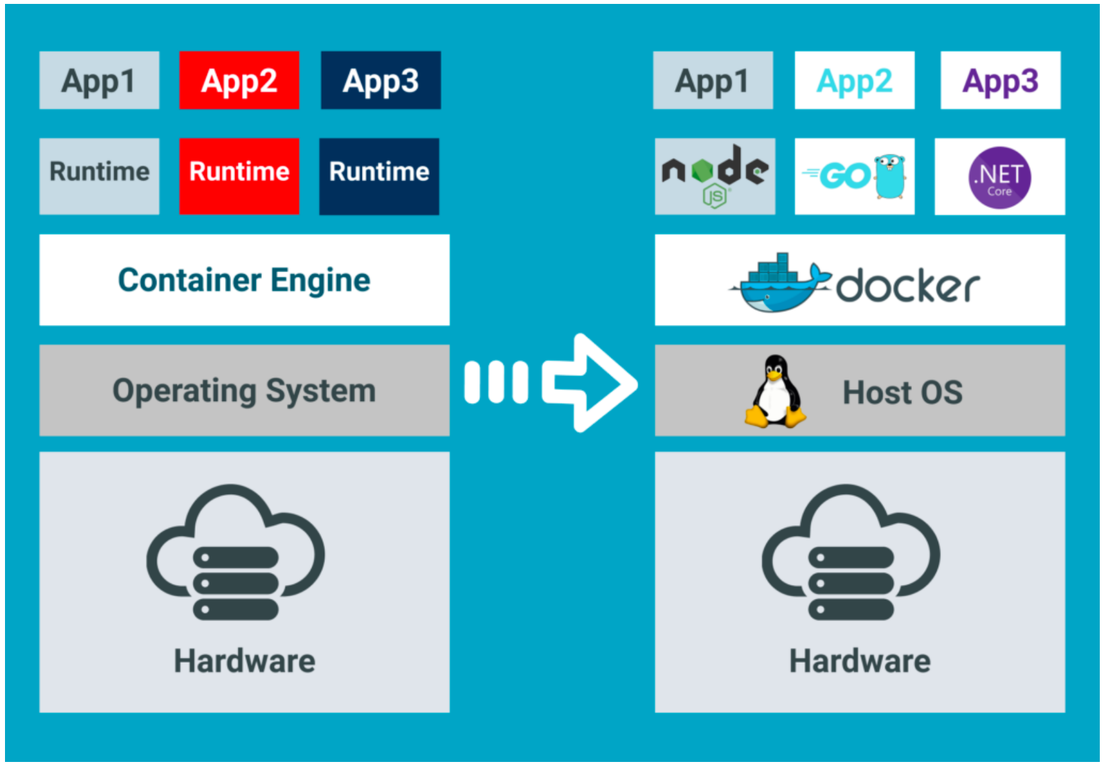
About Docker

docker

도커가 무엇일까?

도커는 리눅스의 응용 프로그램들을 **프로세스 격리** 기술들을 사용해 **컨테이너**로 실행하고 관리하는 오픈소스 프로젝트이다.

- 위키피디아 -



3) 각 애플리케이션을 '컨테이너'라는 단위로 격리시킨 가상화 시스템

1) 도커라는 친구는..

2) Host Os에 영향을 받지 않고

도커를 공부하기 전 도커의 기반이 되는 기술인
LXC(Linux Container)에 대해 알아야 합니다

컨테이너(Container)

컨테이너의 개념은 리눅스의 내장된 LXC(LinuX Container)
기술로부터 처음 소개됨.

LXC는 단일 머신 상 여러 개의 독립된 리눅스 커널
컨테이너를 실행하기 위한 OS레벨의 가상화 개념.

프로세스 격리 기술

리눅스 네임스페이스

리눅스의 리소스 접근을 제어하기 위해 사용, 리소스를 격리, 분리 할 수 있게 해준다

Chroot(Change root directory)

프로세스가 바라보는 루트 디렉토리를 파일 시스템상의 특정한 디렉토리로 변경시킴

컨트롤 그룹(Cgroups)

프로세스에서 사용가능한 CPU,메모리, 네트워크 대역폭, disk i/o 등 을 그룹 단위로 제어

Linux Capabilities, Union Mount...

