전남대학교
소프트웨어중심대학사업단

도커와 쿠버네티스

도커 컨테이너 개념 및 활용

유재명(wesleyok@jnu.ac.kr)

자료실 https://github.com/yoojaemyeong/docker-project

목차

✦ 인터넷 미래 기술

분야	기술	설명
초고속·저지연 네트워크	**5G, 6G, 초저지연 인터넷**	밀리초(ms) 단위 지연, 초당 Tbps급 속도 제공, 실시간 XR·자율주행 지원
차세대 IP·네트워크 아키텍처	**IPv6, SDN, NFV**(Network Functions Virtualization)	무한대 주소 체계, 소프트웨어 정의 네트워크로 유연한 트래픽 관리
AI 기반 지능형 네트워크	**자율 네트워킹, 예측 라우팅**	AI로 트래픽 최적화, **자동 보안 대응**, 사용자 맞춤형 QoS 제공
사물인터넷(IoT) 확장	**스마트홈, 스마트시티**	수십억 대 디바이스가 인터넷에 연결, 센서 데이터 기반 자동화
엣지 컴퓨팅 & 클라우드	**분산형 컴퓨팅, MEC**	데이터 처리 지연을 최소화, 사용자 가까이에서 실시간 데이터 분석
양자 인터넷	**양자 암호통신(QKD)**	절대 보안성을 가진 양자 키 분배, 양자 컴퓨터 간 네트워크 구축
메타버스·Web 3.0	**탈중앙화, 블록체인 기반 웹**	가상 세계와 현실을 융합한 초연결 인터넷, 사용자 주도 데이터 소유권

1-2. 미래 인터넷의 특징

- 초연결 사회(Hyper-Connectivity)
  - 모든 사람, 기기, 서비스가 네트워크로 연결
  - IoT, V2X(Vehicle to Everything), 6G기반 실시간 데이터 공유
  - 예시: 스마트시티, 자율주행차량, 스마트팩토리
- 지능형 네트워크(Intelligent Network)
  - AI가 네트워크 트래픽과 보안을 실시간으로 제어
  - 사용자 소비 패턴에 따라 최적 경로 제공
  - 예시: AI 라우터, 자가 치유(Self-Healing) 네트워크
- 탈중앙화(Web 3.0 분산형 클라우드 서비스, 양자 암호 통신)
- 실감형 서비스 확산(메타버스, AR/VR/XR 실시간 스트리밍)
- 지속가능성과 에너지 효율성

1-3. 6G 특성(1)

✦ 메타버스, 홀로그램 통신

- 기술적 요구: 초고속, 저지연, 초실감 데이터 전송(AR/VR, XR)
- 사례
  - 실시간 3D 홀로그램 회의
  - 가상 콘서트, 교육, 온라인 박람회에서 초실감 체험 제공
  - AR/VR 기반 스마트 제조, 원격 현장 지원

✦ 자율주행차(V2X, Vehicle-to-Everything)

- 기술적 요구: 0.1ms 수준의 초저지연
- 사례
  - 차량-차량(V2X)간 실시간 충돌 방지 정보 교환
  - 무인차량과 드론택시가 도로, 공중에서 실시간 경로 협력

✦ 원격 의료, 원격 로봇 수술

- 기술적 요구: 초고속 데이터 전송 + 초저지연 + 절대 보안
- 사례
  - 전문의가 수천 km 떨어진 환자에게 로봇팔로 0.1ms 반응속도로 수술 진행
  - 초고해상도 MRI 데이터 3D 전송하여 정확한 진단 지원

1-3. 6G 특성(2)

✦ 위성 기반 글로벌 인터넷 커버리지

- 기술적 요구: 지상망 + 위성망 통합, 초연결
- 사례
  - 해양, 사막, 극지방 등 5G 사각지대에서 초고속 인터넷 제공
  - 우주여행, 항공기 내 실시간 스트리밍 서비스
  - 글로벌 재난 상황에서 긴급 통신망 복구

✦ 초실감형 미디어, 실시간 클라우드

- 기술적 요구: Tbps급 전송 속도, 고신뢰성
- 사례
  - 영화, 게임을 다운로드 없이 실시간 클라우드에서 16K 품질로 스트리밍
  - 스포츠 경기 현장을 실시간 360도 VR로 관람
  - 무대 공연 실시간 동기화

✦ 클라우드 서비스 개요

- **인터넷을 통해 컴퓨팅 자원(서버, 스토리지, 네트워크, 애플리케이션 등)을 온디맨드(On-Demand) 방식으로 제공하는 기술**
  - 사용자 요청에 따라 컴퓨팅 자원(CPU, 메모리, 스토리지 등) 제공
  - 예시: AWS EC2, 구글 클라우드, 네이버 클라우드 등
- 전통적인 방식(로컬 서버 운영 방식)보다 유연성, 확장성, 비용 효율성에 큰 장점 있음
- 인터넷만 연결되어 있으면 시간, 장소, 디바이스에 구애받지 않고 서비스 이용
  - PC, 모바일, 태블릿 등 다양한 클라이언트 기기에서 접근 가능
  - 표준 네트워크 프로토콜(HTTP, HTTPS) 기반 통신
  - 예시: 구글 Docs, MS 365문서 등을 웹브라우저로 편집
- 리소스 풀링 및 멀티테넌시
  - 여러 고객이 하나의 클라우드 인프라 자원을 공유하면서, 논리적으로 분리된 환경에서 작업
- **사용한 자원에 따라 자동으로 사용량 모니터링 및 종량제 과금 적용**

1-5. 컨테이너 인프라 환경

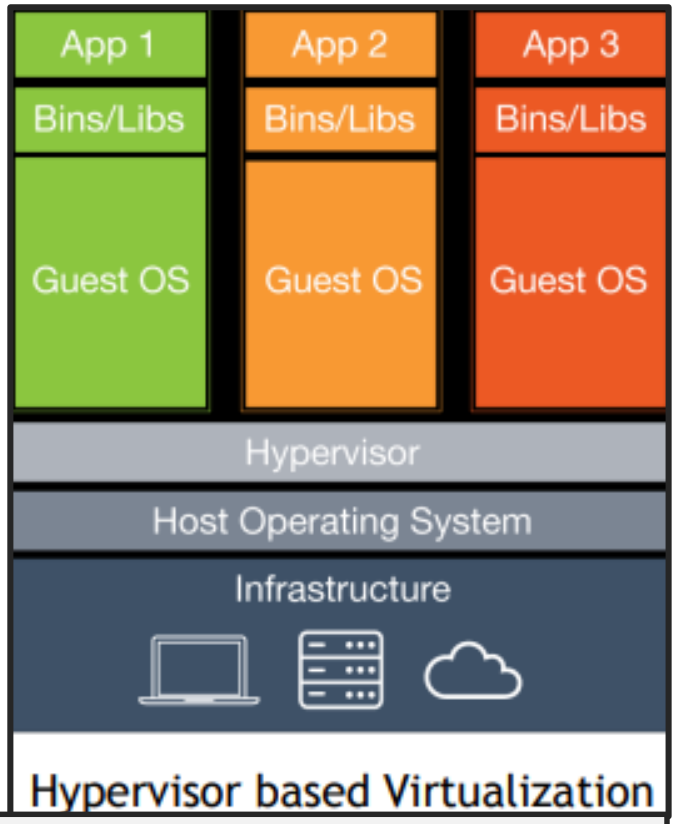
✦ 아키텍처

- Monolithic Service : 하나의 큰 목적이 있는 서비스 또는 애플리케이션에 여러 기능이 **통합**되어 있는 구조
- **Micro Service** : 시스템 전체가 하나의 목적을 지향하는 바는 같으나, 개별 기능 각각을 개발해 연결하는 데서 차이가 있음. 곧 보안, 인증 등과 관련된 기능이 **독립**된 서비스를 구성하고 있으며 다른 서비스들도 독립적으로 동작할 수 있는 구조

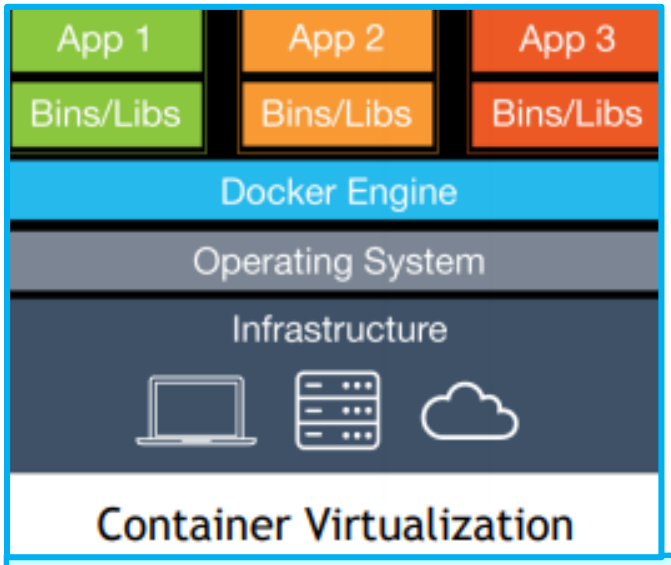
✦ Micro Service: 컨테이너 인프라 환경

- 컨테이너를 중심으로 구성된 인프라 환경
- 컨테이너란 하나의 운영체제안에서 다른 프로세스에 영향을 받지 않고 독립적으로 실행되는 프로세스 상태
  - 격리 환경: **Cgroup(Control Group), Namespace 같은 커널 생성 기술** 활용
    - Cgroup: 시스템의 CPU, 메모리, 네트워크 대역폭과 같은 자원을 제한하고 격리하는 기능
    - Namespace: 시스템 리소스를 프로세스 전용 자원처럼 분리하는 기능
      - 구성요소 : Mount, PID, Network, IPC, UTS(Unix Time Sharing), USER

1-6. 모놀리틱 vs. 마이크로서비스

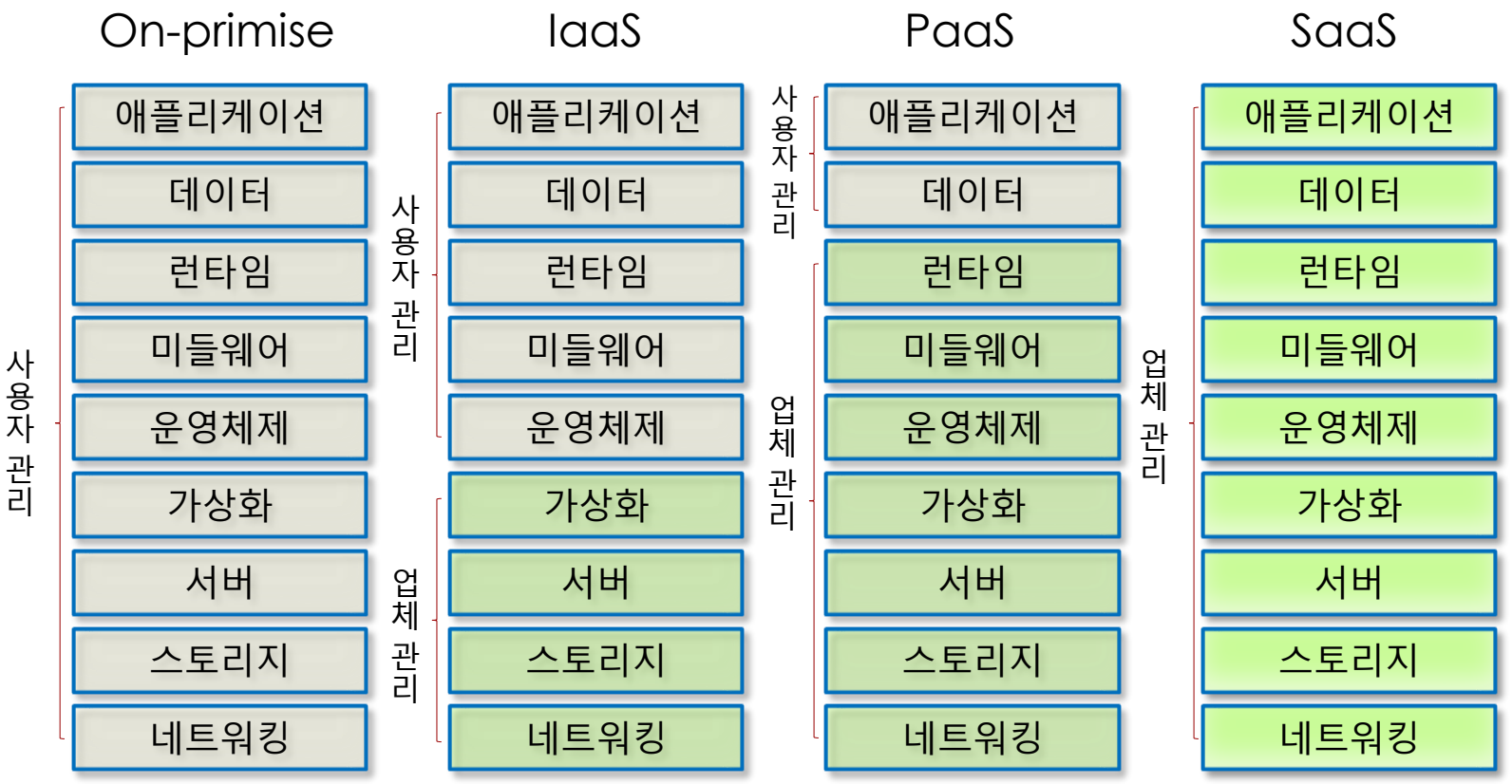


- 소프트웨어가 하나의 결합된 코드로 구성
- 초기 단계에서 설계가 용이, 단순하나 규모가 커지고, 수정이 많을 경우, 연관성이 복잡해지고, 추가 인프라 증설이 어려움



