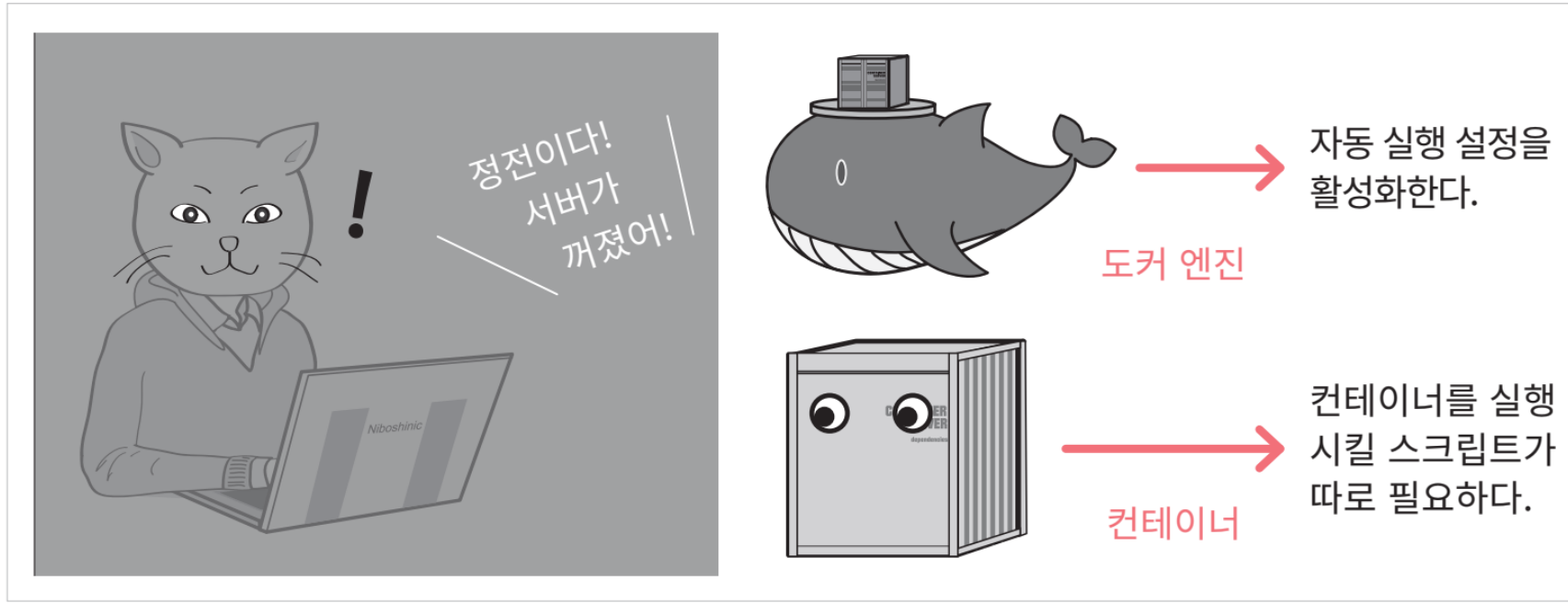


그림 4–1–1 도커 엔진은 자동 실행 설정이 가능하지만 컨테이너는 그렇지 않다.

도커 기본적인 명령어 정리

도커 명령어의 기본 형태 : **docker + 커맨드 + (옵션) + 대상 + (인자)**

- 커맨드

: '무엇을', '어떻게'를 할 것인지 지정하는 부분

: '컨테이너를'(무엇을) '실행'(어떻게)하고 싶다면 container run 을 사용

: start나 run 처럼 container를 붙이지 않아도 실행 가능한 명령어가 있으며, 관행상 생략하는 경우가 많음

- 옵션

: 커맨드에 세세한 설정을 지정하는 용도

: - 또는 --으로 시작하는 것이 일반적이지만 기호를 붙이지 않는 경우도 존재

-d : 백그라운드로 실행

-i, -t : 키보드로 조작

--name : 커맨드에 값 전달

-dit : -d + -i + -t를 합친 옵션

- 대상

: 커맨드와 달리 구체적인 이름을 지정

- 인자

: 대상에 전달할 값을 지정

: 문자 코드 또는 포트 번호 등을 전달할 수 있으며, 인자를 지정하는 경우는 많지 않음

: 옵션과 마찬가지로 - 또는 --로 시작하는 경우가 많음

명령 프롬프트

```
C:\Users\mit-305>docker version
Client:
 Version:           28.0.4
 API version:       1.48
 Go version:        go1.23.7
 Git commit:        b8034c0
 Built:             Tue Mar 25 15:07:48 2025
 OS/Arch:           windows/amd64
 Context:           desktop-linux

Server: Docker Desktop 4.40.0 (187762)
 Engine:
  Version:          28.0.4
  API version:      1.48 (minimum version 1.24)
  Go version:       go1.23.7
  Git commit:       6430e49
  Built:            Tue Mar 25 15:07:22 2025
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.7.26
  GitCommit:        753481ec61c7c8955a23d6ff7bc8e4daed455734
 runc:
  Version:          1.2.5
  GitCommit:        v1.2.5-0-g59923ef
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0

C:\Users\mit-305>
```

도커 명령어의 기본 형태 :

docker + 커맨드 + (옵션) + 대상 + (인자)

