns `Restricted`, then run `Set-ExecutionPolicy AllSigned` or `Set-ExecutionPolicy Bypass -Scope Process`.

Now run the following command:

```
> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.Web
```

3. Paste the copied text into your shell and press Enter.

4. Wait a few seconds for the command to complete.

5. If you don't see any errors, you are ready to use Chocolatey! Type `choco` or `choco -?` now, or see [Getting Started](#) for usage instructions.

Chocolatey Licensed Install:

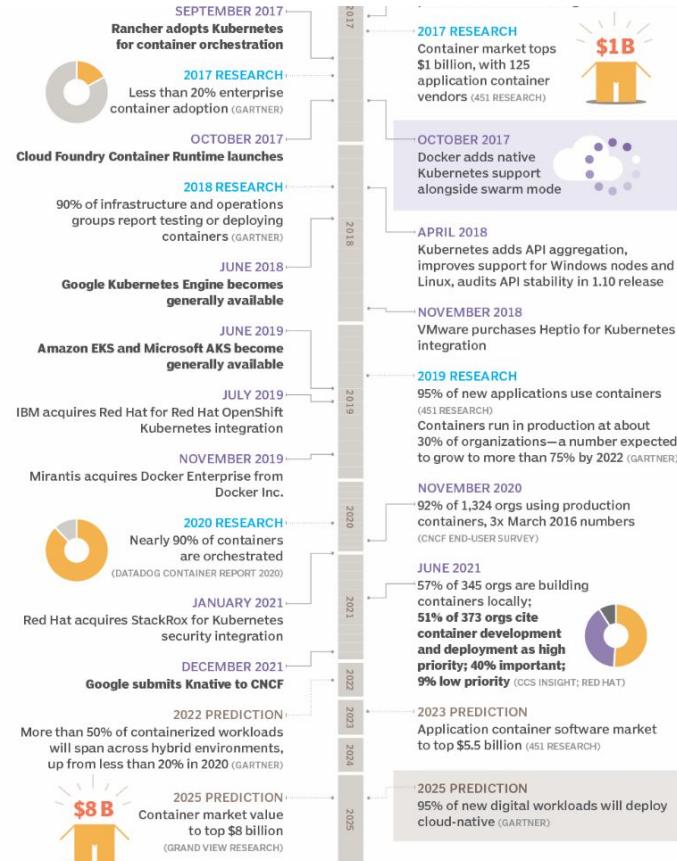
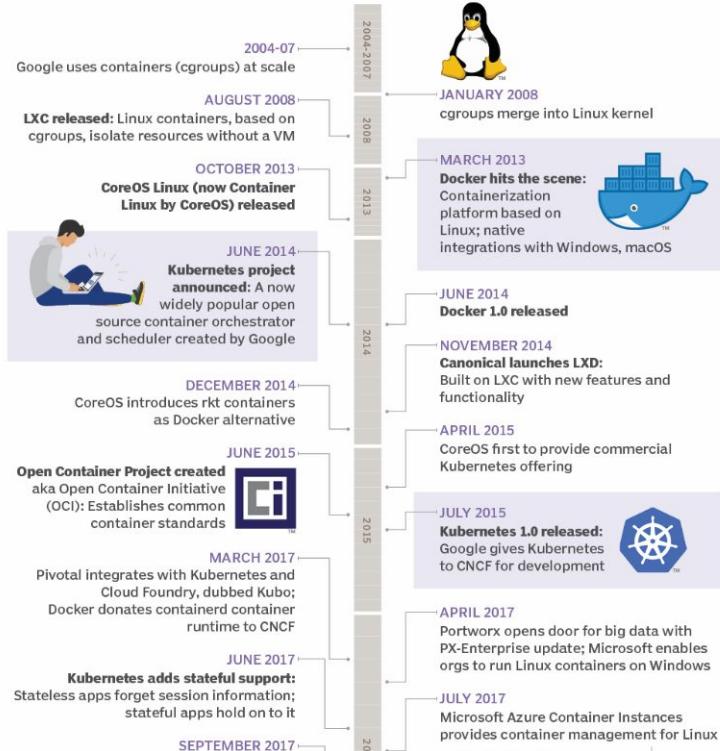
chocolatey.org uses cookies to enhance the user experience of the site. [I accept](#)

Prerequisite

- docker
 - [Docker Desktop \(Windows\)](#)
 - [Install Docker Engine on Ubuntu](#)
 - | [Docker Docs](#)
- docker hub account
- Test Editor VS Code
- Git

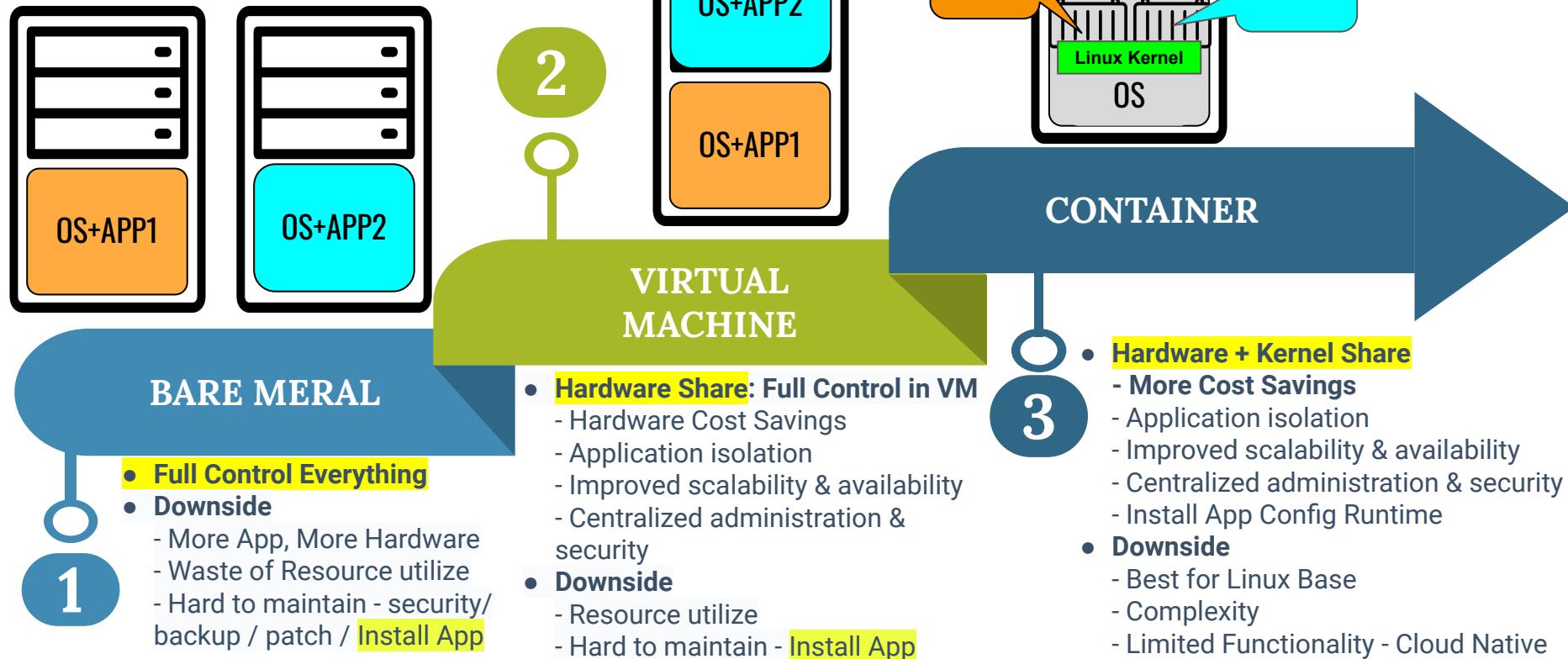
The evolution of containers

Container technology has come a long way from its chroots, starting with Google's exploration into cgroups and working up into widespread organizational adoption.



Ref: [The evolution of containers: Docker, Kubernetes and the future | TechTarget](#)
[/ Something Missed? : History of Container Technology | by Ozan Eren | Medium](#)

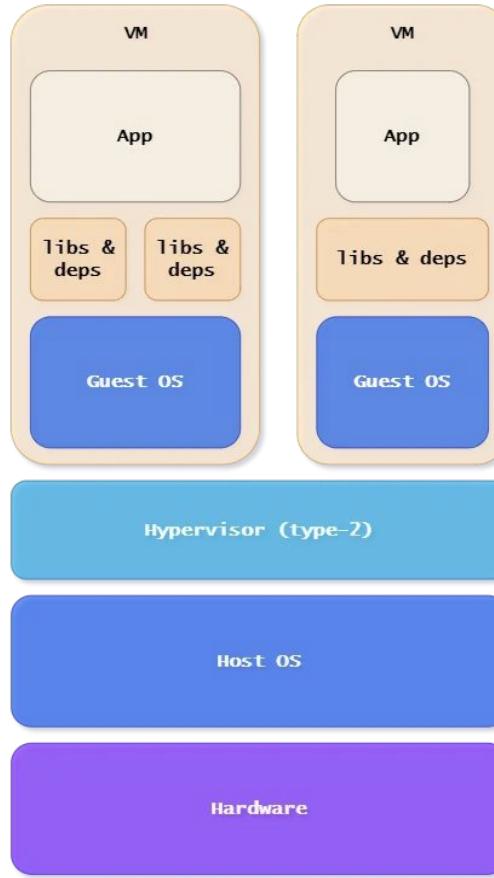
Why Container ?



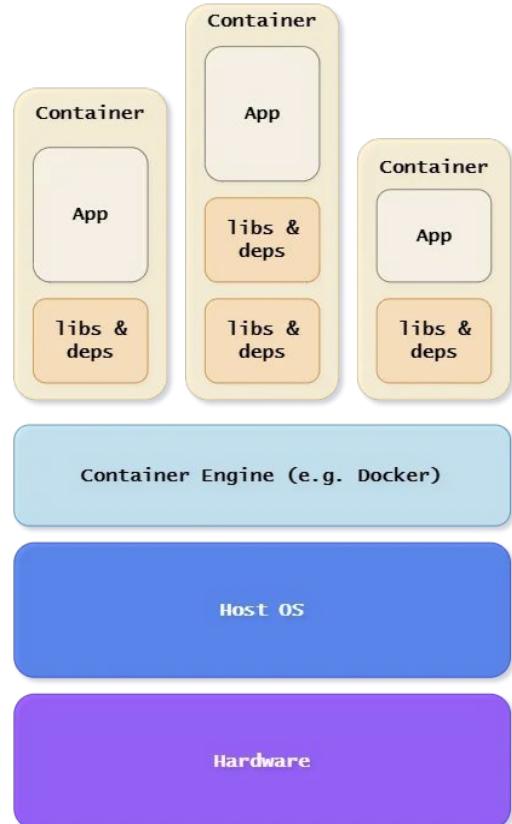
Why Container ?



BARE METAL



VIRTUAL MACHINE



CONTAINERS

Container Runtime

High Level /
Low Level



podman



Linux Container (LXC)



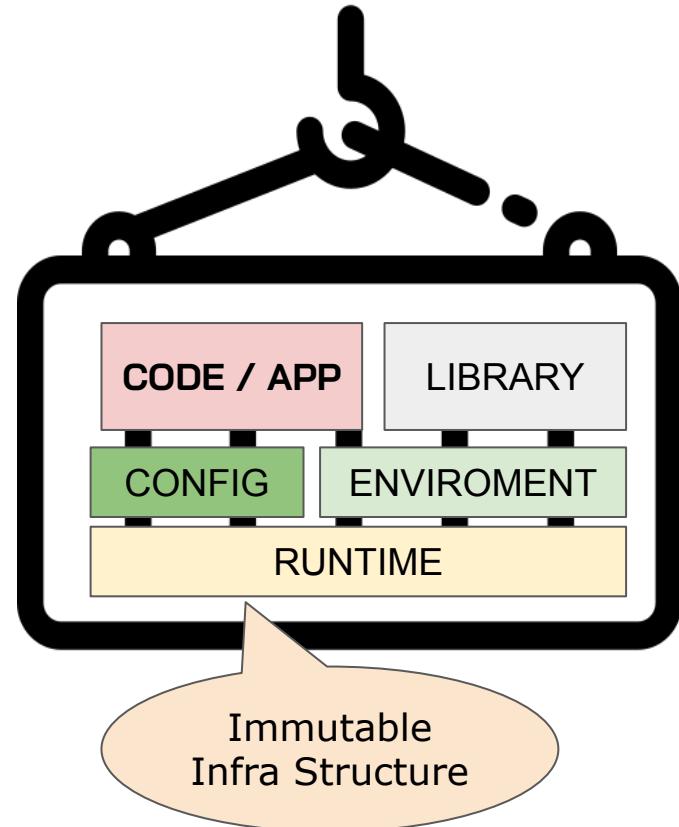
containerd

What is a Container ?

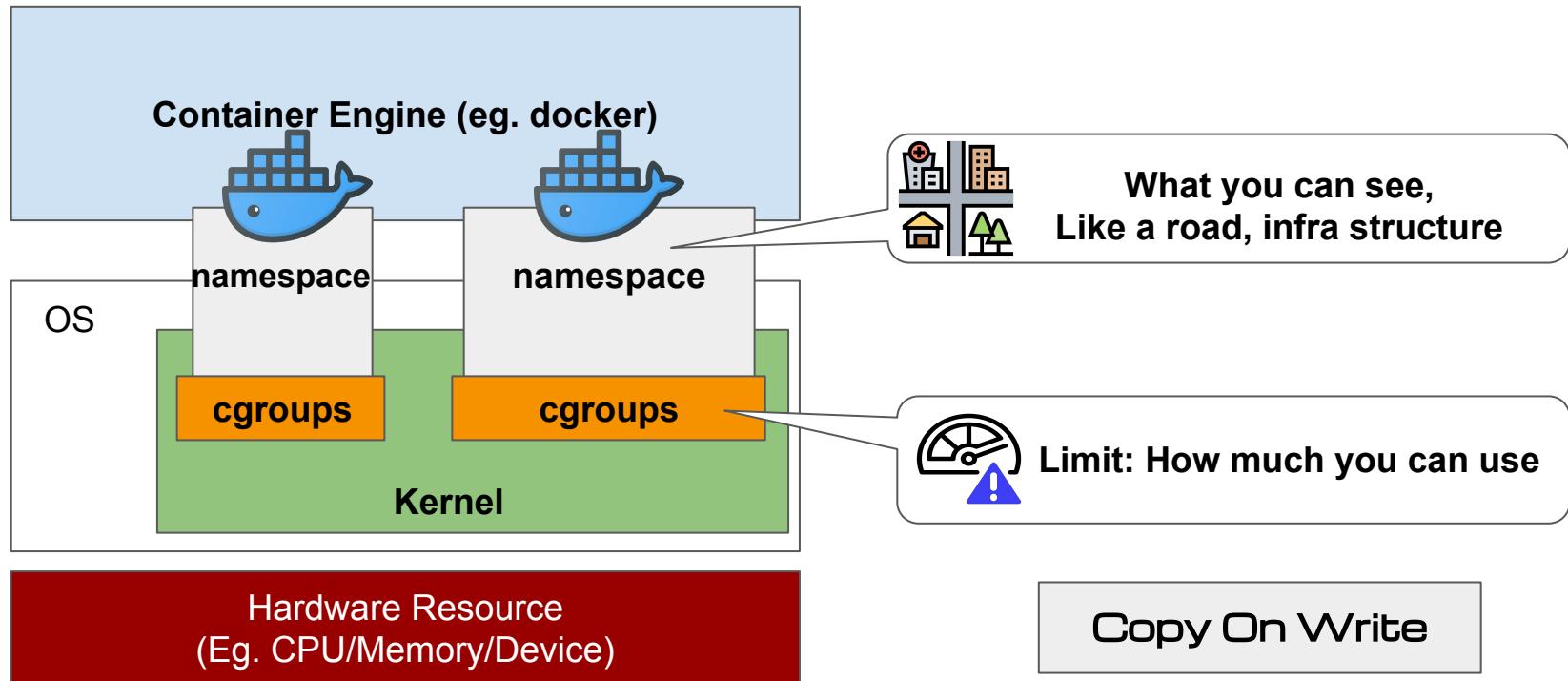
Containers are packages of software that contain all of the necessary elements to run in any environment

Benefits of using Containers

- Efficiency
- Application Isolation - and its dependencies from the host system
- Portability
- Separation of Responsibility - Dev + Ops / Create / Deploy
- Faster Application Development - CI / CD
- Easy Scaling

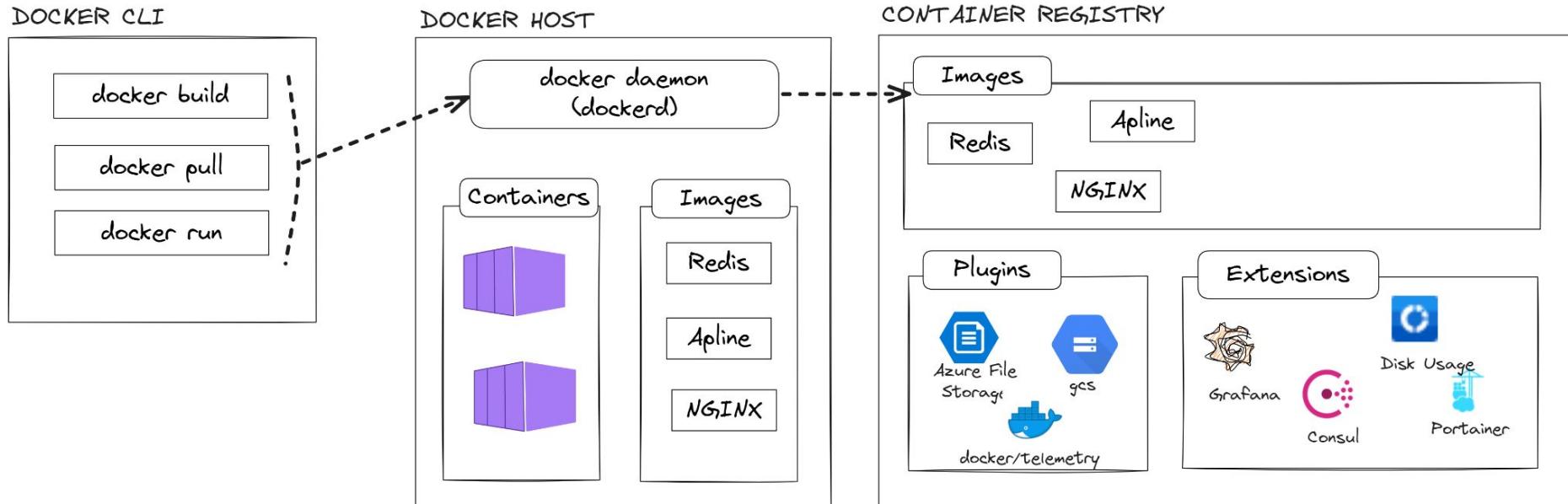


Magic of Container, Why Linux ?



Ref: [linux - difference between cgroups and namespaces - Stack Overflow](#)

Docker Architecture and Components



Docker Architecture

- **Docker Client (cli)**- Send Command via TCP (2375)
- **Docker Host** - Server Docker (เครื่องที่ลง docker)
 - Docker Deamon (dockerd) - process for handle request from docker cli eg.
 1. Pull Image from docker regiistry (If Notcvexist)
 - 2.Create Container
 - ...
 - Image - Local Storage
- **Container Registry** (eg. docker hub / RedHat Quay)
 - Share Image / Free / Not Free (Private Repo)
 - Increase Automation
 - Other - Share Extension / Plugin



Search Docker Hub

ctrl+K



Sign In

Sign up

Explore / Official Images / postgres



postgres

Docker Official Image • 1B+ • 10K+

The PostgreSQL object-relational database system provides reliability and data integrity.

docker pull postgres



Overview Tags

Sort by

Newest

Filter Tags



TAG

12.18-bullseye

Last pushed 2 days ago by dojank

docker pull postgres:12.18-bull...

DIGEST

2301be069271

9da6f511affc

87a3c60f040c

+5 more...

OS/ARCH

linux/386

linux/amd64

linux/arm/v5

VULNERABILITIES

2	32	14	41	0
---	----	----	----	---

2	31	14	41	0
---	----	----	----	---

2	32	14	41	0
---	----	----	----	---

COMPRESSED SIZE

135.97 MB

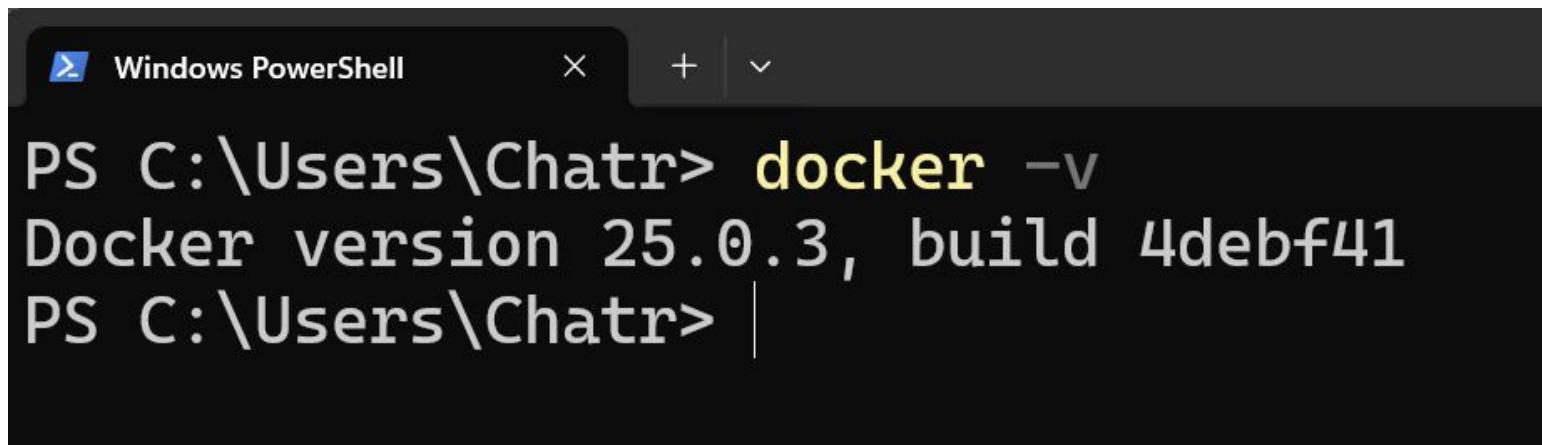
133.53 MB

126.73 MB

Basic Docker Command

Check Version

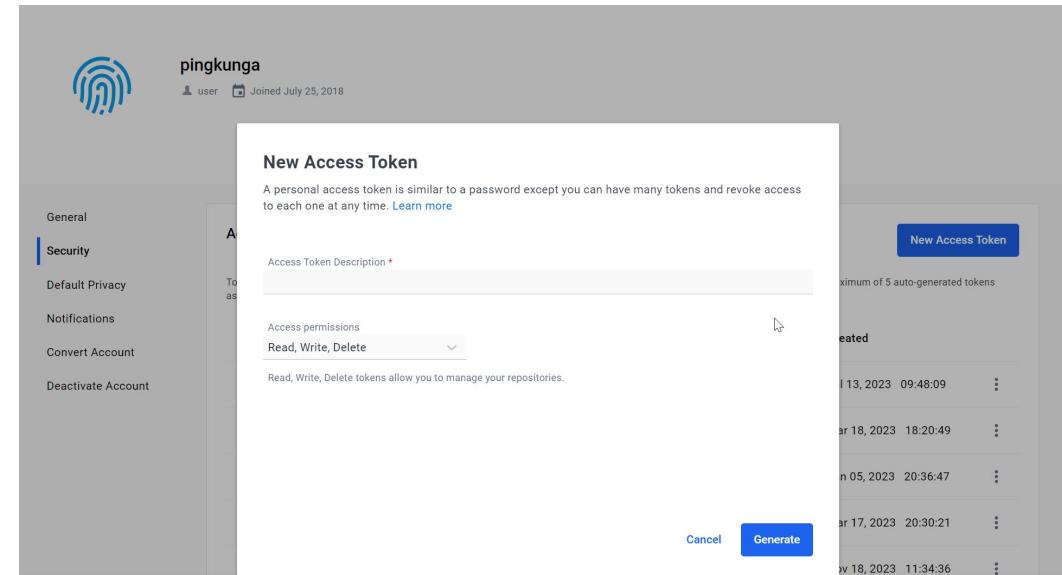
```
docker -v
```

A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "docker -v" is entered, and the output shows the Docker version as 25.0.3. The window has a dark theme with white text and a light gray header bar.

```
PS C:\Users\Chatr> docker -v
Docker version 25.0.3, build 4debff41
PS C:\Users\Chatr>
```

docker login

- Login docker hub
 - Generate token first :
<https://hub.docker.com/settings/security>
 - Keep Token
 - docker login with user + token



```
docker login  
username: <docker username>  
password: <Token>
```