- 개별 기능에 따라 각각 개발
- 각 모듈별 초기 개발이 어려우나, 추후 재사용이 쉽고, 추가 서비스 생성 또는 확장성 좋음

1-7. IaaS, PaaS, SaaS의 구분



※ On-premise : 기업이나 개인이 자체적으로 서버, SW 등 IT인프라를 구매, 설치, 운영하는 방식

✦ 주요 클라우드 플랫폼

- AWS(Amazon Web Services)
  - 주요 서비스: EC2, S3, Lambda, RDS, EKS 등
- Microsoft Azure
  - 주요 서비스: VM, Blob Storage, AKS, Azure ML 등
- Google Cloud Platform
  - 주요 서비스: Compute Engine, GKE, BigQuery, Vertex AI 등
- Oracle Cloud Infrastructure
  - 주요 서비스: OCI compute, Autonomous DB 등
- IBM Cloud
  - 주요 서비스: IBM Watson, Code Engine, Cloud Functions 등

1-8. 도커 컨테이너 개발 환경 구축

1. 윈도우 기반 도커 설치 및 활용

- Docker Desktop 설치
- 도커 구조 및 명령어 소개
- 실습

2. 리눅스 기반 도커 설치 및 활용

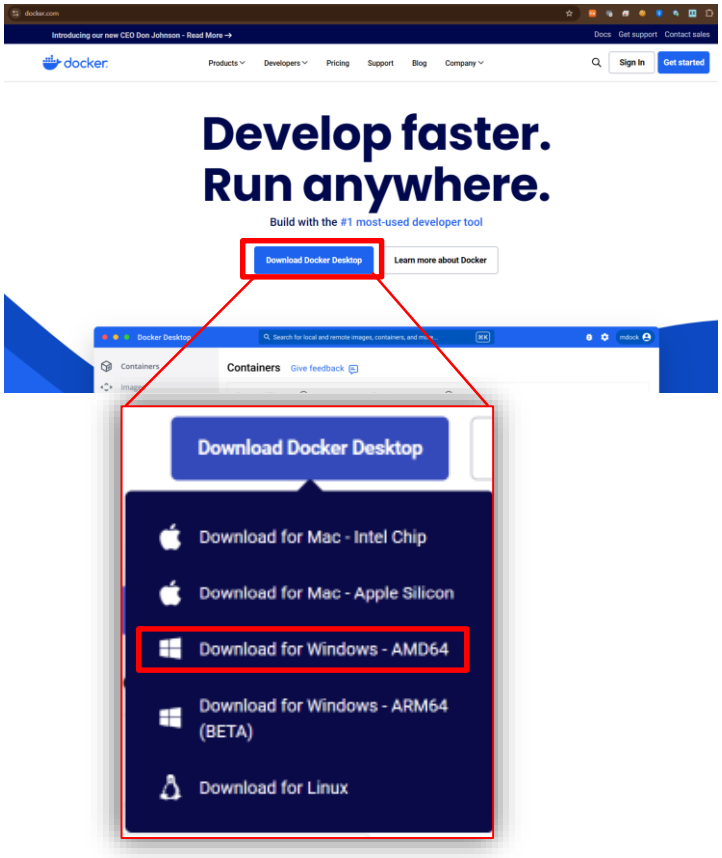
- VirtualBox, Vagrant, Putty 설치
- Vagrant 사용 방법
- 도커 설치 및 활용
  - 사전준비
    - ca-certificates: 인증서 관련 패키지
    - curl : 파일 다운로드 관련 패키지
    - gnupg : 디지털 서명 관련 패키지(GNU Privacy Guard)
    - lsb-release : 리눅스 배포판 식별 패키지
  - 도커설치
    - docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
- 실습

윈도우 기반

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ Docker Desktop 다운로드

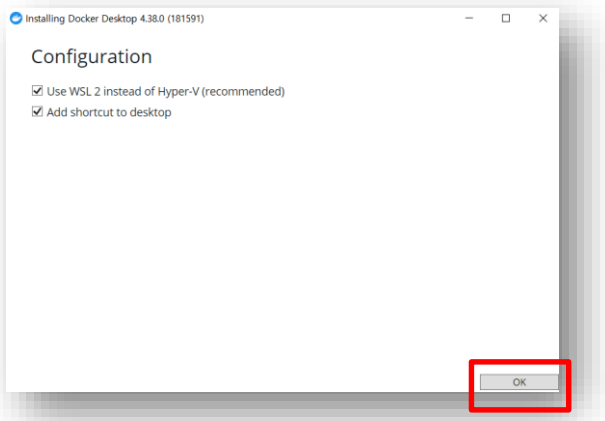
✦ https://www.docker.com/

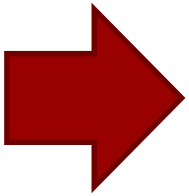


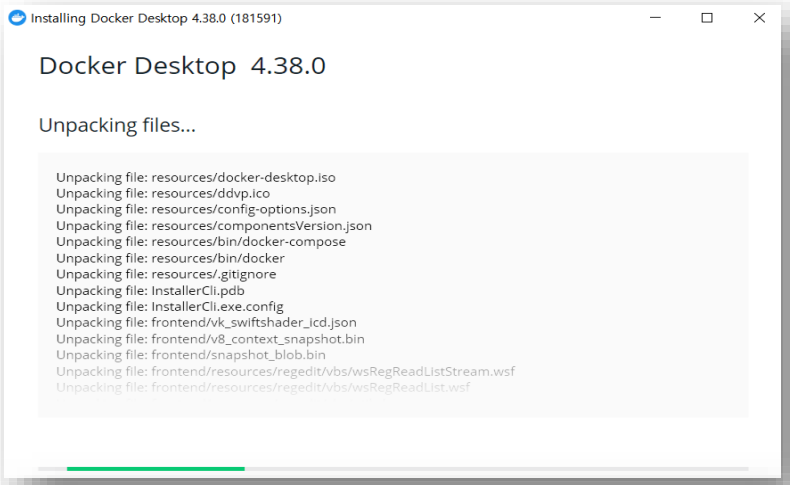
다운로드 및 설치

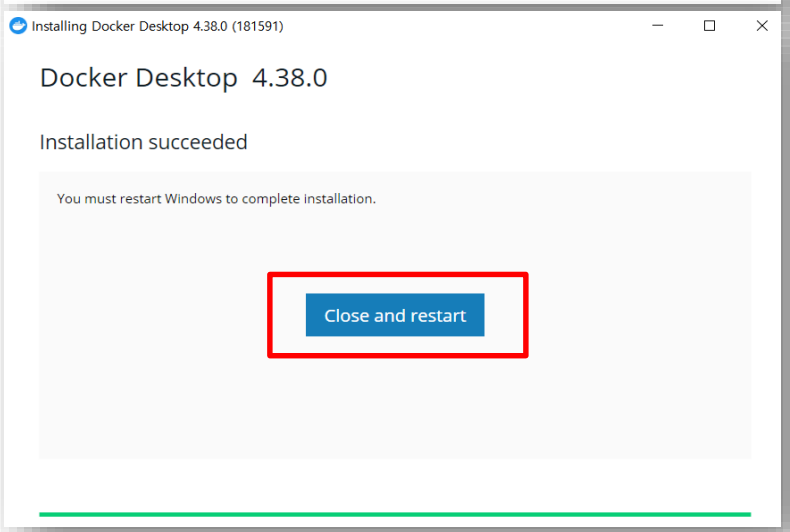
1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 다운로드 및 **설치**

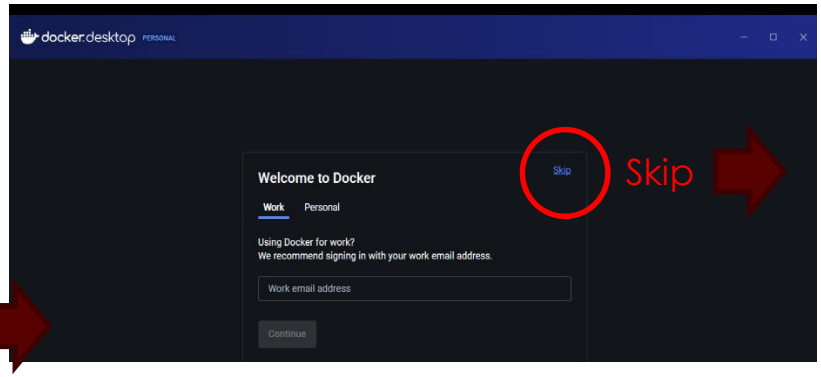




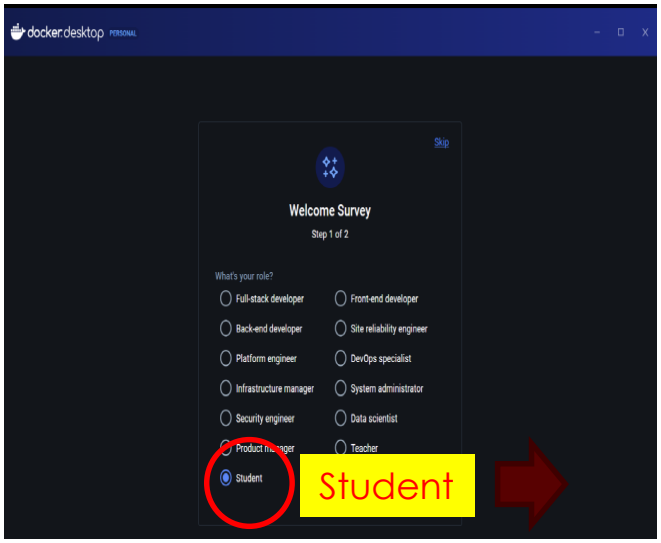


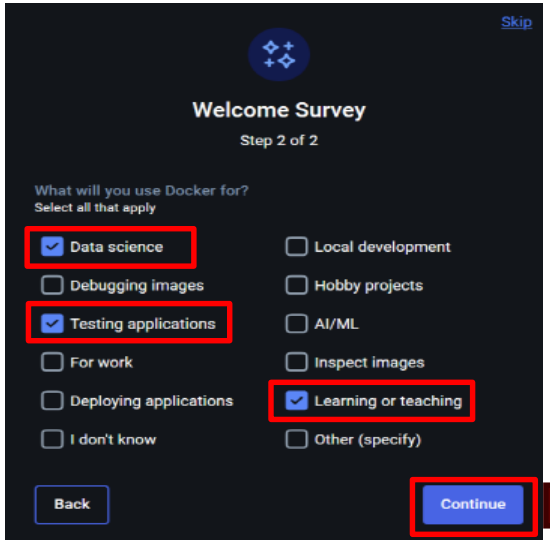


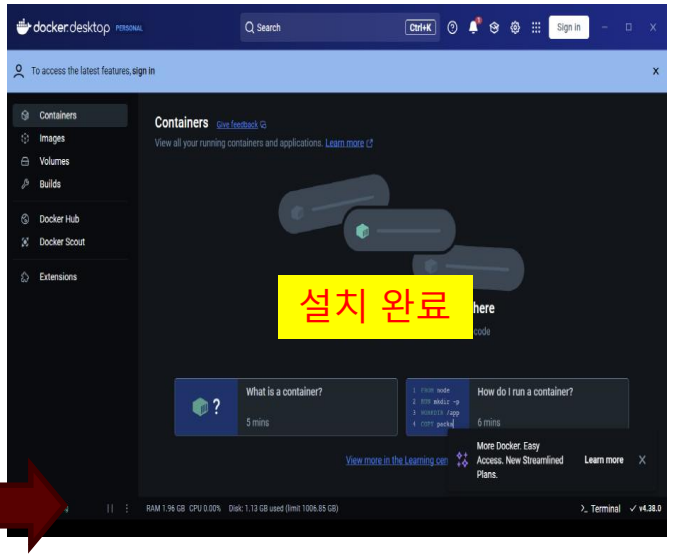
1-8. 윈도우 기반 도커 설치 및 활용(1)



Skip(권장)