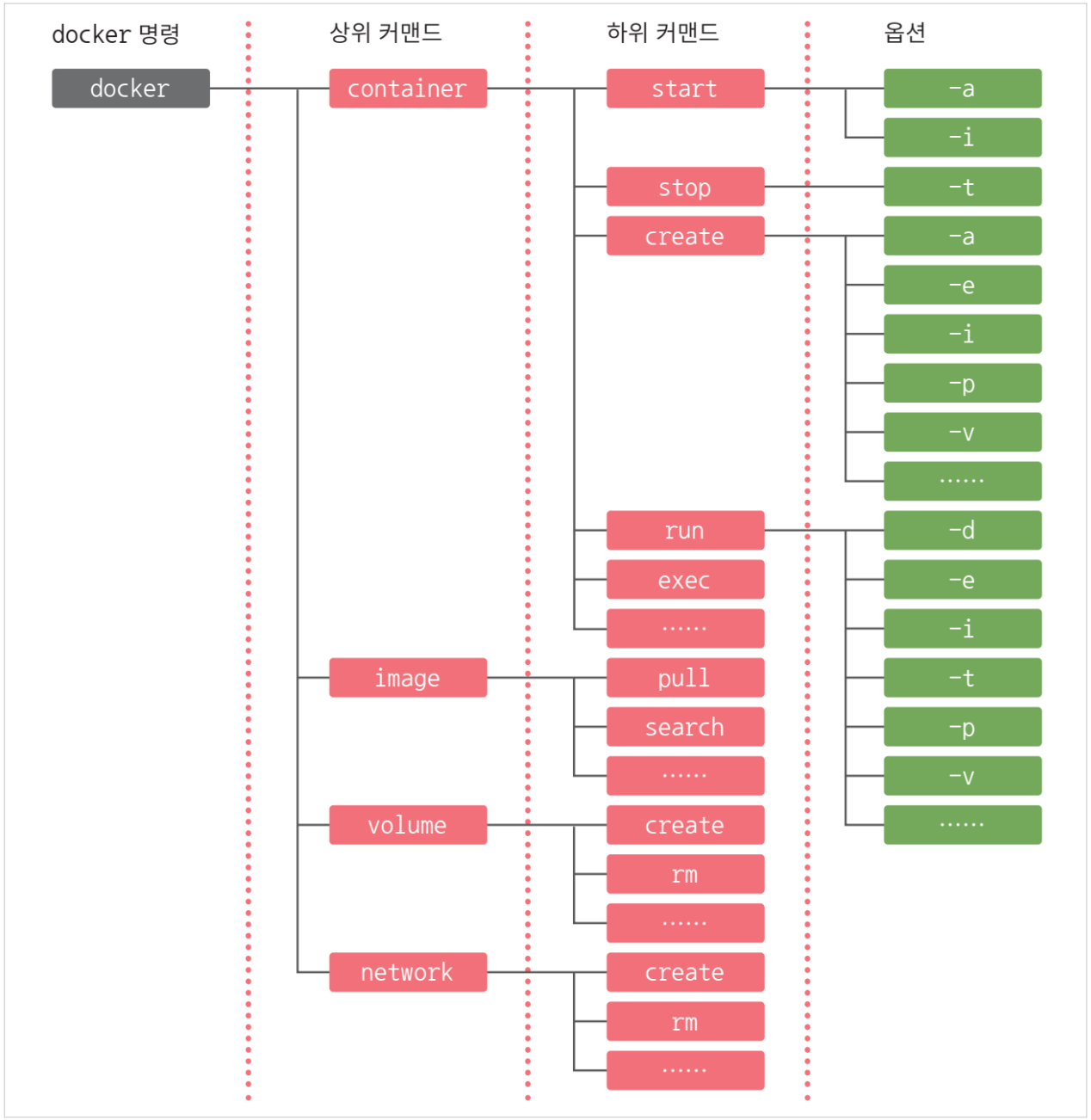


그림 4-2-1 대표적인 명령어

docker container 하위_커맨드 옵션

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
start	컨테이너를 실행	O	-i
stop	컨테이너를 정지	O	거의 사용하지 않음
create	도커 이미지로부터 컨테이너를 생성	O	--name -e -p -v
run	도커 이미지를 내려받고 컨테이너를 생성해 실행함(다운로드는 필요한 경우에만). docker image pull, docker container create, docker container start라는 세 개의 명령을 하나로 합친 것과 같다.	O	--name -e -p -v -d -i -t
rm	정지 상태의 컨테이너를 삭제	O	-f -v
exec	실행 중인 컨테이너 속에서 프로그램을 실행	O	-i -t
ls	컨테이너 목록을 출력	*1	-a
cp	도커 컨테이너와 도커 호스트 간에 파일을 복사	O	거의 사용하지 않음
commit	도커 컨테이너를 이미지로 변환	O	거의 사용하지 않음

생략 가능 커맨드는 'docker container 하위_커맨드'가 아니라 'docker 하위_커맨드'와 같이 실행한다. 이 방법은 예전 표기법과의 호환성을 위한 것이다.

*1: 생략형은 docker ps

docker image 하위_커맨드 옵션

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
pull	도커 허브 등의 리포지토리에서 이미지를 내려받음	O	거의 사용하지 않음
rm	도커 이미지를 삭제	*2	거의 사용하지 않음
ls	내려 받은 이미지의 목록을 출력	X	거의 사용하지 않음
build	도커 이미지를 생성	O	-t

생략 가능 커맨드는 'docker image 하위_커맨드'가 아니라 'docker 하위_커맨드'와 같이 실행한다. 이 방법은 예전 표기법과의 호환성을 위한 것이다.

*2: 생략형은 docker rmi

docker volume 하위_커맨드 옵션

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
`create`	볼륨을 생성	X	`--name`
`inspect`	볼륨의 상세 정보를 출력	X	거의 사용하지 않음
`ls`	볼륨의 목록을 출력	X	`-a`
`prune`	현재 마운트되지 않은 볼륨을 모두 삭제	X	거의 사용하지 않음
`rm`	지정한 볼륨을 삭제	X	거의 사용하지 않음

docker network 하위_커맨드 옵션

주요 하위 커맨드

하위 커맨드	내용	생략 가능 여부	주요 옵션
connect	컨테이너를 도커 네트워크에 연결	X	거의 사용하지 않음
disconnect	컨테이너의 도커 네트워크 연결을 해제	X	거의 사용하지 않음
create	도커 네트워크를 생성	X	거의 사용하지 않음
inspect	도커 네트워크의 상세 정보를 출력	X	거의 사용하지 않음
ls	도커 네트워크의 목록을 출력	X	거의 사용하지 않음
prune	현재 컨테이너가 접속하지 않은 네트워크를 모두 삭제	X	거의 사용하지 않음
rm	지정한 네트워크를 삭제	X	거의 사용하지 않음