A screenshot of a Windows PowerShell window. The title bar says 'Windows PowerShell'. The command 'PS C:\Users\Chatr> docker login' is typed in, followed by 'Authenticating with existing credentials...'. The final output is 'Login Succeeded'. The background of the slide has a red footer bar at the bottom.

docker logout

- Logout from docker hub

docker logout

ปกติไม่ค่อยได้ใช้ รอ Session หมดมาก
กว่า หรือไม่ไป revoke token

docker login / logout

another registry eg. Nexus / gchr.io / ACR

```
docker login <your_registry_url>
username: <your_username>
password: <Token>
```

```
docker logout <registry_url>
```

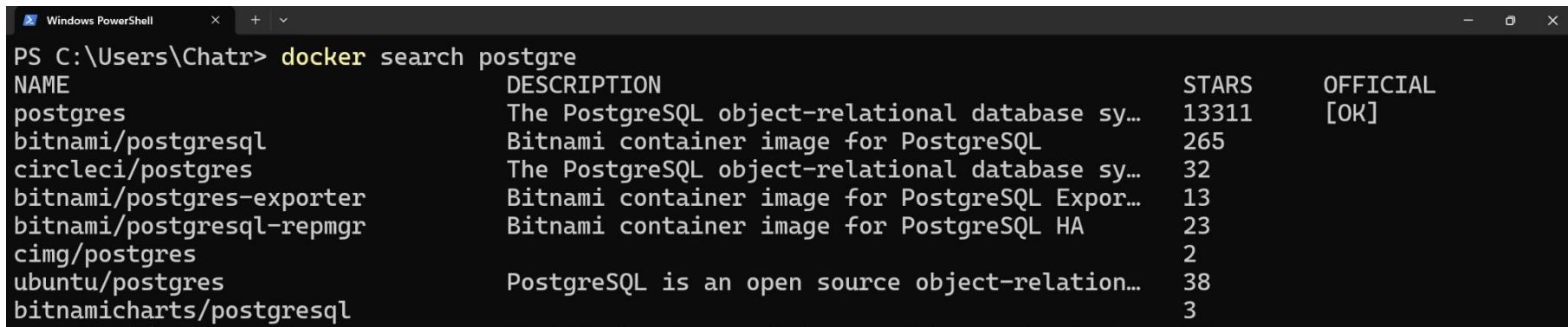
Nexus: [sonatype/nexus3 - Docker Image | Docker Hub](#)

docker search

- Search Image

ปกติไม่ค่อยได้ใช้เหมือนกัน
ดูจาก hub เอาจากกว่า

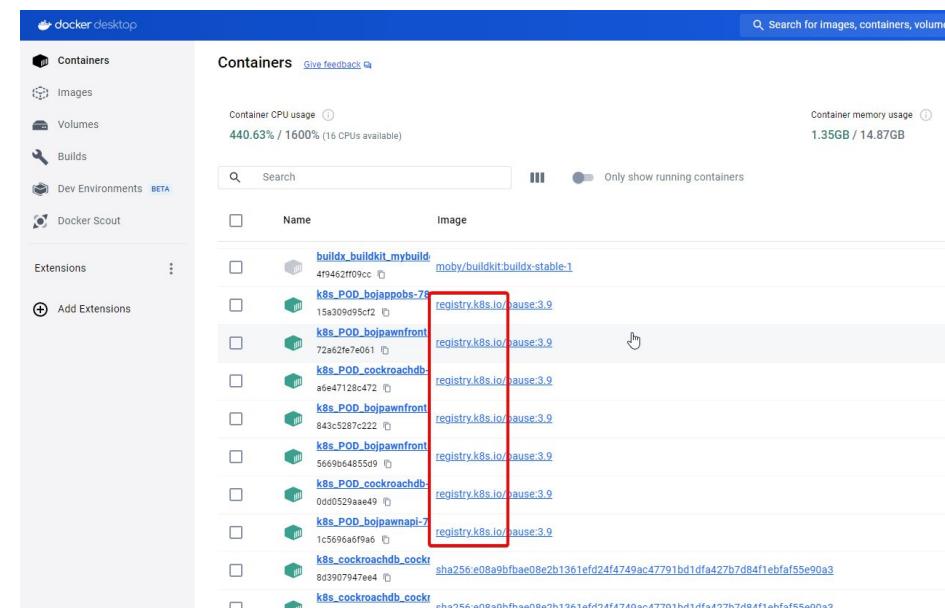
```
docker search <KEYWORD>
```



```
PS C:\Users\Chatr> docker search postgres
NAME                           DESCRIPTION                                     STARS      OFFICIAL
postgres                       The PostgreSQL object-relational database sy... 13311      [OK]
bitnami/postgresql              Bitnami container image for PostgreSQL           265
circleci/postgres               The PostgreSQL object-relational database sy... 32
bitnami/postgres-exporter       Bitnami container image for PostgreSQL Export... 13
bitnami/postgresql-repmgr      Bitnami container image for PostgreSQL HA        23
cimg/postgres                   PostgreSQL is an open source object-relati... 2
ubuntu/postgres                 PostgreSQL is an open source object-relati... 38
bitnamicharts/postgresql       PostgreSQL is an open source object-relati... 3
```

Container Image Name Pattern

- Pattern



Explore Docker's Container Image Repository | Docker Hub

The screenshot shows the Docker Hub interface for the PostgreSQL image. At the top, there is a blue header bar with the Docker Hub logo, a search bar containing "Search Docker Hub", and various navigation links including "ctrl+K", "?", "Sign In", and "Sign up". Below the header, the URL "Explore / Official Images / postgres" is visible. The main content area features the PostgreSQL logo and the text "postgres" followed by "Docker Official Image", "1B+", and "10K+ stars". A brief description states: "The PostgreSQL object-relational database system provides reliability and data integrity." To the right, there is a button labeled "docker pull postgres" with a copy icon. Below this, there are two tabs: "Overview" and "Tags", with "Tags" being the active tab. Under "Tags", there is a "Sort by" dropdown set to "Newest", a "Filter Tags" input field, and a search icon. A section titled "TAG" lists "12.18-bullseye" as the latest tag, pushed 2 days ago by user "doijank". Below this, there are three more tags: "2301be069271", "9da6f151affc", and "87a3c60f040c", each with its corresponding OS/ARCH and vulnerability statistics. A "Pattern" box is overlaid on the left side. At the bottom, there are two input fields: one containing "registry_name/imageName:Tag" and another containing "imageName:Tag", both with a Docker ship icon.

DIGEST	OS/ARCH	VULNERABILITIES	COMPRESSED SIZE
2301be069271	linux/386	2 32 14 41 0	135.97 MB
9da6f151affc	linux/amd64	2 31 14 41 0	133.53 MB
87a3c60f040c	linux/arm/v5	2 32 14 41 0	126.73 MB

Pattern

registry_name/imageName:Tag

imageName:Tag

docker pull

- Download Image from Container Registry to Local

```
docker pull <registry_url><image_name>:<tag>
```



```
docker pull <image_name>:<tag>
```

```
docker pull postgres:16.2-bookworm
```

```
docker pull redis
```

```
docker pull registry.k8s.io/kube-apiserver:v1.29.2
```

```
docker pull yourcompany.com/invs-apiserver:8.23.1
```

```
docker pull ghcr.io/someorg/someimage:sometag
```

docker pull - image type

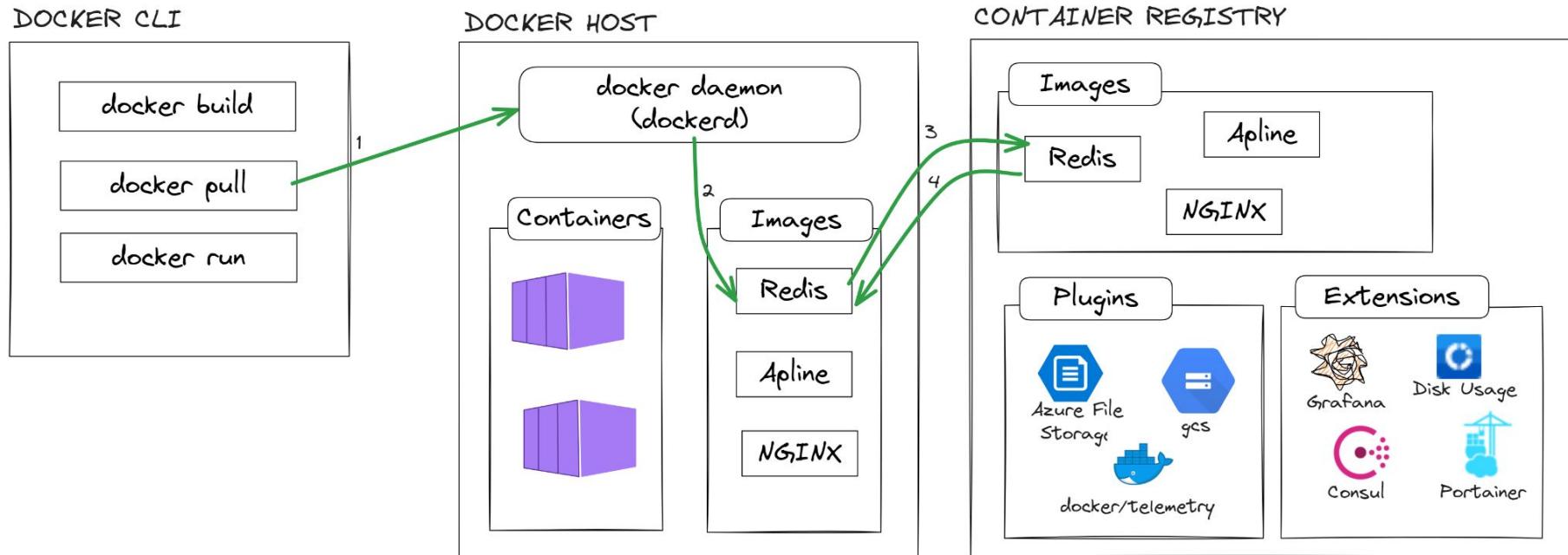
```
docker pull postgres:16.2-bookworm
```

Base Image:

- Alpine
- [bookworm \(ubuntu\)](#)
- Slim vs Minimal
- Secured by Design: [distroless](#) / [chiselled](#) / [chainguard](#)
- etc.

Docker Pull : How it works

docker pull redis



docker scout

- A tool analyzes image contents and generates a detailed report of packages and vulnerabilities that it detects. It can provide you with suggestions for how to remediate issues discovered by image analysis.

```
docker scout quickview <image_name:tag>
docker scout quickview mongo-express:latest
```

What's Next?

View a summary of image vulnerabilities and recommendations → `docker scout quickview mongo-express:latest`
PS C:\Users\Chatr> `docker scout quickview mongo-express:latest`
i New version 1.8.0 available (installed version is 1.6.3) at <https://github.com/docker/scout-cli>
v Image stored for indexing
v Indexed 593 packages

Target

digest

Base image

Updated base image

`mongo-express:latest`

`870141b735e7`

`node:18-alpine3.18`

`node:22-alpine`

1C

3H

7M

1L

2?

0C

0H

1M

0L

2?

0C

0H

4M

0L

2?

+3

What's Next?

View vulnerabilities → `docker scout cves mongo-express:latest`

View base image update recommendations → `docker scout recommendations mongo-express:latest`

Ref:

- [Docker Scout | Docker Docs](#)
- [docker/scout-cli: Docker Scout CLI \(github.com\)](#)

docker scout - Docker Desktop

The screenshot shows the Docker Desktop interface with the Docker Scout extension active. The main window displays the 'mongo-express:latest' image details, including its image hierarchy and layers. A large green arrow points from the 'View packages and CVEs' button in the Docker Desktop UI to the 'Vulnerabilities (14)' tab in the Docker Scout sidebar.

Image hierarchy:

- FROM alpine:3.18, 3.18.6
- FROM node:18-alpine3.18, 18.20-alpine3.18, 18.20.3-alpine3.18, hydrogen-alpine3.18
- ALL mongo-express:latest

Layers (19):

- 0 ADD file:8729f9c0258836b640e9e789c7ab029cf4547e0596557d54dd4a4d7d8... 7.34 MB
- 1 CMD ['./bin/sh'] 0 B

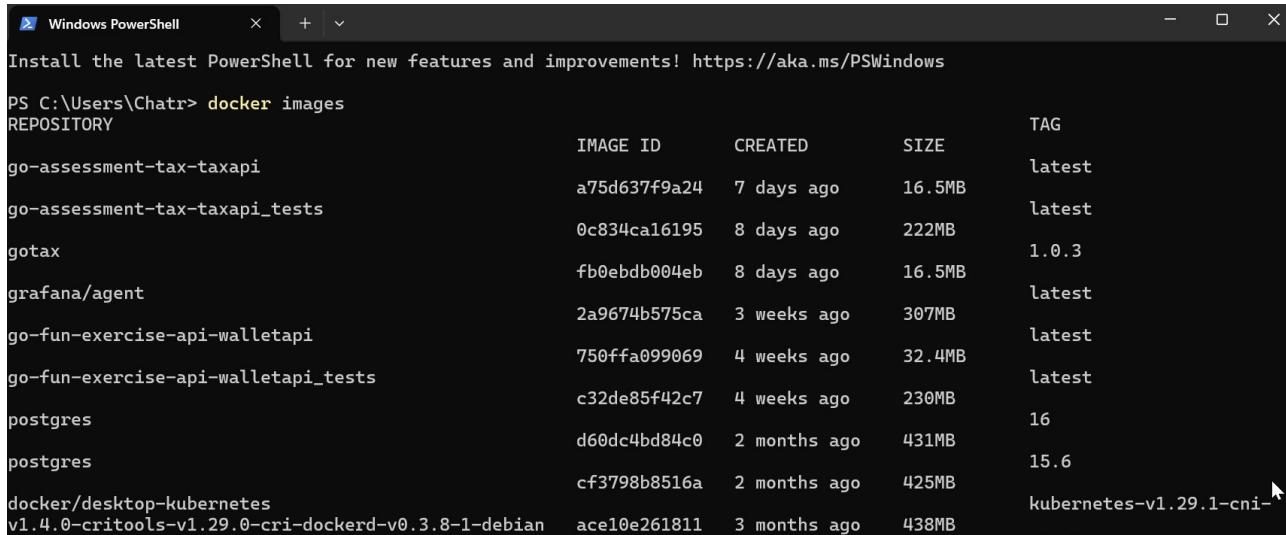
Vulnerabilities (14):

Package	Vulnerabilities
@babel/traverse 7.19.6	1 0 0 0 0
mongo-express 1.0.2	0 1 1 0 0
fast-xml-parser 4.0.11	0 1 1 0 0
json5 2.2.1	0 1 0 0 0
semver 6.3.0	0 0 1 0 0
ip 2.0.0	0 0 1 0 0
express 4.18.2	0 0 1 0 0
alpine/busybox 1.36.1-r5	0 0 1 0 0
es5-ext 10.6.2	0 0 0 1 0

docker images

- List image at local

```
docker images  
docker images --no-trunc // Show full image id
```



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "PS C:\Users\Chatri> docker images" is run, followed by "REPOSITORY" and then a list of Docker images with their details. The table includes columns for REPOSITORY, IMAGE ID, CREATED, SIZE, and TAG. A mouse cursor is hovering over the last entry.

REPOSITORY	IMAGE ID	CREATED	SIZE	TAG
go-assessment-tax-taxapi	a75d637f9a24	7 days ago	16.5MB	latest
go-assessment-tax-taxapi_tests	0c834ca16195	8 days ago	222MB	latest
gotax	fb0ebdb004eb	8 days ago	16.5MB	1.0.3
grafana/agent	2a9674b575ca	3 weeks ago	307MB	latest
go-fun-exercise-api-walletapi	750ffa099069	4 weeks ago	32.4MB	latest
go-fun-exercise-api-walletapi_tests	c32de85f42c7	4 weeks ago	230MB	latest
postgres	d60dc4bd84c0	2 months ago	431MB	16
postgres	cf3798b8516a	2 months ago	425MB	15.6
docker/desktop-kubernetes_v1.4.0-critools-v1.29.0-cri-dockerd-v0.3.8-1-debian	ace10e261811	3 months ago	438MB	kubernetes-v1.29.1-cni-1

docker images - docker desktop

The screenshot shows the Docker Desktop interface with the 'Images' tab selected. The sidebar on the left includes 'Containers', 'Images' (which is highlighted with a red border), 'Volumes', 'Builds', 'Dev Environments BETA', and 'Docker Scout'. The main area displays 97 images across 5.35 GB of used space (out of 26.18 GB). A search bar and a refresh indicator ('Last refresh: 7 hours ago') are at the top. A 'Delete' button with a note 'Space to be reclaimed 9.11 MB' is visible. The table lists images with columns for Name, Tag, Status, Created, Size, and Actions.

Name	Tag	Status	Created	Size	Actions
go-assessment-tax-taxapi	latest	In use	8 days ago	16.49 MB	
go-assessment-tax-taxapi_tests	latest	In use	8 days ago	221.74 MB	
0c34ca16195	latest	In use	8 days ago	221.74 MB	
go-fun-exercise-api-walletapi	latest	In use	1 month ago	32.35 MB	
750ffa099069	latest	In use	1 month ago	32.35 MB	
go-fun-exercise-api-walletapi_tests	latest	In use	1 month ago	230.13 MB	
c32de85f42c7	latest	In use	1 month ago	230.13 MB	
postgres	16	In use	3 months ago	431.42 MB	
d60dc4bd84c0	16	In use	3 months ago	431.42 MB	
postgres	15.6	In use	3 months ago	425.44 MB	
cf3798b8516a	15.6	In use	3 months ago	425.44 MB	
vsc-cosmos-db-dotnet-bc3f69b7e5601a14c0f65ed684a78b528b6e28671b3fc4c	latest	In use	4 months ago	1.16 GB	
9087e29b3b3c	latest	In use	4 months ago	1.16 GB	
vsc-volume-bootstrap	latest	In use	4 months ago	783.76 MB	
d15c2b95bf4b	latest	In use	4 months ago	783.76 MB	
grafana/tempo	latest	In use	5 months ago	104.43 MB	
6edb4b741c24	latest	In use	5 months ago	104.43 MB	
grafana/grafana	latest	In use	6 months ago	399.09 MB	
066d559b720d	latest	In use	6 months ago	399.09 MB	
grafana/mimir	latest	In use	6 months ago	69.23 MB	
8ecbe325755	latest	In use	6 months ago	69.23 MB	
dpage/pgadmin4	latest	In use	8 months ago	544.75 MB	
ce8ae42da4a0	latest	In use	8 months ago	544.75 MB	

Selected 1 of 97

Engine running RAM 6.05 GB CPU 10.47% Signed in v4.29.0

docker history

- Show the history of an image

```
docker history <repo_url><image_name>:<image_tag>
```

```
docker history jenkins:2.440.3
```

```
docker history --no-trunc 128.1.0.12:5000/jenkins/jenkins-ds:2.440.3
```

IMAGE	CREATED	CREATED BY	SIZE	COMMENT
62e79e9f9139	3 hours ago	USER jenkins	0B	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c dotnet nuget add source "http...	4.25kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cp \$JAVA_HOME/lib/security/ne...	217kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd \$JAVA_HOME/lib/security &...	171kB	buildkit.dockerfile.v0
<missing>	3 hours ago	ADD certs/nexus.crt /opt/java/openjdk/lib/se...	1.38kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd /opt && tar -xvf OpenJDK21...	360MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd /opt && tar -xvf OpenJDK17...	330MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd /opt && tar -xvf OpenJDK11...	321MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd /opt && tar -xvf OpenJDK8U...	204MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c cd /opt && dpkg -i jdk-11.0.4...	304MB	buildkit.dockerfile.v0
<missing>	3 hours ago	ADD ./jdk /opt/ # buildkit	1.41GB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c apt-get update && apt-get ins...	509MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c rm packages-microsoft-prod.de...	0B	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c dpkg -i packages-microsoft-pr...	553kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c wget https://packages.microso...	3.31kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c usermod -u 1001 -g 999 jenkin...	2.9kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c groupadd docker && usermod -...	3.91kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c apt-get update -y && apt-get ...	186MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c add-apt-repository "deb [arch...	15.1MB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c curl -fsSL https://download.d...	2.76kB	buildkit.dockerfile.v0
<missing>	3 hours ago	RUN /bin/sh -c apt-get update -y && apt-get ...	247MB	buildkit.dockerfile.v0
<missing>	3 hours ago	USER root	0B	buildkit.dockerfile.v0
<missing>	5 weeks ago	LABEL ora.opencontainers.image.vendor=Jenkin...	0B	buildkit.dockerfile.v0

docker run - start container / docker create

- Create Container + Run

```
docker run [options] <image_name>:<tag>
```

- Create Container Only !!

```
docker create [options] <image_name>:<tag>
```

[options]

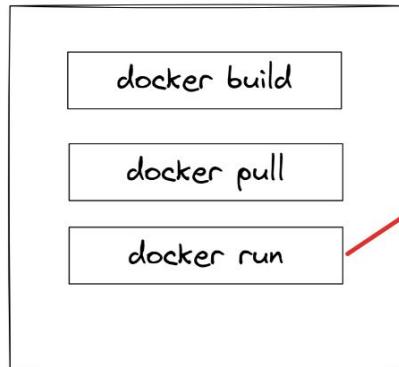
- p <host>:<container port> mapping port
- d background mode (**docker run only**)
- v mount data between Host / Docker
- n set container name / default random
- e <key:value> set environment variable
- network <network_name> set container network
- it interactive mode
- rm auto remove when stop
- label your.key=value
- restart containers restart policy

เราจะรู้ได้ยังไงว่าต้องใส่อะไรบ้าง ?