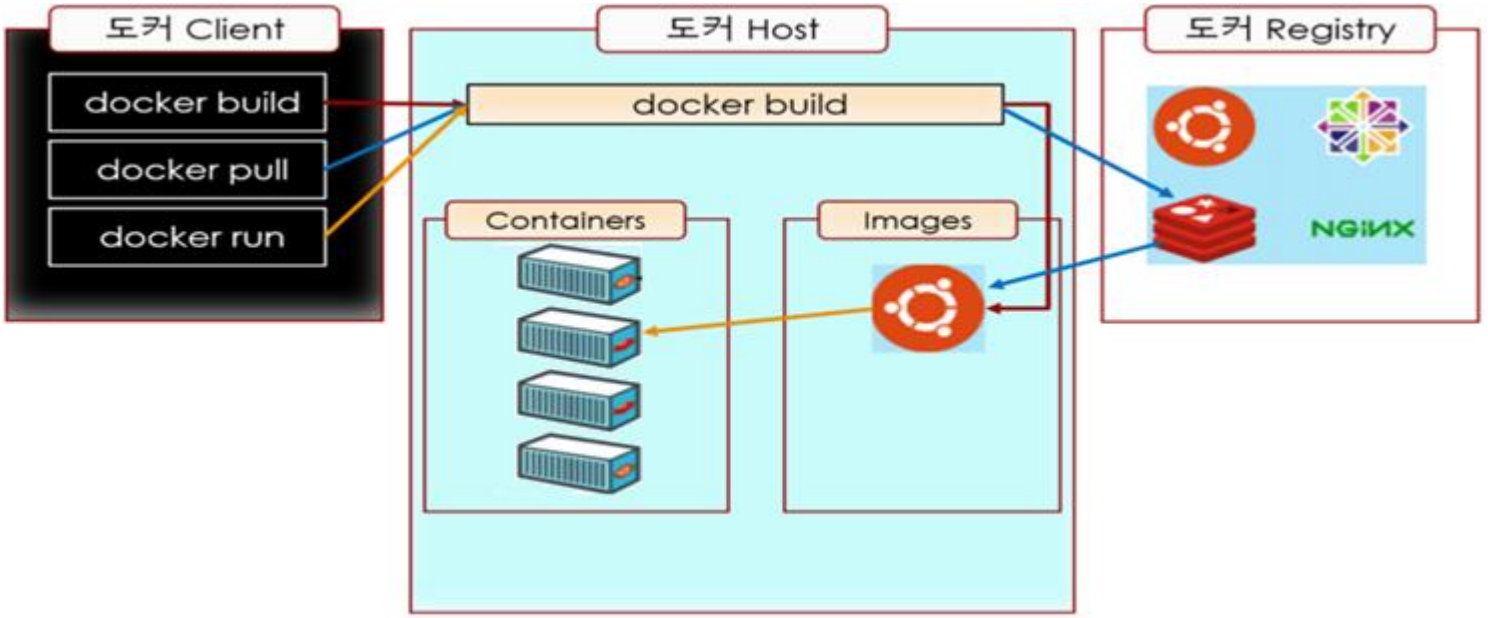


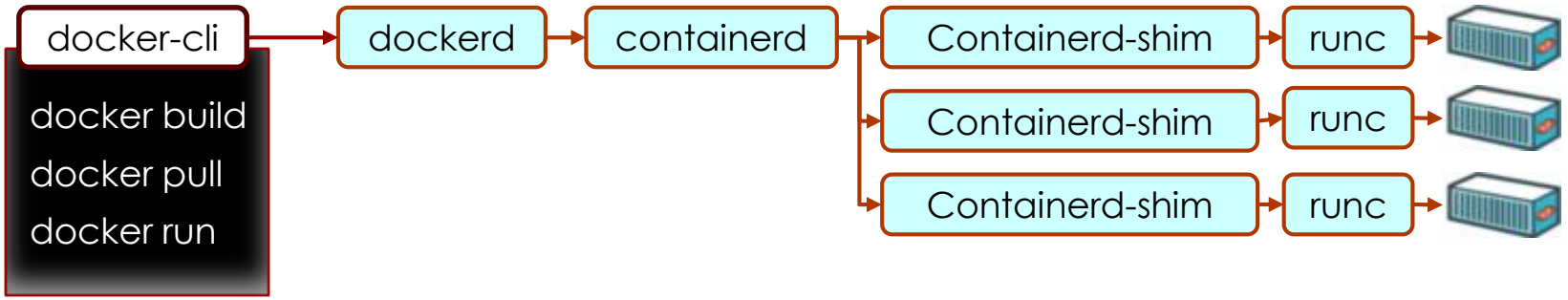




1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 구조





1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 명령어

도커 Client

docker info

docker version

```
Client:
 Version:    27.2.0
 Context:    desktop-linux
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.16.2-desktop.1
    Path:     C:\Program Files\Docker\cli-plugins\docker-buildx.exe
  compose: Docker Compose (Docker Inc.)
    Version:  v2.29.2-desktop.2
    Path:     C:\Program Files\Docker\cli-plugins\docker-compose.exe
  debug: Get a shell into any image or container (Docker Inc.)
    Version:  0.0.34
    Path:     C:\Program Files\Docker\cli-plugins\docker-debug.exe
  desktop: Docker Desktop commands (Alpha) (Docker Inc.)
    Version:  v0.0.15
    Path:     C:\Program Files\Docker\cli-plugins\docker-desktop.exe
  dev: Docker Dev Environments (Docker Inc.)
    Version:  v0.1.2
    Path:     C:\Program Files\Docker\cli-plugins\docker-dev.exe
  extension: Manages Docker extensions (Docker Inc.)
    Version:  v0.2.25
    Path:     C:\Program Files\Docker\cli-plugins\docker-extension.exe
  feedback: Provide feedback, right in your terminal! (Docker Inc.)
    Version:  v1.0.5
    Path:     C:\Program Files\Docker\cli-plugins\docker-feedback.exe
  init: Creates Docker-related starter files for your project (Docker Inc.)
    Version:  v1.3.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-init.exe
  sbom: View the packaged-based Software Bill Of Materials (SBOM) for an image (Anchore Inc.)
    Version:  0.6.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-sbom.exe
  scout: Docker Scout (Docker Inc.)
    Version:  v1.13.0
    Path:     C:\Program Files\Docker\cli-plugins\docker-scout.exe
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 명령어

도커 Client

docker images

```
REPOSITORY                          TAG                 IMAGE ID       CREATED          SIZE
java-master                         latest              9eb135da99c1   4 days ago       6.29GB
quay.io/keycloak/keycloak           latest              70c6d7716be5   2 months ago     451MB
ghcr.io/coder/coder                 latest              e1be9bdabab2   2 months ago     386MB
lscr.io/linuxserver/code-server     latest              d786a1fe5643   2 months ago     654MB
codercom/code-server                latest              379e736696ee   2 months ago     772MB
digmatic/digma-compound             0.3.324             554ee39a0038   3 months ago     725MB
mariadb                             latest              dae0c92b7b63   4 months ago     329MB
domjudge/domserver                  latest              4e1b3bec9c76   4 months ago     636MB
digmatic/digma-persistence          1.3                 769d4d63cda9   7 months ago     134MB
mysql                               latest              be960704dfac   12 months ago    602MB
alpine/java                         22-jdk              0019b75f3232   14 months ago    343MB
jupyter/datascience-notebook        latest              f78a42f3bc9a   24 months ago    5.92GB
digmaai/jaeger-ui                   1.29.1-digma.0.1.4  497be5cc6d84   2 years ago      156MB
jaegertracing/all-in-one            1.45.0              a92326268e14   2 years ago      60.4MB
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 명령어

도커 Client

docker ps

```
CONTAINER ID   IMAGE                                  COMMAND                  CREATED        STATUS                  PORTS                                NAMES
9a536fc1c9f5   java-master:latest                     "tini -g -- start-no…"   4 days ago     Up 4 days (healthy)     0.0.0.0:9998->8888/tcp               java-master
b88bbb7c9842   lscr.io/linuxserver/code-server:latest "/init"                  2 months ago    Up 13 days              0.0.0.0:8443->8443/tcp               code-server
b3dacba2a3ce   mysql                                  "docker-entrypoint.s…"   11 months ago   Up 13 days              0.0.0.0:3306->3306/tcp, 33060/tcp    mysql
```

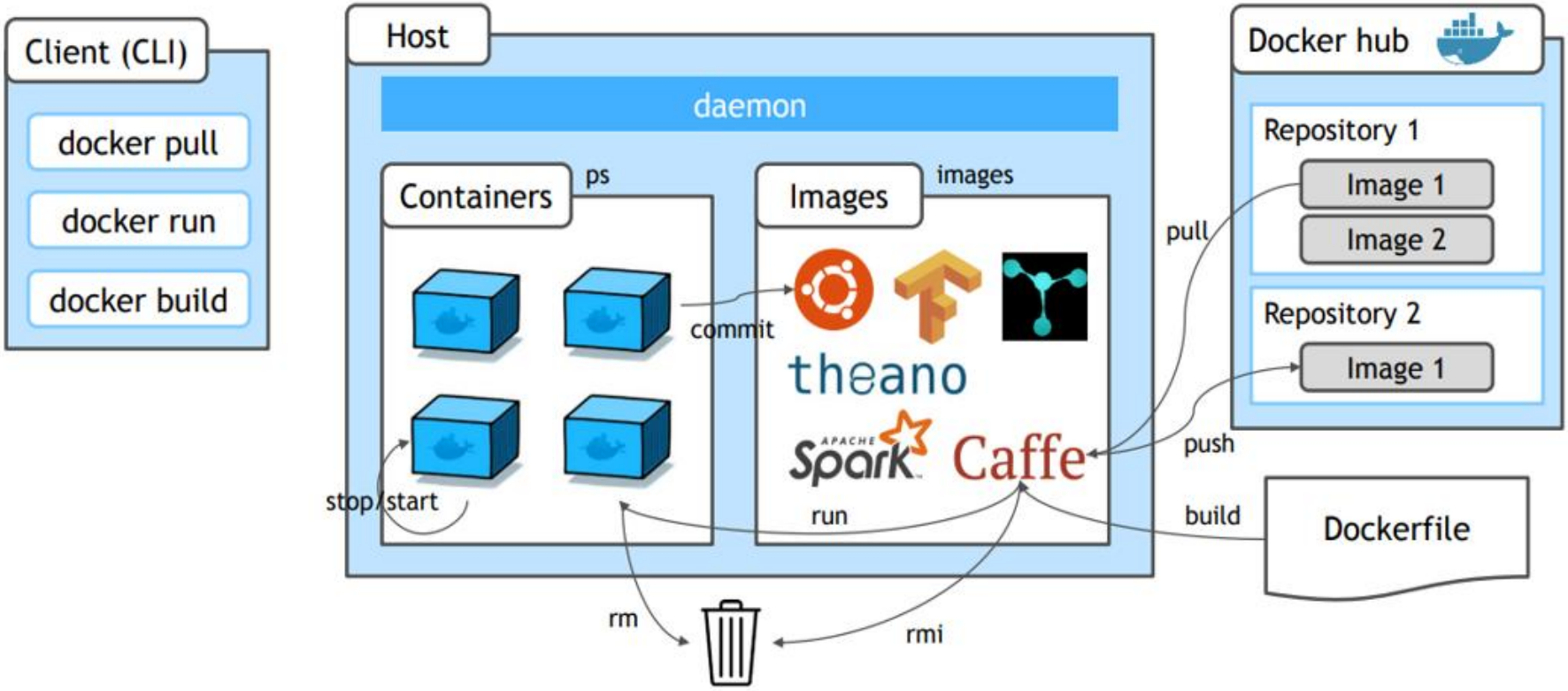
1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 명령어


```
CONTAINER ID   NAME          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O   PIDS
9a536fc1c9f5   java-master   0.96%     2.955GiB / 15.54GiB   19.02%    62.4MB / 67.3MB   0B / 0B     288
b88bbb7c9842   code-server   0.00%     54.66MiB / 15.54GiB   0.34%     1.42kB / 0B       0B / 0B     32
b3dacba2a3ce   mysql         0.18%     504.7MiB / 15.54GiB   3.17%     334kB / 354kB     0B / 0B     47
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 도커 Life Cycle



1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 컨테이너 생성

도커 Client

```
docker run [options] IMAGE[:TAG] [COMMAND] [ARG...]
```

Options

```
--name: container name
-d: run container in background mode
--rm: remove when it exits
-v: volume mount
-p: port forwarding (-P: open all)
-e: set environment in container
-t: terminal
-i: STDIN open (-ti: -t -i)
-u: set user UID
-w: working directory in container
-m: memory limit
--cpuset-cpus: limit cpus to run
--add-host: custom host:ip setting
--privileged: open kernel functions
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 컨테이너 생성

도커 Client

```
docker run –--rm  hello-world
```

```
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
17eec7bbc9d7: Pull complete
Digest: sha256:6dc565aa630927052111f823c303948cf83670a3903ffa3849f1488ab517f891
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 컨테이너 생성

도커 Client

```
docker run –it ubuntu bash
```

```
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
4b3ffd8ccb52: Pull complete
Digest: sha256:66460d557b25769b102175144d538d88219c077c678a49af4afca6fbfc1b5252
Status: Downloaded newer image for ubuntu:latest
root@5c3ff8a61a9a:/#
```

컨테이너내에서 명령어 실행

리눅스버전 확인 : cat /etc/lsb-release

종료 : exit

도커 Client

```
docker run –itd ubuntu
```

```
00ba9235d2475d4606dab64adc19a1146f4e9557de232d089a000c55a46bce3b
```

```
docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
00ba9235d247	ubuntu	"/bin/bash"	8 seconds ago	Up 7 seconds		epic_fermi
9a536fc1c9f5	java-master:latest	"tini -g -- start-no…"	4 days ago	Up 4 days (healthy)	0.0.0.0:9998->8888/tcp	java-master
b88bbb7c9842	lscr.io/linuxserver/code-server:latest	"/init"	2 months ago	Up 13 days	0.0.0.0:8443->8443/tcp	code-server
b3dacba2a3ce	mysql	"docker-entrypoint.s…"	11 months ago	Up 13 days	0.0.0.0:3306->3306/tcp, 33060/tcp	mysql

```
docker exec –it epic_fermi bash
```

```
root@5c3ff8a61a9a:/#
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 컨테이너 삭제

docker ps

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED         STATUS                   PORTS                                       NAMES
00ba9235d247   ubuntu                               "/bin/bash"              6 minutes ago   Up 6 minutes                                                         epic_fermi
9a536fc1c9f5   java-master:latest                   "tini -g -- start-no…"   4 days ago      Up 4 days (healthy)      0.0.0.0:9998->8888/tcp                      java-master
b88bbb7c9842   lscr.io/linuxserver/code-server:latest   "/init"                  2 months ago    Up 13 days               0.0.0.0:8443->8443/tcp                      code-server
b3dacba2a3ce   mysql                                "docker-entrypoint.s…"   11 months ago   Up 13 days               0.0.0.0:3306->3306/tcp, 33060/tcp           mysql

F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker stop epic_fermi
epic_fermi
```

docker stop epic_fermi

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED         STATUS                   PORTS                                       NAMES
9a536fc1c9f5   java-master:latest                   "tini -g -- start-no…"   4 days ago      Up 4 days (healthy)      0.0.0.0:9998->8888/tcp                      java-master
b88bbb7c9842   lscr.io/linuxserver/code-server:latest   "/init"                  2 months ago    Up 13 days               0.0.0.0:8443->8443/tcp                      code-server
b3dacba2a3ce   mysql                                "docker-entrypoint.s…"   11 months ago   Up 13 days               0.0.0.0:3306->3306/tcp, 33060/tcp           mysql

F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker ps -a
```

docker ps -a