그 밖의 상위 커맨드

상위 커맨드	내용
checkpoint	현재 상태를 일시적으로 저장한 후, 나중에 해당 시점의 상태로 되돌릴 수 있다. 현재 실험적 기능이다.
node	도커 스웜의 노드를 관리하는 기능
plugin	플러그인을 관리하는 기능
secret	도커 스웜의 비밀값 정보를 관리하는 기능
service	도커 스웜의 서비스를 관리하는 기능
stack	도커 스웜 또는 쿠버네티스에서 여러 개의 서비스를 합쳐 구성한 스택을 관리하는 기능
swarm	도커 스웜을 관리하는 기능
system	도커 엔진의 정보를 확인하는 기능

단독 커맨드[9]

단독 커맨드	내용	주요 옵션
login	도커 레지스트리에 로그인	-u -p
logout	도커 레지스트리에 로그아웃	거의 사용하지 않음
search	도커 레지스트리를 검색	거의 사용하지 않음
version	도커 엔진 및 명령행 도구의 버전을 출력	거의 사용하지 않음

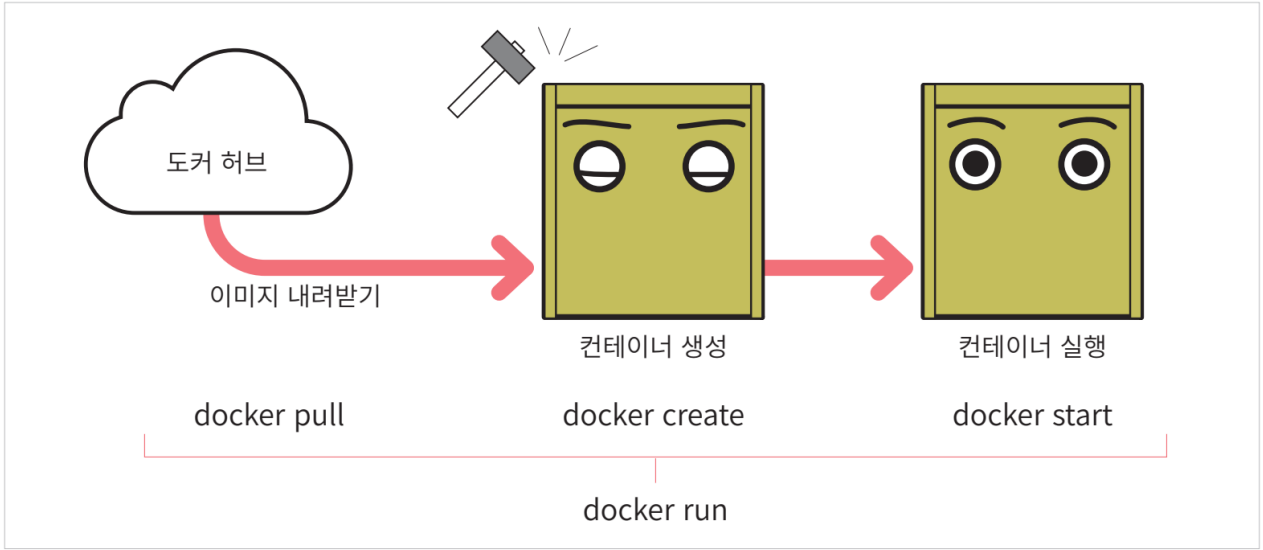


그림 4-3-1 docker run 커맨드는 세 가지 역할을 수행한다.

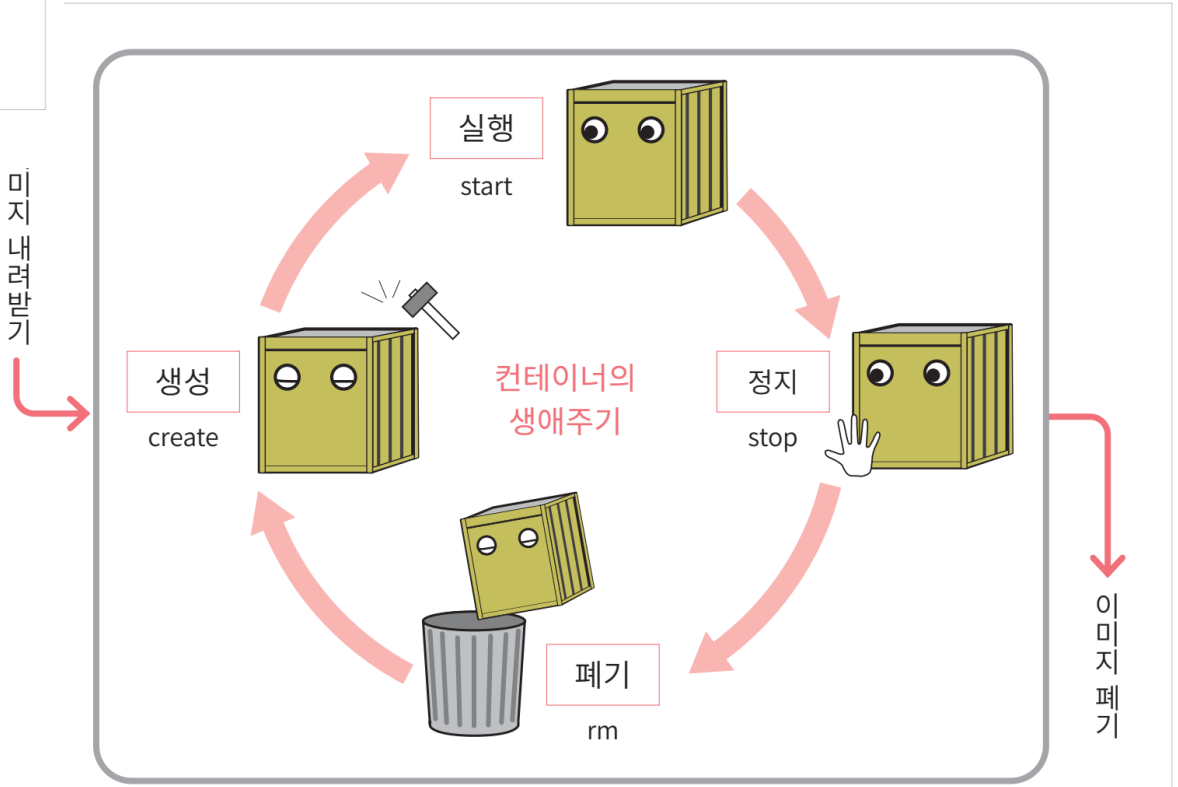


그림 4-3-2 컨테이너의 생애주기

docker run (옵션) 이미지 (인자)

```
C:\Users\mit-305>docker run -d -p 8081:80 --name myhttpd1 httpd
a955a367d7555df636c100bd6b82643630883dc7c094ccdee14d559b16354065
```

