기본적인 도커 클라이언트 명령어

작성일시	@2023년 1월 6일 오전 10:23
강의 번호	Docker
유형	
자료	
복습	☐
학습 소스 출처 1	
학습 소스 출처 2	
날짜	

docker - 도커 클라이언트 언급

run - 컨테이너 생성 및 실행

이미지 이름 - 이 컨테이너를 위한 이미지

ls - 이 위치는 이미지에 있는 명령어를 무시하고 , 실행하게 되는 명령어를 의미

alpine이란 이미지안에는 파일들 중에서 , ls를 실행할 수 있는 파일이 존재한다.

컨테이너 나열 명령어

docker ps (process status)

ps의 항목

- Container ID : 컨테이너 고유한 아이디 해쉬값
- Image : 컨테이너 생성시 사용한 도커이미지
- Command : 컨테이너 시작시 실행될 명령어(대부분 이미지에 내장되어있으므로 별도 설정 x)
- Created : 컨테이너가 생성된 시간
- Status : 컨테이너의 상태 up: 실행중 , Exited: 종료 , Pause :일시정지

- Ports : 컨테이너가 개방한 포트, 호스트가 연결한 포트를 표시
- Names : 컨테이너 고유한 이름. 생성시 —name 옵션으로 설정 가능하지만, 도커엔진이 임의로 이름 설정

create 하게되면 스냅샷까지 생성

kill 과 stop

둘다 컨테이너의 작업을 중지시키지만 ,

stop 은 자비롭게(Gracefully) 중지 , 즉 현재 진행중인 작업을 정리하고 중지 시킨다.

docker stop 명령을 받은 후에 SIGTERM 명령을 받아 grace Peroid (정리기간)을 동안 정리하고 SIGKILL 처리한다.

하지만 kill은 정리하는 시간을 주지 않고 바로 꺼버린다.

즉 바로 SIGKILL 처리를 끝낸다.

도커의 생명주기

docker rm <아이디 /이름>

해당 아이디 혹은 이름을 가지는 컨테이너를 제거

docker rmi 이미지 이름

해당 이미지를 제거

docker system prune

모든 메모리 , 네트워크 , 이미지 등 전부 삭제

실행중인 컨테이너에 명령전달

docker exec <컨테이너 아이디> <명령>

exec 는 exercise 를 의미한다. 해당 컨테이너에서 실행할 수 있는 <명령>을 실행하게 된다.

run 을 사용할때 이미지 이름 다음에 명령을 쓰는것과 명령 형태가 유사한데

run과의 차이는 run은 새로 컨테이너를 만들어서 명령을 실행하고 , exec는 이미 실행중인 컨테이너에 명령을 전달하는 방식이다.

레디스를 이용한 컨테이너 이해

docker run redis

를 이용해 redis를 실행하게 된다면 해당 클라이언트에선 아무것도 입력할수 없게 되고

다른 클라이언트에서도 더이상 run으로는 접근할수 없게된다.

이때는 위에서 배웠던 exec을 이용하여 이미 동작하고 있는 redis 컨테이너에 명령을 전달하는 방식으로 해야한다.

docker exec -it <컨테이너아이디> 명령어

여기서 -it 는 명령어를 실행한후 계속 명령어를 적을 수 있게하는 명령어이다. -i : interactive , -t :terminal

보통은 명령어를 실행한후 해당작업에서 나오기때문에

더 작업을 해야하는 경우 -it를 사용해야한다.

실행중인 컨테이너에서 터미널 생활 즐기기

컨테이너에서 원하는 작업을 하려할때마다 복잡하고 긴 키워드를 전부 쳐야했었다.

하지만 컨테이너 안에서 쉘이나 터미널환경으로 접속하면 편해진다.

마지막 명령에 -sh 를 입력하게 되면

위에서 썼던 ls 나 echo등 리눅스 명령들을 해당컨테이너 ‘안’에서 편하게 사용할 수 있다.

ex) docker exec -it < 컨테이너아이디> sh

를 사용하면 쉘환경으로 들어가게 되고

나올때는 ctrl+D로 나올 수 있다.