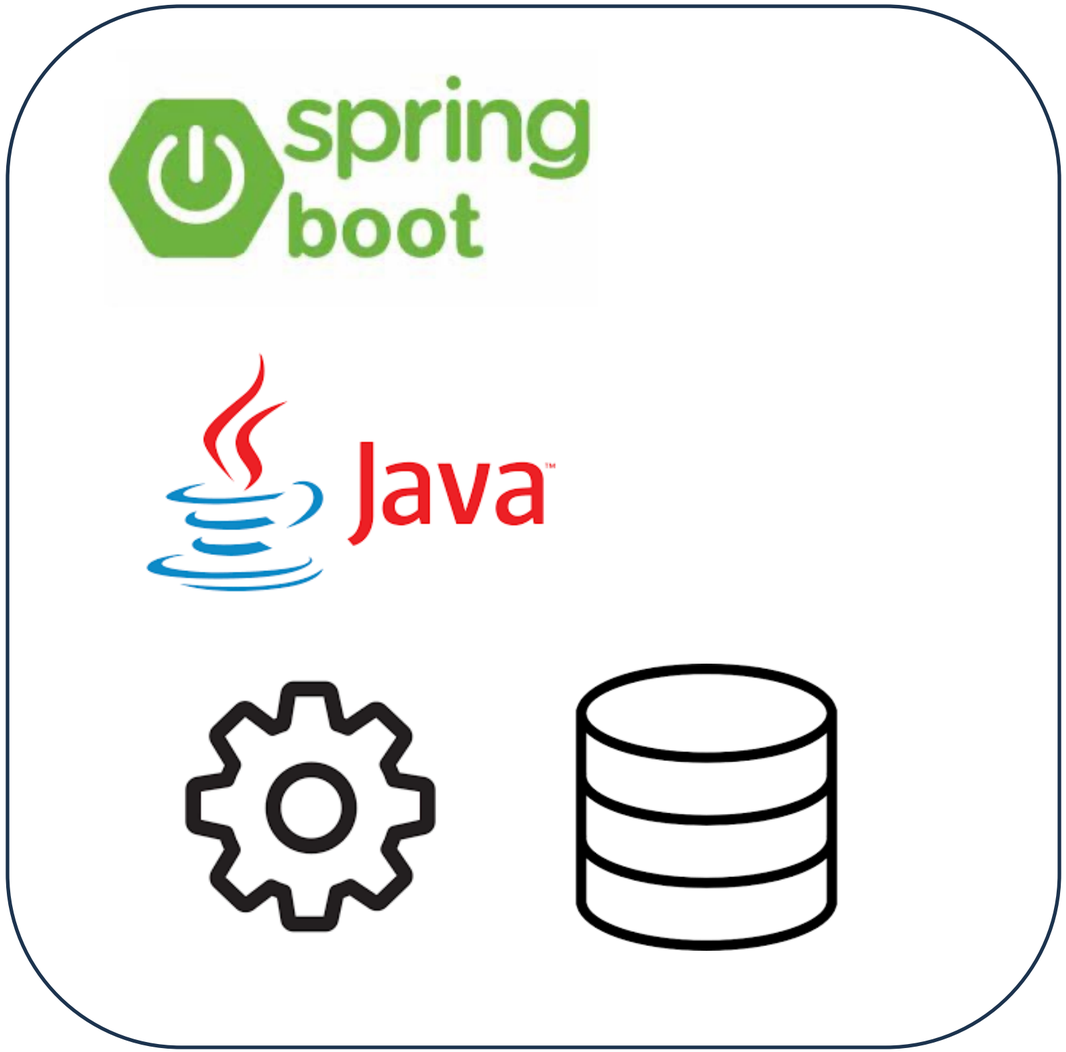
프로세스 권한 제어/ 계층화된 파일 시스템 구현

도커는 리눅스의 여러 기술들을 활용해서 컨테이너 단위로 격리시키며, 우리의 애플리케이션을 가상화 시킨다.

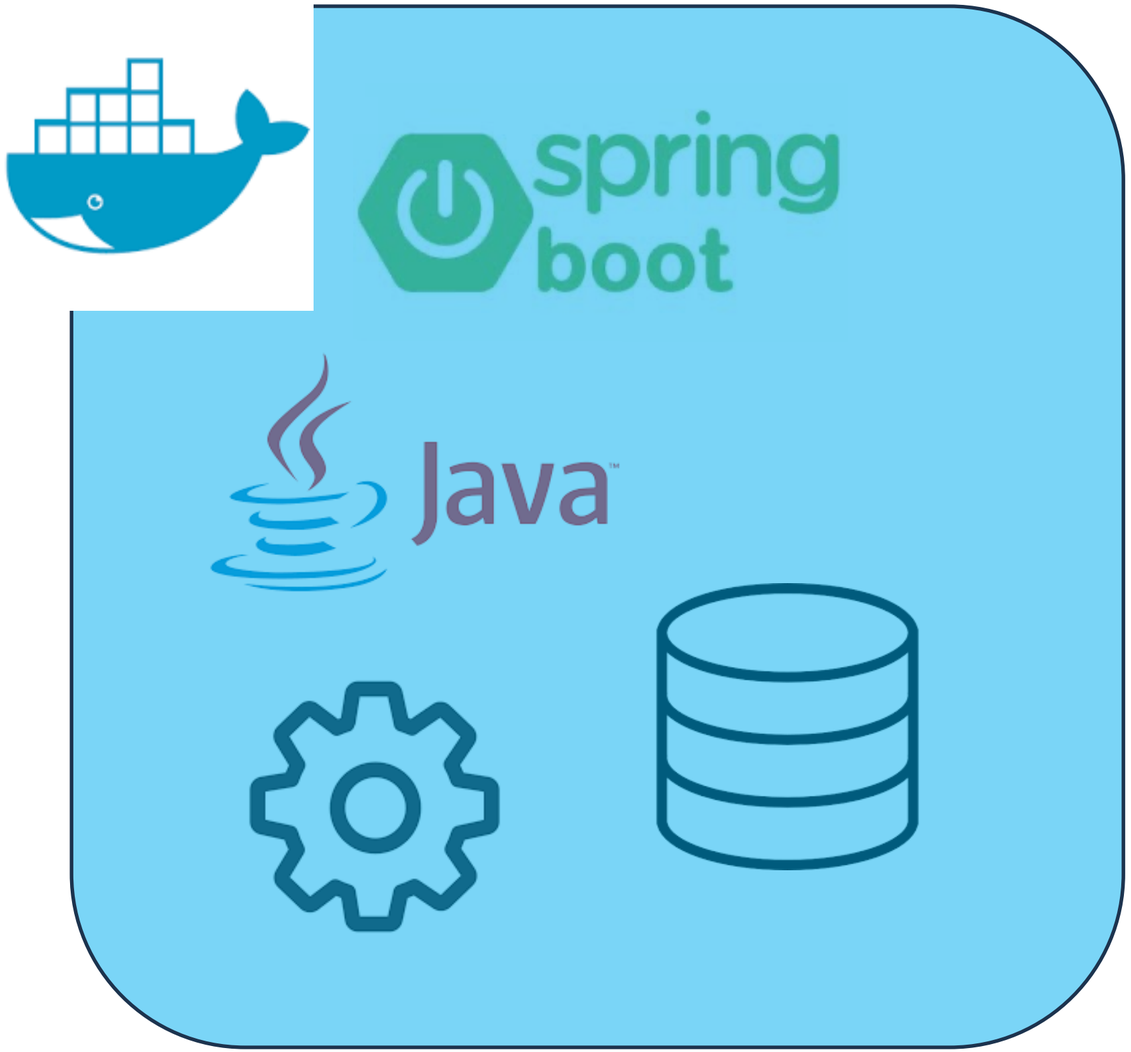
도커를 왜 사용할까?