주요 옵션

옵션 형식	내용
--name 컨테이너_이름	컨테이너 이름을 지정함
-p 호스트_포트번호:컨테이너_포트번호	포트 번호를 지정함
-v 호스트_디스크:컨테이너_디렉터리	볼륨을 마운트함
--net=네트워크_이름	컨테이너를 네트워크에 연결함
-e 환경변수_이름=값	환경변수를 설정함
-d	백그라운드로 실행함
-i	컨테이너에 터미널(키보드)을 연결함
-t	특수 키를 사용 가능하도록 함
-help	사용 방법 안내 메시지를 출력함

-p는 --publish, -v는 --volume, -e는 --env, -d는 --detach, -i는 --interactive, -t는 --tty의 생략형이다.

docker stop 컨테이너_이름

docker rm 컨테이너_이름

docker ps

Docker ps -a

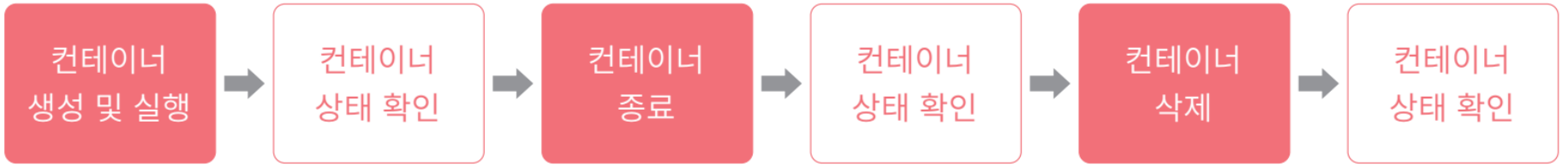
```
C:\Users\mit-305>docker stop myhttpd1
myhttpd1

C:\Users\mit-305>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

C:\Users\mit-305>docker ps -a
CONTAINER ID   IMAGE     COMMAND              CREATED         STATUS
a955a367d755   httpd     "httpd-foreground"   8 minutes ago   Exited (0) 14
```

컨테이너 목록의 주요 항목

항목	내용
CONTAINER ID	컨테이너 식별자. 무작위 문자열이 할당된다. 본래는 64글자이지만 앞에서부터 12글자만 출력한다. 이 12글자만으로도 식별자 역할을 수행할 수 있다.
IMAGE	컨테이너를 만들 때 사용한 이미지의 이름
COMMAND	컨테이너 실행 시에 실행하도록 설정된 프로그램의 이름. 크게 신경 쓰지 않아도 된다.
CREATED	컨테이너 생성 후 경과된 시간
STATUS	컨테이너의 현재 상태. 실행 중이라면 'Up', 종료된 상태라면 'Exited'가 출력된다.
PORTS	컨테이너에 할당된 포트 번호. '호스트 포트 번호 -> 컨테이너 포트 번호' 형식으로 출력된다. 포트 번호가 동일할 경우 ->의 뒷부분은 출력되지 않는다.
NAMES	컨테이너 이름



항목	값
컨테이너 이름	apa000ex1
이미지 이름	httpd

> docker run -d --name apa000ex1 httpd

옵션 항목

항목	내용
--name apa000ex1	apa000ex1이라는 이름으로 컨테이너를 생성
-d	백그라운드로 실행
httpd	아파치의 이미지 이름. 버전을 지정하지 않았으므로 가장 최신 버전(latest)이 사용된다.

```
C:\Users\mit-305>docker run -d --name apa000ex1 httpd
9cd5de59d01682494f392646c36416174f0d50468799eb2483c81e1449b8c139

C:\Users\mit-305>docker ps
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS          PORTS     NAMES
9cd5de59d016   httpd     "httpd-foreground"   54 seconds ago   Up 54 seconds   80/tcp    apa000ex1

C:\Users\mit-305>docker stop apa000ex1
apa000ex1

C:\Users\mit-305>docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES

C:\Users\mit-305>docker ps -a
CONTAINER ID   IMAGE     COMMAND              CREATED              STATUS                      PORTS     NAMES
9cd5de59d016   httpd     "httpd-foreground"   About a minute ago   Exited (0) 9 seconds ago              apa000ex1
a955a367d755   httpd     "httpd-foreground"   18 minutes ago       Exited (0) 10 minutes ago             myhttpd1

C:\Users\mit-305>docker rm apa000ex1
apa000ex1

C:\Users\mit-305>docker ps -a
CONTAINER ID   IMAGE     COMMAND              CREATED          STATUS                      PORTS     NAMES
a955a367d755   httpd     "httpd-foreground"   18 minutes ago   Exited (0) 10 minutes ago             myhttpd1
```