```
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS                       PORTS                                       NAMES
00ba9235d247   ubuntu                               "/bin/bash"              7 minutes ago    Exited (137) 14 seconds ago                                              epic_fermi
5c3ff8a61a9a   ubuntu                               "bash"                   17 minutes ago   Exited (0) 12 minutes ago                                                fervent_almeida
82e042bfd91f   ubuntu                               "bash"                   21 minutes ago   Exited (127) 17 minutes ago                                              dreamy_feynman
9a536fc1c9f5   java-master:latest                   "tini -g -- start-no…"   4 days ago       Up 4 days (healthy)          0.0.0.0:9998->8888/tcp                      java-master
1ac6379907f3   jupyter/datascience-notebook:latest  "tini -g -- start-no…"   13 days ago      Exited (0) 4 days ago                                                    cprogram-master
f373dac5bda4   quay.io/keycloak/keycloak:latest     "/opt/keycloak/bin/k…"   2 months ago     Exited (255) 2 weeks ago     8443/tcp, 9000/tcp, 0.0.0.0:8880->8080/tcp  keycloak
b88bbb7c9842   lscr.io/linuxserver/code-server:latest   "/init"                  2 months ago     Up 13 days                   0.0.0.0:8443->8443/tcp                      code-server
031ff76434b0   domjudge/domserver:latest            "/scripts/start.sh"      2 months ago     Exited (0) 8 weeks ago                                                   domserver
6c6811b90131   mariadb                              "docker-entrypoint.s…"   2 months ago     Exited (0) 8 weeks ago                                                   dj-mariadb
b3dacba2a3ce   mysql                                "docker-entrypoint.s…"   11 months ago    Up 13 days                   0.0.0.0:3306->3306/tcp, 33060/tcp           mysql

F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker start epic_fermi
epic_fermi
```

docker start epic_fermi

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker stop epic_fermi
epic_fermi
```

docker stop epic_fermi

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker rm epic_fermi
epic_fermi
```

docker rm epic_fermi

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker ps -a
```

docker ps -a

```
CONTAINER ID   IMAGE                                COMMAND                  CREATED          STATUS                       PORTS                                       NAMES
5c3ff8a61a9a   ubuntu                               "bash"                   18 minutes ago   Exited (0) 13 minutes ago                                                fervent_almeida
82e042bfd91f   ubuntu                               "bash"                   22 minutes ago   Exited (127) 18 minutes ago                                              dreamy_feynman
9a536fc1c9f5   java-master:latest                   "tini -g -- start-no…"   4 days ago       Up 4 days (healthy)          0.0.0.0:9998->8888/tcp                      java-master
1ac6379907f3   jupyter/datascience-notebook:latest  "tini -g -- start-no…"   13 days ago      Exited (0) 4 days ago                                                    cprogram-master
f373dac5bda4   quay.io/keycloak/keycloak:latest     "/opt/keycloak/bin/k…"   2 months ago     Exited (255) 2 weeks ago     8443/tcp, 9000/tcp, 0.0.0.0:8880->8080/tcp  keycloak
b88bbb7c9842   lscr.io/linuxserver/code-server:latest   "/init"                  2 months ago     Up 13 days                   0.0.0.0:8443->8443/tcp                      code-server
031ff76434b0   domjudge/domserver:latest            "/scripts/start.sh"      2 months ago     Exited (0) 8 weeks ago                                                   domserver
6c6811b90131   mariadb                              "docker-entrypoint.s…"   2 months ago     Exited (0) 8 weeks ago                                                   dj-mariadb
b3dacba2a3ce   mysql                                "docker-entrypoint.s…"   11 months ago    Up 13 days                   0.0.0.0:3306->3306/tcp, 33060/tcp           mysql
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 이미지 삭제

docker images

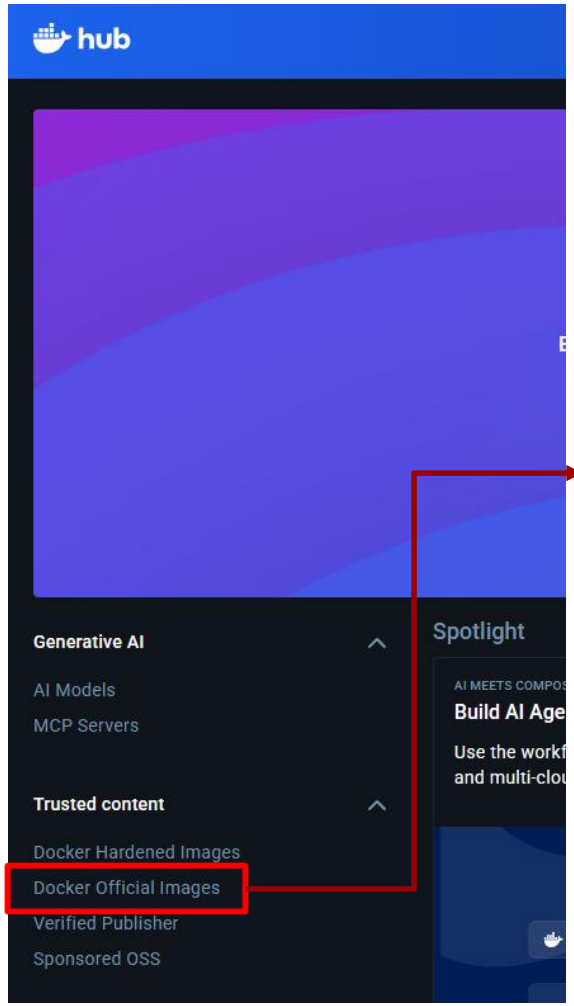
```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker images
REPOSITORY                          TAG                 IMAGE ID       CREATED         SIZE
java-master                         latest              9eb135da99c1   4 days ago      6.29GB
ubuntu                              latest              97bed23a3497   13 days ago     78.1MB
quay.io/keycloak/keycloak           latest              70c6d7716be5   2 months ago    451MB
hello-world                         latest              1b44b5a3e06a   2 months ago    10.1kB
ghcr.io/coder/coder                 latest              e1be9bdabab2   2 months ago    386MB
lscr.io/linuxserver/code-server     latest              d786a1fe5643   2 months ago    654MB
codercom/code-server                latest              379e736696ee   2 months ago    772MB
digmatic/digma-compound             0.3.324             554ee39a0038   3 months ago    725MB
mariadb                             latest              dae0c92b7b63   4 months ago    329MB
domjudge/domserver                  latest              4e1b3bec9c76   4 months ago    636MB
digmatic/digma-persistence          1.3                 769d4d63cda9   7 months ago    134MB
mysql                               latest              be960704dfac   12 months ago   602MB
alpine/java                         22-jdk              0019b75f3232   14 months ago   343MB
jupyter/datascience-notebook        latest              f78a42f3bc9a   24 months ago   5.92GB
digmaai/jaeger-ui                   1.29.1-digma.0.1.4  497be5cc6d84   2 years ago     156MB
jaegertracing/all-in-one            1.45.0              a92326268e14   2 years ago     60.4MB

F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker rmi hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:6dc565aa630927052111f823c303948cf83670a3903ffa3849f1488ab517f891
Deleted: sha256:1b44b5a3e06a9aae883e7bf25e45c100be0bb81a0e01b32de604f3ac44711634
Deleted: sha256:53d204b3dc5ddbc129df4ce71996b8168711e211274c785de5e0d4eb68ec3851
```

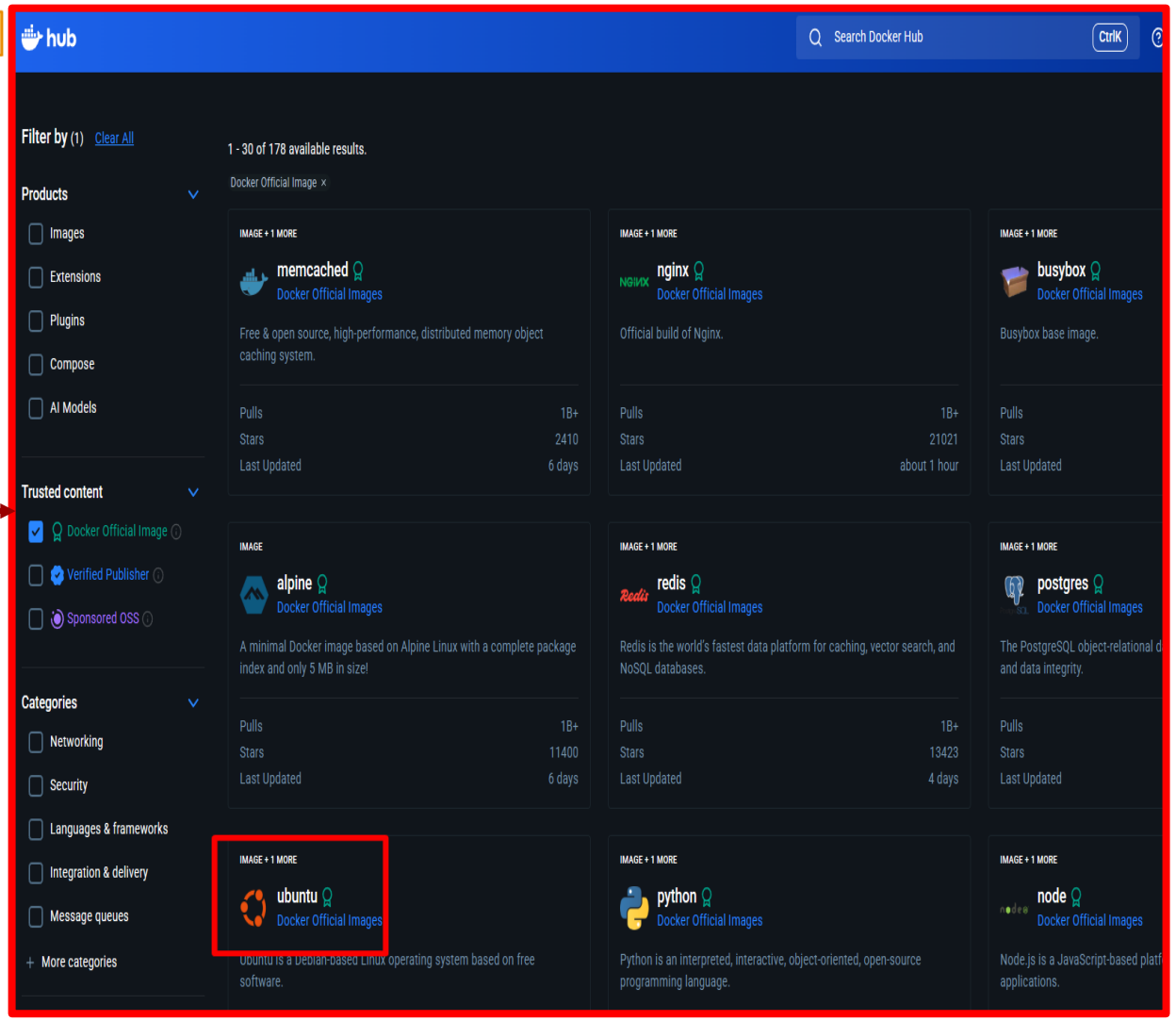
docker rmi hello-world

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 이미지 가져오기 : `docker pull ubuntu:24.04`

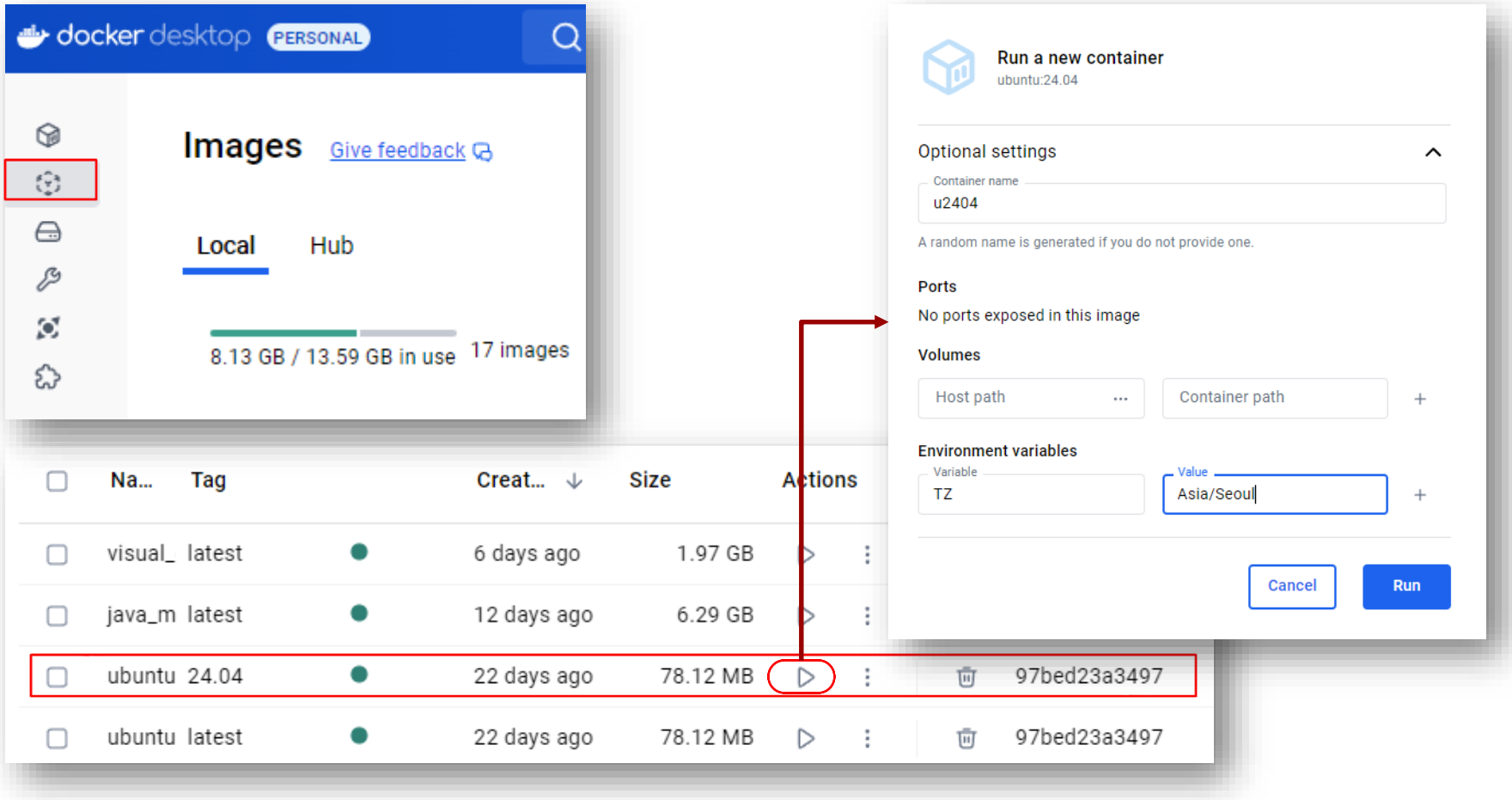
https://hub.docker.com





1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 이미지 가져오기 : `docker pull ubuntu:24.04`



1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 가져온 이미지 활용하기

- `docker run -itd -name u2404 -e TZ=Asia/Seoul ubuntu:24.04`

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker run -itd --name u2404 -e TZ=Asia/Seoul ubuntu:24.04
e01e9db7d4e778c9be19bae048e056511b596e944e7fe5c8f82766893506ab7b
```