Java 개발자들은 JVM만 있으면
어디든지 실행 가능한데 굳이 Docker를
써야 하나요?

실제 배포 서버를 위해서..



실제 배포 서버를 위해서..



docker info

라는 명령어를 실행시켜 봅시다

도커를 설치하면 도커 클라이언트 CLI와
도커 서버(데몬) 으로 구성된다

docker run hello-world

라는 명령어를 실행시켜 봅시다

```
inhyeokjo  ~  docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:a13ec89cdf897b3e551bd9f89d499db6ff3a7f44c5b9eb8bca40da20eb4ea1fa
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/
```

```
inhyeokjo  ~  docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:a13ec89cdf897b3e551bd9f89d499db6ff3a7f44c5b9eb8bca40da20eb4ea1fa
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/
```

CLI에 명령어를 입력하면 클라이언트에서
도커 서버로 요청을 보낸다.

이미지를 현재 locally에서 찾을 수 없기에 다운로드(pull)
을 진행한다.

도커 이미지

코드, 런타임, 시스템 도구, 라이브러리 및 설정 과 같은 응용 프로그램을 실행하는데 필요한 모든 것을 포함하는 가볍고 독립적이며 실행 가능한 소프트웨어 패키지

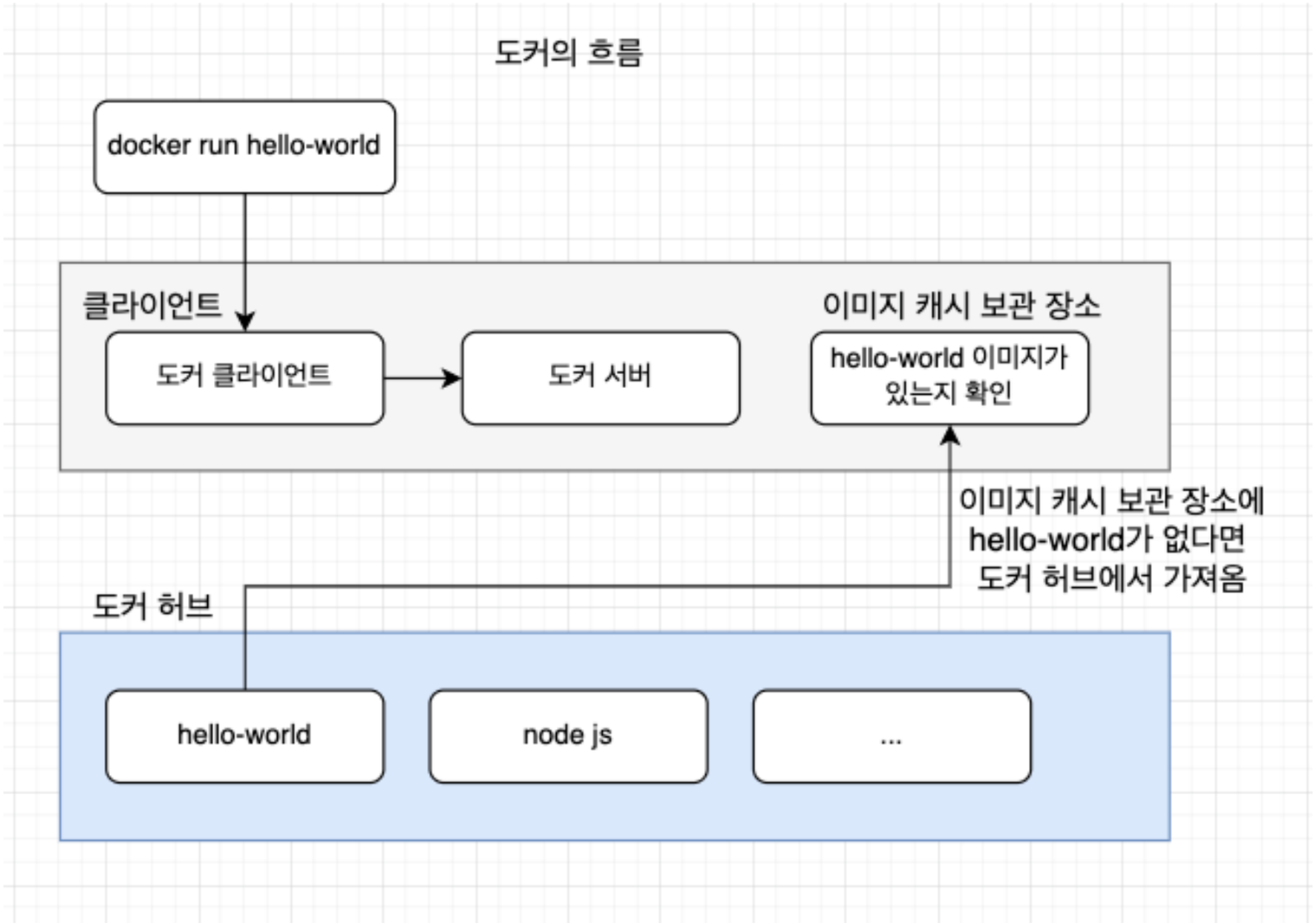
패키지를 이용해 표준 단위를 만들고 실행시킨다 ->

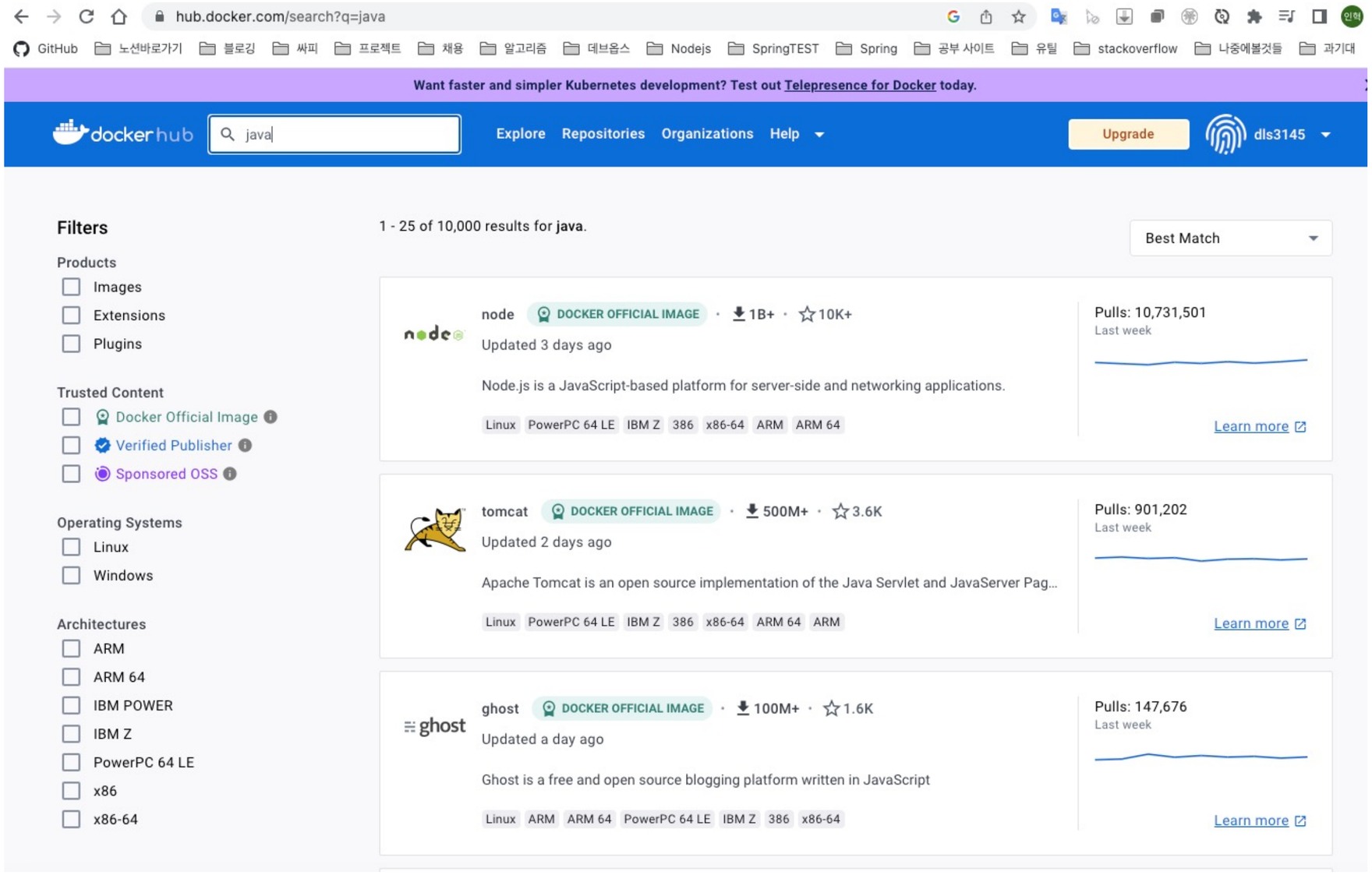
이미지를 이용해 컨테이너를 실행시킨다.