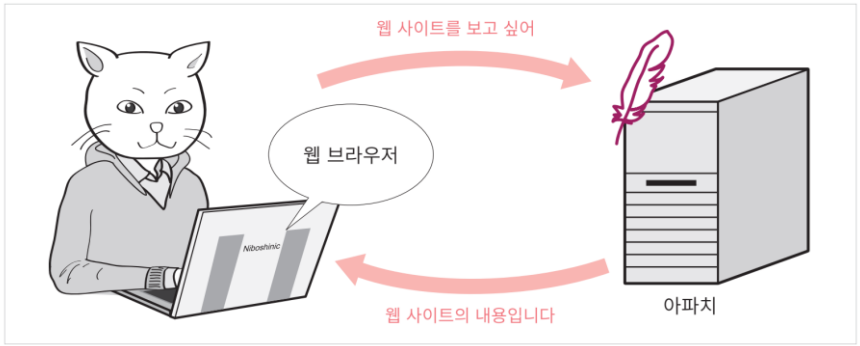


그림 4-4-1 컨테이너에서 동작하는 아파치의 웹 사이트를 열람

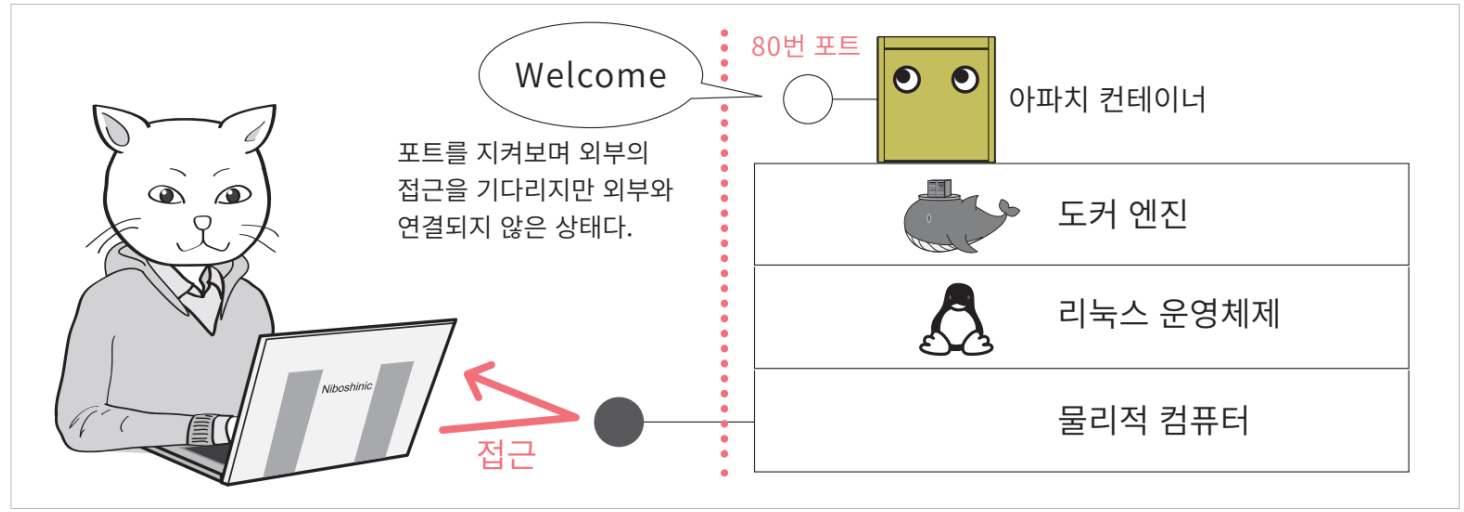


그림 4-4-2 컨테이너의 통신

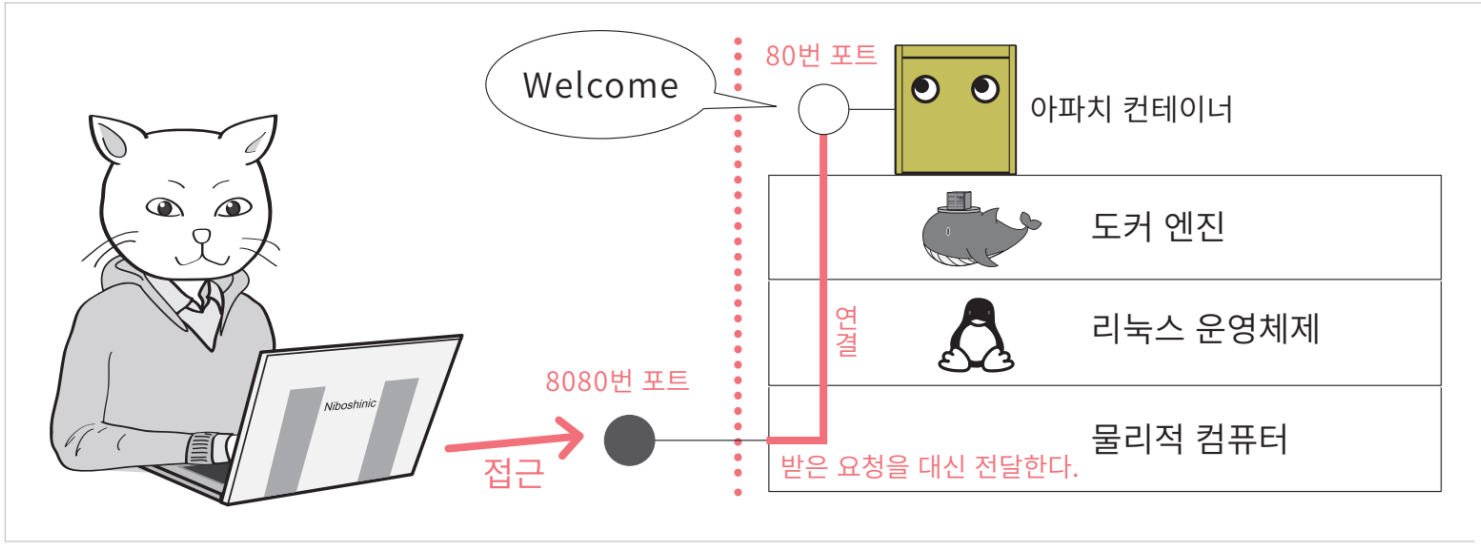


그림 4-4-3 컨테이너를 실행 중인 물리적 컴퓨터의 포트를 통해 컨테이너와 통신한다.