- `docker exec -it u2404 bash ; date`

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker exec -it u2404 bash
root@e01e9db7d4e7:/# date
Wed Oct 15 07:44:37 Asia 2025
```

- `apt-get update && apt-get install -y tzdata`

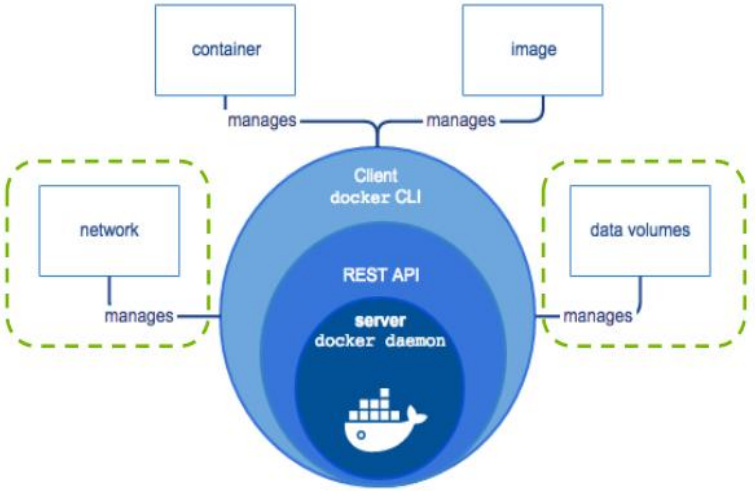
```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  tzdata
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 276 kB of archives.
After this operation, 1420 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 tzdat
Fetched 276 kB in 2s (146 kB/s)
```

- `dpkg-reconfigure tzdata` <= timezone 재구성
- `date` <= 날짜 시간 확인

```
PS C:\Users\master> docker exec -it u2404 bash
root@e01e9db7d4e7:/# date
Thu Oct 23 11:14:39 KST 2025
root@e01e9db7d4e7:/#
```

1-8. 윈도우 기반 도커 설치 및 활용(1)

✦ 자원 관리(로컬 폴더 공유 및 포트 포워딩)



✦ Volume mount:

- -v {host-volume}:{container-volume}[:ro]
- docker run -v "%cd%:/workspace" image-name <= 윈도우
- docker run -v $(pwd):/workspace image-name <= 리눅스

✦ Port forwarding:

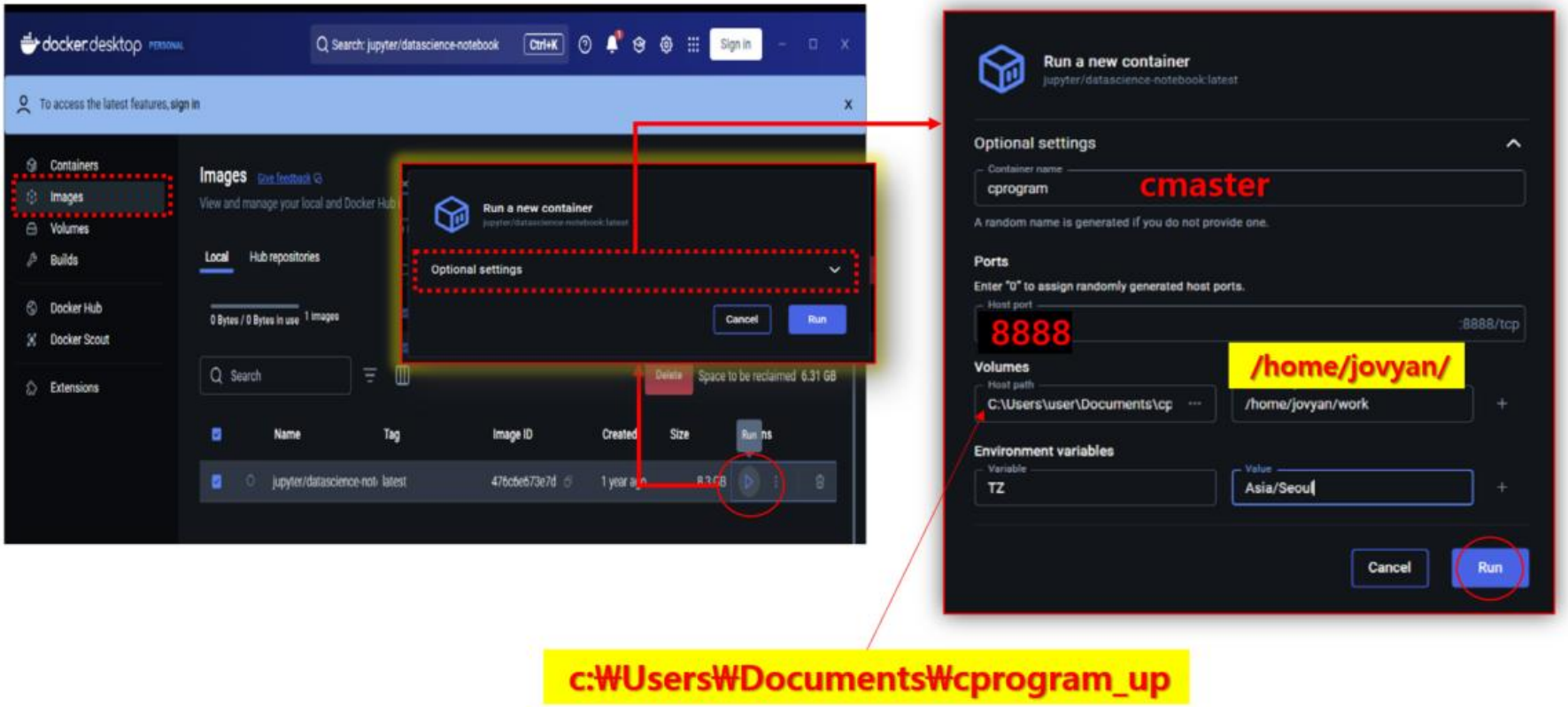
- -p {host-port}:{container-port}
- docker run -p 8899:8888 image-name

✦ User mount(-u: UID:GID로 컨테이너 실행), 환경변수 설정

- docker run -u $(id -u):$(id -g) -e TZ=Asia/Seoul image-name

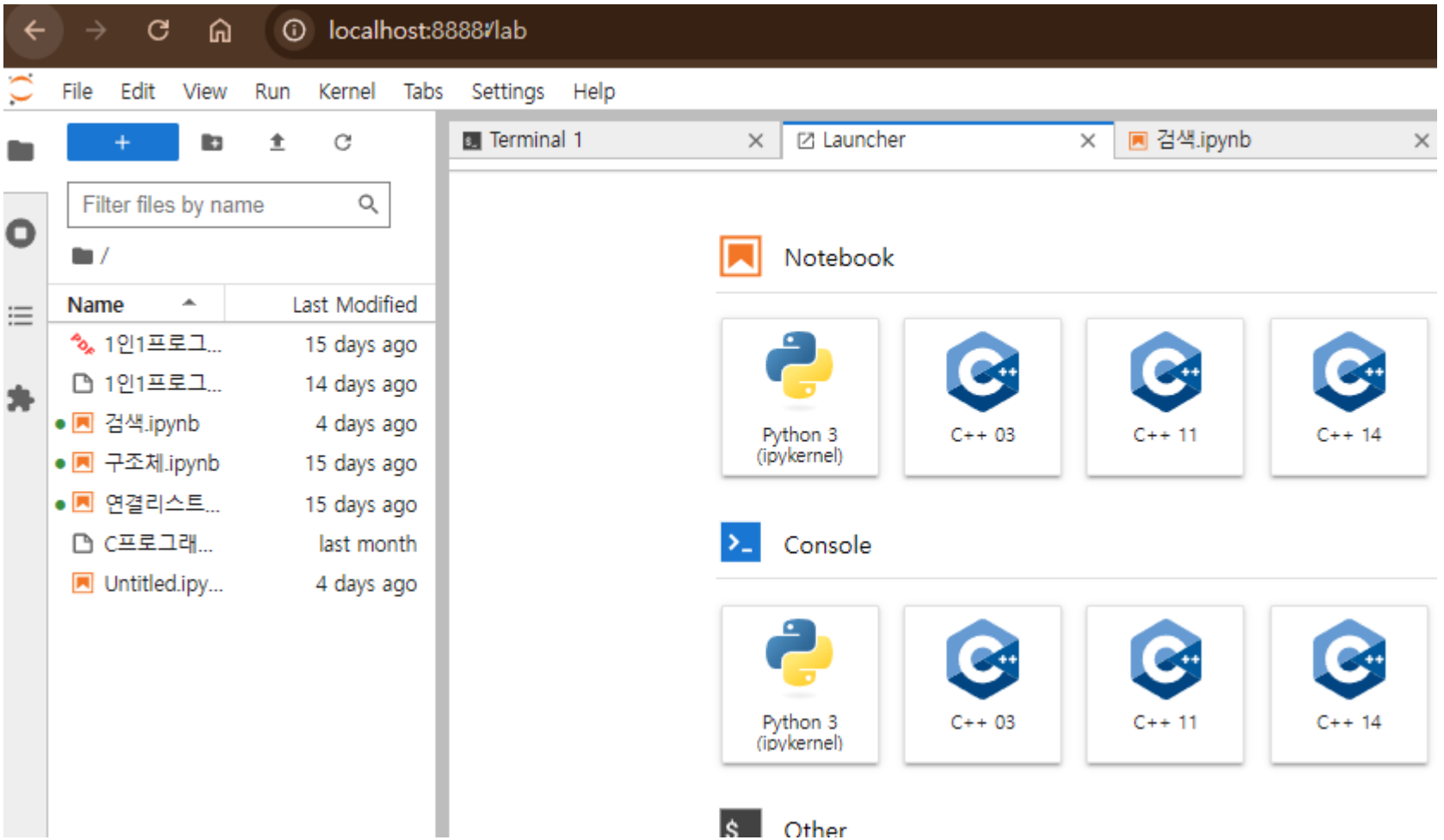
1-8. 컨테이너 생성 예시(1)

- c프로그래밍을 위한 주피터 노트북 컨테이너 만들기
  - jupyter/datascience-notebook 이미지 활용(파이썬 커널이 기본 환경)



  - 컨테이너 생성 후 pip install -y jupyter-cpp-kernel 커널 추가

1-8. 컨테이너 생성 예시(2)



1-8. 컨테이너 생성 예시(3)

docker images

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker images
REPOSITORY                          TAG                 IMAGE ID       CREATED         SIZE
java-master                         latest              9eb135da99c1   4 days ago      6.29GB
ubuntu                              25.04               ee6708774003   12 days ago     77MB
ubuntu                              24.04               97bed23a3497   13 days ago     78.1MB
ubuntu                              latest              97bed23a3497   13 days ago     78.1MB
quay.io/keycloak/keycloak           latest              70c6d7716be5   2 months ago    451MB
ghcr.io/coder/coder                 latest              e1be9bdabab2   2 months ago    386MB
lscr.io/linuxserver/code-server     latest              d786a1fe5643   2 months ago    654MB
codercom/code-server                latest              379e736696ee   2 months ago    772MB
digmatic/digma-compound             0.3.324             554ee39a0038   3 months ago    725MB
mariadb                             latest              dae0c92b7b63   4 months ago    329MB
domjudge/domserver                  latest              4e1b3bec9c76   4 months ago    636MB
digmatic/digma-persistence          1.3                 769d4d63cda9   7 months ago    134MB
mysql                               latest              be960704dfac   12 months ago   602MB
alpine/java                         22-jdk              0019b75f3232   14 months ago   343MB
jupyter/datascience-notebook        latest              f78a42f3bc9a   24 months ago   5.92GB
```

docker run –itd –e TZ=Asia/Seoul –v “%cd%:/home/jovyan” --name cprogram –p 18888:8888 jupyter/da~~~

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker run -itd -e TZ=Asia/Seoul -v "%cd%:/home/jovyan" --name cprogram -p 18888:8888 jupyter/datascience-notebook
c43d7ee64ad2f748f0765e815cd3ede8f24ffe77d61d1c357debf595090e859f
```

docker ps

```
F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker ps
CONTAINER ID   IMAGE                                  COMMAND                  CREATED             STATUS                    PORTS                                      NAMES
c43d7ee64ad2   jupyter/datascience-notebook           "tini -g -- start-no…"   6 seconds ago       Up 6 seconds (healthy)    0.0.0.0:18888->8888/tcp                    cprogram
e01e9db7d4e7   ubuntu:24.04                           "/bin/bash"              About an hour ago   Up About an hour                                                     u2404
9a536fc1c9f5   java-master:latest                     "tini -g -- start-no…"   4 days ago          Up 4 days (healthy)       0.0.0.0:9998->8888/tcp                     java-master
b88bbb7c9842   lscr.io/linuxserver/code-server:latest  "/init"                  2 months ago        Up 13 days                0.0.0.0:8443->8443/tcp                     code-server
b3dacba2a3ce   mysql                                  "docker-entrypoint.s…"   11 months ago       Up 13 days                0.0.0.0:3306->3306/tcp, 33060/tcp          mysql