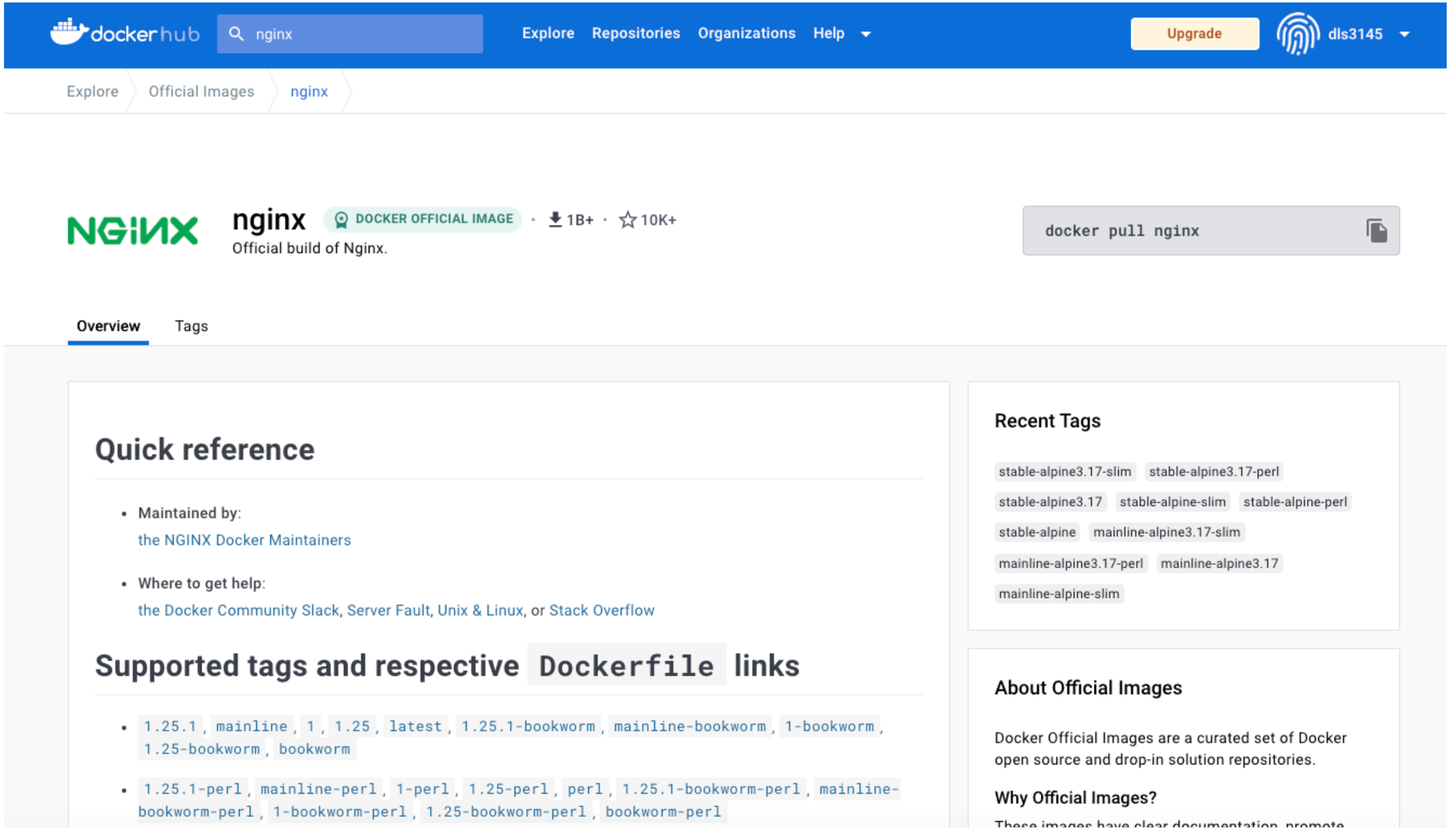
```
inhyeokjo  ~  docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
719385e32844: Pull complete
Digest: sha256:a13ec89cdf897b3e551bd9f89d499db6ff3a7f44c5b9eb8bca40da20eb4ea1fa
Status: Downloaded newer image for hello-world:latest

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/
```

이미지를 다운로드 완료 후 실행시킨다

컨테이너 실행 결과물


도커의 흐름
docker run hello-world
클라이언트
도커 클라이언트
도커 서버
이미지 캐시 보관 장소
hello-world 이미지가
있는지 확인
이미지 캐시 보관 장소에
hello-world가 없다면
도커 허브에서 가져옴
도커 허브
hello-world
node js
...


hub.docker.com/search?q=java
GitHub 노션바로가기 블로깅 싸피 프로젝트 채용 알고리즘 데브옵스 Nodejs SpringTEST Spring 공부 사이트 유팁 stackoverflow 나중에볼것들 과기대
Want faster and simpler Kubernetes development? Test out Telepresence for Docker today.
dockerhub
java
Explore
Repositories
Organizations
Help
Upgrade
dls3145
Filters
Products
Images
Extensions
Plugins
Trusted Content
Docker Official Image
Verified Publisher
Sponsored OSS
Operating Systems
Linux
Windows
Architectures
ARM
ARM 64
IBM POWER
IBM Z
PowerPC 64 LE
x86
x86-64
1 - 25 of 10,000 results for java.
Best Match
node
DOCKER OFFICIAL IMAGE
1B+
10K+
Updated 3 days ago
Node.js is a JavaScript-based platform for server-side and networking applications.
Linux PowerPC 64 LE IBM Z 386 x86-64 ARM ARM 64
Pulls: 10,731,501
Last week
Learn more
tomcat
DOCKER OFFICIAL IMAGE
500M+
3.6K
Updated 2 days ago
Apache Tomcat is an open source implementation of the Java Servlet and JavaServer Pag...
Linux PowerPC 64 LE IBM Z 386 x86-64 ARM 64 ARM
Pulls: 901,202
Last week
Learn more
ghost
DOCKER OFFICIAL IMAGE
100M+
1.6K
Updated a day ago
Ghost is a free and open source blogging platform written in JavaScript
Linux ARM ARM 64 PowerPC 64 LE IBM Z 386 x86-64
Pulls: 147,676
Last week
Learn more


dockerhub
nginx
Explore
Repositories
Organizations
Help
Upgrade
dls3145
Explore
Official Images
nginx
nginx
DOCKER OFFICIAL IMAGE
1B+
10K+
Official build of Nginx.
docker pull nginx
Overview
Tags
Quick reference
Maintained by:
the NGINX Docker Maintainers
Where to get help:
the Docker Community Slack, Server Fault, Unix & Linux, or Stack Overflow
Supported tags and respective Dockerfile links
1.25.1, mainline, 1, 1.25, latest, 1.25.1-bookworm, mainline-bookworm, 1-bookworm, 1.25-bookworm, bookworm
1.25.1-perl, mainline-perl, 1-perl, 1.25-perl, perl, 1.25.1-bookworm-perl, mainline-bookworm-perl, 1-bookworm-perl, 1.25-bookworm-perl, bookworm-perl
Recent Tags
stable-alpine3.17-slim
stable-alpine3.17-perl
stable-alpine3.17
stable-alpine-slim
stable-alpine-perl
stable-alpine
mainline-alpine3.17-slim
mainline-alpine3.17-perl
mainline-alpine3.17
mainline-alpine-slim
About Official Images
Docker Official Images are a curated set of Docker open source and drop-in solution repositories.
Why Official Images?