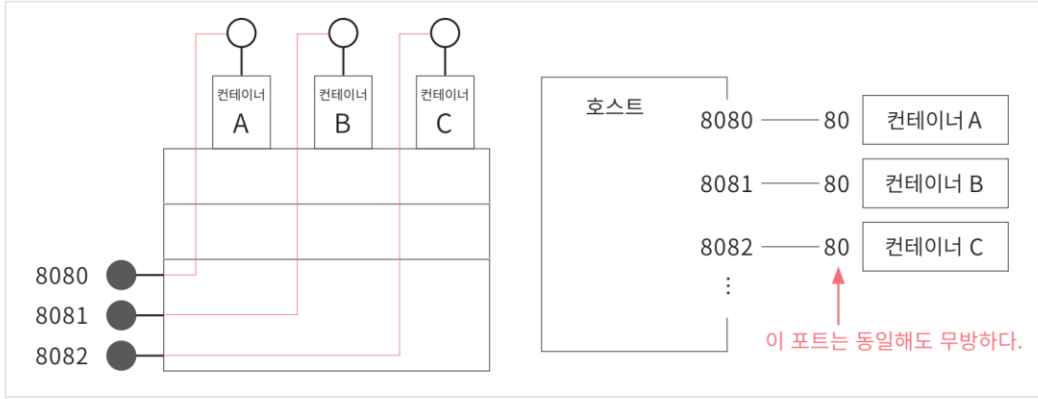
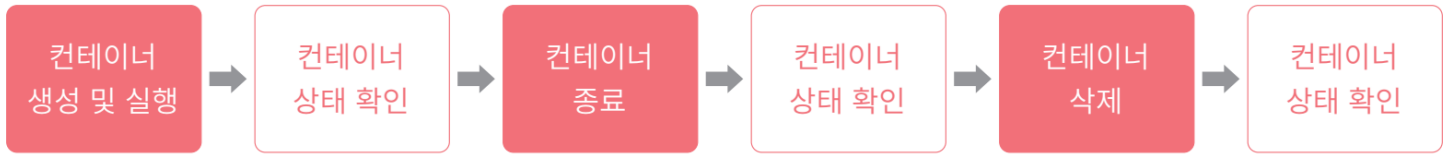
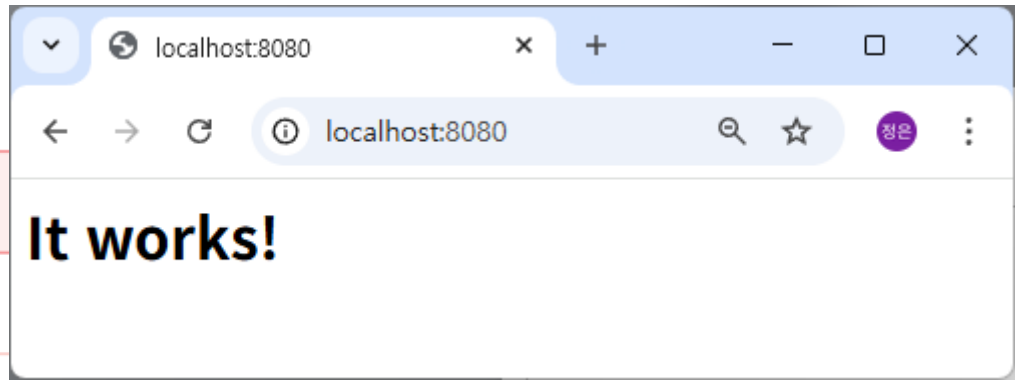


그림 4-4-4 컨테이너와 연결하는 호스트의 포트 번호를 겹치지 않게 설정한다.



항목	값
컨테이너 이름	apa000ex2
이미지 이름	httpd
포트 설정	8080:80

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



옵션 항목

항목	내용
--name apa000ex2	apa000ex2라는 이름으로 컨테이너를 생성
-d	백그라운드로 실행
-p 8080:80	호스트의 포트 8080을 컨테이너 포트 80으로 포워딩
httpd	아파치의 이미지 이름. 버전을 지정하지 않았으므로 최신 버전(latest)이 사용된다.

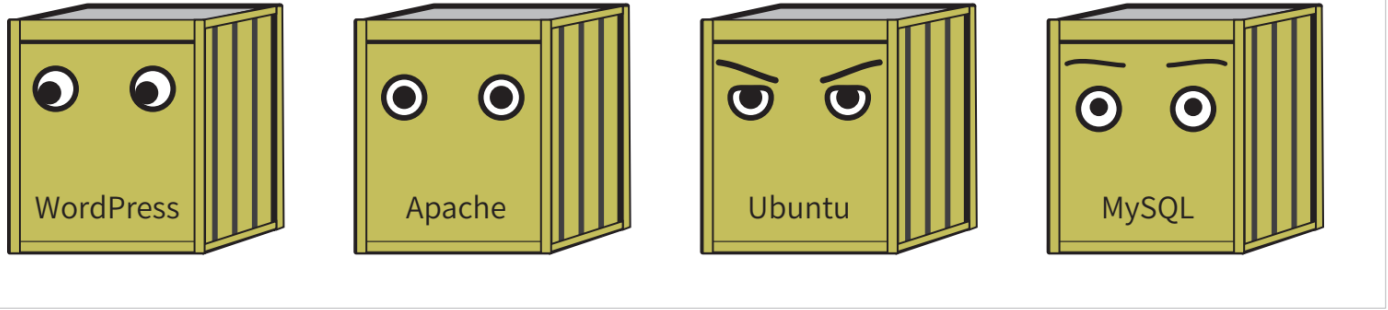


그림 4-5-1 다양한 유형의 컨테이너

리눅스 운영체제가 담긴 컨테이너의 종류

이미지 이름	컨테이너의 내용	컨테이너 실행에 주로 사용되는 옵션 및 인자
ubuntu	우분투	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.
centos	CentOS	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.
debian	데비안	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.
fedora	페도라	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.
busybox	BizyBox	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.
alpine	알파인 리눅스	-d 없이 -it 옵션만 사용. 인자로는 /bin/bash 등 셸 명령어를 지정한다.

