도커 이미지 만들기

최 혁

도커 허브에 공유된 이미지 사용하기

- 소프트웨어 배포 기능을 도커 플랫폼이 완전히 내장했기에 이미지를 내려받는 과정을 도커에게 맡길 수 있지만, 도커 CLI를 통해 원시적으로 이미지를 내려받을 수도 있다.
- 무료로 공개되는 이미지를 제공하는 저장소(도커 허브): 공개 레지스트리
- 도커 이미지는 논리적으로는 하나의 대상이지만, 물리적으로는 여러 개의 작은 파일로 구성돼 있다.
    - 이들 각각의 파일을 이미지 레이어라고 한다.
    - 도커가 이들 파일을 조립해 컨테이너의 내부 파일 시스템을 만든다.
- 도커 컨테이너는 별도의 환경 변수를 가질 수 있다. 환경 변수는 호스트 운영체제의 것을 가져오는 것이 아닌, 도커가 부여해준다.

Dockerfile 작성하기

- 일련의 Instruction으로 구성돼 있다.
- 도커파일 문법은 매우 유연하고, 자주 쓰이는 작업은 별도의 명령으로 마련돼 있으며, 원하는 작업을 직접 작성할 수 있고, 표준 셸 문법도 사용 가능하다.

```
FROM diamol/node

ENV TARGET="blog.sixeyed.com"
ENV METHOD="HEAD"
ENV INTERVAL="3000"

WORKDIR /web-ping
COPY app.js .

CMD ["node", "/web-ping/app.js"]
```

FROM: 출발 이미지

- 모든 이미지는 다른 이미지로부터 출발한다.

ENV: 환경 변수 값 지정

- key="value" 형식을 따른다.

WORKDIR: 컨테이너 이미지 파일 시스템에 디렉터리를 만들고, 해당 디렉터리를 작업 디렉터리로 지정

- 리눅스와 윈도우 모두 구분자로 /(슬래시)를 사용한다.

COPY: 로컬 파일 시스템의 파일 or 디렉터리를 컨테이너 이미지로 복사

- COPY [원본 경로] [복사 경로] 형식이다.

CMD: 도커가 이미지로부터 컨테이너를 실행했을 때 실행할 명령어

컨테이너 이미지 빌드하기

```
docker image build --tag [이미지 이름] [이미지에 들어갈 파일이 위치한 디렉터리]
```

도커 이미지와 이미지 레이어 이해하기

- 이미지에는 자신에 대한 여러 메타데이터 정보가 들어있어 어떻게 빌드됐는지 등 다양한 정보를 볼 수 있다.
    - docker image history [image name]
- dockerfile script의 instruction은 각각 하나의 이미지 레이어와 1:1로 연결된다.
- 레이어는 도커 엔진의 캐시에 물리적으로 저장된 파일이다.
    - 따라서 이미지 레이어는 여러 이미지와 컨테이너에서 공유된다.
    - docker image ls로 나오는 이미지 용량은 이미지가 실제로 차지하는 디스크 용량이 아닌 논리적인 용량이다.
- 이미지를 빌드하면서 레이어가 만들어지면 레이어는 읽기 전용으로 만들어 수정 불가능하다.

```
# 이미지 저장에 실제로 사용된 디스크 용량을 확인할 수 있다.
docker system df
```

이미지 레이어 캐시를 이용한 Dockerfile 스크립트 최적화

- instruction의 결과가 이전 빌드와 같다면, 이전에 캐시된 레이어를 재사용한다.
- 도커는 캐시에 일치하는 레이어가 있는지 확인하기 위해 해시값을 이용한다.
  - 해시값은 instruction과 instruction에 의해 복사되는 파일의 내용으로 계산된다.
  - 기존 이미지 레이어에 해시값이 일치하는 것이 없다면 캐시 미스가 발생하고 instruction이 실행된다.
  - 한 번 instruction이 실행되면 그 다음에 오는 instruction은 수정된 것이 없어도 실행된다.
  - 따라서 dockerfile script는 잘 수정되는 instruction이 뒤로 가야 한다.

최적화한 도커 파일

```
FROM diamol/node

CMD ["node", "/web-ping/app.js"]

ENV TARGET="blog.sixeyed.com" \
    METHOD="HEAD" \
    INTERVAL="3000"

WORKDIR /web-ping

COPY app.js .
```

연습문제

```
도커허브에 공유된 diamol/ch03-lab 이미지가 있고, 이 이미지 안에 ch03.txt가 있다.
ch03.txt 파일 수정 이후, 수정된 파일을 포함하는 새로운 이미지를 빌드하자.
(dockerfile 스크립트 금지)
```

```
docker container run -it diamol/ch03-lab
# ch03.txt 파일 수정
exit
docker container commit ch03 ch03-change # 컨테이너 상태 그대로 이미지 생성
docker container run -it ch03-change
cat ch03.txt
```

플래그 정리

--name: 컨테이너에 원하는 이름을 붙이고 그 이름으로 컨테이너를 지정할 수 있다.
--env / -e: key=value 형태로 환경변수 지정