- 도커 컴포즈는 여러 개의 Docker 컨테이너를 관리하는 도구로서 Docker 데스크탑 설치시 기본으로 설치된다.
- 한 번의 명령어로 여러 개의 컨테이너를 한번에 실행하거나 종료할 수 있다.
- 단일 서버에서 여러 개의 컨테이너를 하나의 서비스로 정의해 컨테이너의 묶음으로 관리할 수 있는 작업 환경을 제공하는 관리 도구이다.

- 시스템 구축과 관련된 명령어를 하나의 텍스트 파일(정의 파일, Compose File)에 기자 한 번의 명령으로 시스템 전체를 실행하거나 종료한다.

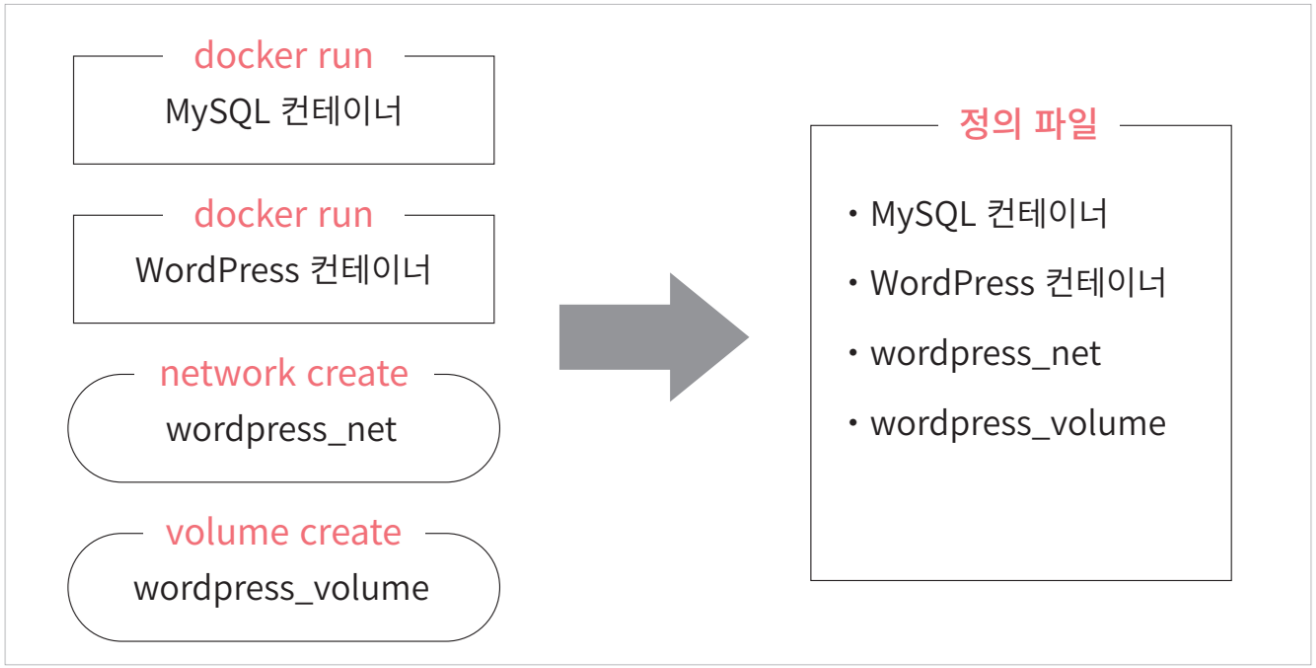


그림 7-1-1 도커 컴포즈를 사용하면 여러 개의 명령어를 하나의 정의 파일로 합쳐 실행할 수 있다.