F:\2025년 오픈소스SW교육\11.6(목)-도커 컨테이너 개념 및 활용>docker exec -it cprogram bash
jovyan@c43d7ee64ad2:~$ date
Wed Oct 15 05:49:48 PM KST 2025
jovyan@c43d7ee64ad2:~$
```

docker exec –it cprogram bash

1-8. 컨테이너 생성 예시(4)

✦ Web Visual Code[Java, C, Python] 컨테이너 만들기

Dockerfile

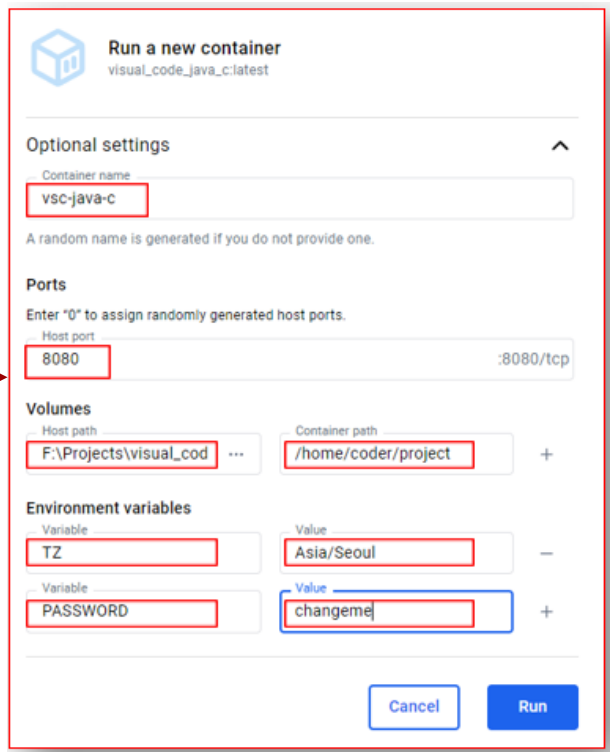
```
FROM ghcr.io/coder/code-server:latest

# coder-server에는 기본 JDK17까지만 제공
USER root
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && \
    apt-get install -y --no-install-recommends \
      python3 python3-pip python3-venv \
      build-essential gcc g++ gdb cmake make git curl ca-certificates \
    && rm -rf /var/lib/apt/lists/*
# JDK 21 추가
RUN curl -fsSL "https://api.adoptium.net/v3/binary/latest/21/ga/linux/x64/jdk/hotspot/normal/eclipse" -o /tmp/jdk21.tar.gz \
 && mkdir -p /opt/jdk-21 \
 && tar -xzf /tmp/jdk21.tar.gz -C /opt/jdk-21 --strip-components=1 \
 && rm -f /tmp/jdk21.tar.gz \
 && update-alternatives --install /usr/bin/java  java  /opt/jdk-21/bin/java  1 \
 && update-alternatives --install /usr/bin/javac javac /opt/jdk-21/bin/javac 1

ENV JAVA_HOME=/opt/java-21
ENV PATH=$JAVA_HOME/bin:$PATH

USER coder

# VS Code 확장 (cpptools는 Open VSX 이슈가 잦으니 clangd로 대체 권장)
RUN code-server --install-extension vscjava.vscode-java-pack \
 && code-server --install-extension ms-python.python \
 && code-server --install-extension ms-toolsai.jupyter \
 && code-server --install-extension llvm-vs-code-extensions.vscode-clangd \
 && code-server --install-extension MS-CEINTL.vscode-language-pack-ko \
 && code-server --install-extension webfreak.debug

WORKDIR /home/coder/project
```

1-8. 컨테이너 생성 예시(5)

이미지 생성 `docker build –t visual_code_java_c .`

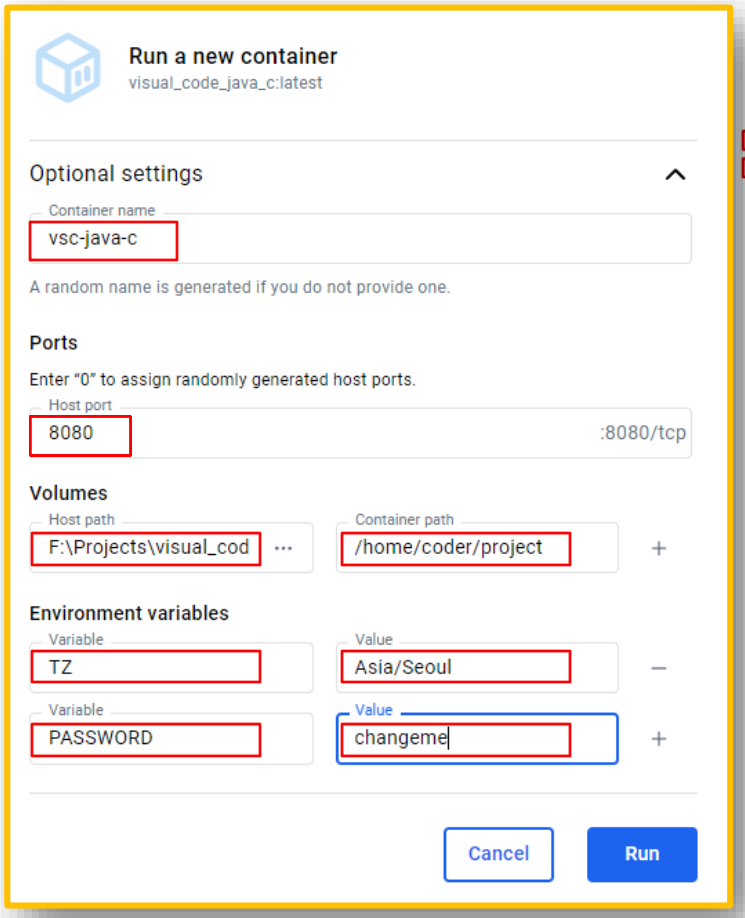
```
PS F:\Projects\visual_code_c_java> docker build -t visual_code_java_c .
[+] Building 294.6s (9/9) FINISHED                                  docker:desktop-linux
 => [internal] load build definition from Dockerfile                                0.0s
 => => transferring dockerfile: 1.34kB                                              0.0s
 => [internal] load metadata for ghcr.io/coder/code-server:latest                   0.9s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 2B                                                     0.0s
 => CACHED [1/5] FROM ghcr.io/coder/code-server:latest@sha256:8a4f9ec985bcb022de1ae70fab74e3a2c  0.0s
```

docker run 명령어

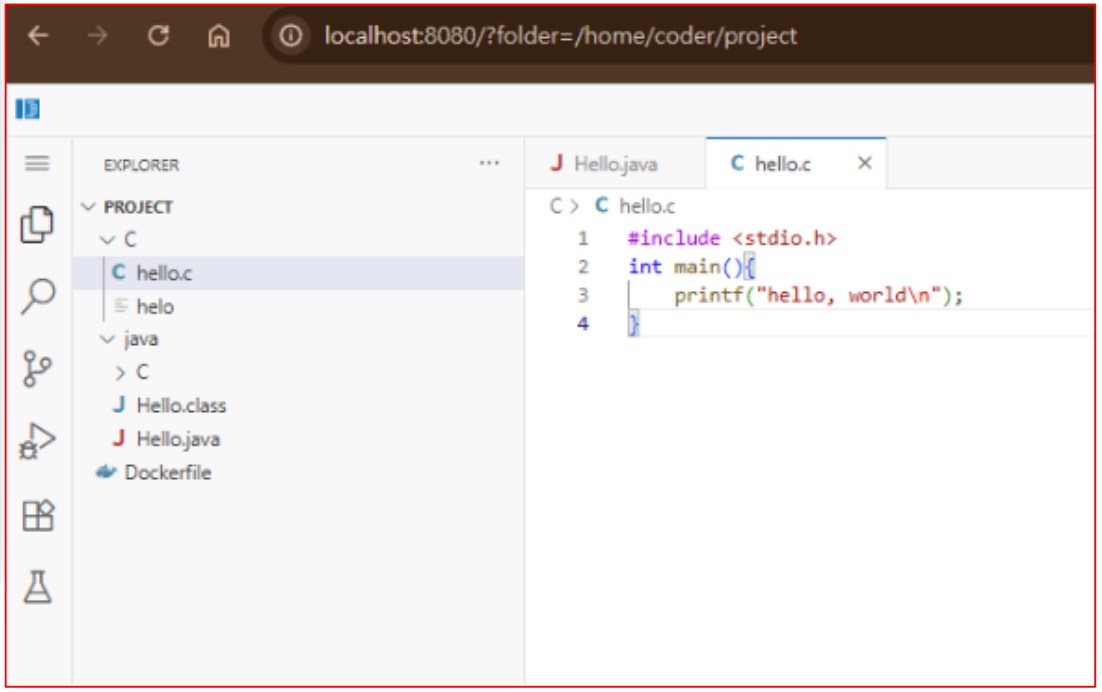


1-8. 컨테이너 생성 예시(6)

✦ 컨테이너 생성 및 활용




```
docker run -itd --name vsc-java-c                    \
    -p 8080:8080                                     \
    -v "%cd%:/home/coder/project"                    \
    -e TZ=Asia/Seoul -e PASSWORD=changeme            \
    visual_code_java_c
```



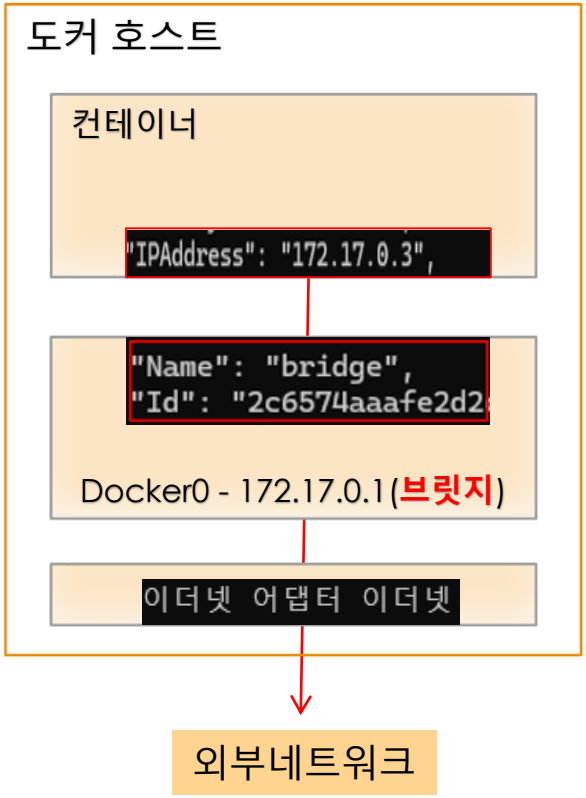
1-8. 컨테이너 생성 예시(7)

✦ 도커 네트워크: docker network ls

```
PS C:\Users\master> docker network ls
NETWORK ID     NAME                          DRIVER    SCOPE
2c6574aaafe2   bridge                        bridge    local
51e28647ec15   code-server-watcher_default    bridge    local
1087e4dce1ef   host                          host      local
3b27e5afeaca   none                          null      local
```

```
PS C:\Users\master> docker inspect 2c6574aaafe2
[
    {
        "Name": "bridge",
        "Id": "2c6574aaafe2d2cf840d6264596cd8ecbd6012dc44db26
        "Created": "2025-10-22T03:14:44.116321227Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16",
                    "Gateway": "172.17.0.1"
                }
            ]
        },
```

```
PS C:\Users\master> docker inspect u2404
[
    {
        "NetworkID": "2c6574aaafe2d2cf840d6
        "EndpointID": "ba9095fc038c00864de5
        "Gateway": "172.17.0.1",
        "IPAddress": "172.17.0.3",
```



1-8. 컨테이너 생성 예시(8)

✦ Docker compose 활용

```
docker run -name apa000ex2 -d -p 8080:80 httpd
```

```
version: "3"

services:
   apa000ex2:
      image: httpd
      ports:
        -8080:80
      restart: always
```

docker compose up –d

```
docker run -itd --name vsc-java-c -p 8080:8080  -v
"%cd%:/home/coder/project"  \
    -e TZ=Asia/Seoul -e PASSWORD=changeme  visual_code_java_c
```

docer-compose.yml

```
version: "3.8"

services:
  vsc-java-c:
    image: visual_code_java_c
    container_name: jsc-java-c
    environment:
      - TZ=Asia/Seoul
      - PASSWORD=changeme
    ports:
      - "8080:8080"
    volumes:
      - ./:/home/coder/project
    stdin_open: true        # -i
    tty: true               # -t
    restart: unless-stopped  # 종료 시 자동 재시작(선택)
```

docker compose up –d

✦ 도커컨테이너실습.ipynb 파일 참조

4. Docker compose 스크립트(컨테이너명: mysql-coffee)

[ ]:
```
%%writefile c:\coffee-mysql\Dockerfile
FROM mysql:8.4
COPY my.cnf /etc/mysql/conf.d/my.cnf
RUN chmod 644 /etc/mysql/conf.d/my.cnf
```

[ ]:
```
%%writefile c:\coffee-mysql\docker-compose.yml
services:
  mysql:
    build: .
    image: mysql-coffee:local
    container_name: mysql-coffee
    restart: unless-stopped
    environment:
      MYSQL_ROOT_PASSWORD: rootpw123!          # 원하는 비번으로 변경
      MYSQL_DATABASE: coffee                   # 최초 생성 DB (옵션)
      TZ: Asia/Seoul
    ports:
      - "3306:3306"
    volumes:
      - ./data:/var/lib/mysql
      - ./import:/import
      - ./sql:/docker-entrypoint-initdb.d
```

1-8. Docker Private Registry

- Docker허브 로그인 및 로그아웃