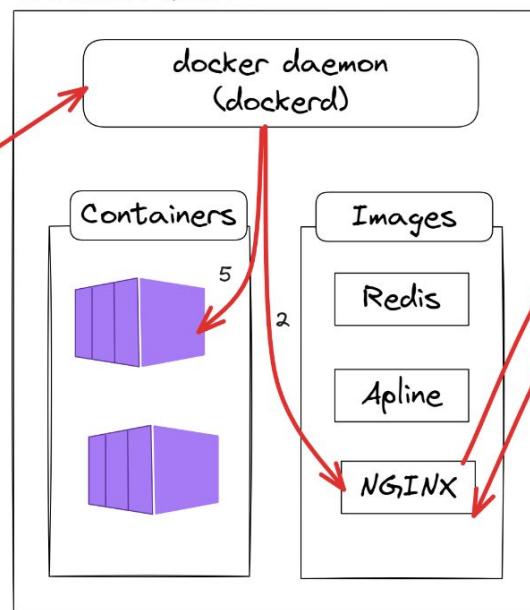
docker run: How it works

```
docker run --rm -d -p 8080:80 --name web nginx
```

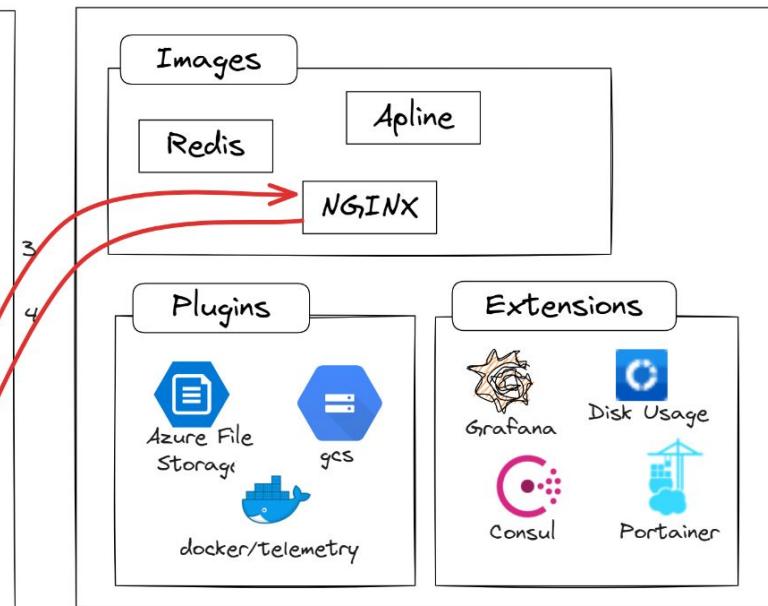
DOCKER CLI



DOCKER HOST



CONTAINER REGISTRY



docker run: How it works

```
docker run --rm -d -p 8080:80 --name web nginx
```



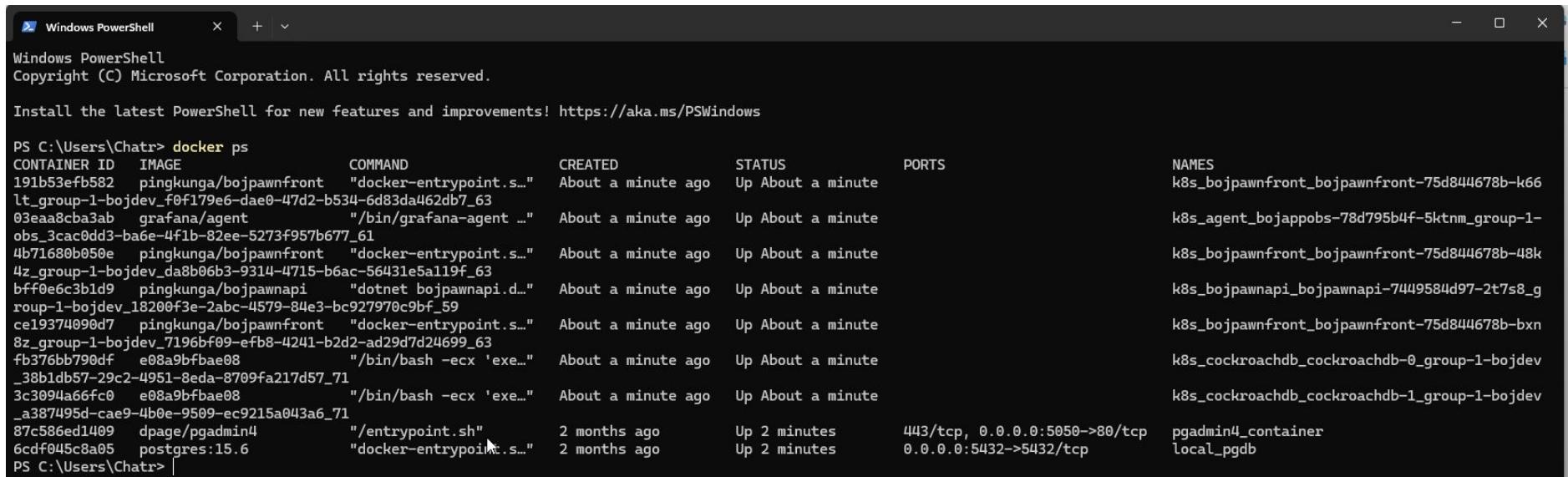
docker ps - list container

- List Running Container

```
docker ps
```

- List All Container (Run and Stop)

```
docker ps -a
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

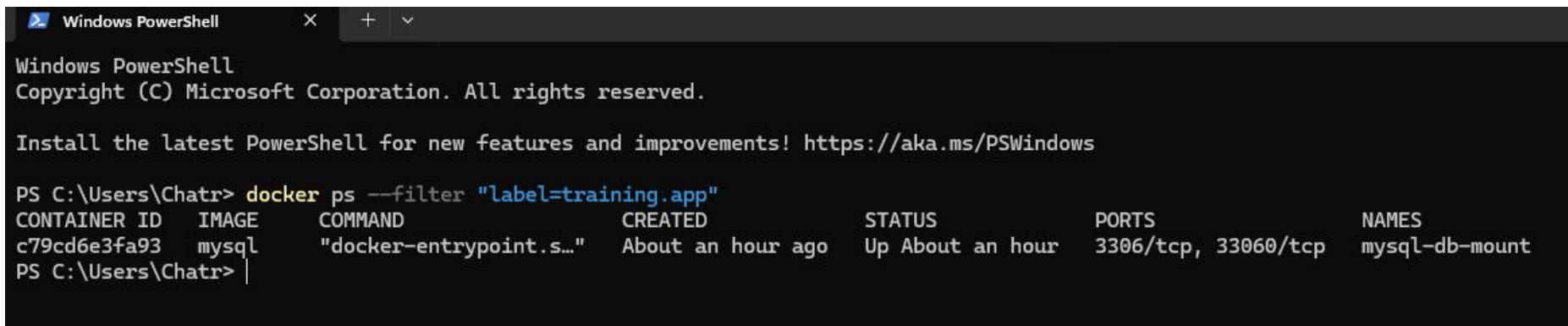
PS C:\Users\Chatr> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
191b53efb582   pingkunga/bojpawnfront   "docker-entrypoint.s..."   About a minute ago   Up About a minute
lt_group-1-bojdev_f0f179e6-dae0-47d2-b534-6d83da462db7_63
03eaa8cba3ab   grafana/agent           "/bin/grafana-agent ..."   About a minute ago   Up About a minute
obs_3cac0dd3-ba6e-4f1b-82ee-5273f957b677_61
4b71680b050e   pingkunga/bojpawnfront   "docker-entrypoint.s..."   About a minute ago   Up About a minute
4z_group-1-bojdev_da8b06b3-9314-4715-b6ac-56431e5a119f_63
bfff0e6c3b1d9   pingkunga/bojpawnapi   "dotnet bojpawnapi.d..."   About a minute ago   Up About a minute
roup-1-bojdev_18200f3e-2abc-4579-84e3-bc927970c9bf_59
ce19374090d7   pingkunga/bojpawnfront   "docker-entrypoint.s..."   About a minute ago   Up About a minute
8z_group-1-bojdev_7196bf09-efb8-4241-b2d2-ad29d7d24699_63
fb376bb790df   e08a9fbfae08         "/bin/bash -ecx 'exe..."   About a minute ago   Up About a minute
_38b1db57-29c2-4951-8eda-8709fa217d57_71
3c3094a66fc0   e08a9fbfae08         "/bin/bash -ecx 'exe..."   About a minute ago   Up About a minute
_a387495d-cae9-4b0e-9509-ec9215a043a6_71
87fc586ed1409  dpage/pgadmin4       "/entrypoint.sh"        2 months ago      Up 2 minutes    443/tcp, 0.0.0.0:5050->80/tcp pgadmin4_container
6cdf045c8a05   postgres:15.6       "docker-entrypoi...t.s..."   2 months ago      Up 2 minutes    0.0.0.0:5432->5432/tcp local_pgdb

PS C:\Users\Chatr>
```

docker ps - list container

- List Container by Label

```
docker ps --filter "label=training.app"
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Chatr> docker ps --filter "label=training.app"
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
c79cd6e3fa93      mysql              "docker-entrypoint.s..."   About an hour ago   Up About an hour   3306/tcp, 33060/tcp   mysql-db-mount
PS C:\Users\Chatr>
```

Ref: [docker ps --filter | Docker Docs](#)

docker ps - docker desktop

The screenshot shows the Docker Desktop interface with the 'Containers' tab selected. The sidebar on the left includes options for Containers, Images, Volumes, Builds, Dev Environments (Beta), Docker Scout, Extensions, and Add Extensions. The main area displays container statistics: Container CPU usage (0.00% / 1600%) and Container memory usage (0B / 14.87GB). A search bar and a filter for 'Only show running containers' are also present. The table lists 33 items, showing columns for Name, Image, Status, CPU (%), Port(s), Last started, and Actions.

Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
bf00e6c3b1d9						
k8s_bojapawnfront_bojapawnfront-75d844678b-48k4z_group-1-4b71680b050e	pingkunga/bojapawnfront	Running	0%	51 seconds ago		
k8s_agent_bojapapps-78d795b4f-5ktnm_group-1-obs_3cac0c	grafana/agent	Running	0%	49 seconds ago		
191b53efb582	pingkunga/bojapawnfront	Running	0%	46 seconds ago		
go-assessment-tax		Exited	0%	8 days ago		
go-fun-exercise-api		Exited	0%	1 month ago		
postgres-1	postgres:16.0	Exited	0%	5432:5432	1 month ago	
walletapi_tests-1	go-fun-exercise-api-walletapi_tests	Exited	0%	1 month ago		
walletapi-1	go-fun-exercise-api-walletapi	Exited (2)	0%	1323:1323	1 month ago	
db-1	postgres:16.0	Exited	0%	5432:5432	1 month ago	
postresql		Running (2/2)	0%	2 minutes ago		
pgadmin4_container	dpage/pgadmin4	Running	0%	5050:80	2 minutes ago	
local_pgdb	postgres:15.6	Running	0%	5432:5432	2 minutes ago	

Showing 33 items

v4.29.0

docker inspect - Get Container Info

```
docker inspect <container_name / container_id>
```



```
> docker inspect local_pgdb
[{"Id": "6cdf045c8a052994e693d2dfa7eed75f7ebe2491742f42df605edde5e3f78", "Created": "2024-02-14T08:47:29.151903864Z", "Path": "docker-entrypoint.sh", "Args": ["postgres"], "State": {"Status": "running", "Running": true, "Paused": false, "Restarting": false, "OOMKilled": false, "Dead": false, "Pid": 745, "ExitCode": 0, "Error": "", "StartedAt": "2024-05-06T14:16:12.666257801Z", "FinishedAt": "2024-05-06T14:16:11.335008508Z"}, "Image": "sha256:cf3798b8516a0a36744738d39cafd58ec26cb96fb88086ac0fae4004ef79a989", "ResolvConfPath": "/var/lib/docker/containers/6cdf045c8a052994e693d2dfa7eed75f7ebe2491742f42df605edde5e3f789/resolv.conf"}, {"LogConfig": {"Type": "json-file", "Config": {}}, "NetworkMode": "postgresql_default", "PortBindings": {"5432/tcp": [{"HostIp": "", "HostPort": "5432"}]}, "RestartPolicy": {"Name": "always", "MaximumRetryCount": 0}}].
```

docker inspect - Docker Desktop

The screenshot shows the Docker Desktop interface. On the left, the sidebar has 'Containers' selected (highlighted with a red box). The main area displays a search bar with the command 'docker inspect <container_name / container_id>' and a status message '0B / 14.87GB'. Below the search bar is a list of containers. A red arrow points from the search bar down to the inspect output window. The inspect output window shows the JSON structure of the local_pgdb container.

```
local_pgdb
postres:15.6
Logs Inspect Bind mounts Exec Files Stats
Platform Cmd State Image PortBindings Runtime Mounts Volumes Env Labels Networks
{
  "Id": "6cdf045c8a052994e693d2dfa7eed75f7eb2491742f42d2f605edde5e3f789",
  "Created": "2024-02-14T08:47:29.151903864Z",
  "Path": "docker-entrypoint.ah",
  "Args": [
    "postgres"
  ],
  "State": {
    "Status": "running",
    "Running": true,
    "Paused": false,
    "Restarting": false,
    "OOMKilled": false,
    "Dead": false,
    "Pid": 745,
    "ExitCode": 0,
    "Error": "",
    "StartedAt": "2024-05-06T14:16:12.666257801Z",
    "FinishedAt": "2024-05-06T14:16:11.335008508Z"
  },
  "Image": "sha256:c3799851ea36744739d3caf59ec26cb96fb88086acf0fae400fe79a999",
  "ResolveConfigPath": "/var/lib/docker/containers/6cdf045c8a052994e693d2dfa7eed75f7eb2491742f42d2f605edde5e3f789/resolv.conf",
  "HostnamePath": "/var/lib/docker/containers/6cdf045c8a052994e693d2dfa7eed75f7eb2491742f42d2f605edde5e3f789/hostname",
  "HostPath": "/var/lib/docker/containers/6cdf045c8a052994e693d2dfa7eed75f7eb2491742f42d2f605edde5e3f789/hosts",
  "LogPath": "/var/lib/docker/containers/6cdf045c8a052994e693d2dfa7eed75f7eb2491742f42d2f605edde5e3f789/json.log",
  "Name": "/local_pgdb",
  "RestartCount": 0,
  "Driver": "overlay2",
  "Platform": "linux",
  "MountLabel": "",
  "ProcessLabel": "",
  "AppArmorProfile": "",
  "ExecIDs": null,
  "HostConfig": {
    "Binds": null
  }
}
```

Docker101 by Chatri Ngambenewong

41

Hand-on 1

Hand-On1: Get your first Container

- Run PostgreSQL Container with requirement
 - version: 15.6 + small image
 - username: docker101
 - password: dockerPwd
 - expose: 5432
 - container name: pg_dev_test
 - run background mode
- Create NGINX contain with requirement
 - expose: 18080
 - container name: webapp
- Check Container Status & Inspect

PS C:\Users\Chatr> docker ps -a					
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
	NAMES				
c761f0d189fe	nginx	"/docker-entrypoint..."	22 seconds ago	Created	
7fb7ec1847f1	postgres:15.6-alpine3.19	"docker-entrypoint.s..."	48 minutes ago	Up 47 minutes	0.0.0.0:5432->5432/tcp
	pg_dev_test				

Hand-on 1

Solution: [GetYourFirstContainer.md at main · pingkunga/github.com](#)

Hand-On1: Get your first Container (Solution)

- Run PostgreSQL Container with requirement

- version: 15.6 + small image
- username: docker101
- password: dockerPwd
- expose: 5432
- container name: pg_dev_test
- run background mode

```
docker run -d -e POSTGRES_USER=docker101 -e POSTGRES_PASSWORD=dockerPwd  
-p 5432:5432 --name pg_dev_test postgres:15.6-alpine3.19
```

Hand-On1: Get you first Container (Solution)

- Create NGINX container with requirement
 - expose: 18080
 - container name: webapp
 - auto remove when container stop

```
docker create --rm -p 18080:80 --name webapp nginx
```

- Check Container Status & Inspect

```
docker ps -a  
  
#inspect  
docker inspect webapp  
docker inspect pg_dev_test
```

Basic Docker Command #02

docker exec - Access Container

```
docker exec -it <container_name / container_id> <command>
```

Run Command in a Container - such as create file / manage app / other operation etc.

```
docker exec -it pg_dev_test /bin/bash  
psql -U docker101  
docker101-# \l
```

Access PostgreSQL
(Hand-on1)

List of databases									
Name	Owner	Encoding	Collate	Ctype	ICU Locale	Locale Provider	Access privileges		
docker101	docker101	UTF8	en_US.utf8	en_US.utf8		libc			
postgres	docker101	UTF8	en_US.utf8	en_US.utf8		libc			
template0	docker101	UTF8	en_US.utf8	en_US.utf8		libc	=c/docker101	+	
						docker101=CTc/docker101			
template1	docker101	UTF8	en_US.utf8	en_US.utf8		libc	=c/docker101	+	
						docker101=CTc/docker101			

(4 rows)

```
docker101-# \q  
7fb7ec1847f1:/# exit
```

docker exec - Docker Desktop

The screenshot shows the Docker Desktop interface with the following details:

- Containers:** pg_dev_test (postgres:15.6-alpine3.19, 7fb7ec1847f1, 5432:5432)
- Image:** postgres:15.6-alpine3.19
- Status:** Running
- CPU (%):** 0.01%
- Ports:** 5432
- Actions:** View details, View image packages and CVEs, Copy docker run, Open in terminal (highlighted with a red box), View logs, Pause, Restart, Open with browser
- Resource usage:** Container CPU usage: 6.74% / 1600%, Container memory usage: 2.97GB / 14.87GB
- Logs:** Docker Debug brings the tools you need to debug your container with one click. Requires a paid Docker subscription. Learn more.
- Terminal:** / # ls
bin etc media proc sbin
dev home mnt root srv
docker-entrypoint-initdb.d lib opt sys tmp usr var
/ # pwd
/
/ # whoami
root
/ #

A large arrow points from the "Exec" button in the container details to the "Open in terminal" option in the context menu.

Inside docker

Docker desktop interface showing the file system of a running container named 'webapp'. The 'Files' tab is selected. The file system structure is as follows:

- ..
- .dockercfg
- bin -> usr/bin
- boot
- dev
- docker-entrypoint.d
- docker-entrypoint.sh
- etc
 - .pwd.lock
 - adduser.conf
 - alternatives
 - awk > Edit file
 - awk.1 Delete
 - awk.1.gz Save
 - builtins > /man/man7/bash-builtins.7.gz
 - awk > /usr/share/man/man1/mawk.1.gz
 - pager > /bin/more
 - pager.1.gz > /usr/share/man/man1/more.1.gz
 - README
 - rmt > /usr/sbin/rmt-tar
 - rmt.8.gz > /usr/share/man/man8/rmt-tar.8.gz
 - which > /usr/bin/which.debianutils
 - which.1.gz > /usr/share/man/man1/which.deb
 - which.de1.gz > /usr/share/man/de/man1/which
 - awk.1.gz -> /usr/share/man/man1/mawk.1.gz
 - pager > /bin/more
 - pager.1.gz > /usr/share/man/man1/more.1.gz
 - README
 - rmt > /usr/sbin/rmt-tar
 - rmt.8.gz > /usr/share/man/man8/rmt-tar.8.gz
 - which > /usr/bin/which.debianutils
 - which.1.gz > /usr/share/man/man1/which.deb
 - which.de1.gz > /usr/share/man/de/man1/which

cmd

docker exec + linux command such as ls

docker cp - transfer file / folder between host ⇔ container

- Copy File

```
#FROM HOST TO CONTAINER  
docker cp [OPTIONS] [SRC_PATH TO FILE] [CONTAINER_ID:DEST_PATH + FILE]
```

```
#FROM CONTAINER TO HOST  
docker cp [OPTIONS] [CONTAINER_ID:SRC_PATH _TO FILE] [DEST_PATH OF HOST+ FILE]
```

- Copy Folder

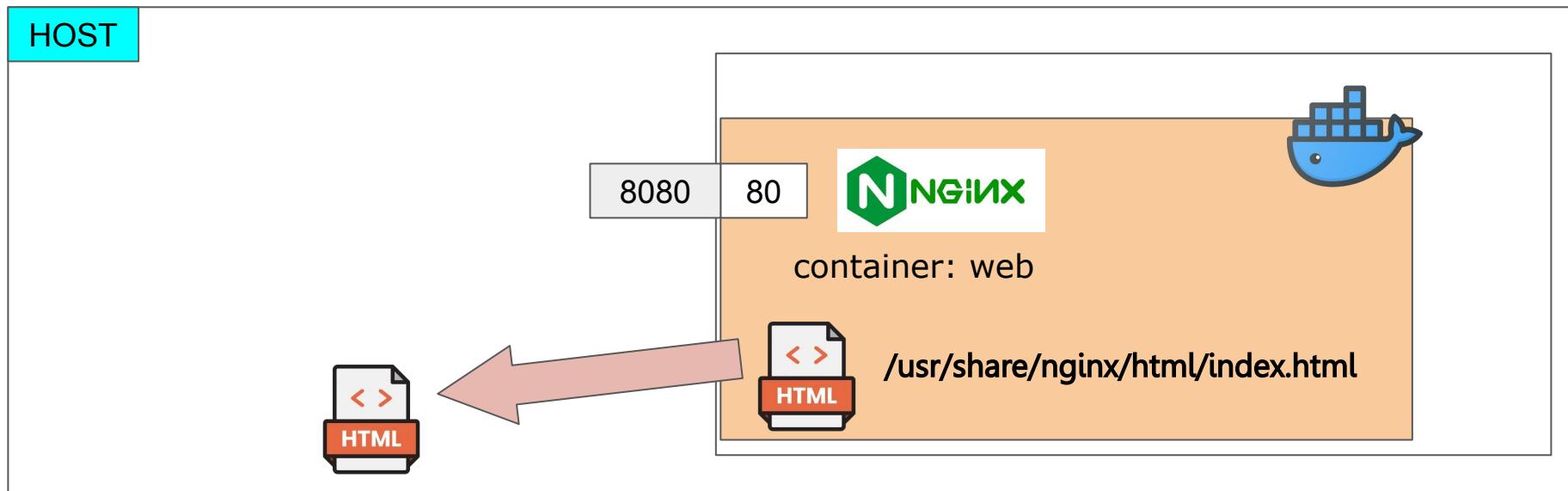
```
#FROM HOST TO CONTAINER  
docker cp [OPTIONS] [SRC_PATH OF HOST] [CONTAINER_ID:DEST_PATH]
```

```
#FROM CONTAINER TO HOST  
docker cp [OPTIONS] [CONTAINER_ID:SRC_PATH] [DEST_PATH OF HOST]
```

docker cp: Visualize

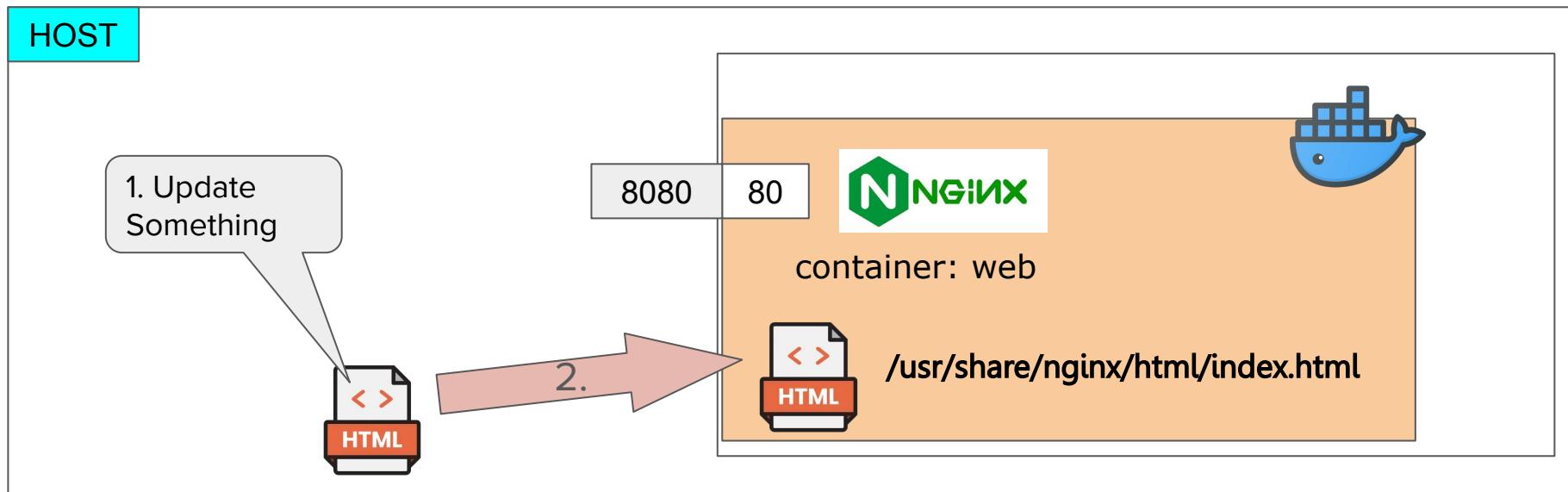
```
docker cp webapp:/usr/share/nginx/html/index.html .
```

Current Dir



docker cp: Visualize

```
docker cp index.html webapp:/usr/share/nginx/html/index.html
```



Control Container State - 1

- start container

```
docker start <container_name / container_id>
```

- stop container (graceful shutdown)

```
docker stop <container_name / container_id>
```

- restart container (stop + start)

```
docker restart <container_name / container_id>
```

- kill - kill hang process

```
docker kill<container_name / container_id>
```

Control Container State - 2

- pause - container process / keep resource (Stop Clear Resource)

```
docker pause <container_name / container_id>
```

Why Pause

- Resource Management
- Troubleshooting - Inspect Log
- Test & Development

- unpause

```
docker unpause <container_name / container_id>
```

docker logs - Investigate Case a Crimescene

```
docker logs <container_name / container_id>
```

```
> docker logs webapp --tail 5
2024/05/09 15:29:57 [notice] 1#1: start worker process 34
2024/05/09 15:29:57 [notice] 1#1: start worker process 35
2024/05/09 15:29:57 [notice] 1#1: start worker process 36
2024/05/09 15:29:57 [notice] 1#1: start worker process 37
PS C:\>
> docker logs webapp
/docker-entrypoint.sh: Looking for script in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/05/08 15:27:51 [notice] 1#1: using the "epoll" event method
2024/05/08 15:27:51 [notice] 1#1: nginx/1.25.5
2024/05/08 15:27:51 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/05/08 15:27:51 [notice] 1#1: OS: Linux 5.15.146.1-microsoft-standard-WSL2
2024/05/08 15:27:51 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/05/08 15:27:51 [notice] 1#1: start worker processes
```

```
docker logs mem-limit > log.txt
```

docker logs - Docker Desktop

The screenshot shows the Docker Desktop interface. On the left, a sidebar lists 'Containers' (webapp), 'Images', 'Volumes', 'Builds', 'Dev Environments BETA', 'Docker Scout', 'Extensions', and an 'Add Extensions' button. The main area displays the logs for the 'nginx' container of the 'webapp' service. The log output is as follows:

```
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 37 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 29 (SIGIO) received
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 17 (SIGCHLD) received from 31
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 31 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 29 (SIGIO) received
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 17 (SIGCHLD) received from 44
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 44 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 29 (SIGIO) received
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 17 (SIGCHLD) received from 42
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 35 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 42 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 29 (SIGIO) received
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 17 (SIGCHLD) received from 35
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 38 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 41 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 29 (SIGIO) received
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: signal 17 (SIGCHLD) received from 30
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 30 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 36 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 39 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: worker process 43 exited with code 0
2024-05-08 22:50:32 2024/05/08 15:50:32 [notice] 1#1: exit
```

Basic Docker Command (Clean Up)

- Delete Containers

```
docker rm <container_name or container_id>
docker rm -f <container_name or container_id>      //Force Delete
docker rm $(docker ps -a -q)                          //Delete All
docker rm $( docker ps -q -f status=exited)          //Delete All Stop (Status Exited)
```

- Delete Images

Note: <repo_url> ใส่ในกรณีที่มาจากการ pull ที่ไม่ใช่ docker hub เช่น gchr.io (github) หรือ repo กลางขององค์กร

```
docker rmi <repo_url>/<image_name>:<tag>
docker rmi -f <repo_url>/<image_name>:<tag>    //Force Delete
docker rmi -f $(docker images -a -q)              //Delete All
```

- Clear Everything > [docker system prune | Docker Docs](#)

15-20 minute

Try

```
docker exec -it  
docker cp > CopySample at main · pingkunga\(github.com\)  
docker start / stop / kill / pause / unpause  
docker logs
```

Hand On - PostgresSQL

- Pull PostgresSQL
- Check Image
- Run PostgresSQL Container
- List Container Image
- Access VIA CLI
- Create Sample DB
- Inspect Container info
- Delete PostgresSQL Container
- Delete PostgresSQL Image

Hand On - PostgresSQL (Soln)

- Pull PostgreSQL

```
docker pull postgres:16.3-alpine3.18
```

- Check Image

```
docker images
```

- Run PostgreSQL Container

```
docker run --name invs_postgresdb -e  
POSTGRES_USER=postgres -e  
POSTGRES_PASSWORD=postgres -p 5432:5432 -d  
postgres:16.3-alpine3.18
```

- List Container Image

```
docker ps -a
```

Hand On - PostgreSQL (Soln- Con.)