웹 서버 및 데이터베이스 서버용 컨테이너의 종류

이미지 이름	컨테이너의 내용	컨테이너 실행에 주로 사용되는 옵션 및 인자
httpd	Apache	-d로 백그라운드로 실행. -p로 포트 번호 지정
nginx	Nginx	-d로 백그라운드로 실행. -p로 포트 번호 지정
mysql	MySQL	-d를 사용. 실행 시 -e MYSQL_ROOT_PASSWORD와 같이 루트 패스워드를 지정
postgres	PostgreSQL	-d를 사용. 실행 시 -e POSTGRES_ROOT_PASSWORD와 같이 루트 패스워드를 지정
mariadb	MariaDB	-d를 사용. 실행 시 -e MYSQL_ROOT_PASSWORD와 같이 루트 패스워드를 지정

프로그래밍 언어의 런타임 컨테이너의 종류

이미지 이름	컨테이너의 내용	컨테이너 실행에 주로 사용되는 옵션 및 인자
openjdk	자바 런타임	-d를 사용하지 않고, 인자로 java 명령 등을 지정해 도구 형태로 사용한다.
python	파이썬 런타임	-d를 사용하지 않고, 인자로 python 명령 등을 지정해 도구 형태로 사용한다.
php	PHP 런타임	웹 서버가 포함된 것과 실행 명령만 포함된 것으로 나뉘어 제공된다.
ruby	루비 런타임	웹 서버가 포함된 것과 실행 명령만 포함된 것으로 나뉘어 제공된다.
perl	펄 런타임	-d를 사용하지 않고, 인자로 perl 명령 등을 지정해 도구 형태로 사용한다.
gcc	C/C++ 컴파일러	-d를 사용하지 않고, 인자로 gcc 명령 등을 지정해 도구 형태로 사용한다.
node	Node.js	-d를 사용하지 않고, 인자로 app 명령 등을 지정해 도구 형태로 사용한다.
registry	도커 레지스트리	-d 옵션을 사용해 백그라운드로 실행하며, -p 옵션으로 포트 번호를 지정한다.
wordpress	WordPress	-d 옵션을 사용해 백그라운드로 실행하며, -p 옵션으로 포트 번호를 지정한다. MySQL 또는 MariaDB가 필요하다. 접속에 필요한 패스워드는 -e 옵션으로 지정한다.
nextcloud	NextCloud	-d 옵션을 사용해 백그라운드로 실행한다. -p 옵션으로 포트 번호를 지정한다.
redmine	Redmine	-d 옵션을 사용해 백그라운드로 실행하며, -p 옵션으로 포트 번호를 지정한다. PostgreSQL 또는 MySQL이 필요하다.

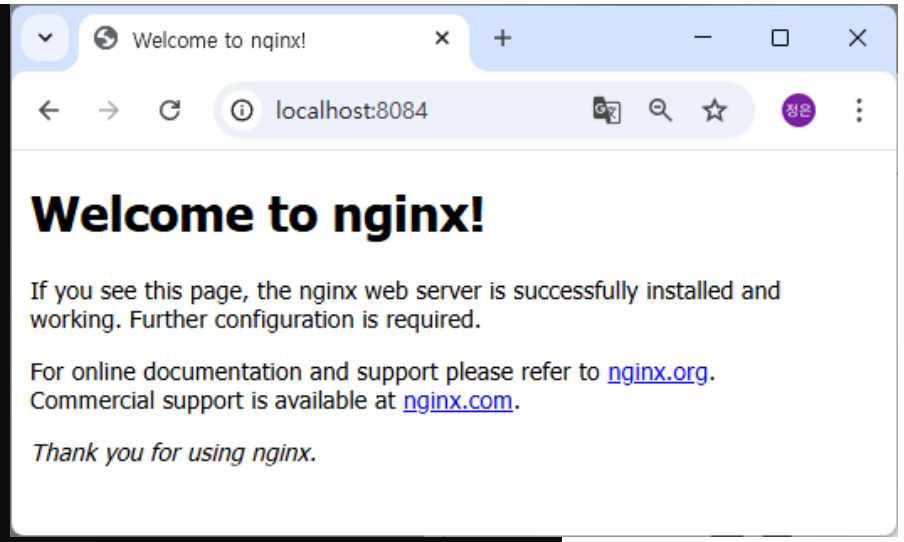
컨테이너 생성 및 실행 → 컨테이너 상태 확인 → 웹 브라우저를 통한 접근 확인 → 컨테이너 종료 → 컨테이너 삭제 → 컨테이너 상태 확인

> docker run -d -p 8084:80 --name nginx000ex6 nginx

항목	값	값	값
컨테이너 이름	apa000ex3	apa000ex4	apa000ex5
이미지 이름	httpd	httpd	httpd
포트 설정	8081:80	8082:80	8083:80

```
C:\Users\mit-305>docker run -d -p 8084:80 --name nginx000ex6 nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
8a628cdd7ccc: Already exists
b0c073cda91f: Pull complete
e6557c42ebea: Pull complete
ec74683520b9: Pull complete
6c95adab80c5: Pull complete
ad8a0171f43e: Pull complete
32ef64864ec3: Pull complete
Digest: sha256:5ed8fcc66f4ed123c1b2560ed708dc148755b6e4cbd8b943fab094f2c6bfa91e
Status: Downloaded newer image for nginx:latest
adcd62c762c6b70b995ae77431b1ce651b85600fd072368c22b5eaf01694bb27

C:\Users\mit-305>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                    NAMES
adcd62c762c6   nginx     "/docker-entrypoint.…"   7 seconds ago   Up 6 seconds   0.0.0.0:8084->80/tcp     nginx000ex6
```



항목	값
컨테이너 이름	mysql000ex7
이미지 이름	mysql
MySQL 루트 패스워드	myrootpass

>docker run -d --name mysql000ex7 -dit -e MYSQL_ROOT_PASSWORD=1234 mysql

항목	내용
--name mysql000ex7	mysql000ex7이라는 이름으로 컨테이너를 생성
-dit	백그라운드에서 실행 및 키보드를 통해 컨테이너 내부의 파일 시스템을 조작
-e MYSQL_ROOT_PASSWORD=	MySQL의 루트 패스워드를 지정
mysql	MySQL 이미지 이름. 버전을 지정하지 않았으므로 최신 버전(latest)이 사용된다.