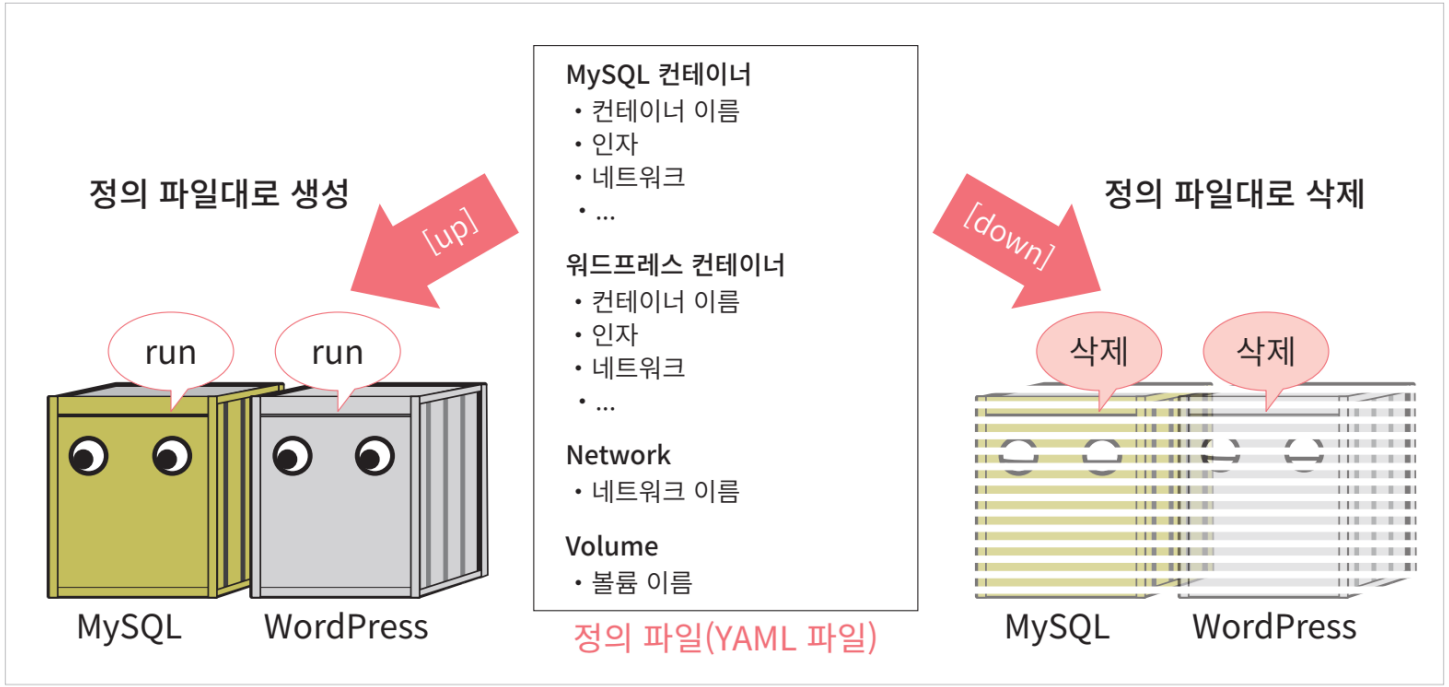


그림 7-1-2 도커 컴포즈는 시스템의 모든 정보를 정의 파일에 기재한다.