- Access VIA CLI

```
docker exec -it invs_postgresdb psql -U postgres
```

- Create Sample DB

```
CREATE DATABASE invs_db;
```

คำสั่งพื้นฐานกับ psql

```
---  
\q ออกจากรสурс  
\! cls , \! clear เคลียร์สกรีน  
\d ดูรายการฐานข้อมูล  
\du ดู user  
\c ดูว่าปัจจุบันเลือกฐานข้อมูลอะไรและใช้ user ไหนอยู่  
\d ดูรายการข้อมูลทั้งหมดใน schema  
\dt, \dt+ ดูเฉพาะตาราง  
\d <table_name> ดูโครงสร้างตาราง  
\dx ดูว่ามี Extensions อะไร
```

```
postgres=# CREATE DATABASE invs_db ENCODING unicode;  
CREATE DATABASE
```

```
postgres-# \l
```

Name	Owner	Encoding	Locale Provider	Collate	Ctype	ICU Locale	ICU Rules	Access privileges
invs_db	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres + postgres=CTc/postgres
postgres	postgres	UTF8	libc	en_US.utf8	en_US.utf8			=c/postgres + postgres=CTc/postgres
template0	postgres	UTF8	libc	en_US.utf8	en_US.utf8			
template1	postgres	UTF8	libc	en_US.utf8	en_US.utf8			

(4 rows)

Hand On - PostgresSQL (Soln- Con.)

- Inspect Container info

```
docker inspect invs_postgresdb
```

- Delete PostgreSQL Container
- Delete PostgreSQL Image

```
docker stop invs_postgresdb  
docker rm invs_postgresdb  
docker rmi postgres:16.3-alpine3.18
```

Container Resource Monitoring & Limit Resource

Ref: [Runtime options with Memory, CPUs, and GPUs | Docker Docs](#)

Resource Monitoring

Why

- Prevent OOME
- Resource Planing / Allocation

- get each container resource

```
docker stats
```

- get per container

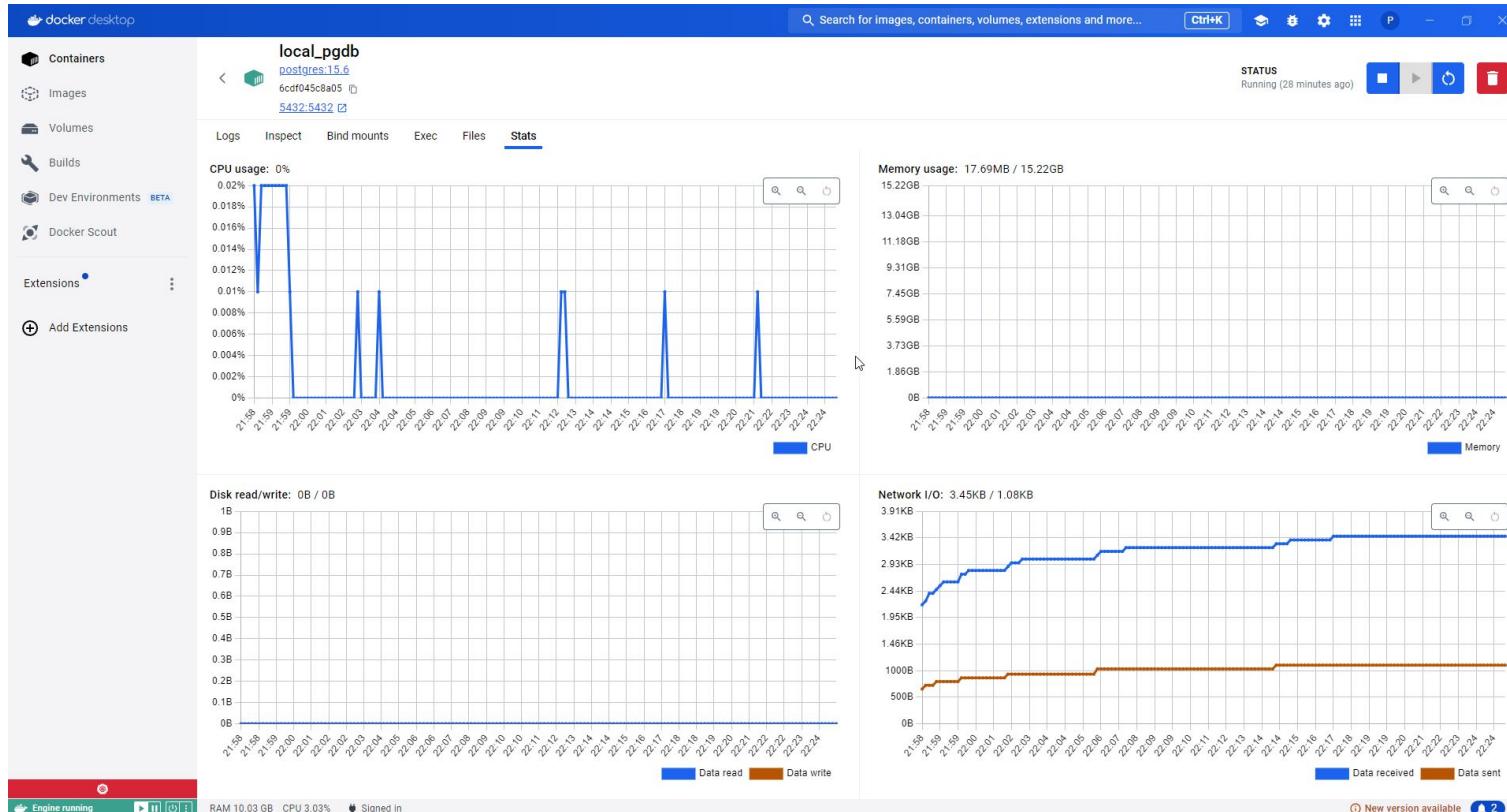
```
docker stats <<container_name/id>>
```

- filter

```
docker stats <<container_name/id>> --no-stream --format "{{ json . }}" | jq  
.
```

```
#For Windows Install jq first  
choco install jq
```

Resource Monitoring - Docker Desktop

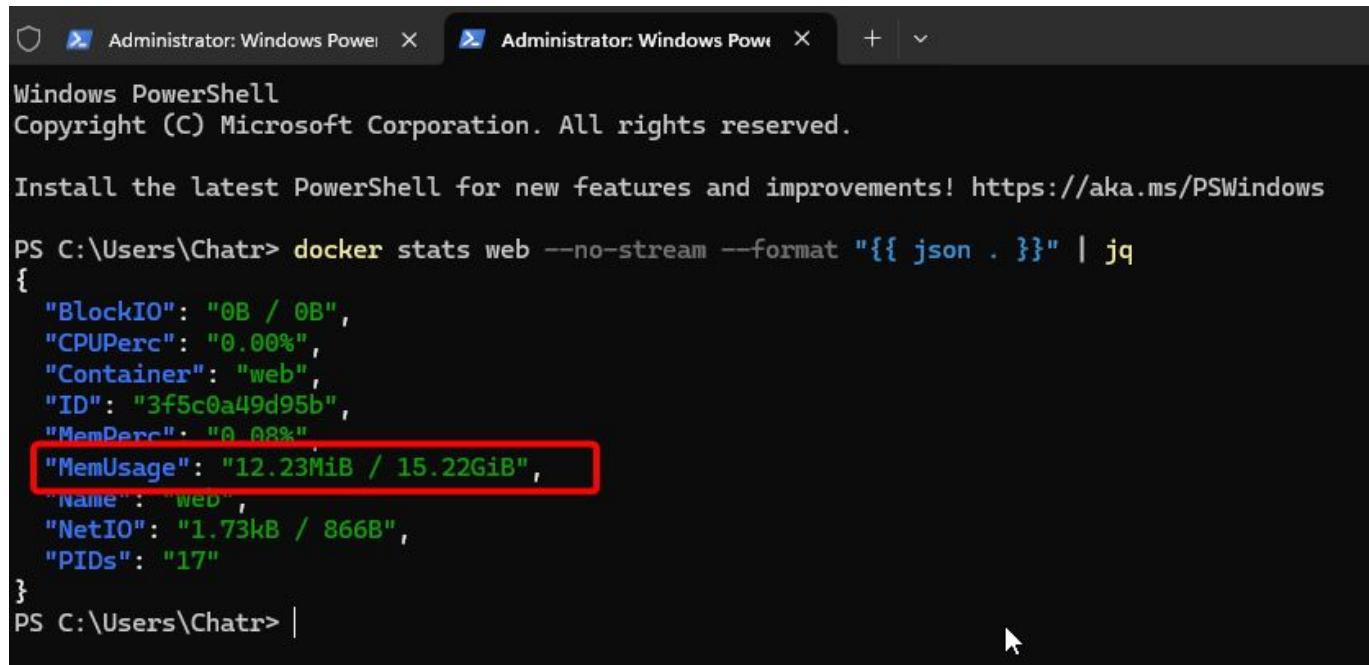


Resource Monitoring - Other Tools

- Command
 - docker stats
 - [Glances](#)
- Web Base
 - [uptime-kuma](#)
 - [Portainer](#)
 - [Zabbix](#) - (Ref: [Docker monitoring and integration with Zabbix](#))
 - [Checkmk Raw](#)
 - [LGTM](#) (More than monitoring > Observability)

Resource Monitoring - Limit Resource

Why - Default is Max of Host Resource



The screenshot shows two PowerShell windows running as Administrator. The left window displays the PowerShell prompt and the command being run. The right window shows the resulting JSON output, with the 'MemUsage' field highlighted by a red rectangle.

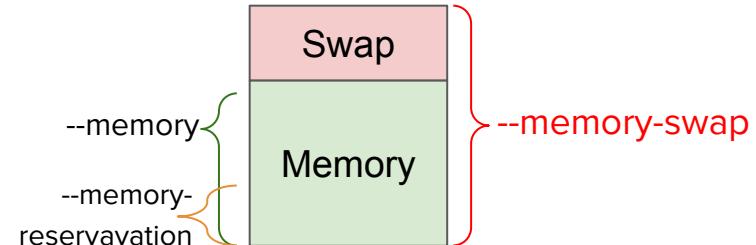
```
Administrator: Windows PowerShell X Administrator: Windows PowerShell X + v
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Chatr> docker stats web --no-stream --format "{{ json . }}" | jq
{
  "BlockIO": "0B / 0B",
  "CPUPerc": "0.00%",
  "Container": "web",
  "ID": "3f5c0a49d95b",
  "MemPerc": "0.08%",
  "MemUsage": "12.23MiB / 15.22GiB",
  "Name": "web",
  "NetIO": "1.73kB / 866B",
  "PIDs": "17"
}
PS C:\Users\Chatr>
```

Resource Monitoring - Limit Resource (Memory)

- Memory (Hard / Soft / Swap)
 - memory = Maximum Memory (Hard Limit)



```
docker run -d --name mem-limit --memory=256m nginx:alpine
```

- memory-reservation = Minimum Memory (Soft Limit)

```
docker run -d --name mem-limit --memory=256m --memory-reservation=128m nginx:alpine
```

- memory-swap = Excess Memory (Critical Case use disk like linux swap)

```
docker run -d --name mem-limit --memory=256m --memory-reservation=128m  
--memory-swap=384m nginx:alpine
```

Resource Monitoring - Limit Resource (CPU)

- CPU (Req. No of vCPU)
 - --cpuset-cpus = Fixed CPU Core

```
docker run -d --name cpu-limit --cpuset-cpus 0,3 nginx:alpine
```

0	1	2	3
█			█

```
docker run -d --name cpu-limit --cpuset-cpus 0-3 nginx:alpine
```

0	1	2	3
█	█	█	█

- --cpus = Fixed by Resource Size (Percent)

```
docker run -d --name cpu-limit --cpus=0.8 nginx:alpine
```

0	1	2	3
█			

```
docker run -d --name cpu-limit --cpus=1.5 nginx:alpine
```

0	1	2	3
	█		

```
docker run -d --name cpu-limit --cpus=2.0 --cpuset-cpus=0,1 nginx:alpine
```

0	1	2	3
	█		

```
docker run -d --name cpu-limit --cpus=1.5 --cpuset-cpus=0,1 nginx:alpine
```

0	1	2	3
█	█		

แบ่ง Core
ไหนเท่าไหร่
Random

Resource Monitoring - Limit Resource (CPU)

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Chatr> docker run -d --name cpu-limit --cpuset-cpus 2-3 --cpus=1.5 nginx:alpine
36936aebde1de2b779983ad33409830469df17b26af00f3fae532a7b76201d36
PS C:\Users\Chatr> docker exec -it cpu-limit cat /sys/fs/cgroup/cpuset/cpuset.cpus
2-3

What's next?
Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug cpu-limit
Learn more at https://docs.docker.com/go/debug-cli/
PS C:\Users\Chatr> docker inspect cpu-limit | select-string Cpus

    "CpuShares": 0,
    "NanoCpus": 15000000000,
    "CpusetCpus": "2-3",
    "CpusetMems": "",
```

Resource Monitoring - Hand-On

- [Day1ResourceLimitSample.md at main · pingkunga \(github.com\)](#)

Container Networking

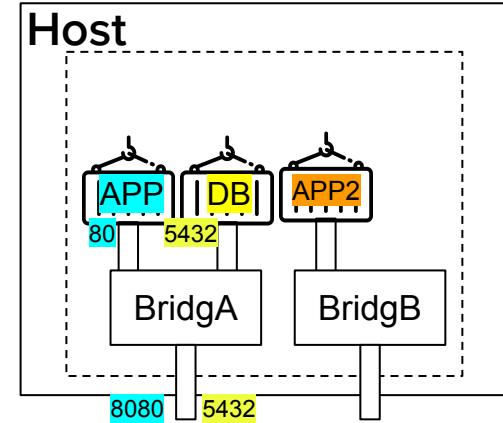
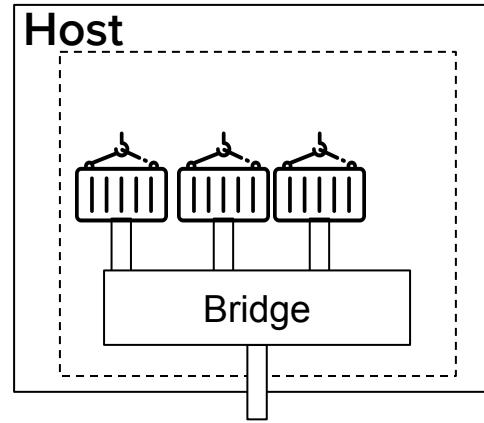
Networking

- Allow containers to communicate with each other / host / system / external etc.
- Access by Container Name, Docker Resolve IP automatically.
- Type
 - **bridge (default) - private network**
 - **host - use host netwok / port (Linux Only)**
 - **none - no network**
 - overlay - share network between docker hosts (Recommend K8S)
 - ipvlan
 - macvlan - assign mac address

Ref: [Day in the Life: Docker Networking - YouTube](#)

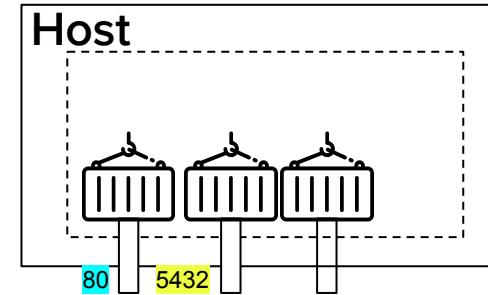
Networking

- Type
 - bridge (default)
 - host
 - none

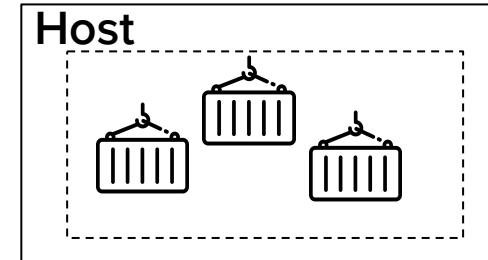


Networking

- Type
 - bridge (default)
 - host
 - none



-
- Type
 - bridge (default)
 - host
 - none



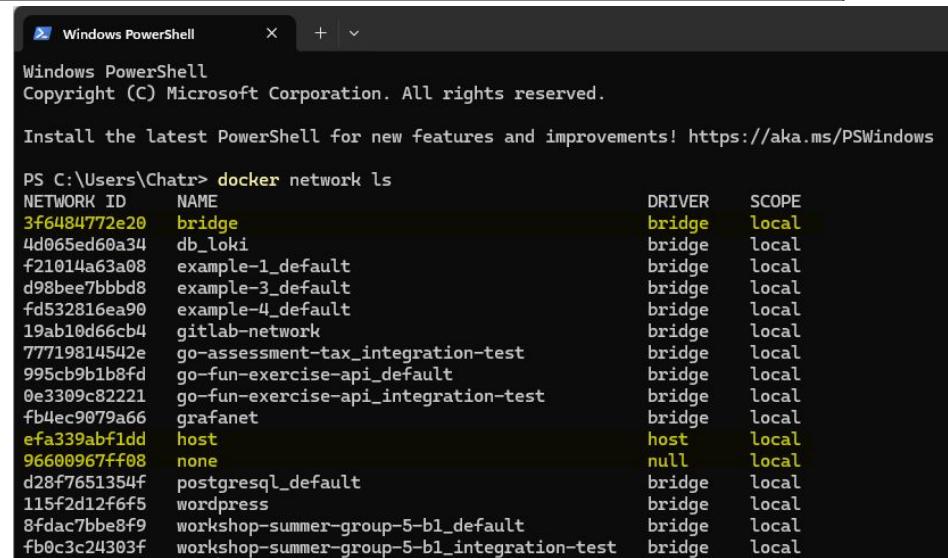
Networking

- Create Network

```
docker network create <<network_name>>
docker network create -d <<driver name>> <<network_name>>
```

- List Network

```
docker network ls
```



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "docker network ls" is run, displaying a table of network configurations. The table has columns for NETWORK ID, NAME, DRIVER, and SCOPE. The output shows various networks like bridge, db_loki, example-1_default, etc., each with its respective driver (bridge or host) and scope (local).

NETWORK ID	NAME	DRIVER	SCOPE
3f6484772e20	bridge	bridge	local
4d065ed60a34	db_loki	bridge	local
f21014a63a08	example-1_default	bridge	local
d98bee7bbbd8	example-3_default	bridge	local
fd532816ea90	example-4_default	bridge	local
19ab10d66cb4	gitlab-network	bridge	local
77719814542e	go-assessment-tax_integration-test	bridge	local
995cb9b1b8fd	go-fun-exercise-api_default	bridge	local
0e3309c82221	go-fun-exercise-api_integration-test	bridge	local
fb4ec9079a66	grafanet	bridge	local
efa339abf1dd	host	host	local
96600967ff08	none	null	local
d28f7651354f	postgresql_default	bridge	local
115f2d12f6f5	wordpress	bridge	local
8fdac7bbe8f9	workshop-summer-group-5-b1_default	bridge	local
fb0c3c24303f	workshop-summer-group-5-b1_integration-test	bridge	local

Networking

- Add Container to Network

```
docker network connect <<network_name>> <<container_name>>
```

- Create Container with Network

```
docker run -d --name webapp --network <<network_name>> nginx:alpine
```

Networking

- Inspect Network

```
docker network inspect <<network_name>>
```

```
PS C:\Users\Chatr> docker network inspect wordpress
[{"Name": "wordpress",
 "Id": "115f2d12f6f5e55d700b1270e69ccbcb155ce50fd45cfce88bfd85c12c1048b3",
 "Created": "2023-10-14T02:24:55.372433513Z",
 "Scope": "local",
 "Driver": "bridge",
 "EnableIPv6": false,
 "IPAM": {
   "Driver": "default",
   "Options": {},
   "Config": [
     {
       "Subnet": "172.18.0.0/16",
       "Gateway": "172.18.0.1"
     }
   ]
 },
 "Internal": false,
 "Attachable": false,
 "Ingress": false,
 "ConfigFrom": {
   "Network": ""
 },
 "ConfigOnly": false,
 "Containers": {},
 "Options": {},
 "Labels": {}
}
```

```
PS C:\Users\Chatr> docker network inspect postgresql_default
[{"Name": "postgresql_default",
 "Id": "d28f7651354f02844676e4ec0b257b170f9b6b44141c7870b057313876c6c730",
 "Created": "2024-02-14T08:47:29.030309282Z",
 "Scope": "local",
 "Driver": "bridge",
 "EnableIPv6": false,
 "IPAM": {
   "Driver": "default",
   "Options": null,
   "Config": [
     {
       "Subnet": "172.25.0.0/16",
       "Gateway": "172.25.0.1"
     }
   ],
   "Internal": false,
   "Attachable": false,
   "Ingress": false,
   "ConfigFrom": {
     "Network": ""
   },
   "ConfigOnly": false,
   "Containers": {
     "6cdf045c8a052994e693d2dfaf7eed75f7ebe2491742f42df605edde5e3f789": {
       "Name": "local_pgdb",
       "EndpointID": "7d9a2c9bd055e92a2d271545cf653f4cec64746087fc3e0c1a0ca4353e9206c3",
       "MacAddress": "02:42:ac:19:00:03",
       "IPv4Address": "172.25.0.3/16",
       "IPv6Address": ""
     },
     "87c586ed1409c27dff61083ca70c36a3b3b52f5b23fa4053c93e0c3f36634e33": {
       "Name": "pgadmin4_container",
       "EndpointID": "9b9d5157a31886d124b008edf3676956ff12e1f698b8b18f0626b89129294ecc",
       "MacAddress": "02:42:ac:19:00:02",
       "IPv4Address": "172.25.0.2/16",
       "IPv6Address": ""
     }
   },
   "Options": {},
   "Labels": {
     "com.docker.compose.network": "default",
     "com.docker.compose.project": "postgresql",
     "com.docker.compose.version": "2.23.0"
   }
}
```

Networking

- Disconnect from network

```
docker network disconnect <<network_name>> <<container_name>>
```

- Remove Network

