01

AWS EC2 배포

01. SpringBoot 배포 순서

- AWS EC2 인스턴스 생성
  - 키페어 저장
- Spring Boot Build
  - Gradle – tasks- build- bookjar
- AWS EC2 인스턴스 접속
  - chmod 400 "20241211_ServerKey.pem"
  - ssh -i "20241211_ServerKey.pem" ubuntu@ec2-43-202-46-116.ap-northeast-2.compute.amazonaws.com
- Sudo su
- Apt update
- Apt upgrade
- Java –v

01. SpringBoot 배포 순서

- apt install openjdk-17-jre-headless
- Apt install mysql-server
- 다른 터미널을 열어 EC2에 파일 복사
  - scp -i 20241211_ServerKey.pem “springboot build 파일명.jar” ubuntu@ec2-43-202-46-116.ap-northeast-2.compute.amazonaws.com:/home/ubuntu
  - scp -i 20241211_ServerKey.pem “export 한 데이터베이스 테이블명.sql” ubuntu@ec2-43-202-46-116.ap-northeast-2.compute.amazonaws.com:/home/ubuntu

- EC2에서 Mysql –u root –p 접속
- Create database shop;
- Use shop;

01. SpringBoot 배포 순서

- Source “export한 데이터베이스 테이블.sql”;
- ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY ‘1234';
- Quit;
- Java –jar “springboot build 파일명.jar”

02. React 배포 순서

- EC2에서 탄력적 아이피 생성 인스턴스에 연결
- Frontend 요청 Ip 주소를 EC2의 아이피로 변경
- SpringBoot의 Frontend 아이피를 EC2의 아이피로 변경
- Frontend에서 npm run build
- SprinfBoot 빌드 파일 .jar를 EC2 /home/ubuntu에 복사
- Frontend 빌드 폴더안의 모든 내용을 EC2 /home/ubuntu/build안에 복사
- 서버의 Backgroud실행을 위해 nohup설치
- apt update
- apt install coreutils

02. React 배포 순서

- nohup java -jar Auth2JWT-0.0.1-SNAPSHOT.jar &
- Serve 설치를 위한 node.js 설치
- apt install -y nodejs
- apt install npm
- npm install -g serve
- serve -s build

03. Nginx로 React 배포

- Apt update
- apt install nginx
- systemctl start nginx
- systemctl status nginx
- rm /etc/nginx/sites-available/default
- rm /etc/nginx/sites-enabled/default
- cd /etc/nginx/sites-available/
- vi react-project.conf

03. Nginx로 React 배포

```
server {
    listen 80;
    server_name ec2 아이피 주소;

    root /var/www/html;
    index index.html index.htm;

    location / {
        try_files $uri $uri/  /index.html;
    }
}
```

- ln -s /etc/nginx/sites-available/react-project.conf /etc/nginx/sites-enabled/react-project.conf

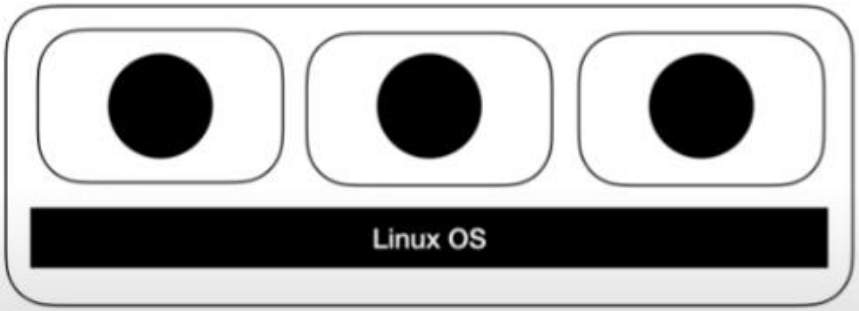
- systemctl restart nginx
- systemctl status nginx
- chmod -R 755 /var/www/html