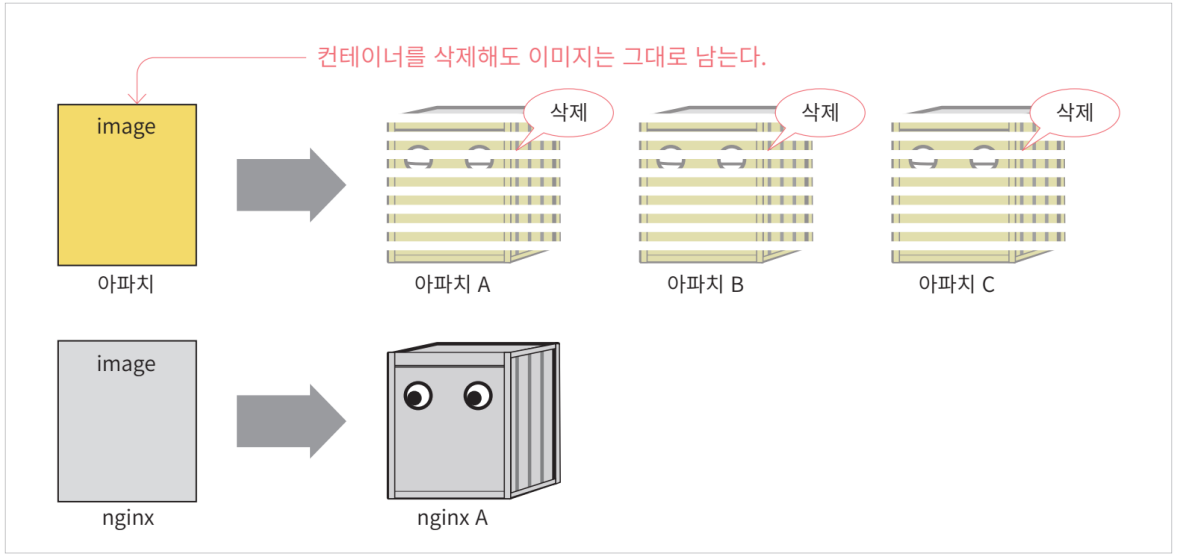


그림 4-6-1 컨테이너의 삭제와 이미지의 삭제

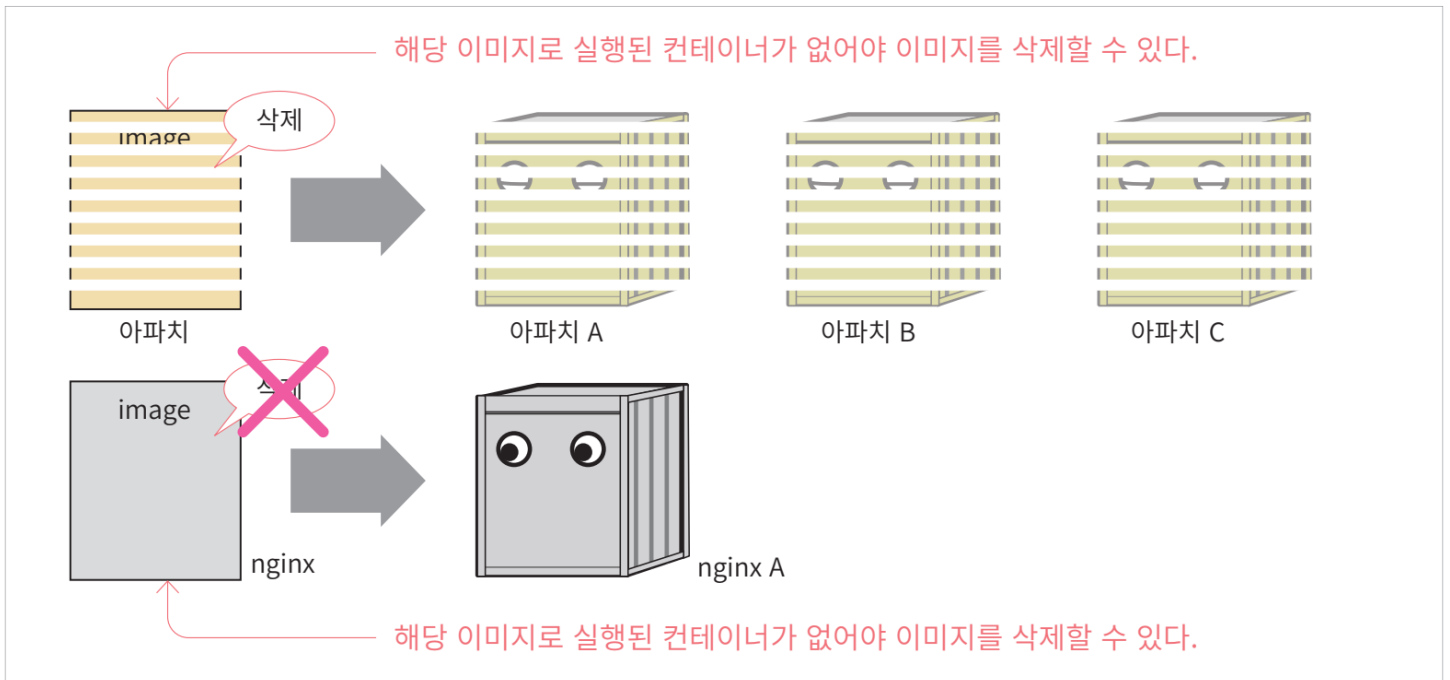


그림 4-6-2 해당 이미지로 실행된 컨테이너가 있으면 이미지를 삭제할 수 없다.

이미지 목록의 주요 항목

항목	내용
REPOSITORY	이미지 이름
TAG	버전 정보. 이미지를 내려받을 때 따로 지정하지 않으면 latest(최신 버전)를 내려받는다.
IMAGE ID	이미지 식별자. 본래는 64글자이지만 앞에서부터 12글자만 출력한다. 이 12글자만으로도 식별자 역할을 수행할 수 있다.
CREATED	이미지 생성 후 경과된 시간
SIZE	이미지의 전체 용량