```
docker login –u 아이디
Password:

Login Succeeded

 docker logout
```

- Docker 이미지 빌드

```
docker build –t user/image:tag .
```

- Docker허브(또는 사설 저장소) 업로드

```
docker push user/image:tag
```

- Runtime컨테이너를 이미지로 저장

```
docker commit [옵션] 컨테이너 저장소:tag
```

예: docker commit u2404 wesleyok/ubuntu-custom:1.0

- Docker 다운로드

```
docker pull user/image:tag
```

1-8. Docker Private Registry(1)

✦ 사전준비: 웹서버 설치

- Ubuntu/Debian
  - Sudo apt update && sudo apt install -y apache2-utils
- Fedora/CentOS/RHEL(레드헷)
  - Sudo dnf -y install httpd-tools
- 윈도우: https://www.apachelounge.com/download/
  - Httpd-2.4.x-win64-VS17.zip 압축해제 및 환경설정 , **htpasswd** 명령어 활용하기 위해

✦ 디렉토리 및 계정 생성

- mkdir -p ~/reg/registry ~/reg/auth
- cd ~/reg/auth && htpasswd -Bc registry.passwd userid
  - 비밀번호 입력, bcrypt로 생성(-B), 새 파일 생성(-c)

✦ 레지스트리 컨테이너 실행

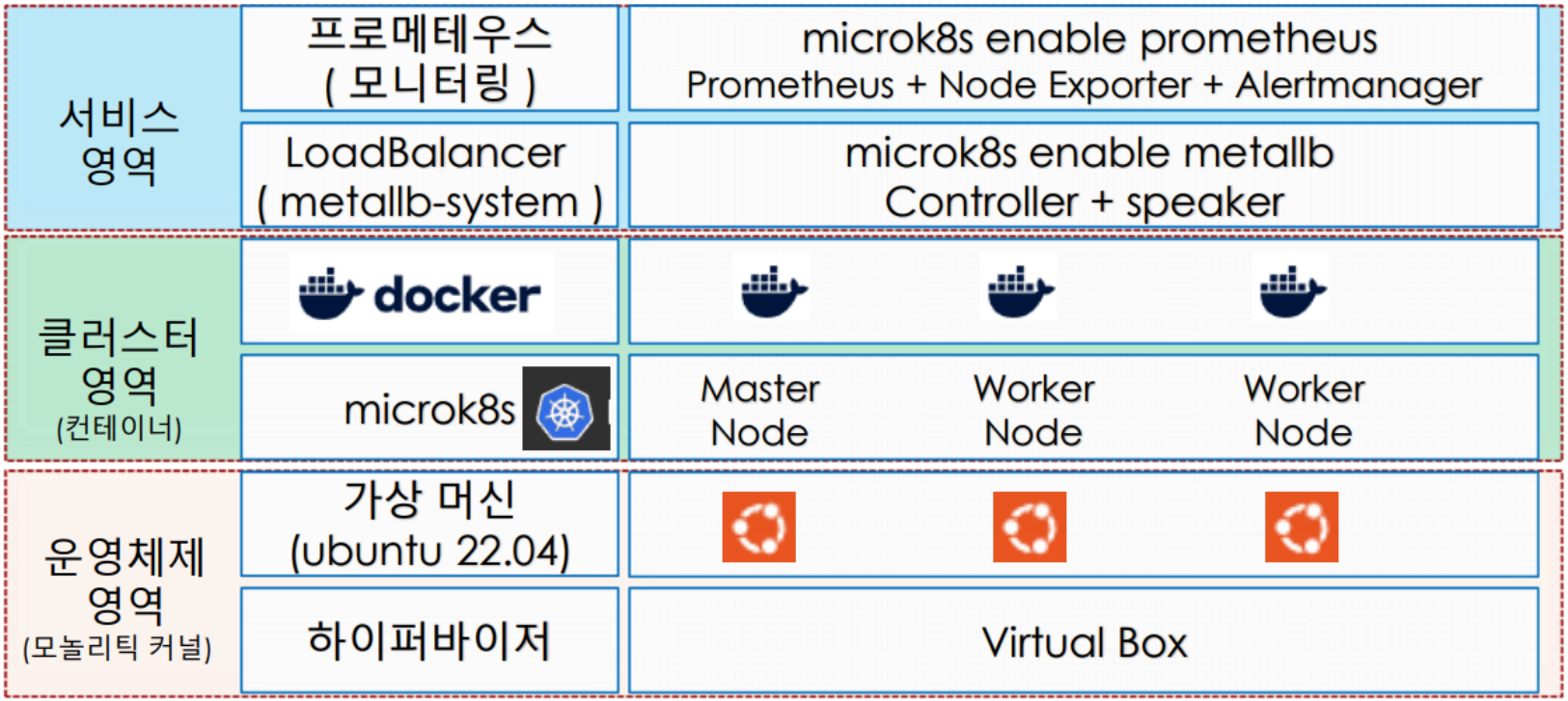
```
docker run -d -p 5000:5000 --restart=always --name jnureg1 ₩
                           -v ~/reg/registry:/var/lib/registry ₩
                           -v ~/reg/auth:/auth ₩
                           -e REGISTRY_AUTH=htpasswd ₩
                           -e REGISTRY_AUTH_HTPASSWD_REALM="Registry Realm" ₩
                           -e REGISTRY_AUTH_HTPASSWD_PATH="/auth/registry.password"    registry:latest
```

1-8. Docker Private Registry(2)

✦ 로그인 및 사설 저장소에 이미지 저장, 활용

- 로그인
  - `docker login localhost:5000`
  - id 및 passswd(htpasswd로 만든 암호)
- 이미지 태깅
  - `docker tag myapp:1.0 localhost:5000/myapp:1.0`
- 이미지 로컬저장소에 저장
  - `docker push localhost:5000/myapp:1.0`
- 이미지 로컬저장소에서 가져오기
  - `docker pull localhost:5000/myapp:1.0`

리눅스 기반



컨테이너 인프라 환경 구축

리눅스 기반 도커 설치 및 활용

- 개발자와 시스템 관리자들의 기본 실무 지식
- VirtualBox, Vagrant, Putty 설치
- Vagrant 사용 방법
- 도커 설치 및 활용
  - 사전준비
    - ca-certificates: 인증서 관련 패키지
    - curl : 파일 다운로드 관련 패키지
    - gnupg : 디지털 서명 관련 패키지(GNU Privacy Guard)
    - lsb-release : 리눅스 배포판 식별 패키지
  - 도커설치
    - docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
- 실습

✦ 계정과 권한

- root계정 : 모든 권한을 가진 관리자 계정
- 사용자 계정 : 제한된 권한을 가진 계정
- 관련 명령어:

명령어 또는 중요파일	사용예	옵션 설명
chmod [옵션] [파일,폴더]	chmod 755 admin.sh chmod a+rw,u+x admin.sh	rwx | rwx | rwx usr grp other
/etc/sudoers /etc/sudoers.d	root ALL=(ALL:ALL) ALL admin ALL=(ALL:ALL) ALL	사용자에게 sudo 실행 권한을 부여하기 계정 모든호스트=(모든사용자) 모든명령어
chown [옵션] [파일,폴더] chgrp [옵션] [그룹] [파일,폴더]	chown user.group Test chgrp group Test	Test파일의 user와 group 설정 Test파일의 그룹 설정

✦ 중요 명령어

프로세스 확인	디스크 용량 확인	스케줄링
ps –ef | head –n 6 ps aux | grep http kill -9 PID #PID 강제 죽이기 nohup 명령어 # 백그라운드 실행	df –h #파티션 사용량 확인 du –h #디스크, 폴더, 파일 용량 확인 find /root -mtime -1 –type f –ls ls –al | grep f* #f로 시작되는파일 ls –al | grep sh$ #sh로 끝나는 파일	crontab –l #스케줄 보기 10,30 1-3 * * * /data/run-job.sh ------ 분 시 일 월 주 명령어 일: 1 – 31, 월: 1 – 12 주: 0 – 6 (일 - 토)

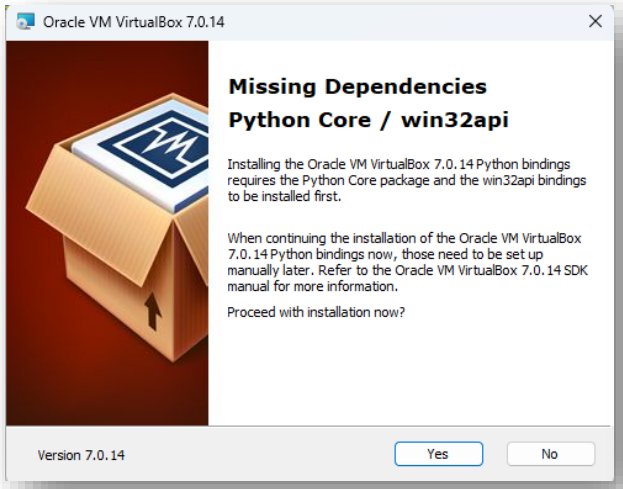
- 그 외에도 alias(별칭),네트워크 확인(ifconfig, netstat 등), curl(웹정보 확인) 등 활용

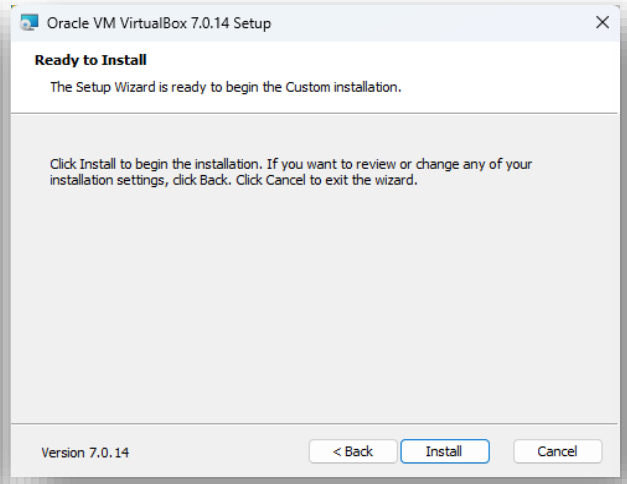
1-9. 클라우드 플랫폼 예시(1)

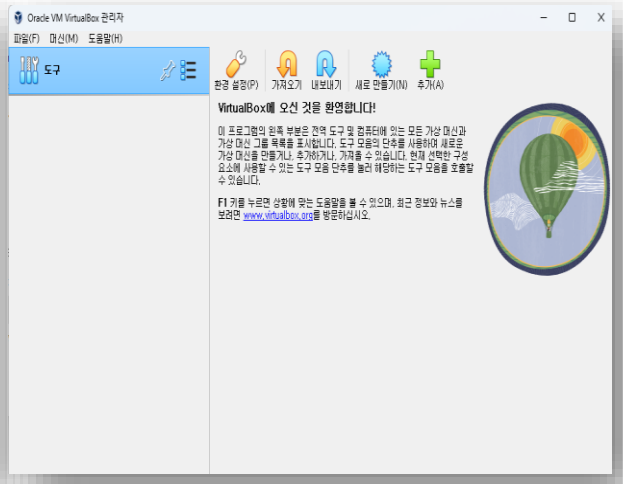
✦ 하이퍼바이저와 클라우드

- 피지컬 컴퓨터 위에 여러 개의 가상머신(VM)을 생성하는 가상화 기술의 핵심 소프트웨어
- 클라우드 플랫폼(AWS, Azure, GCP 등)에서는 내부적으로 하이버바이저 기반의 가상화를 사용하여 수 천대의 VM 구동

✦ Virtualbox 설치(https://www.virtualbox.org/)





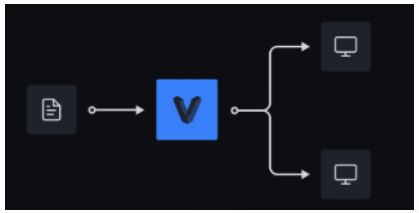


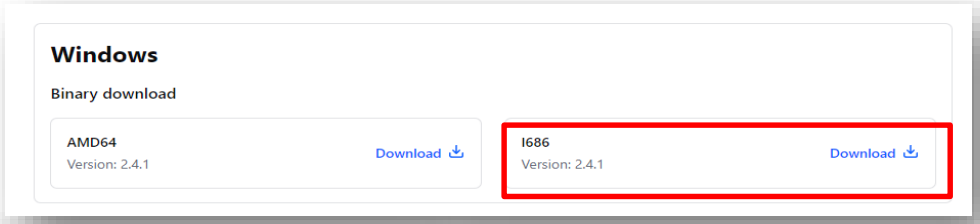
1-9. 클라우드 플랫폼 예시(2)

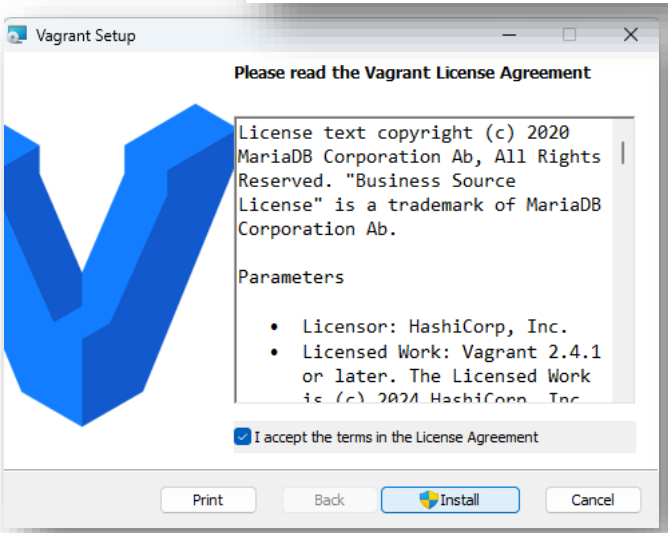
✦ Vagrant 활용하여 가상머신 생성

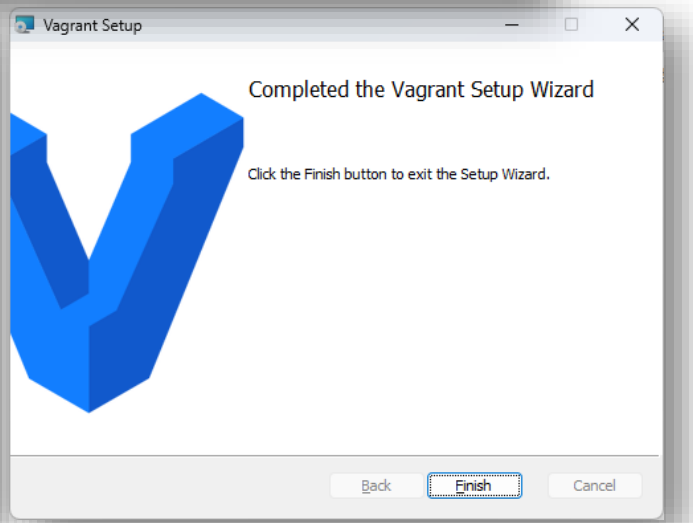
- 가상 머신(VM)을 코드로 관리하는 가상 환경 자동화 도구
- 다양한 백엔드 지원: VirturalBox, VMWare, Hyper-V, Docker 등

✦ Vagrant 설치(https://www.virtualbox.org/)









1-9. 클라우드 플랫폼 예시(3)

✦ Vagrant 명령어

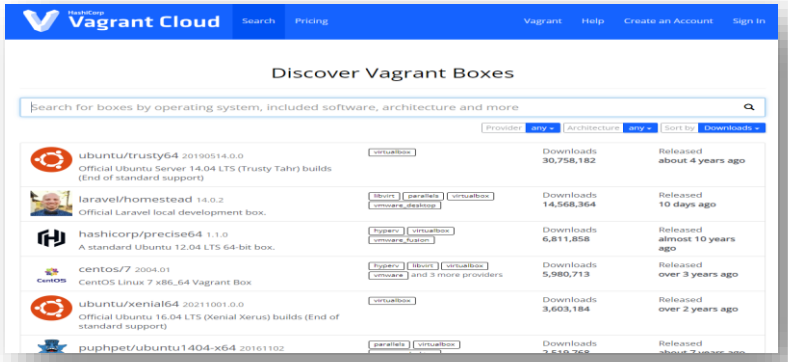
명령어	설명
vagrant init	프로비저닝을 위한 기초 파일 생성
vagrant up	Vagrantfile을 읽어들여 프로비저닝 시작
vagrant halt	Vagrant 가상 머신 종료
vagrant destroy	Vagrant 관리대상 가상 머신 삭제
vagrant ssh	Vagrant 관리 가상 머신에 ssh 접속
vagrant provision	Vagrant 관리하는 가상 머신에 변경된 설정을 적용

✦ Vagrant 작업순서

- 작업폴더 생성: `mkdir future_network`
- 작업폴더 이동: `cd future_network`
- 작업 초기화: `vagrant init`
- 스크립트 편집: `vi Vagrantfile`
- 스크립트 실행: `vagrant up`
- 작업 상태보기: `vagrant status`

Vagrant 클라우드

https://app.vagrantup.com/boxes/search



1-9. 클라우드 플랫폼 예시(3-1)

✦ Guest OS와 통신하기 위해 SSH Client

- Putty 프로그램 다운로드 및 설치
- 웹사이트: https://putty.org

Package files

You probably want one of these. They include versions of all the PuT

(Not sure whether you want the 32-bit or the 64-bit version? Read th

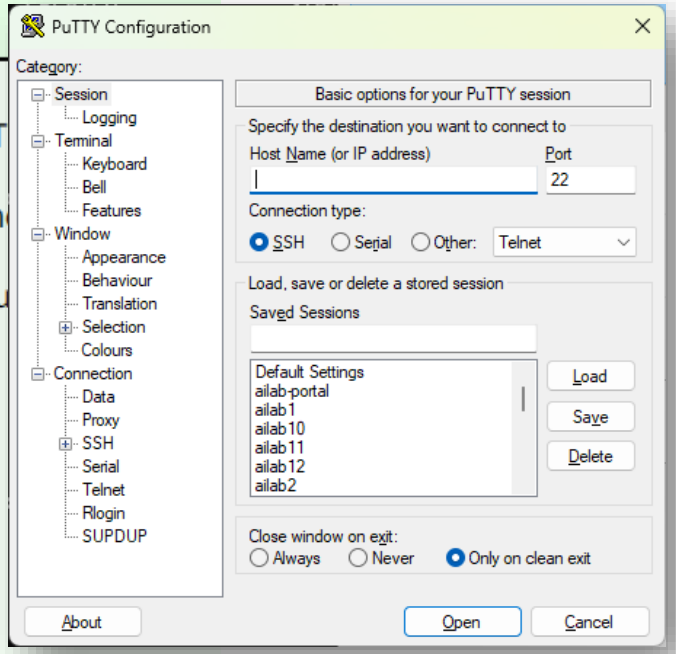
We also publish the latest PuTTY installers for all Windows architectu

MSI ('Windows Installer')

64-bit x86:	putty-64bit-0.83-installer.msi	(signature)
64-bit Arm:	putty-arm64-0.83-installer.msi	(signature)
32-bit x86:	putty-0.83-installer.msi	(signature)

Unix source archive

.tar.gz:	putty-0.83.tar.gz	(signature)



1-9. Vagrant 활용 VM 생성(4)

Vagrant 클라우드

https://app.vagrantup.com/boxes/search

1. `vagrant init` : 프로젝트 폴더 생성/이동 후 초기환경 생성

vi Vagrantfile

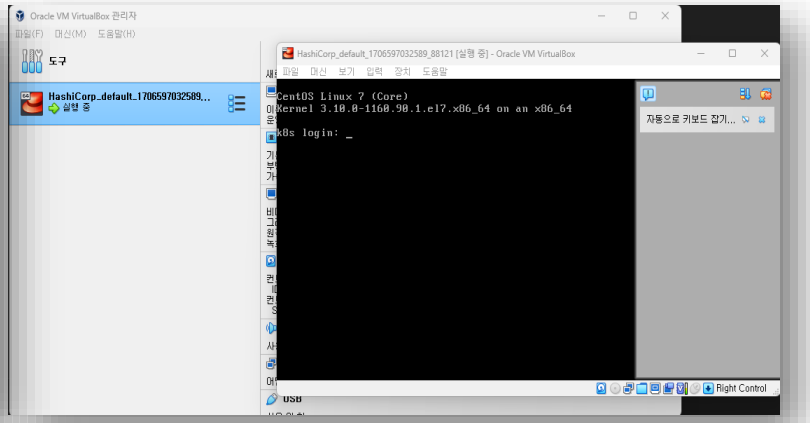
```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do | config |
   config.vm.box = "sysnet4admin/Ubuntu-k8s"
end
```

2. `vagrant up` : virtualbox에 vm 생성

3. `vagrant ssh` : CentOS에 ssh 연결

4. `vagrant destroy -f` : VM 삭제