```
docker network rm <<network_name>>
```

Networking - Sample

- [Day1/NetworkSample/NetworkSample.md at main · pingkunga \(github.com\)](#)

Container - Store Data

Store Data

Container Stateless

- Way to Store Data
 - volume - docker manage
 - mount
- Basic Command docker manage volume

```
# Create a named volume  
docker volume create mysql-files
```

```
# List all volumes  
docker volume ls
```

```
# Connect a volume to a container  
docker run --name mysql-db -d -e MYSQL_ROOT_PASSWORD=secret -v  
mysql-files:/var/lib/mysql --label dev.app=persistent-db mysql
```

```
# Remove a volume (only works if it's not in use)  
docker volume rm mysql-files
```

[How to back up and restore your Docker Desktop data | Docker Docs](#)

[Back Up and Share Docker Volumes with This Extension | Docker](#)

Store Data - docker volume vs mount

- Mount - Map HostPath / Network Path (NFS) with container path

```
#docker volume  
docker run --name mysql-db -d -e  
MYSQL_ROOT_PASSWORD=secret -v  
mysql-files:/var/lib/mysql --label  
dev.app=persistent-db mysql
```

```
# mount  
docker run --name mysql-db-mount -d  
-e MYSQL_ROOT_PASSWORD=secret  
-v D:\DBDEV:/var/lib/mysql  
--label training.app=persistent-db mysql
```

Name	Date modified	Type	Size
#innodb_redo	4/16/2024 5:46 PM	File folder	
#innodb_temp	4/16/2024 5:46 PM	File folder	
mysql	4/16/2024 5:45 PM	File folder	
performance_schema	4/16/2024 5:45 PM	File folder	
sys	4/16/2024 5:45 PM	File folder	
#ib_16384_0 dblwr	4/16/2024 5:46 PM	DBLWR File	192 KB
#ib_16384_1 dblwr	4/16/2024 5:45 PM	DBLWR File	8,384 KB
auto.cnf	4/16/2024 5:45 PM	CNF File	1 KB

Store Data - docker volume in Docker Desktop

The screenshot shows the Docker Desktop interface. On the left, the sidebar includes 'Containers', 'Images', 'Volumes' (which is selected), 'Builds', 'Dev Environments BETA', 'Docker Scout', and 'Extensions'. A 'Create New Volume' button is at the bottom. The main area shows a volume named 'mysql-files' used by a container named 'mysql-db'. The 'Data' tab is selected, showing a list of files under the 'innodb_redo' directory. A context menu is open over one of the files, with options 'Save as...' and 'Delete' highlighted in red. The table below lists the files with their sizes, last modified times, and modes.

Name	Size	Last modified	Mode
#ib_16384_0 dblwr	192 kB	38 seconds ago	-rw-r----
#ib_16384_1 dblwr	8.2 MB	60 seconds ago	-rw-r----
#innodb_redo	100 MB	43 seconds ago	drwxr-x---
#ib_redo01	3.1 MB	43 seconds ago	-rw-r----
#ib_redo02	3.1 MB	43 seconds ago	-rw-r----
#ib_redo03	3.1 MB	43 seconds ago	-rw-r----
#ib_redo04	3.1 MB	43 seconds ago	-rw-r----
#ib_redo05	3.1 MB	43 seconds ago	-rw-r----
#ib_redo06	3.1 MB	43 seconds ago	-rw-r----
#ib_redo07	3.1 MB	43 seconds ago	-rw-r----
#ib_redo08	3.1 MB	43 seconds ago	-rw-r----
#ib_redo09	3.1 MB	43 seconds ago	-rw-r----
#ib_redo10	3.1 MB	43 seconds ago	-rw-r----
#ib_redo11	3.1 MB	43 seconds ago	-rw-r----
#ib_redo12	3.1 MB	43 seconds ago	-rw-r----
#ib_redo13	3.1 MB	43 seconds ago	-rw-r----
#ib_redo14	3.1 MB	43 seconds ago	-rw-r----
#ib_redo15	3.1 MB	43 seconds ago	-rw-r----
#ib_redo16	3.1 MB	43 seconds ago	-rw-r----
#ib_redo17	3.1 MB	43 seconds ago	-rw-r----
#ib_redo18	3.1 MB	43 seconds ago	-rw-r----
#ib_redo19	3.1 MB	43 seconds ago	-rw-r----
#ib_redo20	3.1 MB	43 seconds ago	-rw-r----
#ib_redo21	3.1 MB	43 seconds ago	-rw-r----
#ib_redo22	3.1 MB	43 seconds ago	-rw-r----
#ib_redo23	3.1 MB	43 seconds ago	-rw-r----
#ib_redo24	3.1 MB	43 seconds ago	-rw-r----
#ib_redo25	3.1 MB	43 seconds ago	-rw-r----
#ib_redo26	3.1 MB	43 seconds ago	-rw-r----
#ib_redo27	3.1 MB	43 seconds ago	-rw-r----
#ib_redo28	3.1 MB	43 seconds ago	-rw-r----
#ib_redo29	3.1 MB	43 seconds ago	-rw-r----
#ib_redo30	3.1 MB	43 seconds ago	-rw-r----
#ib_redo31	3.1 MB	43 seconds ago	-rw-r----

```
#docker volume
docker run --name mysql-db -d -e
MYSQL_ROOT_PASSWORD=secret -v
mysql-files:/var/lib/mysql --label
dev.app=persistent-db mysql
```

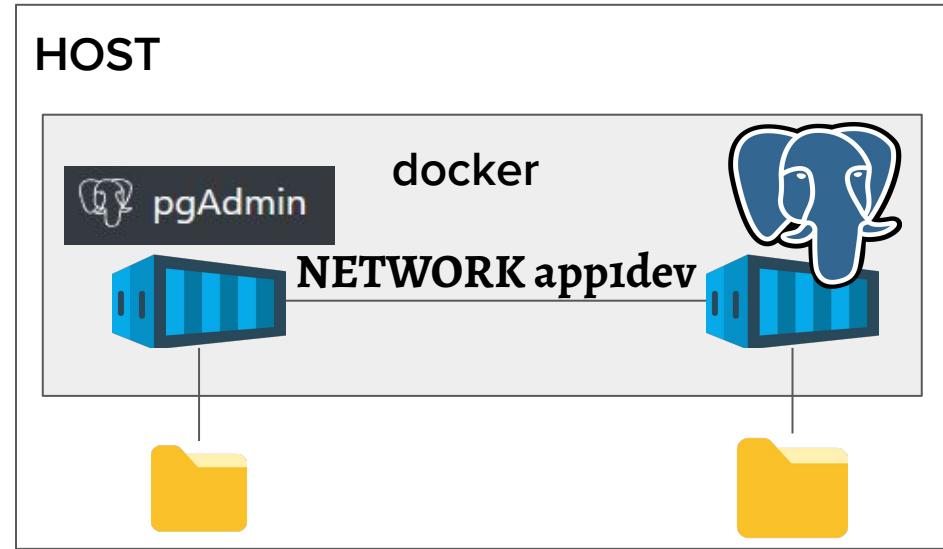
Docker101 by Chatra Agam Senapati 85

Hand On - Volume

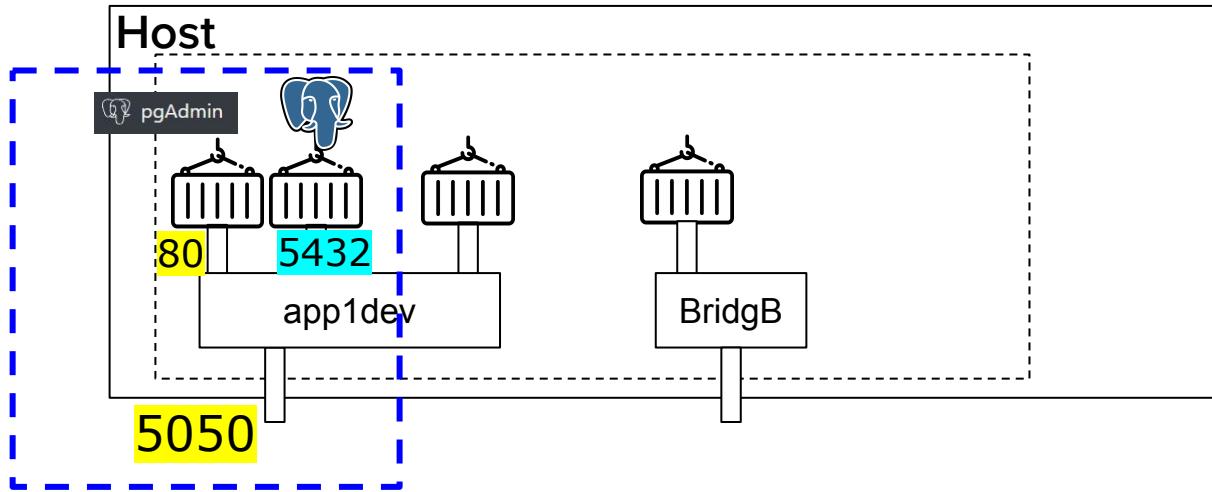
- [Day1/VolumeSample/VolumeSample.md at main · pingkunga/ \(github.com\)](https://github.com/pingkunga/Day1/blob/main/VolumeSample/VolumeSample.md)

Hand On - Multiple Image

- Pull PostgresSQL
- Check Image
- Run PostgresSQL Container with
 - network app1dev+ Volume/Mount
 - Init Database with [init.sql](#)
- List Container Image
- Pull PGAdmin4
- Run PostgresSQL Container with network app1dev + Volume/Mount
- Create Connection
- Delete PostgresSQL/PGAdmin4 Container



Hand On - Multiple Image Solution



Solution: [Day1/MultipleImageWithVolumeNetwork/solution.md at main · pingkunga/github.com](#)

Hand On - Multiple Image Solution

The screenshot displays two main interfaces: pgAdmin (left) and Docker Desktop (right).

Docker Desktop:

- Containers:** Shows a list of running containers. One container, `postgres_app1`, is highlighted with a red border.
- Search bar:** "Search for images, containers, volumes..."
- Filter:** "Only show running containers"
- Table Headers:** Name, Image, Status, CPU (%), Port(s), Last Seen
- Container Details:** CPU usage (7.47% / 1600%), Memory usage (3.55GB / 14.87GB)

Name	Image	Status	CPU (%)	Port(s)	Last Seen
6d49d3fb3b92	grafana/ag	Running	0.72%	20	2023-07-18 10:45:20
749b72025984	pingkunga/ag	Running	0.01%	20	2023-07-18 10:45:20
31dc22904aca	pingkunga/ag	Running	0.01%	20	2023-07-18 10:45:20
c9a5573492ac	pingkunga/ag	Running	0.01%	20	2023-07-18 10:45:20
postgres_app1	postgres	Running	0%	5432:5432	5 minutes ago
pgadmin4.app1	dpage/pgadmin4	Running	0.63%	5050:80	5 minutes ago
workshop-summer-gro		Exited	0%	7 minutes ago	21
go-assessment-tax		Exited	0%	21 minutes ago	11
go-fun-exercise-api		Exited	0%	1 hour ago	1
postgresql		Exited	0%	20 hours ago	20

pgAdmin:

- Servers:** Servers (1) > postgres_app1
- Databases:** Databases (2) > postgres
- Query Editor:** SELECT * FROM postgres_app1
- Connection Tab:** Host name/address: postgres_app1, Port: 5432, Maintenance database: taxdb, Username: postgres, Kerberos authentication?: Off, Role: , Service: .
- Data Output:** Shows a table with columns id, [PK] int, and rows 1, 2, 3.

Hand On - Multiple Image Solution

The screenshot shows the PgAdmin 4 interface. The top navigation bar includes File, Object, Tools, and Help. The main window has tabs for Object Explorer, Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, Processes, and a selected tab for taxdb/postgres@postgres_app1*. The left sidebar shows the schema tree with the tax_deductionconfig table selected. The right pane contains a query editor with the following SQL code:

```
1 SELECT id, deduction_type, deduction_amount, deduction_min, deduction_max
2 FROM public.tax_deductionconfig;
```

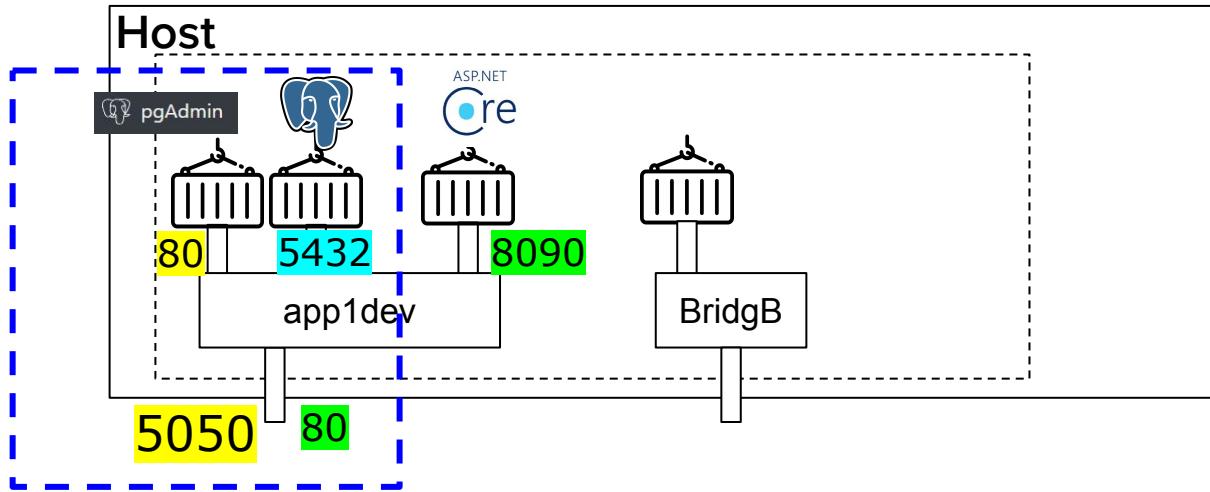
Below the query editor is a Data Output tab showing the results of the query:

	id	deduction_type	deduction_amount	deduction_min	deduction_max
1	1	personal	60000.00	10000.00	100000.00
2	2	donation	0.00	0.00	100000.00
3	3	k-receipt	50000.00	0.00	100000.00

At the bottom, it says "Total rows: 3 of 3" and "Query complete 00:00:00.138".

Ln 1, Col 1

Hand On - Multiple Image Solution (Improve)



Day 2

More Example with Resouce Limit

- ASP NET8 [microsoft-dotnet-samples - Official Image](#)
- Sample curl with powershell
- Hand-On: [Day1/ResourceLimitWithASPNETCore.md at main · pingkunga/\(github.com\)](#)

More Example with Resource Limit

- ASP.NET 8 [microsoft-dotnet-samples - Official Image](#)
- Sample curl with powershell

Initial: `docker run -it --rm -p 8000:8080 --name aspnetcore_sample mcr.microsoft.com/dotnet/samples:aspnetapp`

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
fef8b6d4e08f	aspnetcore_sample	0.01%	26.25MiB / 15.22GiB	0.17%	1.39kB / 586B	0B / 0B	30

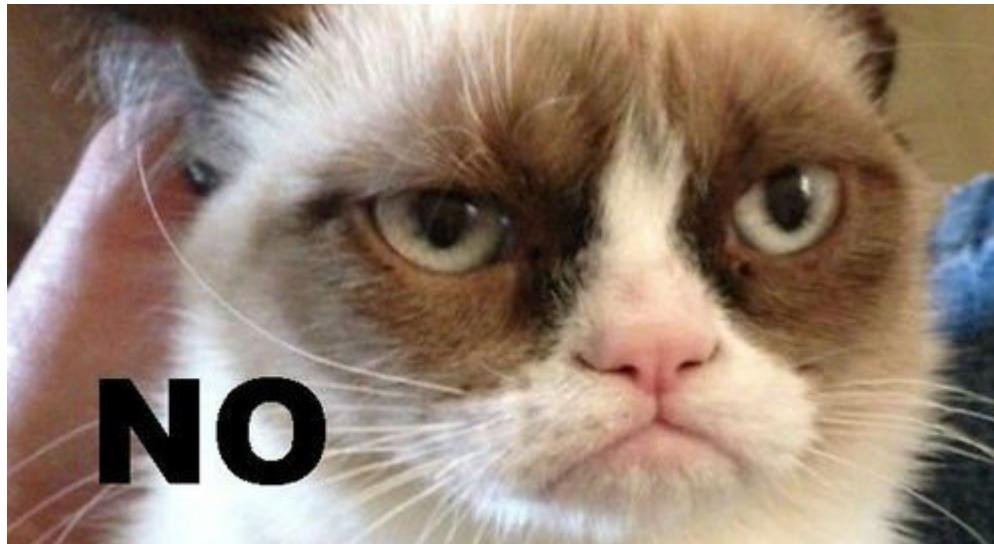
Curl+Wait for
3-5 minutes

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
5993b2a0d118	aspnetcore_sample	31.90%	256.1MiB / 15.22GiB	1.64%	38.6MB / 337MB	0B / 0B	36

Limit Resource: `docker run -it --rm -p 8000:8080 --memory=48m --memory-reservation=32m --memory-swap=64m --cpus=0.8 --name aspnetcore_sample mcr.microsoft.com/dotnet/samples:aspnetapp`

CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
c128ded11d9d	aspnetcore_sample	25.75%	40.81MiB / 48MiB	85.02%	57.5MB / 486MB	0B / 0B	16

How can I add a volume / host directory (disk / NFS) to an existing Docker container?



Update configuration of running Containers

```
docker update <options> <container id/name>
```

- Example

```
docker update -m 300M aspnetcore_sample  
aspnetcore_api
```

Options

Option	Default	Description
--blkio-weight		Block IO (relative weight), between 10 and 1000, or 0 to disable (default 0)
--cpu-period		Limit CPU CFS (Completely Fair Scheduler) period
--cpu-quota		Limit CPU CFS (Completely Fair Scheduler) quota
--cpu-rt-period		API 1.25+ Limit the CPU real-time period in microseconds
--cpu-rt-runtime		API 1.25+ Limit the CPU real-time runtime in microseconds
-c, --cpu-shares		CPU shares (relative weight)
--cpus		API 1.29+ Number of CPUs
--cpuset-cpus		CPUs in which to allow execution (0-3, 0,1)
--cpuset-mems		MEMs in which to allow execution (0-3, 0,1)
-m, --memory		Memory limit
--memory-reservation		Memory soft limit
--memory-swap		Swap limit equal to memory plus swap: -1 to enable unlimited swap
--pids-limit		API 1.40+ Tune container pids limit (set -1 for unlimited)
--restart		Restart policy to apply when a container exits

Ref: [docker container update | Docker Docs](#)
[/ docker update command sample \(github.com\)](#)

How can I add a volume / host directory (disk / NFS) to an existing Docker container? (Work Around)

Solution 1 : docker cp

- docker cp [OPTIONS] OLD_CONTAINER:SRC_PATH DEST_PATH (disk / NFS)
- create new container + mount
- remove old container



How can I add a volume / host directory (disk / NFS) to an existing Docker container? (Work Around)

As [David Maze](#) mentioned, it's almost impossible to change the volume location of an existing container by using normal docker commands.

7 I found an alternative way that works for me. **The main idea is convert the existing container to a new docker image and initialize a new docker container on top of it.** Hope works for you too.

```
# Create a new image from the container
docker commit CONTAINERID NEWIMAGENAME

# Create a new container on the top of the new image
docker run -v HOSTLOCATION:CONTAINERLOCATION NEWIMAGENAME
```

Share Edit Follow Flag

edited Nov 10, 2021 at 14:06



40.2k ● 15 ● 171 ● 498

answered Jul 5, 2020 at 17:43



Bowen Xu
4,026 ● 1 ● 23 ● 25

Add a comment



[Linux - How to mount a host directory into a running docker container - Stack Overflow](#)

Solution 2 : create a new image from old container

- docker commit <ContainerId> <NewImageName>
- create new container from NewImageName + mount
- remove/stop NewImageName Container
- start new container from old image with mount option

!!! Data Loss ถ้า Images นั้นถูกกำหนดว่า ถ้าถูก Deploy ให้ Initial ข้อมูลใหม่

Build Your Own Image

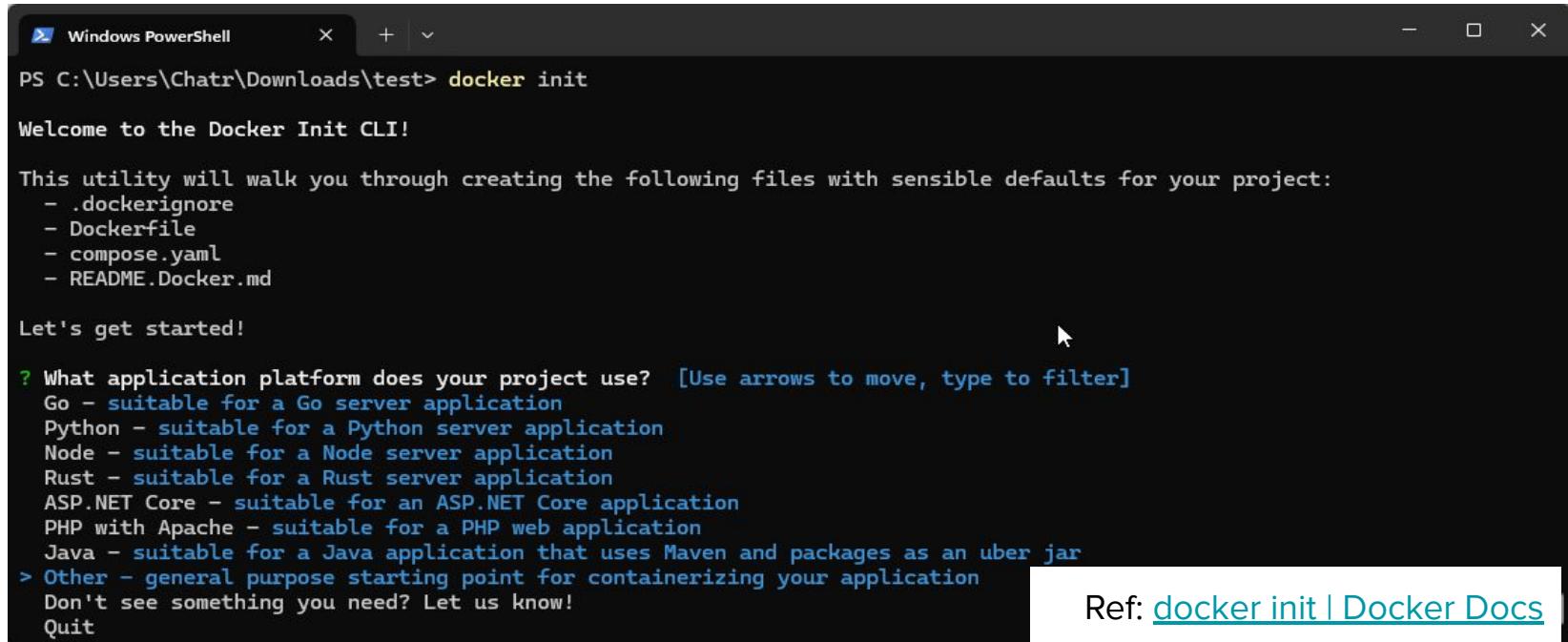
- Building Container (Dockerfile)
- Hand-On

Why Custom image

- Customer Requirement !!!
- Optimize Cost
- Security Concert (non-root)

docker init

- Initialize a project with the files necessary to run the project in a container.



```
PS C:\Users\Chatr\Downloads\test> docker init

Welcome to the Docker Init CLI!

This utility will walk you through creating the following files with sensible defaults for your project:
- .dockerignore
- Dockerfile
- compose.yaml
- README.Docker.md

Let's get started!