- YAML(YAML Ain't a Markup Language)포맷의 정의 파일

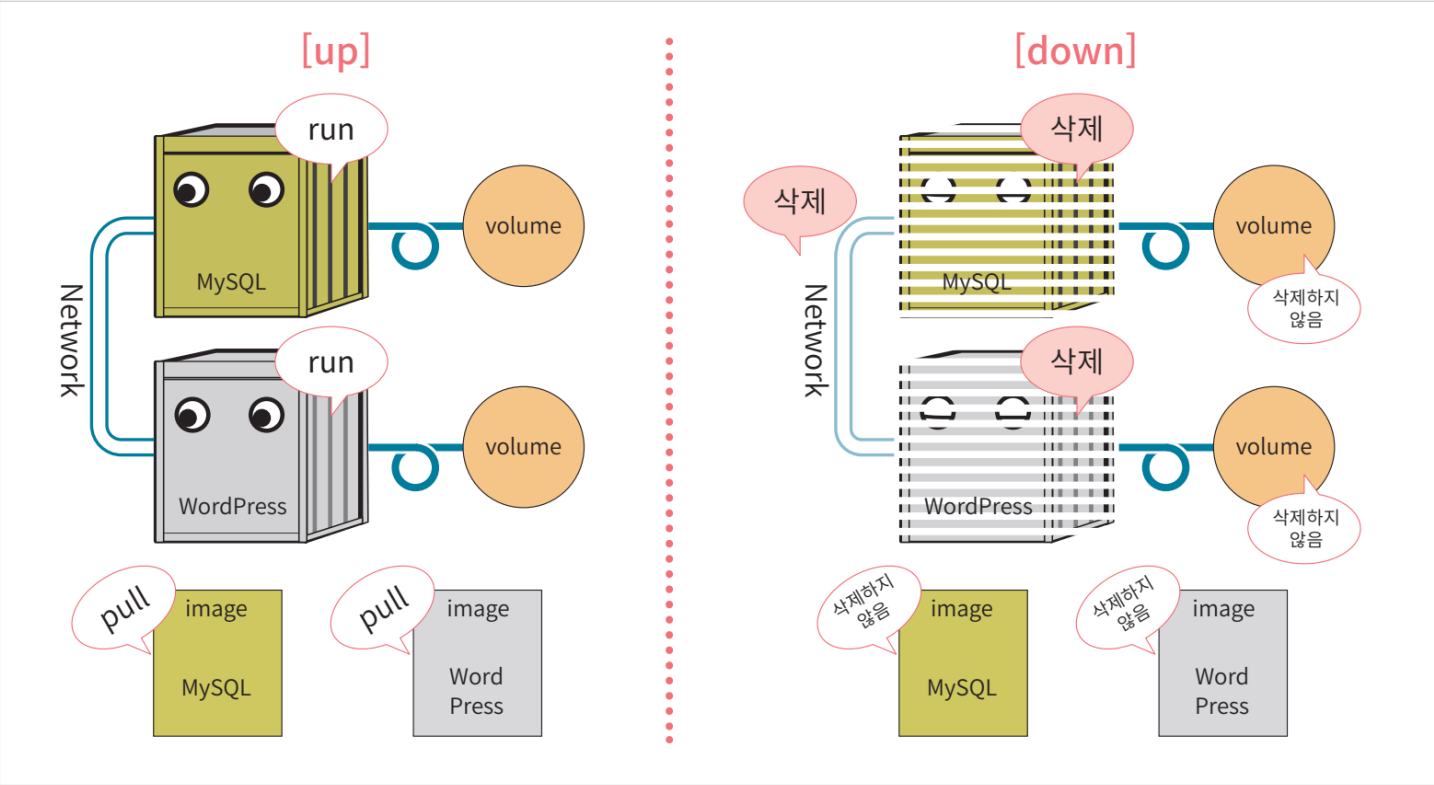


그림 7-1-3 up 커맨드로 생성 및 시작, down 커맨드로 정지 및 삭제한다.

- up 명령 : 정의 파일(docker-compose.yml)의 내용대로 이미지를 내려 받고 컨테이너를 생성 및 실행, 네트워크나 볼륨도 정의 가능
- down 명령 : 컨테이너와 네트워크를 정지 및 삭제
- stop 명령 : 컨테이너와 네트워크를 삭제없이 종료만 진행