```
PS F:\2024년-jenkins\HashiCorp> vagrant ssh
[vagrant@k8s ~]$ uptime
 15:52:26 up 8 min,  1 user,  load average: 0.00, 0.02, 0.03
[vagrant@k8s ~]$ cat /etc/redhat-release
CentOS Linux release 7.9.2009 (Core)
[vagrant@k8s ~]$ exit
logout
Connection to 127.0.0.1 closed.
PS F:\2024년-jenkins\HashiCorp> vagrant destroy -f
==> default: Forcing shutdown of VM...
==> default: Destroying VM and associated drives...
PS F:\2024년-jenkins\HashiCorp>
```



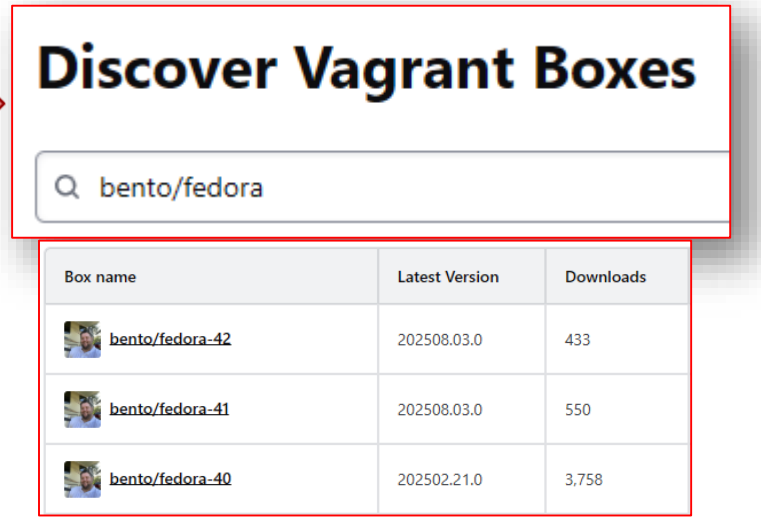
1-9. Vagrant 예시1

Vagrant 클라우드

https://app.vagrantup.com/boxes/search

키워드: bento/fedora

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure("2") do | config |
   config.vm.box = "bento/fedora-latest"
end
```



배포판	계열	패키지 형식 · 관리자	한 줄 특징
Debian	**Debian 계열**	.deb · apt/dpkg	안정성 중시, 보수적 업데이트
Ubuntu	**Debian 계열**	.deb · apt/dpkg (+ snap)	LTS 제공(일반 5년), 데스크톱/클라우드 친화
Fedora	**Red Hat 계열**	.rpm · dnf/rpm	최신 기술 빠른 반영, 릴리스 주기 짧음
CentOS	**Red Hat 계열**	.rpm · dnf/yum/rpm	엔터프라이즈 환경 유사, 지금은 **CentOS Stream** 사용 권장

1-9. Vagrant 예시2


```
[vagrant@fedora-docker ~]$ cat data/Vagrantfile
# -*- mode: ruby -*-
# vi: set ft=ruby :
Vagrant.configure("2") do |config|
  config.vm.box = "bento/fedora-latest"
  config.vm.provider "virtualbox" do |vb|
    vb.name = "fedora-docker"
    vb.memory = "2048"
    vb.cpus = 2
    vb.customize ["modifyvm",:id,"--groups","/Fedora"]
  end
  config.vm.hostname="fedora-docker"
  config.vm.network "forwarded_port", guest: 8888, host: 18888
  config.vm.network "private_network", ip: "192.168.5.20"
  config.vm.synced_folder ".", "/home/vagrant/data"
  config.vm.provision "shell", path: "pro.sh", privileged: true
end
```


```
[vagrant@fedora-docker ~]$ cat data/pro.sh
#!/usr/bin/bash
set -euo pipefail
# 필요시: dos2unix 설치
# sudo dnf -y install dos2unix

# 데이터 디렉토리 생성
if [[ ! -d /home/vagrant/data ]]; then
  mkdir -p /home/vagrant/data
fi

# Fedora에서 Docker CE 설치
sudo dnf -y install dnf-plugins-core
sudo dnf config-manager --add-repo https://download.docker.com/linux/fedora/docker-ce.repo
sudo dnf -y install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

# Docker 서비스 활성화
sudo systemctl enable --now docker

# vagrant 사용자를 docker 그룹에 추가 (다음 로그인부터 반영)
sudo usermod -aG docker vagrant
sudo setfacl -m user:vagrant:rw /var/run/docker.sock
```

Vagrant 예시2


```
# -*- mode: ruby -*-
# vi: set ft=ruby  :

VAGRANT_API_VERSION = "2"

hosts = {
   "n1" => "192.168.77.10",
   "n2" => "192.168.77.11",
   "n3" => "192.168.77.12"
}

Vagrant.configure(VAGRANT_API_VERSION) do |config|
  config.ssh.insert_key             = false
  config.ssh.forward_agent          = true
  check_guest_additions             = false
  functional_vboxsf                 = false
  config.vm.box = "bento/ubuntu-latest"

  hosts.each do |name, ip|
    config.vm.define name do |machine|
      machine.vm.network :private_network, ip: ip
      machine.vm.provider "virtualbox" do |vb|
        # Customize the amount of memory on the VM:
        vb.customize ["modifyvm", :id, "--cpus", 2]
        vb.name = name
      end
    end
  end
end
```

1-9. Vagrant 예시3


```
VAGRANT_API_VERSION = "2"

servers = [
{:hostname => "node1", :cpus => 1, :memory => 512, :box=> "centos/7" , :message=> "custom message for vm 1"},
{:hostname => "node2", :cpus => 1, :memory => 512, :box=> "centos/7" , :message=> "custom message for vm 2"},
{:hostname => "node3", :cpus => 1, :memory => 512, :box=> "centos/7" , :message=> "custom message for vm 3"}
]

Vagrant.configure(VAGRANT_API_VERSION) do |config|

  servers.each do |server|

    config.vm.define server[:hostname] do |nodeconfig|

      nodeconfig.vm.box = server[:box]
      nodeconfig.vm.hostname = server[:hostname]
      nodeconfig.vm.post_up_message = server[:message]
      nodeconfig.vm.provider "virtualbox" do |vb|
        # Display the VirtualBox GUI when booting the machine
        vb.gui = true
        # Customize the amount of memory on the VM:
        vb.memory = server[:memory]
        vb.cpus = server[:cpus]
        vb.customize ["modifyvm", :id, "--nic2", "natnetwork", "--nat-network2", "NatNetwork"]
        # vb.customize ["modifyvm", :id, "--memory", "2048"]
        # vb.customize ["modifyvm", :id, "--cpus", "2"]
      end
    end

  end
end
```

감사합니다.