? What application platform does your project use? [Use arrows to move, type to filter]
Go - suitable for a Go server application
Python - suitable for a Python server application
Node - suitable for a Node server application
Rust - suitable for a Rust server application
ASP.NET Core - suitable for an ASP.NET Core application
PHP with Apache - suitable for a PHP web application
Java - suitable for a Java application that uses Maven and packages as an uber jar
> Other - general purpose starting point for containerizing your application
Don't see something you need? Let us know!
Quit
```

Ref: [docker init | Docker Docs](#)

docker init - output

1. .dockerignore - specifies which files and directories should be excluded from the build such as source code / secret
2. Dockerfile - a script that contains instructions for building a customized docker image
3. compose.yaml - make container working together
4. README.Docker.mdv

Dockerfile - Way to defined Container

- Structure
 - FROM : select the base image
 - WORKDIR : defines a working directory for RUN CMD ENTRYPOINT COPY ADD
 - ADD/**COPY** : move file from host > container image
 - RUN : execute command (linux)
 - ENV : set environment variable
 - USER : defines the user to use to run the RUN CMD ENTRYPOINT command

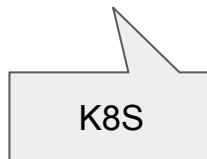
FROM scratch

Security
Issue

Ref: [Dockerfile reference | Docker Docs](#)

Dockerfile - Way to defined Container

- Structure (Cont.)
 - EXPOSE : specifies the container to wait for requests on the specified port. Use it with -p when using docker run.
 - CMD : execute command, but only once in file, if more than one use last one or use as default parameter to ENTRYPOINT
 - ENTRYPOINT : Command to run after starting the container



Ref: [Dockerfile reference | Docker Docs](#)

Sample Docker File

- [dotnet6_db2_dockerfile](#)
[\(github.com\)](https://github.com)

```
dotnet6_db2_dockerfile

1  FROM mcr.microsoft.com/dotnet/aspnet:6.0 AS runtime
2
3  COPY bin/Release/net6.0/publish/ /app
4
5  RUN apt-get update && apt-get install -y libxml2-dev
6  ENV DB2_CLI_DRIVER_INSTALL_PATH="/app/clidriver"
7  ENV LD_LIBRARY_PATH="/app/clidriver/lib:/app/clidriver/lib/libdb2.so"
8  ENV LIBPATH="/app/clidriver/lib"
9
10 ENV PATH=$PATH:"/app/clidriver/bin"
11 ENV PATH=$PATH:"/app/clidriver/adm"
12 ENV PATH=$PATH:"/app/clidriver/lib"
13
14 RUN apk add --no-cache tzdata
15 ENV TZ=Asia/Bangkok
16 RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
17
18 RUN addgroup --system --gid 10001 invsopgp && adduser --system --uid 1001 --ingroup invsopgp --shell /bin/sh invsbch
19 RUN chown -Rf invsbch:invsopgp /app && chmod -Rf 760 /app
20
21 USER invsbch
22 WORKDIR /app
23 RUN mkdir /app/BNZstorage
24
25 ENV ASPNETCORE_URLS=http://+:8080
26 EXPOSE 8080
27
28 ENTRYPOINT ["dotnet", "ds.Invest.WebAPI.dll", "-buc", "/app", "-env", "Production"]
```

Build Image from Dockerfile

~~docker build~~ not support multi-platform

Context - Current Dir

```
docker build -t inv-api:1.0.0 .
```

```
docker build -t inv-api:1.0.0 -f abc.DockerFile
```

```
docker build -t inv-api:1.0.0 --no-cache -f abc.DockerFile
```

Build Image from Dockerfile

docker buildx for multiple platform

- create builder

```
#Create Builder (Container)
docker buildx create --name mybuilder
docker buildx use mybuilder
```

- build

```
#build and save local
docker buildx build --platform linux/amd64,linux/arm64 -t ourapp:latest . --load
docker buildx build --platform linux/amd64 -t my_org_postgres:16.3 . -f .\DockkerfileNotOptimize
--load
```

```
#build and push
docker buildx build --platform linux/amd64,linux/arm64 -t ourapp:latest . --push
```

Build Image from Dockerfile

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

powerShell - BuildingImage + □ 🖌 ... ^ X

```
ERROR: unable to prepare context: path ".DockkerfileNotOptimize" not found
PS D:\2\WarRoom\9\PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> docker buildx build --platform linux/amd64 -t my_org_postgres-1:16.3 DockkerfileNotOptimize --load
[+] Building 0.0s (0/0)
ERROR: unable to prepare context: path "DockkerfileNotOptimize" not found
PS D:\2\WarRoom\9\PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> docker buildx build --platform linux/amd64 -t my_org_postgres-1:16.3 . -f .\DockkerfileNotOptimize --load
[+] Building 25.9s (13/13) FINISHED
  => [internal] load build definition from DockkerfileNotOptimize
  => => transferring dockerfile: 763B
  => [internal] load metadata for docker.io/library/postgres:16.3-bookworm
  => [auth] library/postgres:pull token for registry-1.docker.io
  => [internal] load .dockignore
  => => transferring context: 2B
  => CACHED [1/7] FROM docker.io/library/postgres:16.3-bookworm@sha256:1bf73ccae25238fa555100080042f0b2f9be08eb757e200fe6afc1fc413a1b3c
  => => resolve docker.io/library/postgres:16.3-bookworm@sha256:1bf73ccae25238fa555100080042f0b2f9be08eb757e200fe6afc1fc413a1b3c
  => [2/7] RUN apt-get update
  => [3/7] RUN apt-get install -y curl
  => [4/7] RUN apt-get -y install postgresql-16-pgvector
  => [5/7] RUN apt-get -y install postgresql-16-cron
  => [6/7] RUN echo "shared_preload_libraries='pg_cron'" >> /usr/share/postgresql/postgresql.conf.sample
  => [7/7] RUN echo "cron.database_name='postgres'" >> /usr/share/postgresql/postgresql.conf.sample
  => exporting to docker image format
  => => exporting layers
  => => exporting manifest sha256:1ac4765b500f5710ca5db65820a925087454688ffbd032a7921b3085b9541fea
  => => exporting config sha256:0e7877a249961d01aadb7bbc31de57f806816bdf136ec4dd0637e2126971fdcc
  => => sending tarball
  => importing to docker
  => => loading layer a6a7d9a4e0c9 163.84kB / 15.68MB
  => => loading layer a9b0a56913dc 32.77kB / 1.96MB
  => => loading layer eab3188bfd74 32.77kB / 936.60kB
  => => loading layer 168cdc8e04f4 32.77kB / 715.25kB
  => => loading layer 9278ee4ec784 9.14kB / 9.14kB
```

Build Image from Dockerfile - Docker Desktop

The screenshot shows the Docker Desktop interface. On the left, a sidebar has 'Builds' selected and highlighted with a red box. The main area displays a table of build history:

ID	Name	Builder	Status	Duration
RK0FVY	Day2/BuildingImage	mybuilder	Completed	25.6s 26 r
LD2GBG	Day2/BuildingImage	mybuilder	Completed	7.3s
LC10Z2	Day2/BuildingImage	mybuilder	Completed	7.9s
ISR1BI	Day2/BuildingImage	mybuilder	Completed	6.0s
P8AKMX	Day2/BuildingImage	mybuilder	Completed	7.9s
E7U8B7	Day2/BuildingImage	mybuilder	Completed	8.5s
JWWR1U	Day2/BuildingImage	mybuilder	Completed	8.6s
U41K5X	Day2/BuildingImage	mybuilder	Completed	7.4s
BNKQ42	Day2/BuildingImage	mybuilder	Completed	35.0s

A red arrow points from the 'Selected builder' section to the 'Builder settings' tab in the top navigation bar.

Selected builder
mybuilder
Builder settings

Builds [Give feedback](#)

Build container images and artifacts from source code. [Learn more](#)

Build large and multi-platform Docker images faster in Docker Build Cloud.
[Sign up](#) to improve build speeds for you, your team, and even your CI.

Build history [Active builds](#)

Search

Settings [Give feedback](#)

General Resources Docker Engine Builders Kubernetes Software updates Extensions Features in development Notifications

Selected builder

This is the builder that will be used by default when you start a build. [Learn more](#)

mybuilder docker-container Running 28 minutes ago

Available builders

default docker Running 21 days ago

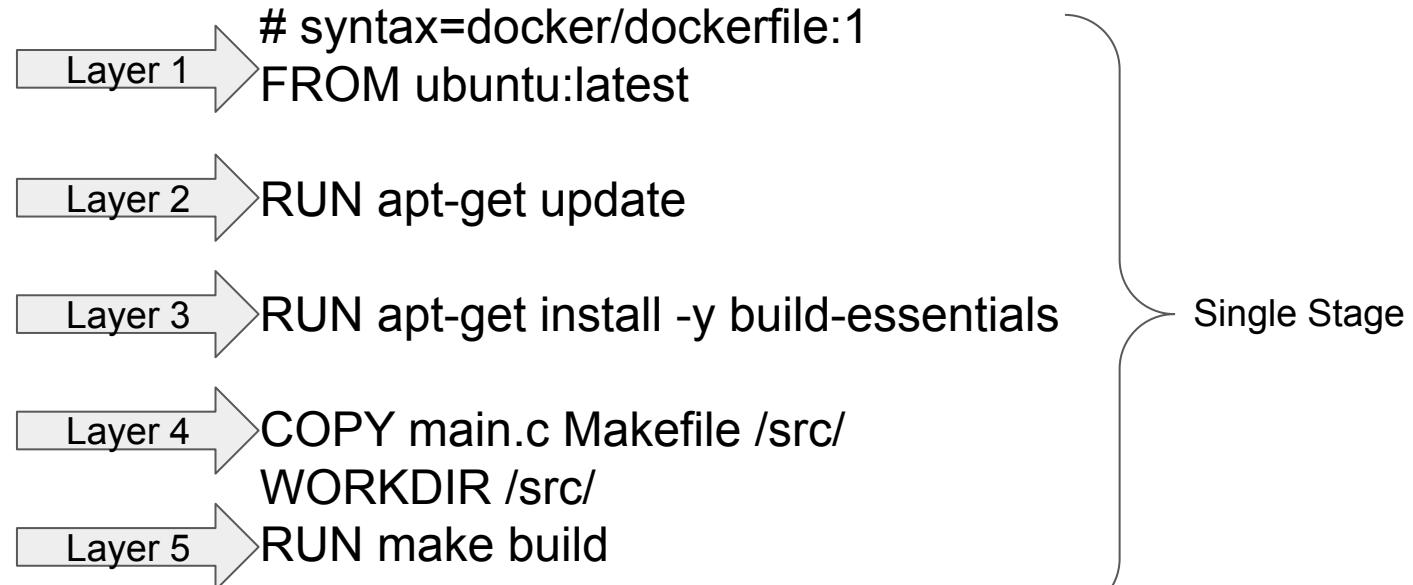
desktop-linux docker Running

[Cancel](#) [Apply & restart](#)

Engine running RAM 11.77 GB CPU 1.19% Signed in RAM 11.82 GB CPU 1.38% Signed in New version available 2

Docker Layer

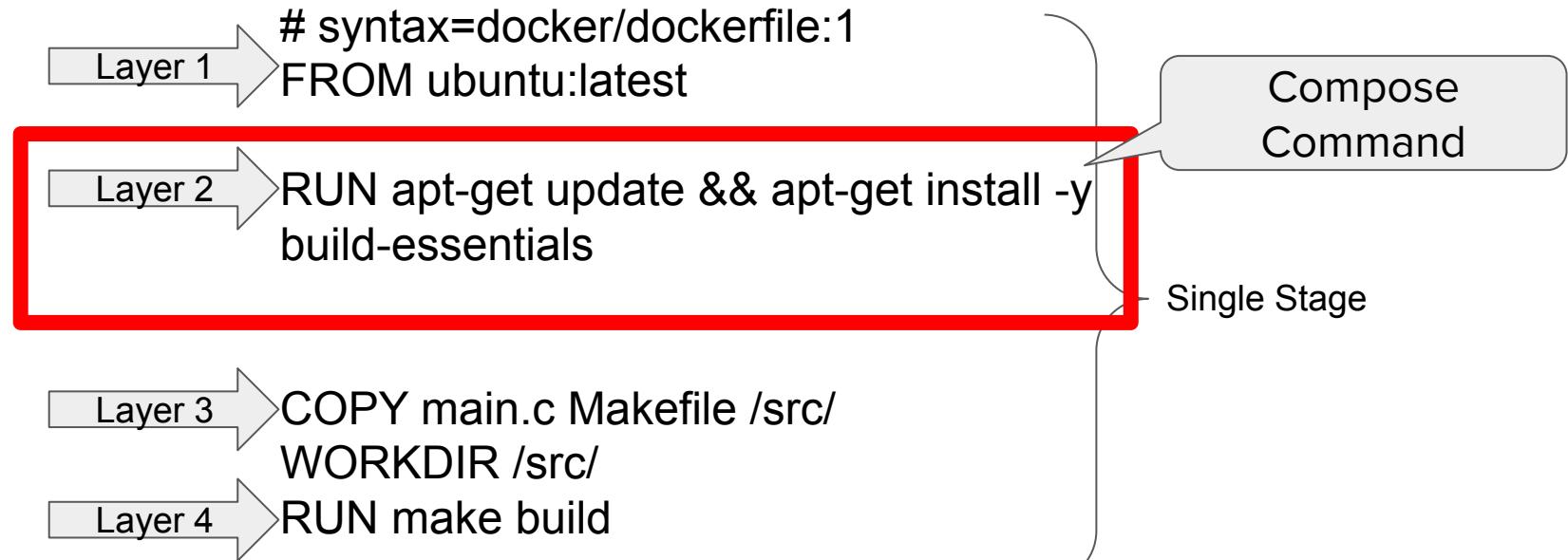
- COW (Copy On Write) Concept - Each Action on docker create layer (Like Code Commit)



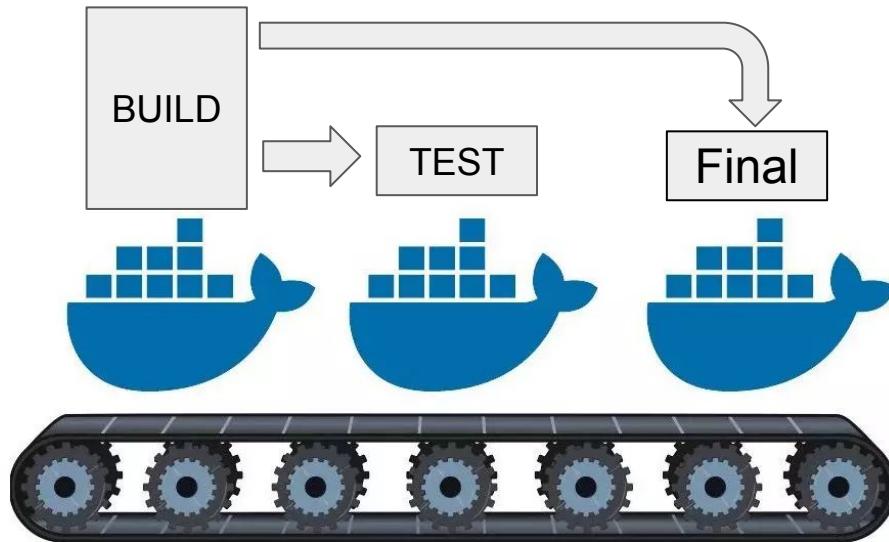
Single Stage - Compose Command

- More Layer > Increase Image Size

Hint: Compose Command into Single Step / or Using Multi-Stage



Multi-Stage



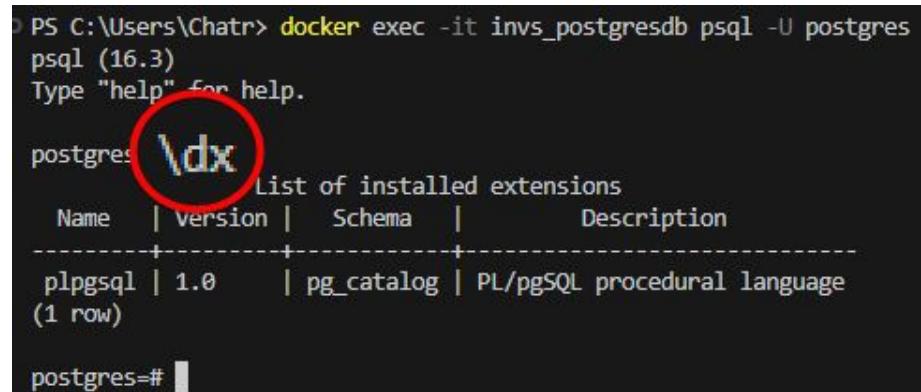
Sample: [go-assessment-tax/Dockerfile at main · pingkunga/go-assessment-tax \(github.com\)](https://github.com/pingkunga/go-assessment-tax)

Blog: [Optimize Container image size | naiwaen@DebuggingSoft](#)

Create Dockerfile-Problem

Problem:

- Manager Ask me for create Postgresql with
 - AI Support ([pgvector/pgvector: Open-source vector similarity search for Postgres \(github.com\)](#))
 - Simple cron-based ([citusdata/pg_cron: Run periodic jobs in PostgreSQL \(github.com\)](#))
- But Postgresql Container Image does not fit with your requirement.



```
PS C:\Users\Chatr> docker exec -it invs_postgresdb psql -U postgres
psql (16.3)
Type "help" for help.

postgres=# \dx
List of installed extensions
Name | Version | Schema | Description
-----+-----+-----+
plpgsql | 1.0 | pg_catalog | PL/pgSQL procedural language
(1 row)

postgres=#
```

Create Dockerfile-Problem

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    AZURE
Error response from daemon: No such container: invs_postgresdb

What's next?
Try Docker Debug for seamless, persistent debugging tools in any container or image → docker debug invs_postgresdb
Learn more at https://docs.docker.com/go/debug-cli/
● PS D:\2WarRoom\9PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> docker exec -it invs_postgresdb_raw psql -U postgres
psql (16.3 (Debian 16.3-1.pgdg120+1))
Type "help" for help.

postgres=# \dx
          List of installed extensions
   Name   | Version | Schema |      Description
-----+-----+-----+
 plpgsql | 1.0     | pg_catalog | PL/pgSQL procedural language
(1 row)

postgres=# Create extension vector;
ERROR: extension "vector" is not available
DETAIL: Could not open extension control file "/usr/share/postgresql/16/extension/vector.control": No such file or directory.
HINT: The extension must first be installed on the system where PostgreSQL is running.
postgres=# CREATE EXTENSION pg_cron;
ERROR: extension "pg_cron" is not available
DETAIL: Could not open extension control file "/usr/share/postgresql/16/extension/pg_cron.control": No such file or directory.
HINT: The extension must first be installed on the system where PostgreSQL is running.
postgres=# exit
```

Create Dockerfile - Custom Docker

```
FROM postgres:16.3-bookworm

#https://packages.debian.org/trixie/amd64/database/postgresql-16-pgvector
#https://packages.debian.org/trixie/amd64/database/postgresql-16-cron

RUN apt-get update \
    && apt-get install -y curl \
    && apt-get -y install postgresql-16-pgvector \
    && apt-get -y install postgresql-16-cron

RUN echo "shared_preload_libraries='pg_cron'" >> /usr/share/postgresql/postgresql.conf.sample

RUN echo "cron.database_name='postgres'" >> /usr/share/postgresql/postgresql.conf.sample

#For your own database name

#RUN echo "shared_preload_libraries='pg_cron'" >> /usr/share/postgresql/postgresql.conf
#RUN echo "cron.database_name='your_db_name'" >> /usr/share/postgresql/postgresql.conf
```

Create Dockerfile - Custom Container Postgres Image

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS AZURE

```
be9b83e393ad169a382ad7512d0367f212019b5b905c75f3fdb66b21bc1f3a16
PS D:\2WarRoom\9PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> docker exec -it invs_postgresdb /bin/sh
# psql -U postgres
psql (16.3 (Debian 16.3-1.pgdg120+1))
Type "help" for help.

postgres=# CREATE EXTENSION pg_cron;
CREATE EXTENSION
postgres=# CREATE EXTENSION vector;
CREATE EXTENSION
postgres=# \nl
postgres=# \dx
          List of installed extensions
   Name   | Version | Schema | Description
   pg_cron | 1.6    | pg_catalog | Job scheduler for PostgreSQL
   plpgsql | 1.0    | pg_catalog | PL/pgSQL procedural language
  vector  | 0.7.0  | public   | vector data type and ivfflat and hnsw access methods
(3 rows)

postgres=#

```

[Top 8 PostgreSQL Extensions
\(timescale.com\)](https://timescale.com)

Create Dockerfile - Custom Container Postgress Image

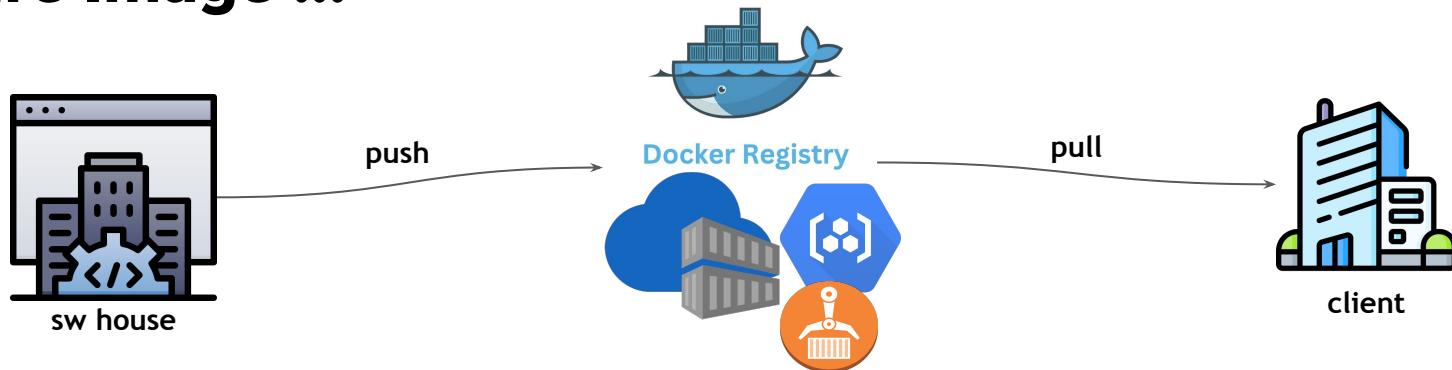
- resource: [Day2/BuildingImage at main · pingkunga/ \(github.com\)](https://github.com/pingkunga/Day2/blob/main/BuildingImage)

Hand On - Building Image

- Download 2 Docker File
 - A [DockerfileA at main · pingkunga/train-docker-at-BRSU \(github.com\)](#)
 - B [DockerfileB at main · pingkunga/train-docker-at-BRSU \(github.com\)](#)
- Build Images
- Use docker images, to compare size

```
PS D:\2WarRoom\9PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> docker images "my_org_postgres*"
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
my_org_postgres    16.3     efdddf767550   3 hours ago  459MB
my_org_postgres-l  16.3     0e7877a24996   7 days ago  462MB
PS D:\2WarRoom\9PublicSpeaker\train-docker-at-BRSU\Day2\BuildingImage> |
```

Share Image !!!



- docker build & push

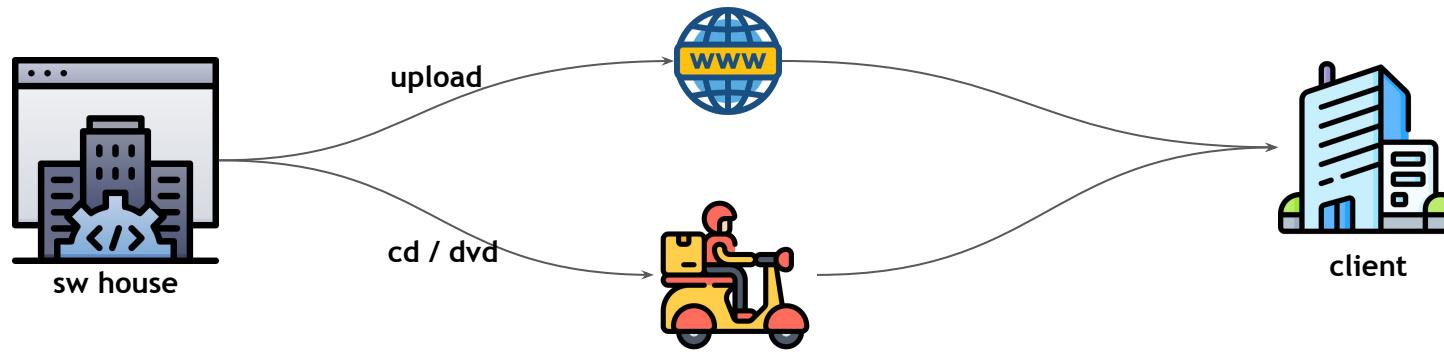
```
#build and push  
docker buildx build --platform linux/amd64,linux/arm64 -t ourapp:latest . --push
```

- some case re-tag & Push

```
docker tag app1:1.0.0-dev some_hub/app1:1.0.0  
docker push some_hub/app1:1.0.0
```

No Internet !!!

THE CUSTOMER SAID YOU **CANNOT USE THE INTERNET**
TO PULL DOCKER IMAGES, AND THERE IS NO
CONTAINER REGISTRY.