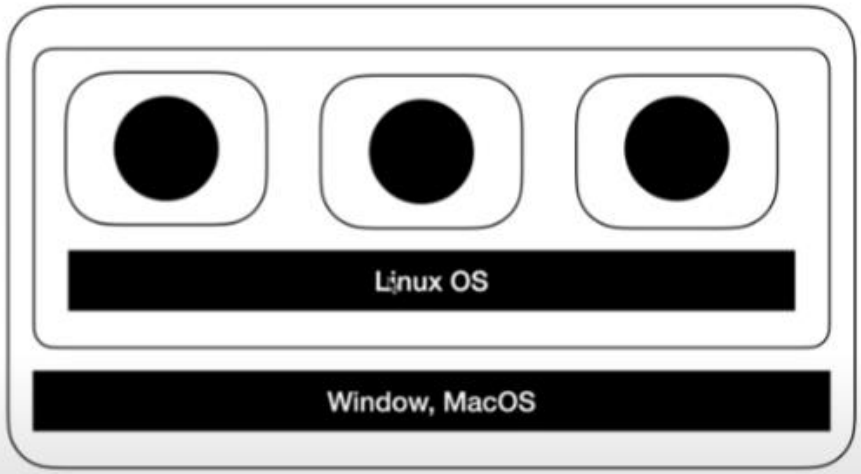
02

Docker

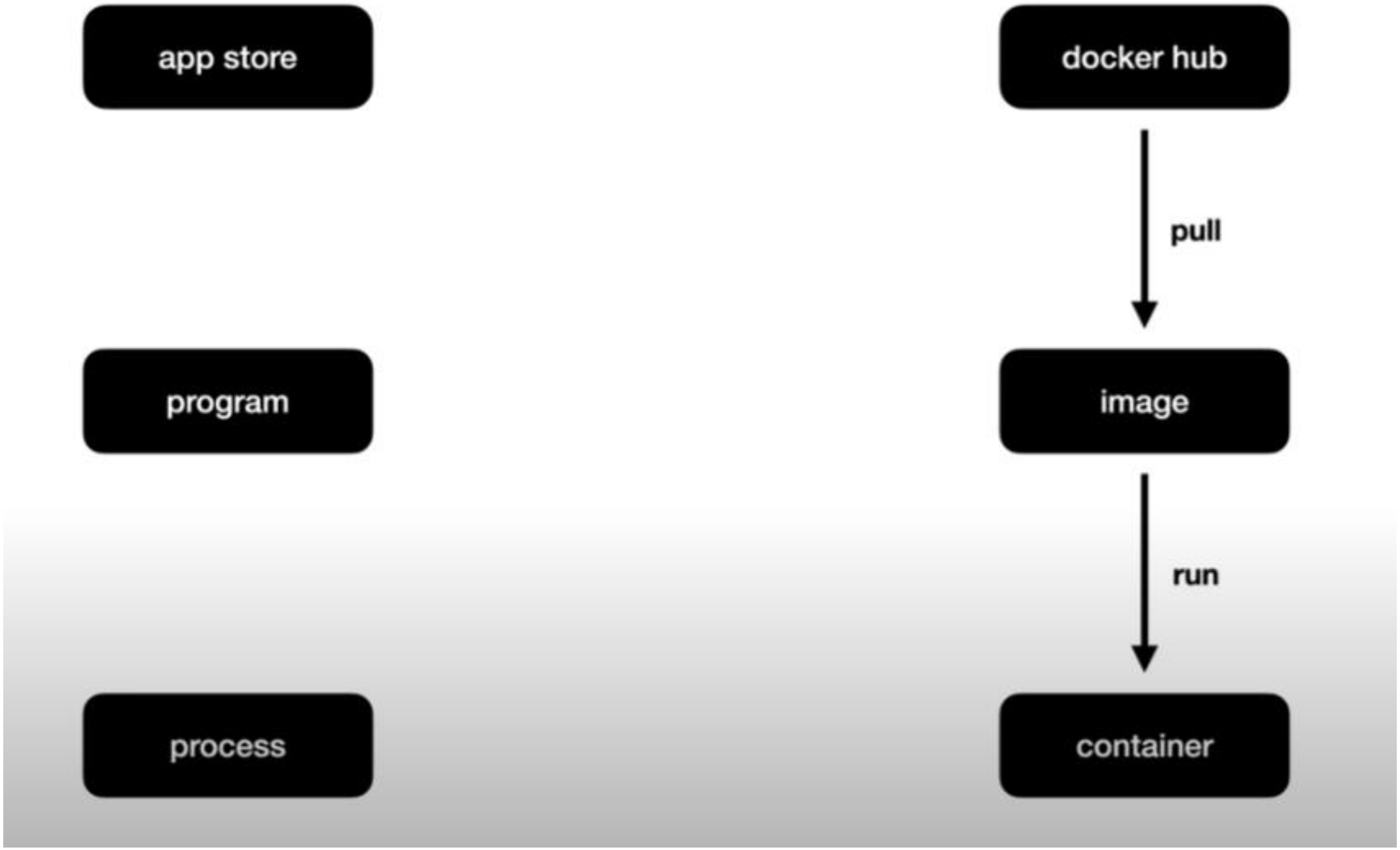
01. Docker

애플리케이션과 그 실행 환경을 컨테이너라는 가벼운 단위로 패키징 하여 어디서나 동일하게 실행할 수 있게 해주는 플랫폼





01. Docker


app store
docker hub
pull
program
image
run
process
container

02. Docker 기본 명령어

- Image 다운로드
  - docker pull 이미지 이름
  - docker pull ubuntu:18.04
- 다운로드된 이미지 보기
  - docker images
- 이미지 삭제
  - Docker rmi 이미지 이름 또는 아이디
  - Docker rmi unbuntu
- 이미지 실행하여 컨테이너 생성
  - Docker run 이미지 이름 또는 아이디
  - Docker run –d tomcat
  - Docker run –d –-name mytomcat tomcat
  - Docker run –d –p 8080:80 tomcat
  - Docker run –dit tomcat

02. Docker 기본 명령어

- 컨테이너 리스트 보기
  - Docker ps
  - Docker ps –a
- 컨테이너 중지
  - Docker stop 컨테이너 이름 또는 아이디
- 컨테이너 다시 시작
  - Docker start 컨테이너 이름 또는 아이디
- 컨테이너 삭제
  - Docker rm 컨테이너 이름 또는 아이디
- 현재 중지된 컨테이너 모두 삭제
  - docker rm $(docker ps -aq -f status=exited)

02. Docker 기본 명령어

- 현재 실행 또는 중지중인 모든 컨테이너 삭제
  - docker rm -f $(docker ps -aq)
- 현재 도커의 모든 이미지 삭제
  - docker rmi $(docker images -q)
- 현재 실행 중인 컨테이너의 로그 보기
  - docker logs 컨테이너 아이디
  - docker logs --follow 컨테이너 아이디