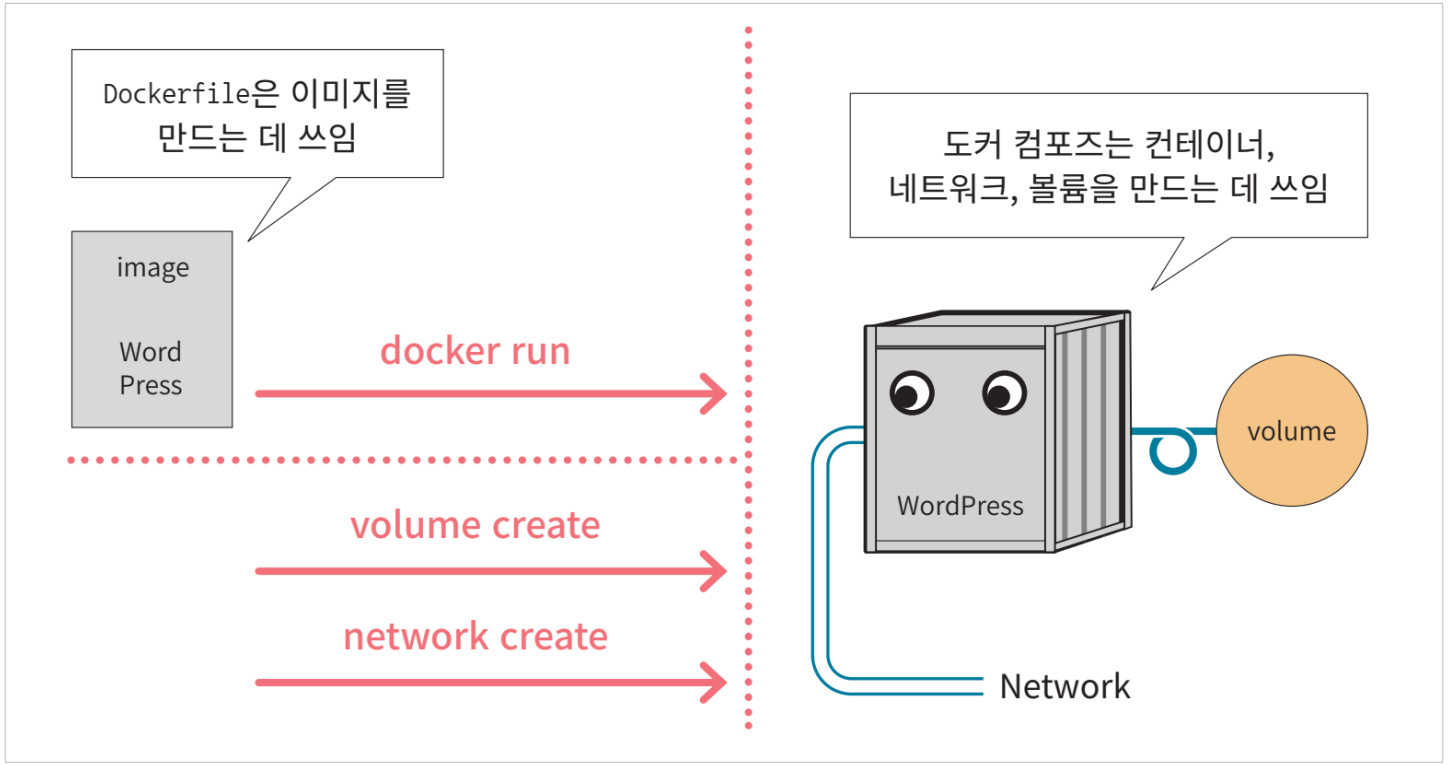


그림 7-1-4 Dockerfile 스크립트와 도커 컴포즈의 차이점

- 도커 컴포즈와 Dockerfile 스크립트 차이점
  - 도커 컴포즈 : 컨테이너와 주변 환경 및 네트워크와 볼륨까지 만들 수 있다.
  - Dockerfile 스크립트 : 이미지를 만들기 위한 것으로 네트워크나 볼륨은 만들 수 없다

- 도커 컴포즈 설치

파이썬으로 작성된 프로그램이다.

•도커 데스크톱에는 도커 컴포즈 함께 설치되어 있음

•Spring Boot 애플리케이션을 도커 컴포즈를 이용하여 리눅스 기반의 AWS의 EC2에 배포하려면 직접 도커 컴포즈는 설치해야 한다.

```
sudo apt install -y python3 python3-pip
sudo pip3 install docker-compose
```

- 확인

Docker-compose --version

```
C:\Users\mit-305>Docker-compose --version
Docker Compose version v2.34.0-desktop.1
```

docker compose up -d

YAML 파일에 정의된 서비스 생성 및 시작

docker compose ps

현재 실행중인 서비스 상태 표시

docker compose build

현재 실행중인 서비스의 이미지만 빌드

docker compose logs

실행 중인 서비스의 로그 표시

docker compose down

YAML 파일에 정의된 서비스 종료 및 제거

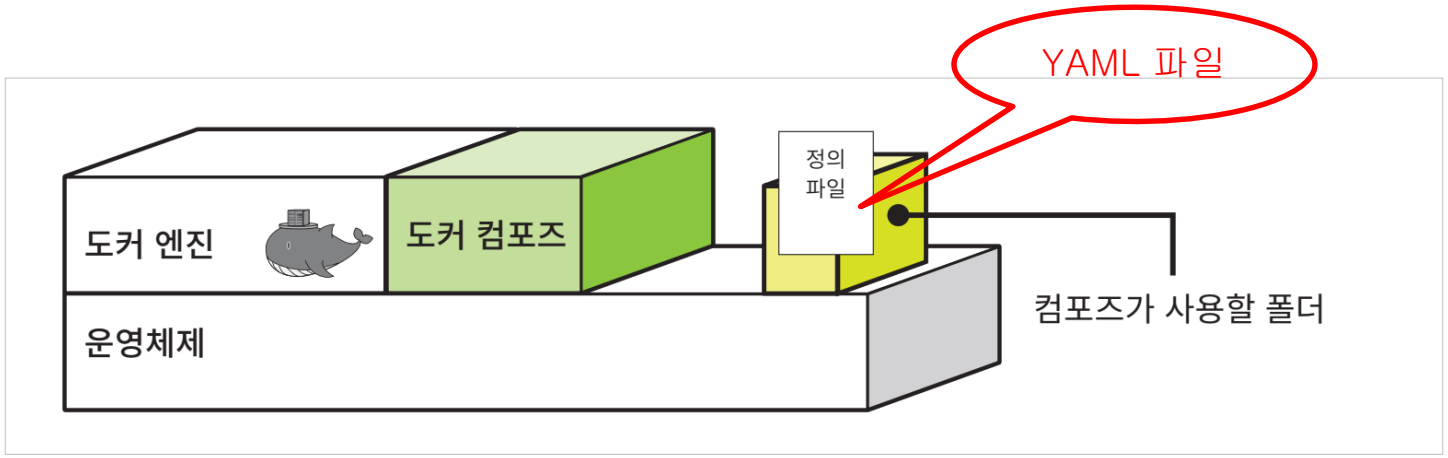


그림 7-2-1 호스트 컴퓨터에 정의 파일을 준비한다.