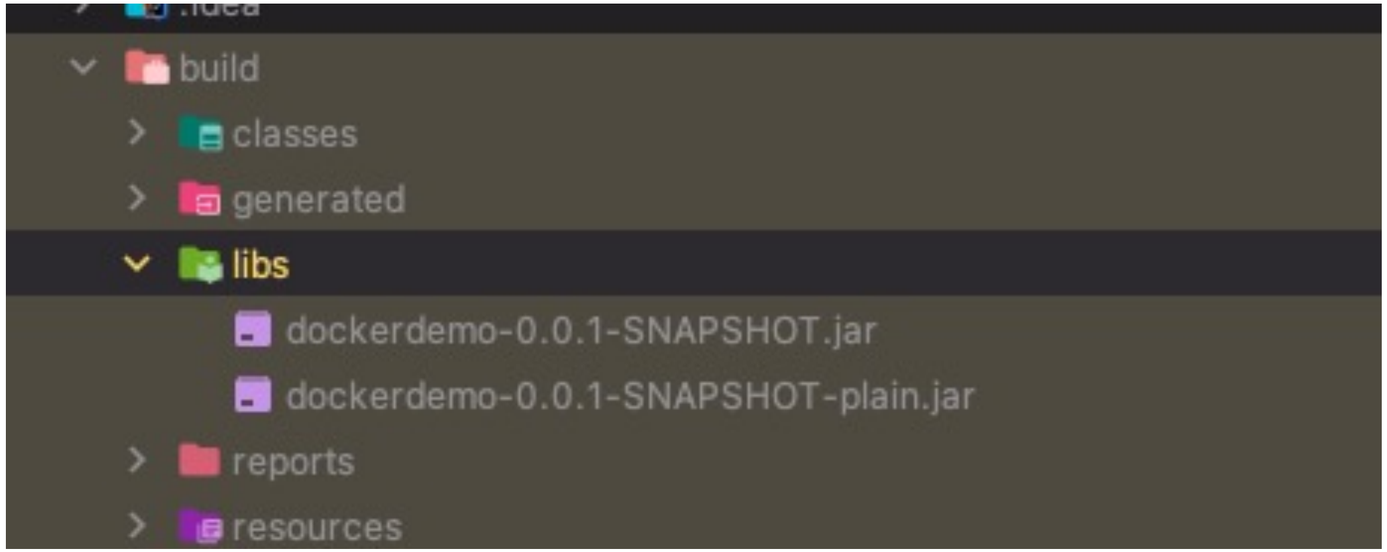
실습---직접 해보기 스프링 프로젝트로 컨트롤러 하나 작성

```java
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class DockerHelloController {

    @GetMapping("/hello-docker")
    public String helloDocker() {
        return "hello-docker";
    }
}
```

실습---직접 해보기 빌드 진행

./gradlew build

라는 명령어를 실행시켜 봅시다



실습---직접 해보기
도커 파일 만들기

루트 디렉토리에 Dockerfile
이라는 파일 이름으로 만들어봅시다.

```
FROM openjdk:17-oracle

WORKDIR app

ARG JAV_FILE=./build/libs/dockerdemo-0.0.1-SNAPSHOT.jar

COPY ${JAV_FILE} app.jar

EXPOSE 8080

ENTRYPOINT ["java","-jar","app.jar"]
```