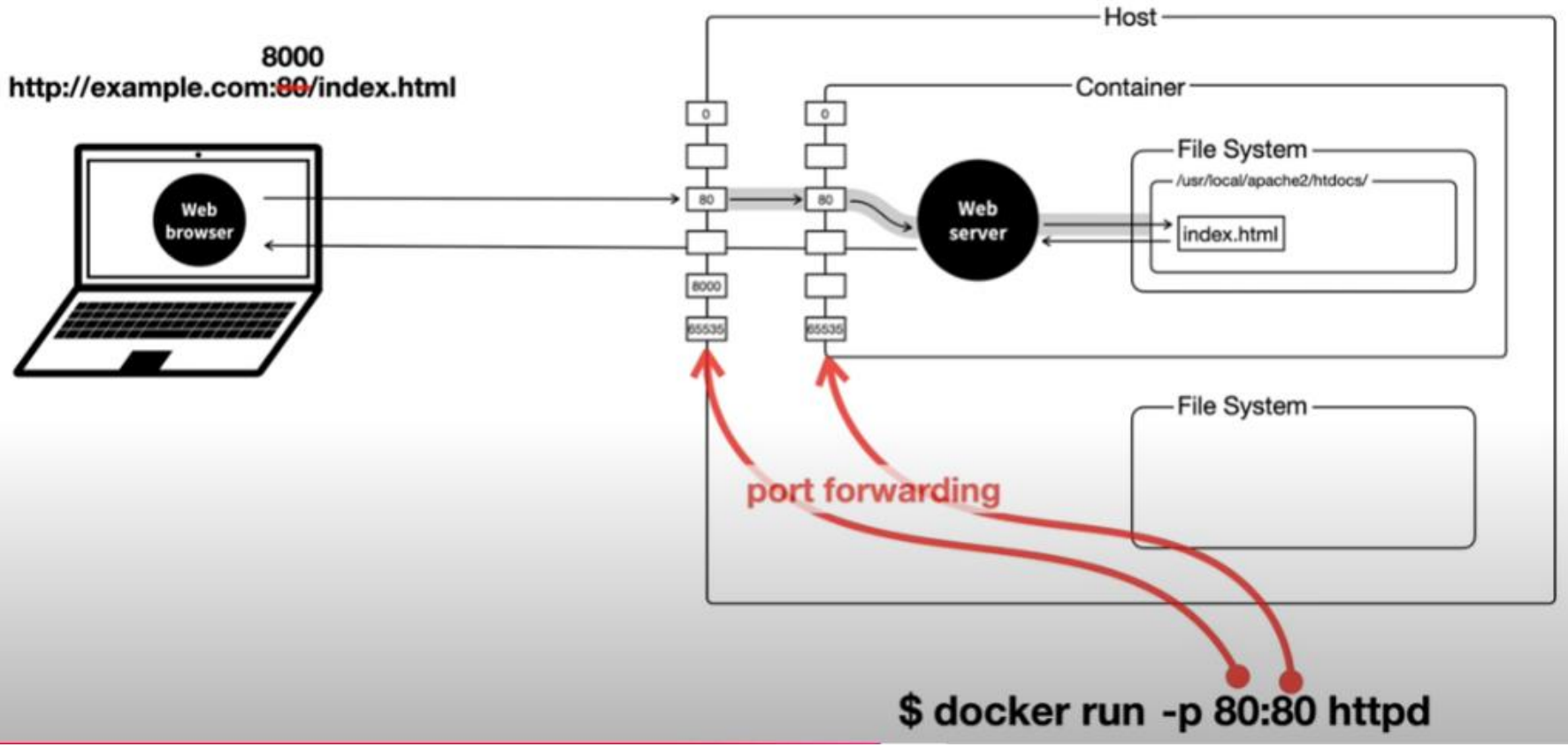
03. Docker 컨테이너의 생명주기

- 운영체제(예:ubuntu)와 같은 정적 프로그램
  - Docker run ubuntu, docker run –d ubuntu 와 같이 실행하면 실행되었다가 바로 종료
  - Docker run –dit ubuntu 로 실행하여야 함
- 웹서버(예: httpd, tomcat, nginx)와 같은 동적 프로그램
  - 주로 server의 역할을 하는 프로그램
  - Docker run –d tomcat 과 같이 Back ground로 실행