- 정의 파일명 : docker-compose.yml
- 도커 엔진에 의해 수행

  사람이 입력하는 수동 명령을 컴포즈가 대신 전달하는 개념

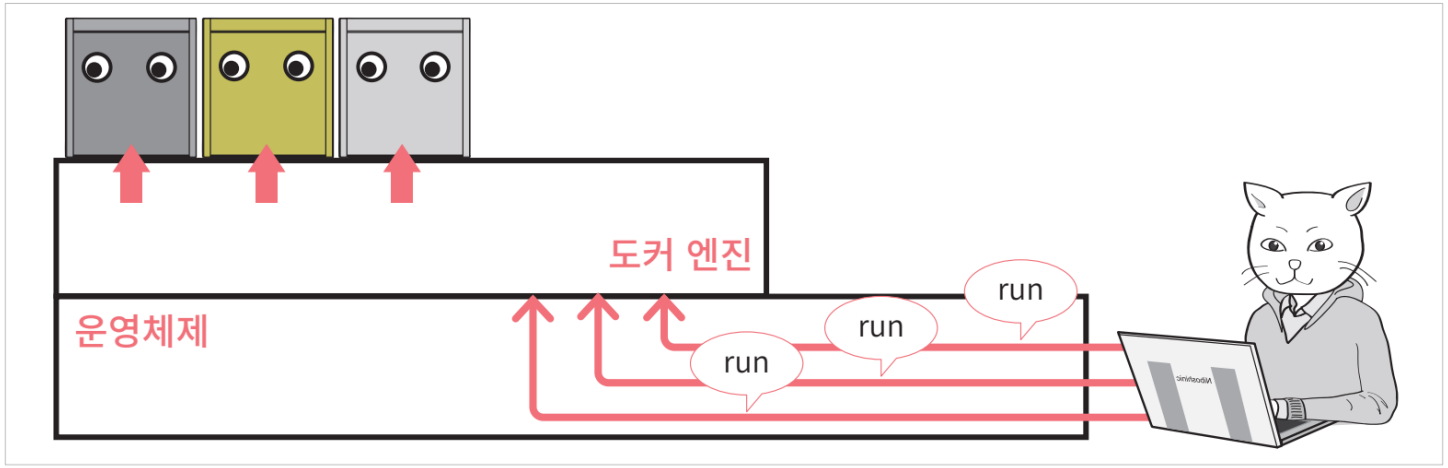


그림 7-2-2 도커 컴포즈가 명령어를 대신 입력해주는 구조

- 서비스와 컨테이너

  도커 컴포즈에서 서비스란 컨테이너가 모인 것을 의미

  일반적으로 컨테이너와 서비스는 동일한 개념으로 봐도 됨

- 컴포즈 파일 작성 방법

  - 주 항목 -> 이름 추가 -> 설정의 순으로 진행

```
services:     # 컨테이너 관련 정보
networks :    # 네트워크 관련 정보
volumes:      # 볼륨 관련 정보
```

- 첫 번째 줄에 도커 컴포즈 버전 작성
- 주 항목 services, networks, volumes 아래에 설정 내용 작성
- 이름 추가 시 주 항목보다 한 단 들여쓰기
- 이름 뒤에는 반드시 ':' 사용

항목	내용
정의 파일의 형식	YAML 형식
파일 이름	docker-compose.yml

- : 문자 뒤에 한 줄에 이어서 설정이 필요한 경우 : 뒤에 공백문자가 1개가 반드시 필요
- 이름 뒤에 여러 개의 설정이 필요한 경우 줄을 바꿔 들여쓰기와 함께 '하이픈(–)+공백문자'와 함께 설정 내역 작성
- 항목 간의 상하 관계는 공백을 사용한 들여쓰기로 정함