- docker save

```
docker save -o sampleapp:25.05.24
```

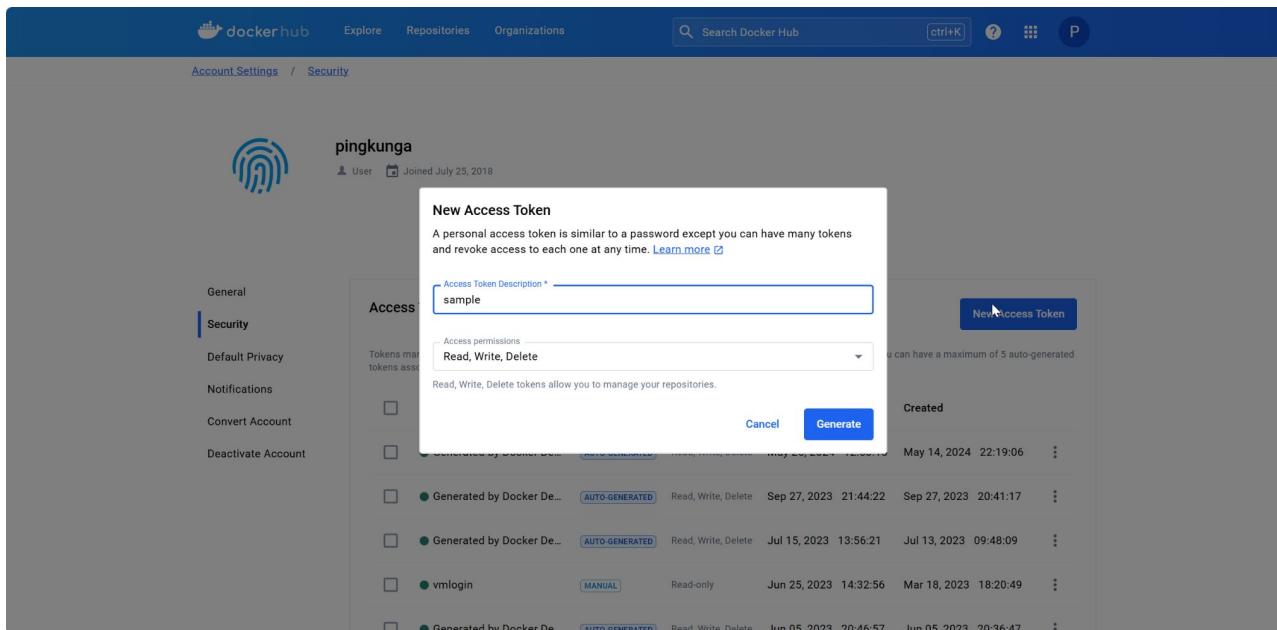
- docker image load

```
docker load --input sampleapp:25.05.24
```

- Note: some case re-tag & Push

Hand-On Push Image to docker hub

- Generate Token & docker login



Hand-On Push Image to docker hub

- Run Command to connect a docker hub

The screenshot shows the Docker Hub security settings page for a user named 'pingkunga'. A modal window titled 'Copy Access Token' is open, providing instructions for using a Docker CLI client. It includes a code block for running 'docker login' and a placeholder for entering the personal access token. A warning message states that the token will only be displayed once and cannot be retrieved later. Below the modal, a table lists existing access tokens with columns for creation date, permissions, and options.

Created	Permissions	Actions
May 14, 2024 22:19:06	Read, Write, Delete	⋮
Sep 27, 2023 20:41:17	Read, Write, Delete	⋮
Jul 13, 2023 09:48:09	Read, Write, Delete	⋮

Hand-On Push Image to docker hub

Case: Build & Push Command

```
docker buildx build --platform linux/amd64 -t  
<docker_username>/my_org_postgres:16.3 . --push
```

Case: Retag & Push Command

```
docker tag my_org_postgres:16.3 <docker_username>/my_org_postgres:16.3
```

```
docker push <docker_username>/my_org_postgres:16.3
```

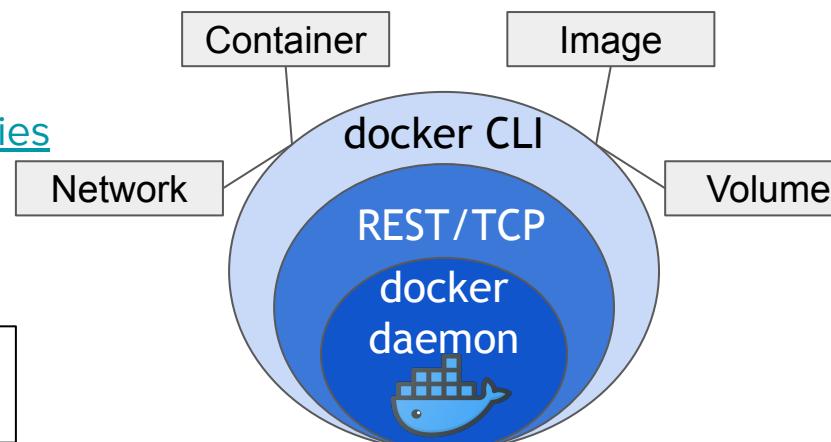
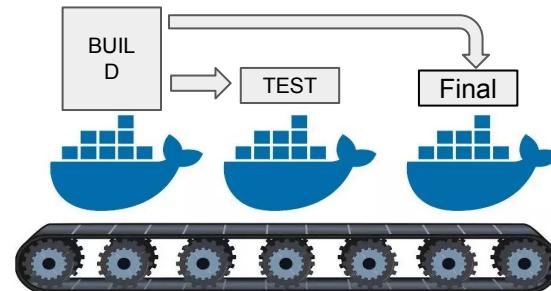
Container Best Practice

- Train your Team !!!
- Use trusted base images
- Keep images up to date
 - Fixed Version
 - With Schedule Update
- Reduce the attack surface (Alpine/ Chiselled / Chainguard Image / Distroless)
- Limit container privileges (Non root user in dockerfile **USER**)
- Limit container resource
- Separate Data from Container (Design Data Path for Mount / Volume)

Container Best Practice

- Reduce Layer
 - Merge Layer
 - Multi-Stage
- Implement Access Controls
 - Host VM
 - Docker
 - o [Docker Security - OWASP Cheat Sheet Series](#)
 - o [CIS Docker Benchmarks \(cisecurity.org\)](#)
 - o [docker/docker-bench-security\(github.com\)](#)
 - K8S

Rootless Container [Enhancing application container security and compliance with Podman \(redhat.com\)](#)



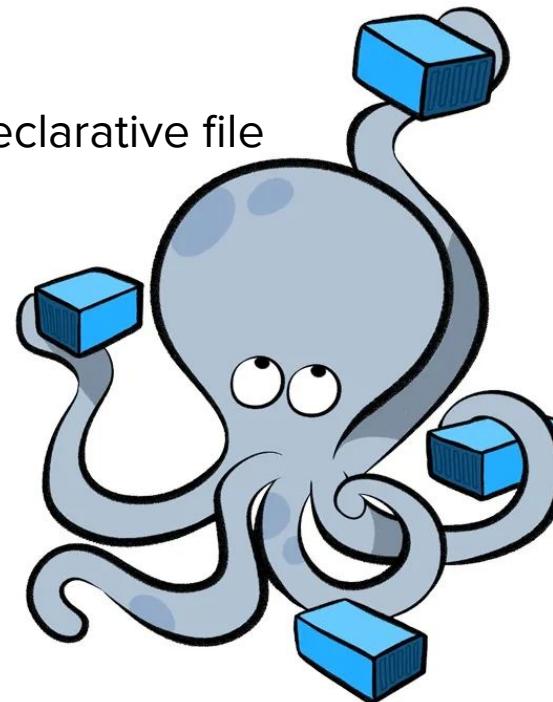
Container Best Practice

- Scan Images for vulnerabilities (Docker Scout / [Trivy](#))
- Container Image Signing - ([Cosign](#) / [docker trust](#)) - supply chain attack
- Implement network security
- Monitor / Audit Activity (Observability)

Docker Compose

Docker Compose

- Old a lot of command
 - Solution: Script .sh / ps
 - Problem: Container Dependency / Human Error
- New way > Define final state of system in declarative file (docker-compose.yaml)



YAML

- YAML is a data-serialization language that is human-friendly format like json
- **Indentation** - recommended indentation for YAML files is two spaces per level

```
Day2 > Compose > postgres > 🚫 docker-compose.yml > {} networks > {} app1dev
      docker-compose.yml - The Compose specification establishes a standard for the definition of multi-container platform-agnostic applications
  1  networks:
  2    #docker network create app1dev
  3    app1dev:
  4      driver: bridge
  5    services:
  6      #docker run --name postgres_app1
  7      #        --network=app1dev
  8      #        -p 5432:5432
  9      #        -e POSTGRES_USER=postgres
 10      #        -e POSTGRES_PASSWORD=postgres
 11      #        -e POSTGRES_DB=taxdb
 12      #        -v ${pwd}/dbdata:/var/lib/postgresql/data
 13      #        -v ${pwd}/init.sql:/docker-entrypoint-initdb.d/init.sql
 14      #        -d postgres:16
 15      postgres_db:
 16        image: postgres:16.0
 17        container_name: local_app1_pgdb
 18        restart: always
 19        #For Expose to External Network
 20        ports:
```

Ref: [A Gentle Introduction to the YAML format - DEV Community](#)

Docker Compose

- Network
- Service
- Config
- Secret
- Volume

Full Sample Compose File

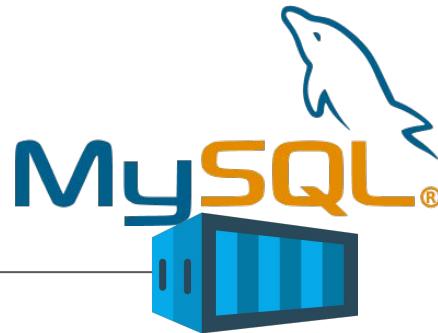
[/Day2/Compose/SampleStructure.yaml](https://Day2/Compose/SampleStructure.yaml)
at main · pingkunga(github.com)

```
Day2 > Compose > ! SampleStructure.yaml > {} volumes > db_data
  1   networks:
  2     wordpress:
  3       driver: bridge
  4
  5   services:
  6   > db: ...
20
21   > wordpress: ...
35
36   # Note Good Practice to use secrets for passwords
37   # config:
38   #   db_root_password:
39   #     file: db_root_password.txt
40   #   db_password:
41   #     file: db_password.txt
42   secrets:
43     db_password:
44       file: db_password.txt
45     db_root_password:
46       file: db_root_password.txt
47
48   volumes:
49     db_data:
```

Convert Docker Command to Compose

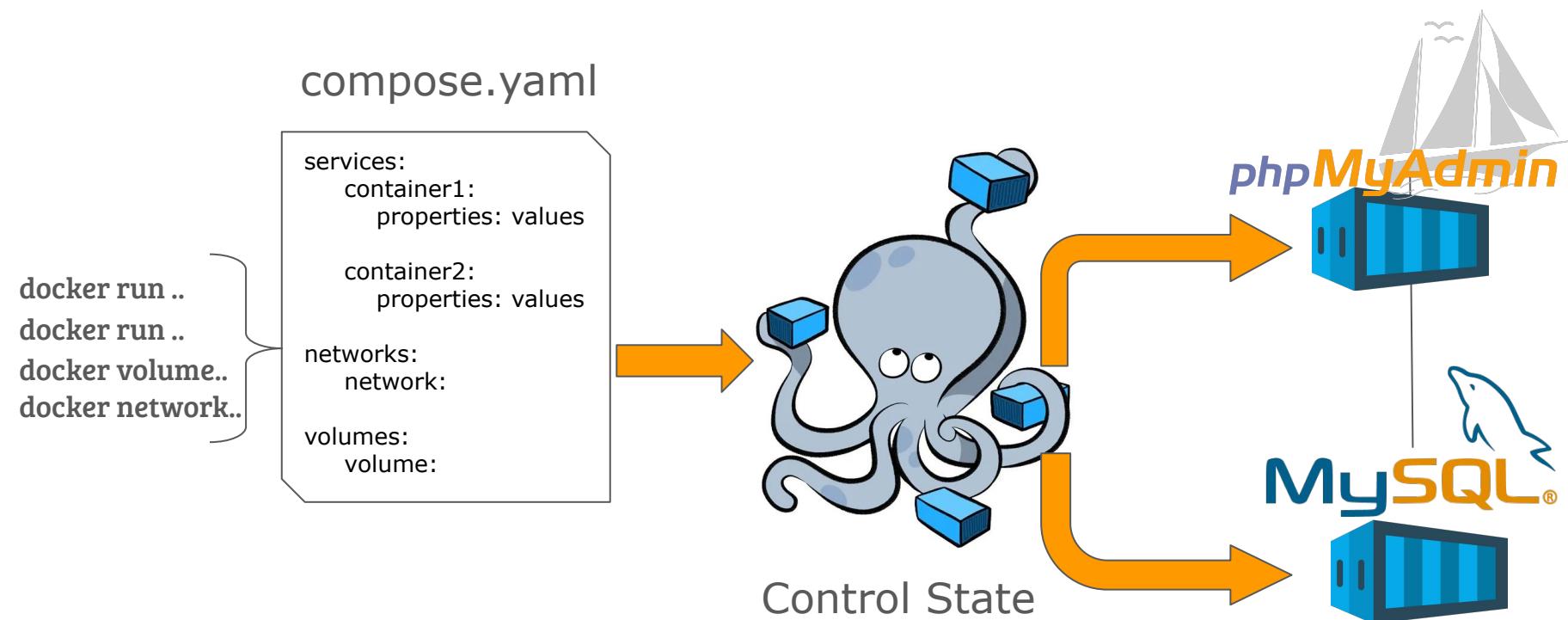


```
docker run -d --name phpmyadmin \
-e PMA_HOST=mysqldb \
-e PMA_USER=root \
-e PMA_PASSWORD=test \
-p 8080:80 \
--restart always \
--link mysqldb\
phpmyadmin/phpmyadmin:5.2
```



```
docker run -d -p 3399:3306 \
--name mysqldb
-e MYSQL_ROOT_PASSWORD=test \
-e MYSQL_DATABASE=dev \
-v mysql-data:/var/lib/mysql \
mysql:8.0
```

Convert Docker Command to Compose



Convert Docker Command to Compose

- Declare Service (Container name and Base Image) create docker-compose.yaml

```
docker run -d -p 3399:3306 \
    --name mysqldb
    -e MYSQL_ROOT_PASSWORD=test \
    -e MYSQL_DATABASE=dev \
    -v mysql-data:/var/lib/mysql \
    mysql:8.0
```

```
services:
  mysqldb:
    image: mysql:8.0
```

Convert Docker Command to Compose

- Next, we bring over the port mapping.

```
docker run -d -p 3399:3306 \
  --name mysqlDb
  -e MYSQL_ROOT_PASSWORD=test \
  -e MYSQL_DATABASE=dev \
  -v mysql-data:/var/lib/mysql \
  mysql:8.0
```

```
services:
  mysqlDb:
    image: mysql:8.0
    ports:
      - 3399:3306
```

Convert Docker Command to Compose

- Next, environment variables.

```
docker run -d -p 3399:3306 \
    --name mysqldb
    -e MYSQL_ROOT_PASSWORD=test \
    -e MYSQL_DATABASE=dev \
    -v mysql-data:/var/lib/mysql \
    mysql:8.0
```

```
services:
  mysqldb:
    image: mysql:8.0
    ports:
      - 3399:3306
    environment:
      - MYSQL_ROOT_PASSWORD=test
      - MYSQL_DATABASE=dev
```

Convert Docker Command to Compose

- Next, define the volume, ensuring our data is persisted across environment restart

```
docker run -d -p 3399:3306 \
    --name mysqldb
    -e MYSQL_ROOT_PASSWORD=test \
    -e MYSQL_DATABASE=dev \
    -v mysql-data:/var/lib/mysql \
    mysql:8.0
```

```
services:
  mysqldb:
    image: mysql:8.0
    ports:
      - 3399:3306
    environment:
      - MYSQL_ROOT_PASSWORD=test
      - MYSQL_DATABASE=dev
    volumes:
      - mysql-data:/var/lib/mysql
volumes:
  mysql-data:
```

Convert Docker Command to Compose

- check point: test docker compose file
 - [Day2/Compose/mysql/Compose1.yaml at main · pingkunga \(github.com\)](#)

Basic Compose commands

```
# Use Compose to reconcile and launch what's defined  
docker compose up
```

```
# Start Compose in background mode  
docker compose up -d
```

```
# Monitor Docker Compose Status  
docker compose ps
```

```
# Tear everything down (leaves volumes by default)  
docker compose down
```

```
# Share logs from all of the application services  
docker compose logs
```

```
# or specific compose file  
docker compose -f <your-compose-file>.yaml up / down / logs
```

Convert Docker Command to Compose

- Next Convert docker command for phpmyadmin to compose

```
docker run -d --name phpmyadmin \
-e PMA_HOST=mysqldb \
-e PMA_USER=root \
-e PMA_PASSWORD=test \
-p 8080:80 \
--restart always \
--link mysql \
phpmyadmin/phpmyadmin:5.2
```

```
services:
  mysqldb:
    ...
  phpmyadmin:
    image: phpmyadmin:5.2
    ports:
      - 8080:80
    restart: always
    depends_on:
      - mysqldb
    links:
      - mysqldb
    environment:
      PMA_HOST: mysqldb
      PMA_USER: root
      PMA_PASSWORD: test
    volumes:
      mysql-data:
```

Convert Docker Command to Compose

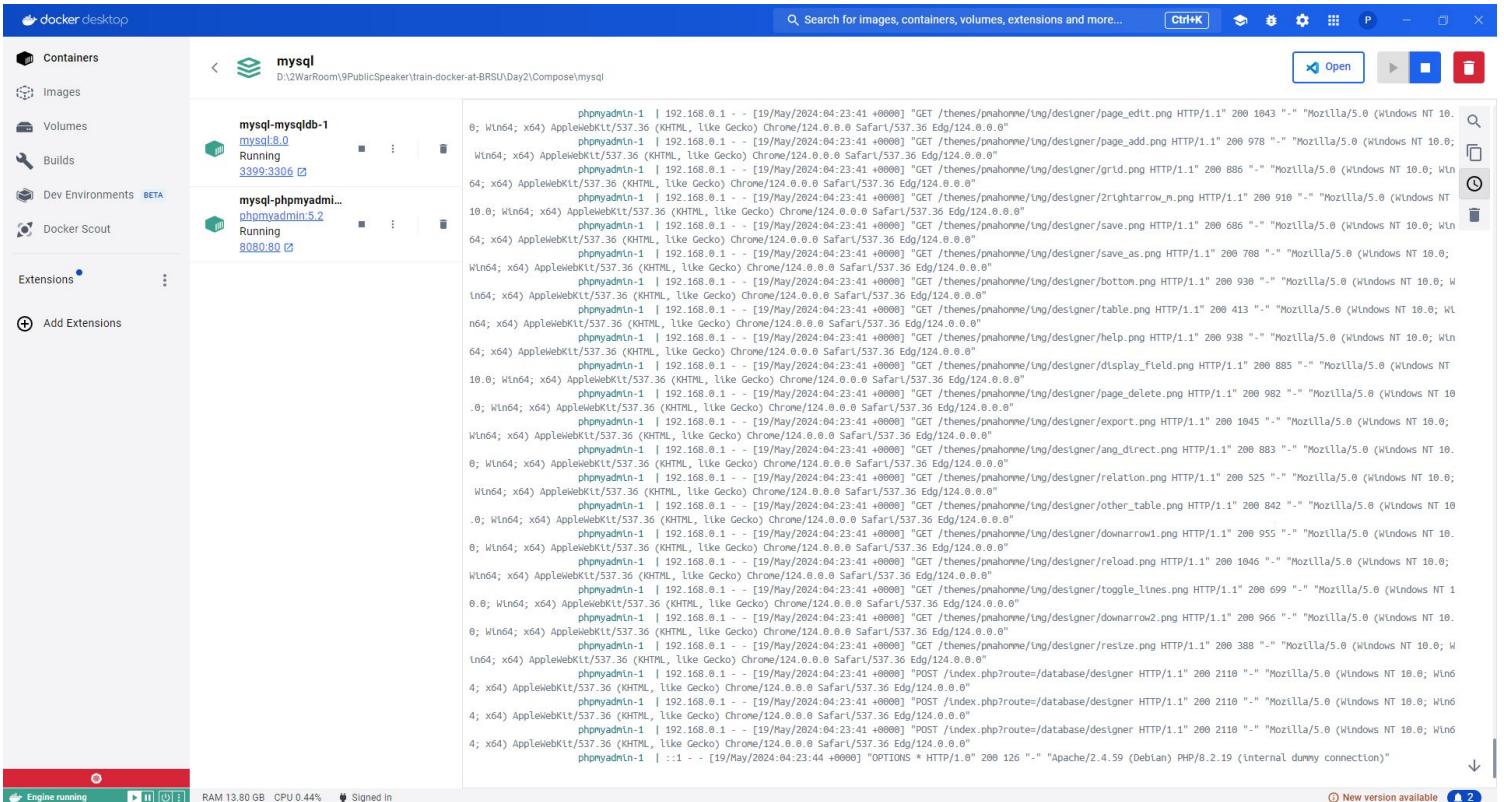
- Full Compose File: [mysql/Compose.yaml at main · pingkunga \(github.com\)](#)
- Test

docker compose - Docker Desktop

The screenshot shows the Docker Desktop interface with the 'Containers' tab selected. The sidebar on the left includes 'Containers', 'Images', 'Volumes', 'Builds', 'Dev Environments BETA', 'Docker Scout', and 'Extensions' (marked as active). The main area displays a table of running containers with columns for Name, Image, Status, CPU (%), Port(s), Last started, and Actions. A search bar and a filter for 'Only show running containers' are at the top. A red box highlights a group of four containers under the name 'mysql': 'mysql' (Running, 0.47%, 34 minutes ago), 'phpmyadmin-1' (Running, 0%, 8080:80, 34 minutes ago), 'mysqldb-1' (Running, 0.47%, 3399:3306, 34 minutes ago), and 'mysql:8.0' (Running, 0.47%, 3399:3306, 34 minutes ago). The status bar at the bottom indicates 'Engine running', system resources (RAM 13.53 GB, CPU 1.63%), and user status (Signed in). A notification for a new version is shown in the bottom right.

Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
2d6554c6538c						
k8s_bojpawnfront_bojpawnfront-75d844678b-k66t_group-1-1	pingkunqa/bojpawnfront	Running	0.01%	15 hours ago	[Actions]	
k8s_agent_bojappobs-78d795b4f-5knm_group-1-obs_3cac0r	grafana/agent	Running	0.87%	15 hours ago	[Actions]	
6d49d3fb3b92						
k8s_bojpawnfront_bojpawnfront-75d844678b-bxn8z_group-1-	pingkunqa/bojpawnfront	Running	0.02%	15 hours ago	[Actions]	
749b7205984						
k8s_bojpawnapi_bojpawnapi-7449584d97-217s8_group-1-boj	pingkunqa/bojpawnapi	Running	0.58%	15 hours ago	[Actions]	
31dc229b4aca						
k8s_bojpawnfront_bojpawnfront-75d844678b-48k4z_group-1-	pingkunqa/bojpawnfront	Running	0.01%	15 hours ago	[Actions]	
c9a5573e92ac						
workshop-summer-group-5-b1		Exited	0%	7 days ago	[Actions]	
go-assessment-tax		Exited	0%	21 days ago	[Actions]	
go-fun-exercise-api		Exited	0%	1 month ago	[Actions]	
postgresql		Exited	0%	15 hours ago	[Actions]	
mysql		Running (2/2)	0.47%	34 minutes ago	[Actions]	
phpmyadmin-1	phpmyadmin:5.2	Running	0%	8080:80	34 minutes ago [Actions]	
mysqldb-1	mysql:8.0	Running	0.47%	3399:3306	34 minutes ago [Actions]	

docker compose - Docker Desktop



DOCKER COMPOSE CHEAT SHEET

File

structure

```
services:
```

```
  container1:
    properties: values
```

```
  container2:
    properties: values
```

```
networks:
```

```
  network:
```

```
volumes:
```

```
  volume:
```

Types

value

```
key: value
```

array

```
key:
  - value
  - value
```

dictionary

```
master:
  key: value
  key: value
```

Properties

build

```
build image from dockerfile
in specified directory
```

container

```
  build: ./path
  image: image-name
```

image

```
use specified image
```

```
image: image-name
```

container_name

```
define container name to access
it later
```

```
container_name: name
```

volumes

```
define container volumes to
persist data
```

volumes:

```
  - /path:/path
```

command

```
override start command for the
container
```

```
command: execute
```

environment

```
define env variables for the
container
```

environment:

```
  KEY: VALUE
```

```
---
```

environment:

```
  - KEY=VALUE
```

env_file

```
define a env file for the
container to set and override
env variables
```

```
env_file: .env
```

```
---
```

env_file:

```
  - .env
```

restart

```
define restart rule
(no, always, on-failure, unless-
stopped)
```

expose:

```
  - "9999"
```

networks

```
define all networks for the
container
```

networks:

```
  - network-name
```

ports

```
define ports to expose to other
containers and host
```

ports:

```
  - "9999:9999"
```

expose

```
define ports to expose only to
other containers
```

expose:

```
  - "9999"
```

network_mode

```
define network driver
(bridge, host, none, etc.)
```

```
network_mode: host
```

depends_on

```
define build, start and stop
order of container
```

depends_on:

```
  - container-name
```

Other

idle container

```
send container to idle state
> container will not stop
```

```
command: tail -f /dev/null
```

named volumes

```
create volumes that can be used in
the volumes property
```

services:

```
  container:
```

```
    image: image-name
```

```
    volumes:
```

```
      - data-
```

```
        volume:/path/to/dir
```

volumes:

```
  data-volume:
```

networks

```
create networks that can be used in
the networks property
```

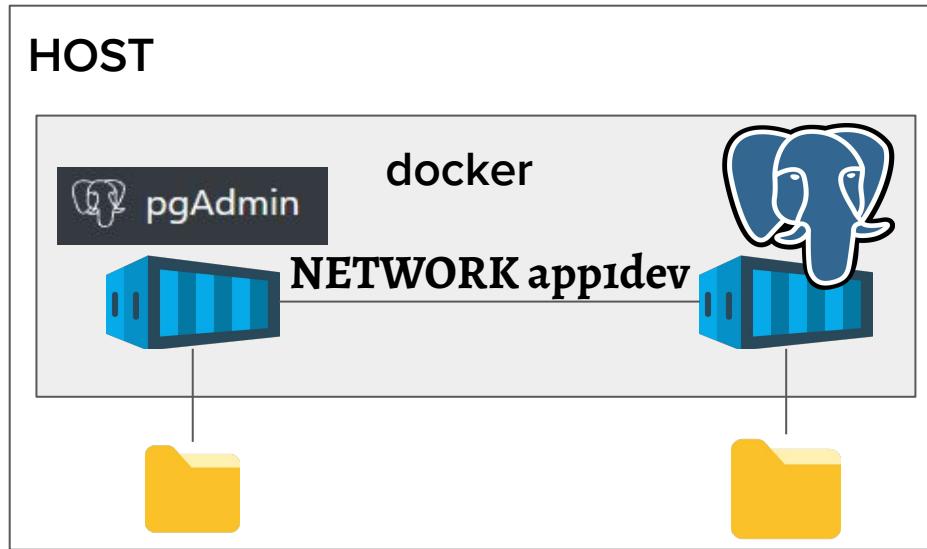
networks:

```
  frontend:
```

```
    driver: bridge
```

- Copy & Paste (Template) [docker/awesome-compose: Awesome Docker Compose samples \(github.com\)](https://github.com/docker/awesome-compose)

Hand On + Exercise



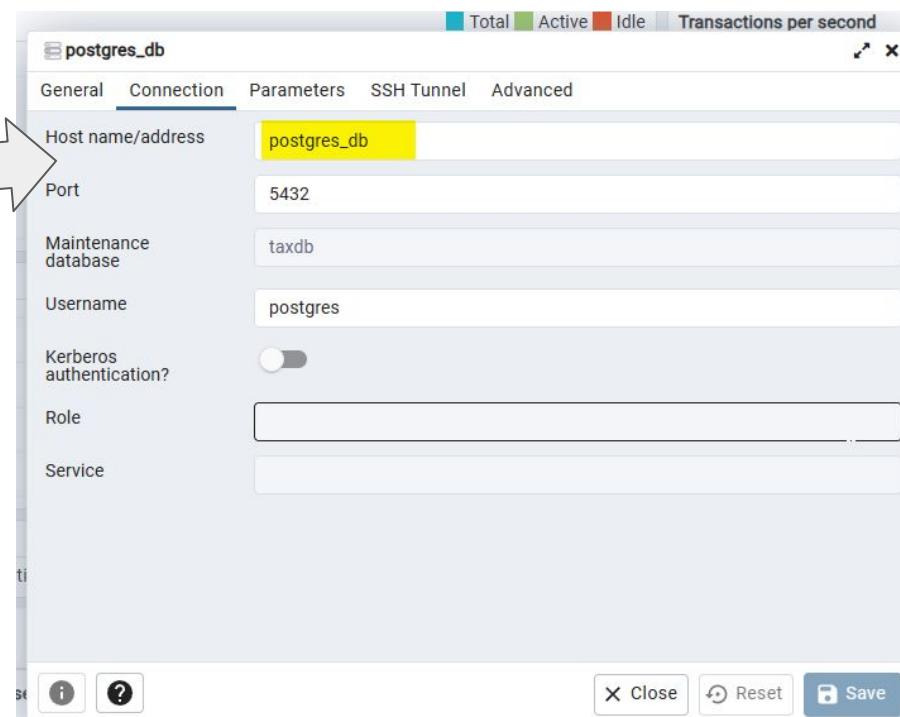
Hand On + Exercise Solution

- Full docker compose file [postgress/docker-compose.yaml at main · pingkunga \(github.com\)](#)

Hand On + Exercise Solution

- Test

```
1 docker-compose.yml - The Compose specification establishes a standard for the definition of
2 networks:
3   app1dev:
4     driver: bridge
5 services:
6   postgres_db:
7     image: postgres:16.0
8     container_name: local_app1_pgdb
9     restart: always
10    ports:
11      - "5432:5432"
12    environment:
13      POSTGRES_USER: postgres
14      POSTGRES_PASSWORD: postgres
15      POSTGRES_DB: taxdb
16    volumes:
17      #- local_app1_pgdata:/var/lib/postgresql/data
18      - ./dbdata:/var/lib/postgresql/data
19      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
20    networks:
21      - app1dev
22    healthcheck:
23      test: ["CMD-SHELL", "pg_isready"]
```



Hand On + Exercise Solution

- health check

```
23 pgadmin:
24   image: dpage/pgadmin4
25   container_name: pgadmin4_app1_container
26   restart: always
27   ports:
28     - "5050:80"
29   environment:
30     PGADMIN_DEFAULT_EMAIL: pingkunga@example.com
31     PGADMIN_DEFAULT_PASSWORD: VeryStrongP@ssw0rd
32   volumes:
33     - pgadmin-app1-data:/var/lib/pgadmin
34   networks:
35     - app1dev
36   depends_on:
37     postgres_db:
38       condition: service_healthy
39   volumes:
40     local_app1_pgdata:
41     pgadmin-app1-data:
```

```
1 docker-compose.yml - The Compose specification establishes a standard for the definition of
2 networks:
3   app1dev:
4     driver: bridge
5   services:
6     postgres_db:
7       image: postgres:16.0
8       container_name: local_app1_pgdb
9       restart: always
10      ports:
11        - "5432:5432"
12      environment:
13        POSTGRES_USER: postgres
14        POSTGRES_PASSWORD: postgres
15        POSTGRES_DB: taxdb
16      volumes:
17        #- local_app1_pgdata:/var/lib/postgresql/data
18        - ./dbdata:/var/lib/postgresql/data
19        - ./init.sql:/docker-entrypoint-initdb.d/init.sql
20      networks:
21        - app1dev
22      healthcheck:
23        test: ["CMD-SHELL", "pg_isready"]
```



Ref: [Services elements - Health Check | Docker Docs](#)

docker desktop

Containers [Give feedback](#)

Container CPU usage: 14.16% / 1600% (16 CPUs available)

Container memory usage: 2.99GB / 14.87GB

Show charts

Search

Only show running containers

	Name	Image	Status	CPU (%)	Port(s)	Last started	Actions
<input type="checkbox"/>	k8s_bojpaawnfront_bojps	pingkunga/bojpaawnfront 7fc6e11a2b7e	Running	0.01%	7 hours ago		
<input type="checkbox"/>	k8s_bojpaawnfront_bojps	pingkunga/bojpaawnfront e67d58d075dd	Running	0.01%	7 hours ago		
<input type="checkbox"/>	k8s_bojpaawnapi_bojpaaw	pingkunga/bojpaawnapi 7d20ec256fc6	Running	0.04%	7 hours ago		
<input type="checkbox"/>	k8s_bojpaawnfront_bojps	pingkunga/bojpaawnfront 45de06db7e12	Running	0.01%	7 hours ago		
<input type="checkbox"/>	go-assessment-tax		Exited	0%	28 days ago		
<input type="checkbox"/>	go-fun-exercise-api		Exited	0%	2 months ago		
<input type="checkbox"/>	postgresql		Exited	0%	7 hours ago		
<input type="checkbox"/>	postgress		Running (1/2)	0%	7 seconds ago		
<input type="checkbox"/>	pgadmin4_app1_cont	dpage/pgadmin4 2b96643218e8	Created	0%	5050:80		
<input type="checkbox"/>	local_app1_pgdb	postgres:16.0 c8778766d5f0	Running	0%	5432:5432	7 seconds ago	

Showing 28 items

Engine running RAM 12.66 GB CPU 3.88% Signed in

New version available

Hand On + Exercise Solution

- Resource Limit

```
docker-compose.yml - The Compose specification establishes a standard for the definition of multi-container platform-agnostic applications (compose-spec.json)
1 networks:
2   app1dev:
3     driver: bridge
4 services:
5   postgres_db:
6     image: postgres:16.0
7     container_name: local_app1_pgdb
8     restart: always
9     #For Expose to External Network
10    ports:
11      - "5432:5432"
12    environment:
13      POSTGRES_USER: postgres
14      POSTGRES_PASSWORD: postgres
15      POSTGRES_DB: taxdb
16    volumes:
17      #- local_app1_pgdata:/var/lib/postgresql/data
18      - ./dbdata:/var/lib/postgresql/data
19      - ./init.sql:/docker-entrypoint-initdb.d/init.sql
20    networks:
21      - app1dev
22    healthcheck:
23      test: ["CMD-SHELL", "pg_isready"]
24      #Sample recommendation not fixed cpuset
25      #cpuset: "0-1"
26    deploy:
27      resources:
28        limits:
29          cpus: '0.50'
30          memory: 512M
31        reservations:
32          cpus: '0.25'
33          memory: 256M
34      > pgadmin:...
35      volumes:
36        local_app1_pgdata:
37        pgadmin-app1-data:
```

Ref: [Compose Deploy+ Resource Specification | Docker Docs](#)

Portainer

Portainer

Upgrade to Business Edition

portainer.io COMMUNITY EDITION

Home

Environment: None selected

Settings

- Users
- Environments
- Registries
- Authentication logs
- Notifications
- Settings

Latest News From Portainer

Portainer version 2.20.3 STS is now available for users on our Short-Term Support branch. As well as bug fixes and performance improvements, this release includes support for MicroK8s 1.30, offline support for our MicroK8s provisioning, and usability improvements to Edge Configurations.

Dismiss

Environments

Click on an environment to manage

Search by name, group, tag, status, URL... Refresh Kubeconfig

Sort By ↑

Environment	Status	Created	Type	IP Address	Live connect	Actions
local	Up	2024-05-26 21:10:04	Standalone	24.0.2 /var/run/docker.sock	Disconnected	Edit Logs
nuc(192.168.1.222)	Up	2024-05-26 21:10:04	Standalone	24.0.5 192.168.1.222:9001	Disconnected	Edit Logs
podman (128.1.0.181)	Up	2024-05-26 21:10:05	Standalone	4.6.1 128.1.0.181:18888	Disconnected	Edit Logs

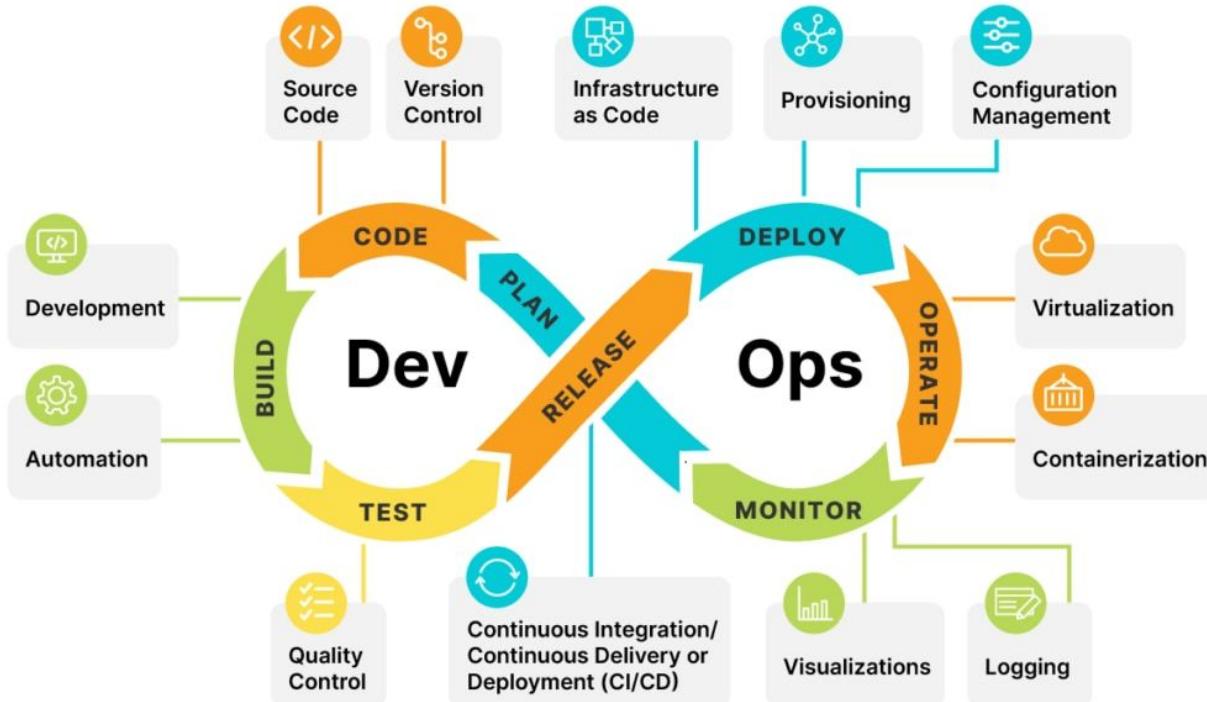
Items per page: 10

Introduction to Portainer

- Setup [Day2/Portainer/docker-compose.yaml at main · pingkunga \(github.com\)](#)
- Add Agent [Install Portainer Agent on Docker Standalone | 2.19 | Portainer Documentation](#)
 - Secure “**-e AGENT_SECRET=yoursecret**”
- Manage & Deploy Stack (Docker Compose)

Ref: [Getting Started - Portainer Knowledge Base](#)

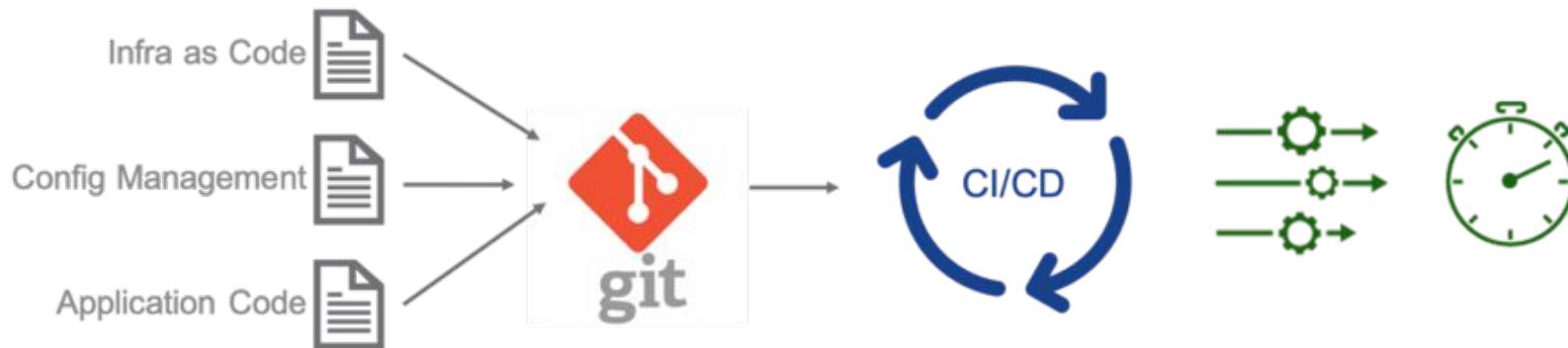
Next:Dev[XXX]Ops



Ref: [What Is DevOps? Complete Guide to Best Practices - Orange Matter \(solarwinds.com\)](https://www.solarwinds.com/resources/what-is-devops)

Next: GitOps

GitOps-in-a-nutshell



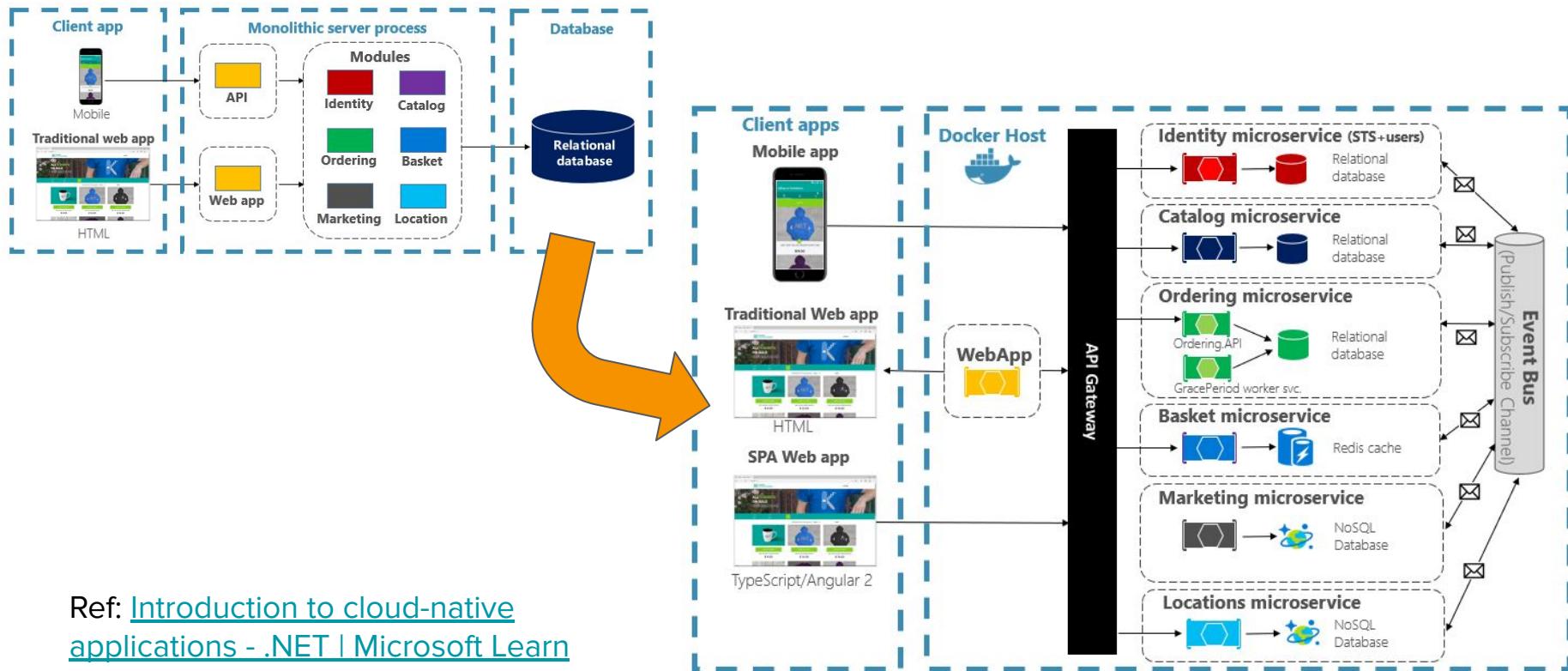
“Source of Truth” for declarative code

Update to code source triggers a pipeline

Pipeline runs a series of tasks, resulting in the update of the runtime environment to match the source

Ref: [GitOps Methodologies, Process, and Best Practices | Spiceworks - Spiceworks](#)

Next: Cloud Native



Ref: [Introduction to cloud-native applications - .NET | Microsoft Learn](#)

Wrap-up

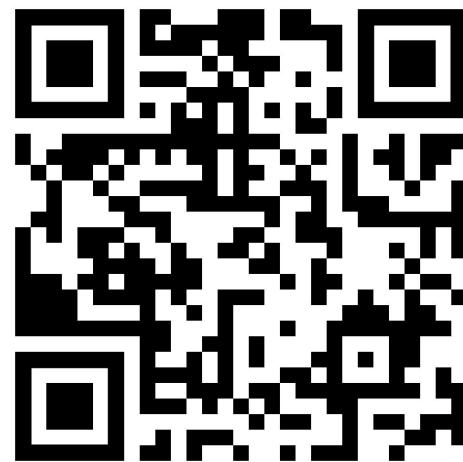
- Container
- Docker [docker cli | Docker Docs](#)
- Docker Compose [docker compose specification | Docker Docs](#)
- Use-Case
 - DevOps
 - GitOps
 - Cloud Native App

Thank you

Resource [pingkunga/train-docker-at-BRSU](https://github.com/pingkunga/train-docker-at-BRSU)
(github.com)



Feedback <https://forms.gle/ySmFcNZawv3MDyQDA>



Find me on:



CONTACT ME

Others

Task Manager

Type a name, publisher, or PID to search

Startup apps

Run new task

Enable

Disable

Properties

Last BIOS time: 17.2 seconds

Name	Publisher	Status	Startup impact
Xbox App Services	Microsoft Corporation	Disabled	None
Spotify	Spotify AB	Disabled	None
Lenovo Hotkeys	LENOVO INC	Disabled	None
SamsungNotesBridgeWPF	Samsung Electronics Co...	Disabled	None
BrotherHelp.exe		Disabled	None
Docker Desktop.exe		Disabled	None
WebexHost.exe		Disabled	None
Notion.exe		Disabled	None
Update.exe		Disabled	None
steam.exe		Disabled	None
Update.exe		Disabled	None
LenovoVantage.exe		Disabled	None
msedae.exe		Disabled	None

Docker Desktop.exe

Disabled

159

docker : invalid reference format - Stack Overflow

28 Answers

Sorted by: Highest score (default)



In powershell you should use `$(pwd)` instead of `$(pwd)`

278

Share Edit Follow Flag

edited Mar 5, 2021 at 2:35



SiddAjmera

39k 5 74 111

answered Jul 17, 2018 at 10:59



Levon Petrosyan

9,345 8 56 68



37 THIS is what caused me to bang my head into the wall (sometimes I hate M\$FT) – [beerwin](#) Jan 15, 2019 at 10:25

1 `docker run --rm -ti --name zalenium -p 4444:4444 -p 5555:5555 \ -e SAUCE_USERNAME -e SAUCE_ACCESS_KEY \ -v /tmp/videos:/home/seluser/videos \ -v /var/run/docker.sock:/var/run/docker.sock \ dosel/zalenium start -- sauceLabsEnabled true` what's wrong with my command? It is also giving same error. – [paul](#) Feb 4, 2019 at 5:45

2 For anyone having problem like @paul for a perfectly correct command, Please refer to this answer stackoverflow.com/a/65690853/807104 – [Mohd Abdul Muiib](#) Jan 12, 2021 at 19:37

docker & syslog

- [จัดการ Log จาก Docker container ด้วย Fluentd \(somkiat.cc\)](#)
- [Fluentd logging driver | Docker Docs](#)
- [An Overview of Syslog Parsing with Fluentd | Tanmay Bhat \(tanmay-bhat.github.io\)](#)

ทด Postgres Note for Legacy DB

- [PostgreSQL : คำสั่ง SQL การ Create Database \(mindphp.com\)](#)
- [สร้าง Database เพื่อให้ใช้การเรียบลำดับ ภาษาไทยได้ | Simple has no boundaries... \(wordpress.com\)](#)
- [แก้ปัญหา update ubuntu server แล้วแจ้ง error ว่า perl: warning: Setting locale failed. – CoP PSU IT Blog](#)
- [How to change DATE command output language locales in Alpine-Linux? - Linux - nixCraft Linux/Unix Forum](#)