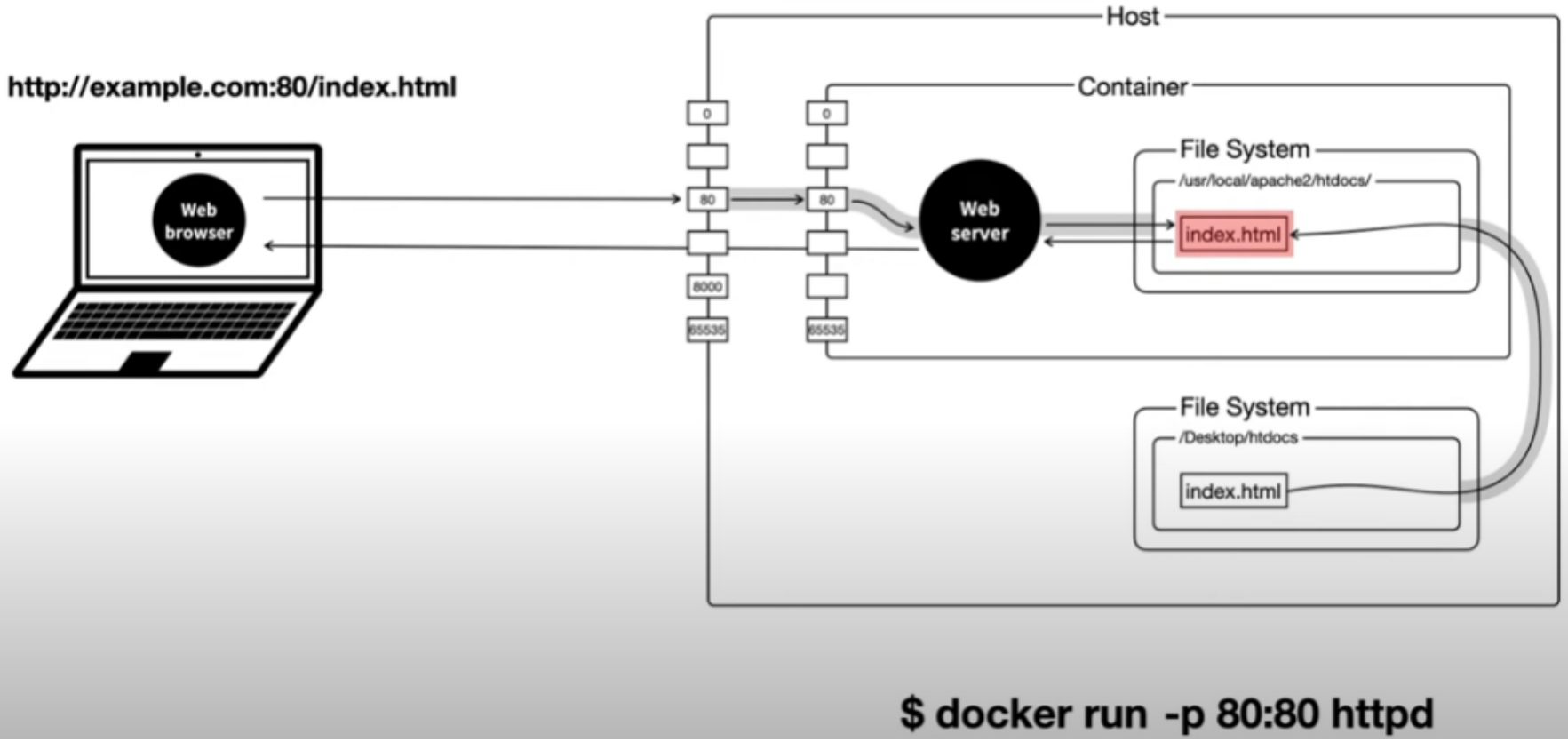
04. Docker 컨테이너 접근

- 정적 프로그램 접근
  - Docker attach 컨테이너 이름 또는 아이디
  - docker run -dit ubuntu
  - Docker attach a961
- 동적 프로그램 접근
  - Docker exec –it 컨테이너 이름 또는 아이디 sh
  - docker run -d -p 8080:80 httpd
  - Docker exec –it ef86 sh
- 컨테이너 접근 종료
  - Exit : 컨테이너가 중지되면서 접근 종료
  - Ctl + P + Q : 컨테이너 중지 없이 접근 종료

05. Docker 작동원리



05. Docker 작동원리



06. Docker Volumn 연결

- 컨테이너와 호스트의 File System을 연결
  - docker run -d -p 8080:80 -v d:₩temp₩html:/usr/share/nginx/html/ nginx
  - d:₩temp₩html와 컨테이너의 /usr/share/nginx/html/를 연결
  - 컨테이너의 /usr/share/nginx/html/에 들어가면 d:₩temp₩html를 접근할 수 있음

07. Docker hub 에 이미지 upload 실습

- Docker hub 가입
- Docker hub에 repository 생성
- docker run -dit ubuntu
- docker attach 93d8(컨테이너 아이디)
- Apt update
- Apt install vim
- Cd home
- Mkdir kim
- Cd kim
- Vi jinah
- ctrl+PQ

07. Docker hub 에 이미지 upload 실습

- docker commit 0419 closer19/vim-ubuntu:1.0
- docker images
- docker push closer19/vim-ubuntu:1.0
- docker rm -f $(docker ps -aq)
- docker rmi $(docker images -q)
- docker run -dit closer19/vim-ubuntu:1.0
- docker attach e934

08. Dockerfile 작성

- **임의의 위치에 Dockerfile 생성**

```
FROM httpd
COPY ./webapp /usr/local/apache2/htdocs
CMD ["httpd-foreground"]
```

- **같은 위치에 webapp폴더 생성**
- **webapp폴더안에 index.html생성**
- **docker build -t webserver:1.0 ./**
- **docker run -d -p 8080:80 webserver:1.0**
- **docker exec -it c667 sh**