- 들여쓰기는 같은 수의 배수만큼의 공백을 사용
- # 은 주석 처리
- 문자열은 ' ' 또는 " "를 이용
- 탭 문자는 사용 못함

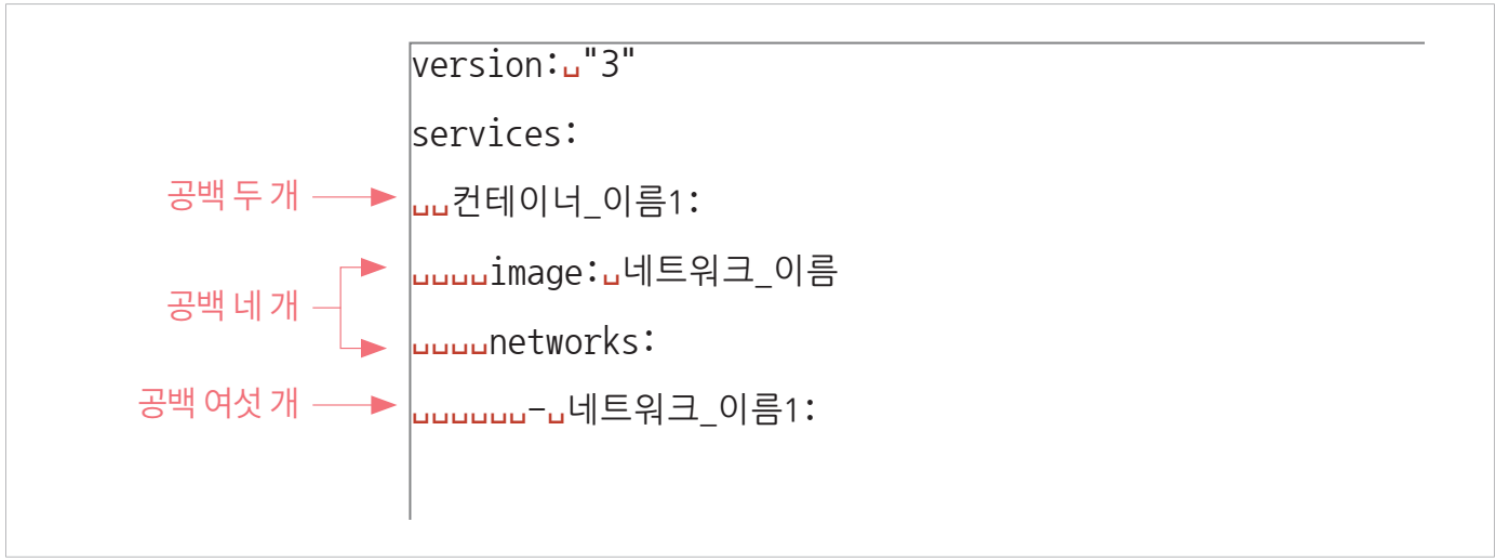


그림 7-3-1 들여쓰기를 일정하게 맞춘다.

- docker-compose.yml 템플릿

```
version: "1"
services:
  컨테이너_이름1:
    image: 이미지_이름
    networks:
      - 네트워크_이름
    ports:
      - 포트_설정
  컨테이너_이름2:
    image: 이미지_이름
  ...
networks:
  네트워크_이름1:
  ...
volumes:
  볼륨_이름1:
  볼륨_이름2:
  ...
```

주 항목

항목	내용
services	컨테이너를 정의한다.
networks	네트워크를 정의한다.
volumes	볼륨을 정의한다.

자주 나오는 정의 내용

항목	docker run 커맨드의 해당 옵션 또는 인자	내용
image	이미지 인자	사용할 이미지를 지정
networks	--net	접속할 네트워크를 지정
volumes	-v, --mount	스토리지 마운트를 설정
ports	-p	포트 설정
environment	-e	환경변수 설정
depends_on	없음	다른 서비스에 대한 의존관계를 정의
restart	없음	컨테이너 종료 시 재시작 여부를 설정

restart의 설정값

설정값	내용
no	재시작하지 않는다.
always	항상 재시작한다.
on-failure	프로세스가 0 외의 상태로 종료됐다면 재시작한다.
unless-stopped	종료 시 재시작하지 않음. 그 외에는 재시작한다.

depends_on : 컨테이너 생성 순서나 연동 여부를 정의한다.

예) depends_on:
  - mysqledu

현재 컨테이너를 mysqledu 컨테이너를 생성한 다음에 생성하라는 의미이다.

- 도커 컴포즈 정의 파일명 : docker-compose.yml

  다른 파일명이거나 다른 디렉토리에 존재하는 경우에는 -f 와 함께 지정 가능

  docker-compose -f 정의파일이름 up 옵션

사용 예)
docker-compose -f docker-compose2.yml up -d

현재 작업 디렉터리에 존재하는 docker-compose.yml 을 컴포즈 정의 파일로 사용할 때에는 따로 콤포즈 파일을 지정하지 않아도 됨

  docker-compose up -d

옵션 항목

옵션	내용
-d	백그라운드로 실행
--no-color	화면 출력 내용을 흑백으로 함
--no-deps	링크된 서비스를 실행하지 않음
--force-recreate	설정 또는 이미지가 변경되지 않더라도 컨테이너를 재생성
--no-create	컨테이너가 이미 존재할 경우 다시 생성하지 않음
--no-build	이미지가 없어도 이미지를 빌드하지 않음
--build	컨테이너를 실행하기 전에 이미지를 빌드
--abort-on-container-exit	컨테이너가 하나라도 종료되면 모든 컨테이너를 종료
-t, --timeout	컨테이너를 종료할 때의 타임아웃 설정. 기본은 10초.
--remove-orphans	컴포즈 파일에 정의되지 않은 서비스의 컨테이너를 삭제
--scale	컨테이너의 수를 변경