실습---이미지 빌드 및 실행

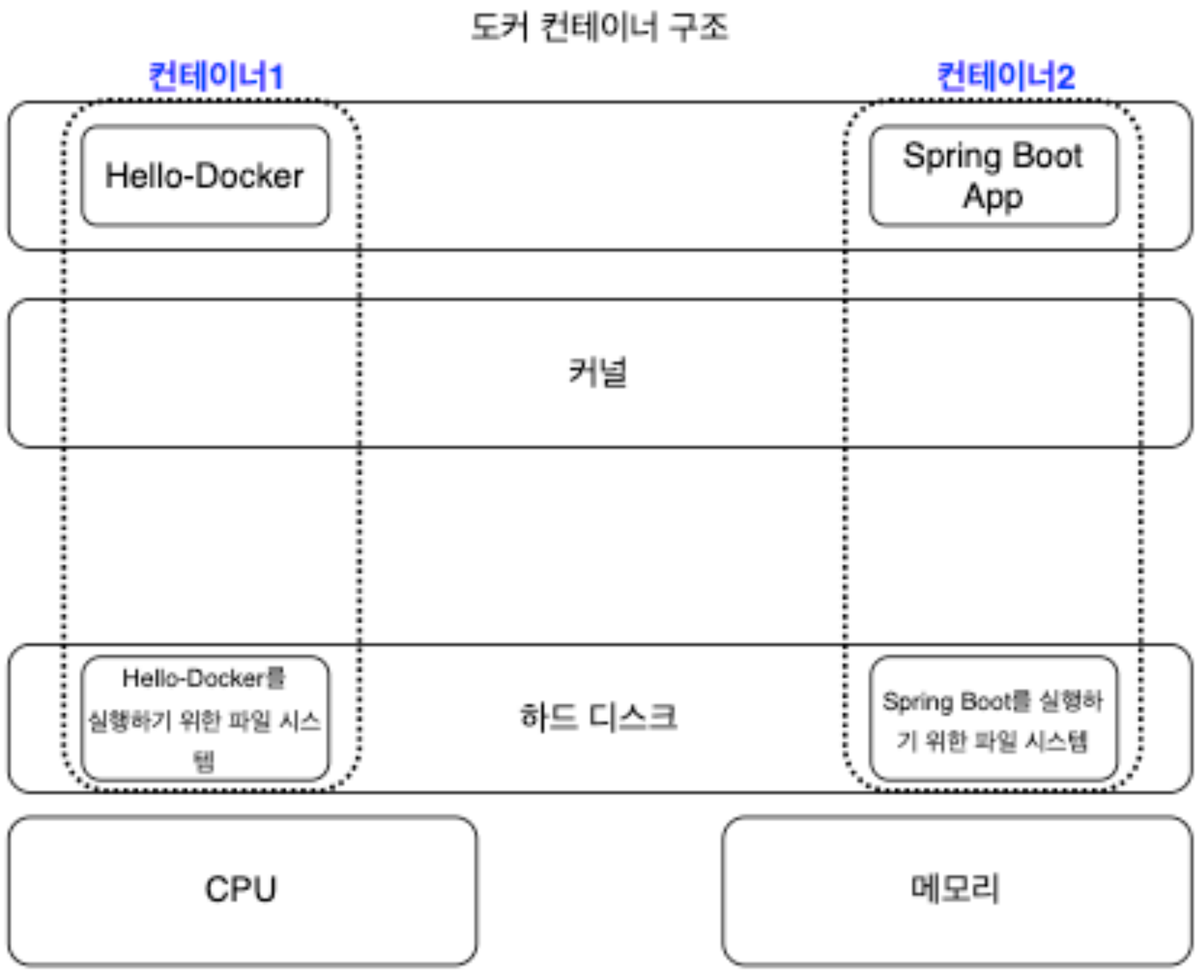
이미지 빌드 : docker build -t springimage .

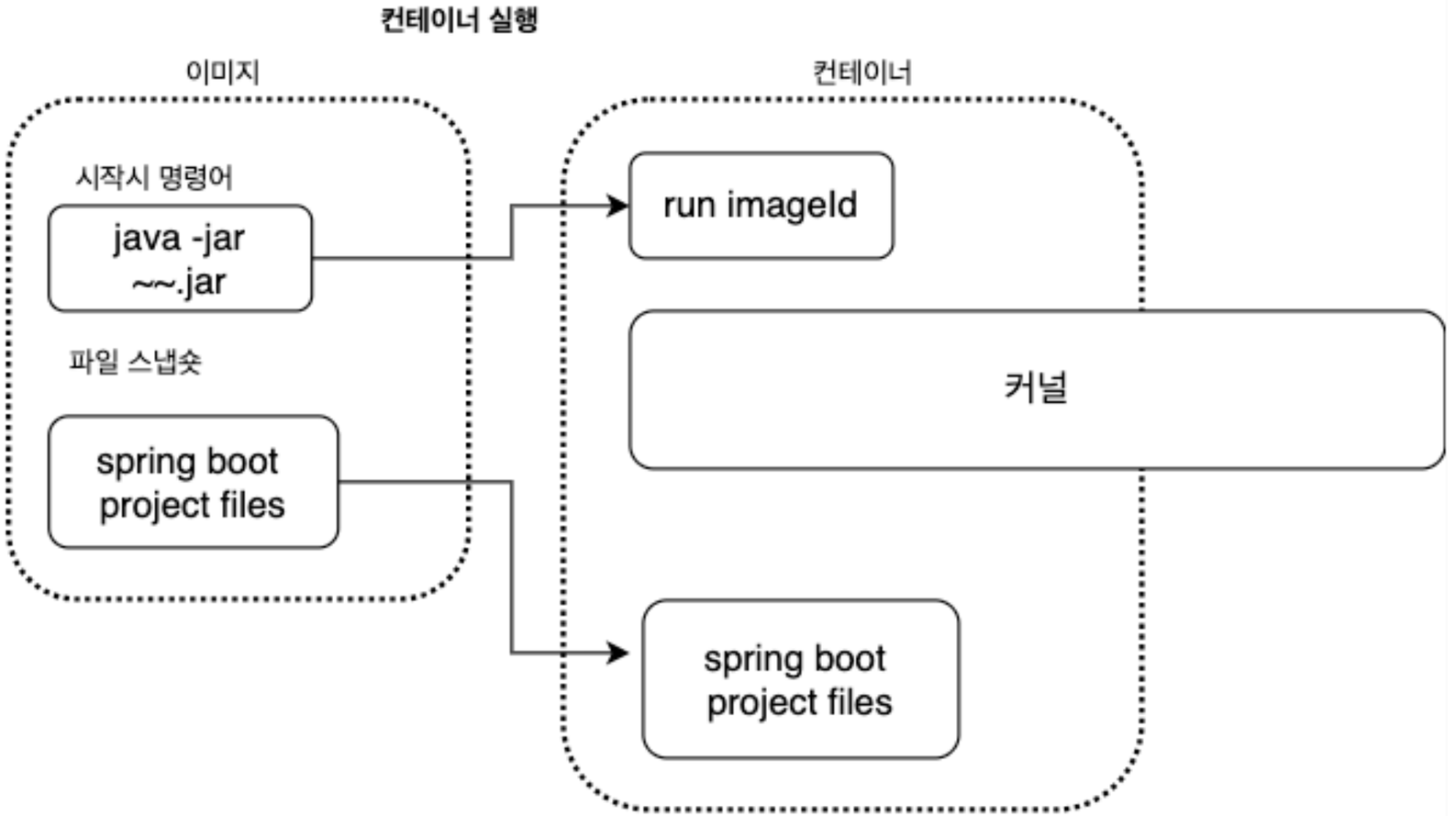
현재 디렉토리 . 기준으로 도커 이미지 빌드

이미지 확인 : docker images

컨테이너 실행 : docker run -d -p 8080:8080 --name mycontainer springimage

무슨 일이 일어난걸까?