- 컨테이너와 네트워크 삭제 명령

  docker-compose -f 정의_파일_이름 down 옵션

옵션 항목

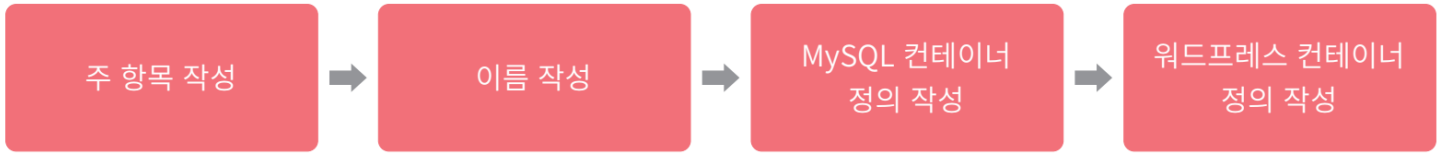
옵션	내용
--rmi 종류	삭제 시에 이미지도 삭제한다. 종류를 all로 지정하면 사용했던 모든 이미지가 삭제된다. local로 지정하면 커스텀 태그가 없는 이미지만 삭제한다.
-v, --volumes	volumes 항목에 기재된 볼륨을 삭제한다. 단, external로 지정된 볼륨은 삭제되지 않는다.
--remove-orphans	컴포즈 파일에 정의되지 않은 서비스의 컨테이너도 삭제한다.

- 컨테이너만 종료하는 명령

  docker-compose -f 정의_파일_이름 stop

- docker-compose 주요 명령 목록

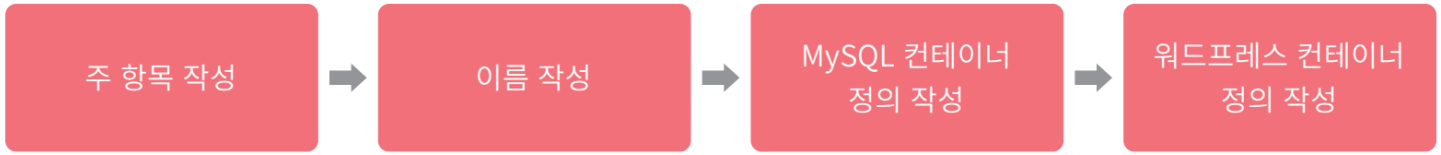
커맨드	내용
up	컨테이너를 생성하고 실행한다.
down	컨테이너와 네트워크를 종료하고 삭제한다.
ps	컨테이너 목록을 출력한다.
config	컴포즈 파일을 확인하고 내용을 출력한다.
port	포트 설정 내용을 출력한다.
logs	컨테이너가 출력한 내용을 화면에 출력한다.
start	컨테이너를 시작한다.
stop	컨테이너를 종료한다.
kill	컨테이너를 강제 종료한다.
exec	명령어를 실행한다.
run	컨테이너를 실행한다.
create	컨테이너를 생성한다.
restart	컨테이너를 재실행한다.
pause	컨테이너를 일시 정지한다.
unpause	컨테이너의 일시 정지를 해제한다.
rm	종료된 컨테이너를 삭제한다.
build	컨테이너에 사용되는 이미지를 빌드 혹은 재빌드한다.
pull	컨테이너에 사용되는 이미지를 내려받는다.
scale	컨테이너의 수를 지정한다.
events	컨테이너로부터 실시간으로 이벤트를 수신한다.
help	도움말 화면을 출력한다.



항목	값
네트워크 이름	wordpress000net1
MySQL 볼륨 이름	mysql000vol11
워드프레스 볼륨 이름	wordpress000vol12
MySQL 컨테이너 이름	mysql000ex11
워드프레스 컨테이너 이름	wordpress000ex12

MySQL 컨테이너(mysql000ex11)의 정의

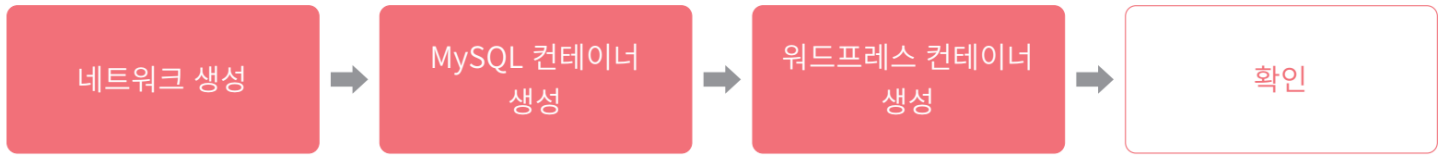
항목	항목 이름	값
MySQL 이미지 이름	image:	mysql:5.7
사용할 네트워크	networks:	wordpress000net1
사용할 볼륨	volumes:	mysql000vol11
마운트 위치		/var/lib/mysql
재시작 설정	restart:	always
MySQL 설정	environment:	★이 붙은 항목 설정
★MySQL 루트 패스워드	MYSQL_ROOT_PASSWORD	myrootpass
★MySQL 데이터베이스 이름	MYSQL_DATABASE	wordpress000db
★MySQL 사용자 이름	MYSQL_USER	wordpress000kun
★MySQL 패스워드	MYSQL_PASSWORD	wkunpass



워드프레스 컨테이너(wordpress000ex12)의 정의

항목	항목 이름	값
의존관계	depends_on:	mysql000ex11
워드프레스 이미지 이름	image:	wordpress
사용할 네트워크	networks:	wordpress000net1
사용할 볼륨	volumes:	wordpress000vol12
마운트 위치		/var/www/html
포트 번호 설정	port:	8085:80
재시작 설정	restart:	always
데이터베이스 관련 정보	environment:	★이 붙은 항목 설정
★데이터베이스 컨테이너 이름	WORDPRESS_DB_HOST	mysql000ex11
★데이터베이스 이름	WORDPRESS_DB_NAME	wordpress000db
★데이터베이스 사용자 이름	WORDPRESS_DB_USER	wordpress000kun
★데이터베이스 패스워드	WORDPRESS_DB_PASSWORD	wkunpass

항목	값
컴포즈 파일 배치 경로(윈도우)	C:\Users\사용자명\Documents\com_folder
컴포즈 파일 배치 경로(macOS)	/Users/사용자명/Documents/com_folder
컴포즈 파일 배치 경로(리눅스)	/home/사용자명/com_folder



항목	값
윈도우	C:\Users\사용자명\Documents\com_folder\docker-compose.yml
macOS	/Users/사용자명/Documents/com_folder/docker-compose.yml
리눅스	/home/사용자명/com_folder/docker-compose.yml

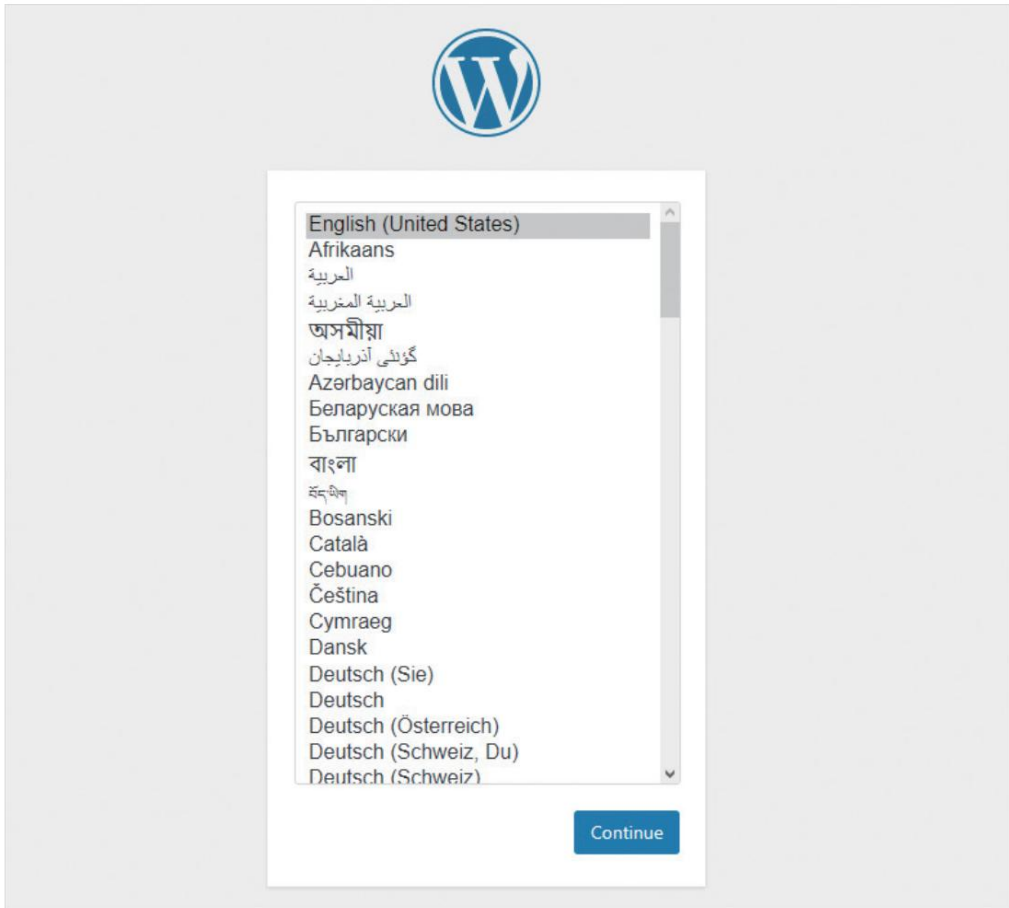


그림 7-4-2 워드프레스의 초기 화면