도커 컨테이너 구조
컨테이너1
컨테이너2
Hello-Docker
Spring Boot App
커널
Hello-Docker를 실행하기 위한 파일 시스템
하드 디스크
Spring Boot를 실행하기 위한 파일 시스템
CPU
메모리


컨테이너 실행
이미지
시작시 명령어
java -jar
~~.jar
파일 스냅숏
spring boot
project files
컨테이너
run imageId
커널
spring boot
project files

도커 파일 명령어 설명

From

이미지 생성 시 기반이 되는 이미지 레이어 명시 <이미지이름>:<태그> 형식으로 작성
태그를 붙이지 않으면 자동으로 가장 최신 버전을 내려 받음

WORKDIR

컨테이너 내부 작업할 디렉토리 지정

RUN

도커 이미지가 생성되기 전에 수행할 셸 명령어

CMD or ENTRYPOINT

컨테이너가 시작할 때 실행할 명령어

추가적으로 run vs cmd vs entrypoint 에 대해 공부해보기!

도커 이미지 허브에 올리기

도커 허브 로그인 : docker login

도커 이미지 다시 빌드 : docker build -t [계정이름]/[이미지이름]:[태그] .

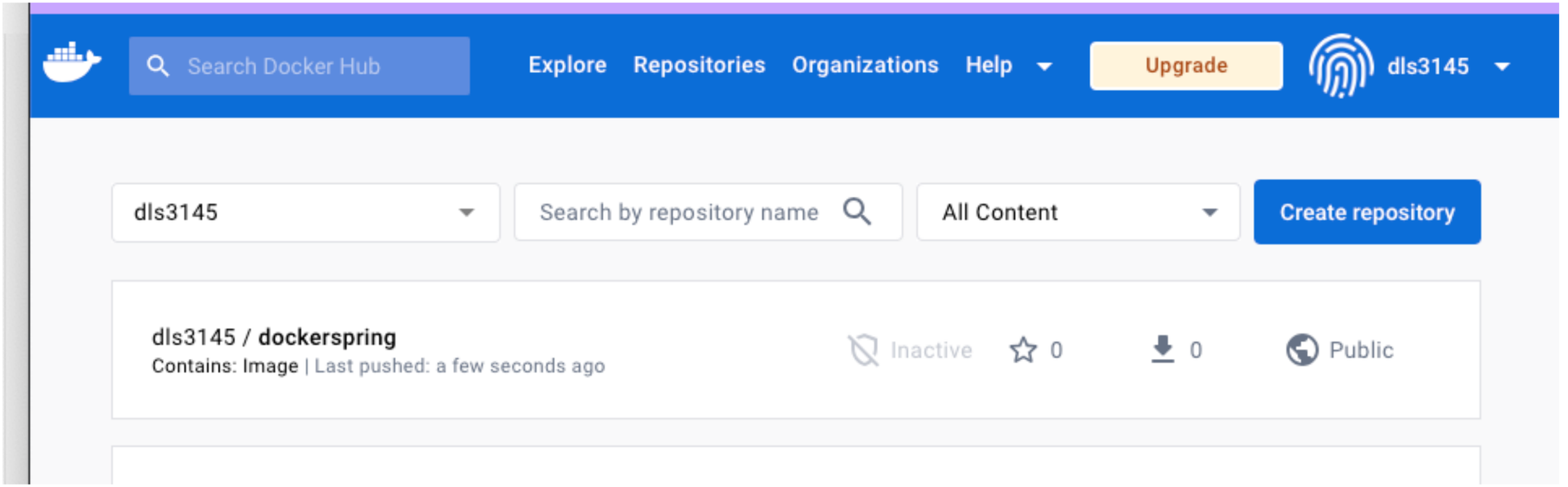
도커 이미지는 기본적으로 [저장소이름]/[이미지이름]:[태그]로 구성

저장소(repository) 이름 : 이미지가 저장된 장소, 이름이 명시되지 않은 이미지는 도커에서 기본적으로 제공하는 이미지 저장소인 도커 허브의 공식 이미지를 뜻한다.

이미지 이름 : 해당 이미지가 어떤 역할을 하는지 나타냄, 생략할 수 없으며 반드시 설정해야 함.

태그 : 이미지의 버전 관리 혹은 리비전(Revision)관리에 사용. 태그를 생략하면 도커 엔진은 이미지의 태그를 latest로 인식함.

도커 허브에 올리기 : docker push [계정이름]/[이미지이름]:[태그]



기존 도커 컨테이너 삭제 및 이미지 삭제

도커 컨테이너 상태 확인 : docker ps
도커 컨테이너 상태 확인 : docker ps -a

기존 컨테이너 종료 : docker stop [컨테이너id] or [컨테이너 이름]
기존 컨테이너 삭제 : docker rm [컨테이너id] or [컨테이너 이름]

이미지 삭제 : docker rmi [이미지id] or [이미지 이름]:[태그]

도커 허브에서 이미지 받아와서 실행하기

docker run -d -p 8080:8080 -name [컨테이너이름] [저장소이름]/[이미지이름]:[태그]

여기까지 도커에 대해서
세미나를 마치겠습니다

추가적으로 공부하기

docker image layer..

docker volume

docker network

docker compose

k8s..