

DOCKER KOMUTLARI

- Komutlar nginx imaj alınarak yazılmıştır.

Komut: `docker run --name <name> --rm -p <port> nginx sleep <time>`

```
C:\Users\asus>docker run --name odev --rm -p 8080:80 nginx sleep 30
```

	odev 74377e0c7820 	nginx	Running	8080:80 
---	--	-----------------------	---------	---

- `docker run [OPTIONS] image [COMMANDS]` formatında yazılır. Nginx burada konteynerımızın imajıdır.
- **--name:** konteynerın ismini belirlememizi sağlar.
- **--rm:** iş sona erdiğinde konteyner kendini siler.
- **-p:** belirtilen portta çalışmasını sağlar.
- **sleep:** belirtilen süre boyunca işlemi duraklatmak için kullanılır.

Komut: `docker run -it --name <name> nginx ls`

```
C:\Users\asus>docker run -it --name odev nginx ls
bin    dev          docker-entrypoint.sh  home  lib64  mnt  proc  run  srv  tmp  var
boot  docker-entrypoint.d  etc          lib   media  opt  root  sbin  sys  usr
```

- **-it:** konteynerın terminaline bağlanmamızı sağlar.
- `ls` komutu yerine herhangi bir terminal komutu yazılabilir.

Komut: `docker run -d --name <name> nginx`

```
C:\Users\asus>docker run -d --name odev nginx
f5c268ebf86cd790719b7578455e700644a971d27c47b2973da56209b06002e4
```

- **-d**: arka planda konteyneri çalıştırmaya devam ederken yeni komut satırına geçer. Bu sayede terminalde başka işlemler gerçekleştirebiliriz.

Komut: `docker ps`

```
C:\Users\asus>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS      NAMES
f5c268ebf86c   nginx    "/docker-entrypoint..." About a minute ago Up About a minute 80/tcp     odev
```

- **ps**: bu komut sayesinde ise arka planda çalışan konteynerları görüntüleyebiliriz.

Komut: `docker ps -a`

```
C:\Users\asus>docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS      NAMES
9dff05067ec5   nginx    "/docker-entrypoint..." About a minute ago Exited (0) About a minute ago odev
1edb0cf49f52   diwa:latest "docker-php-entrypoi..." 2 months ago   Exited (0) 2 months ago laughing_kapitsa
474619b09fea   yavuzlar/vulnlab "/usr/sbin/run.sh"      2 months ago   Exited (137) 9 hours ago infallible_hopper
```

- **-a**: komut sayesinde çalışan ya da çalışmayan tüm konteynerlar görüntülenebilir.

Komut: `docker start <containerName/id>`

```
C:\Users\asus>docker start odev
odev
```



[odev](#)

9dff05067ec5



[nginx](#)

Running

- **start**: istediğimiz konteynerın çalıştırılmasını sağlar.

Komut: `docker stop <containerName/id>`

```
C:\Users\asus>docker stop odev
odev
```

	odev 9dff05067ec5 	nginx	Exited
---	--	-----------------------	--------

- **stop**: istediğimiz konteynerin durdurulmasını sağlar.

Komut: `docker exec <containerName/id> [COMMANDS]`

```
C:\Users\asus>docker exec odev ls
bin
boot
dev
docker-entrypoint.d
docker-entrypoint.sh
etc
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
```

- **exec**: var olan bir konteynerin çalıştırılması sağlanır.
- Terminal komutlarını kullanabiliriz.

DOCKER-COMPOSE

```
version: '3'

services:
  app:
    build:
      context: .
      dockerfile: Dockerfile
    depends_on:
      - db
    ports:
      - "80:80"
    networks:
      - net

  db:
    image: mysql:latest
    environment:
      - MYSQL_DATABASE=yavuzlar
      - MYSQL_ROOT_PASSWORD=1
    volumes:
      - db_data:/var/lib/mysql
      - ./yavuzlar_messages.sql:/docker-entrypoint-initdb.d/yavuzlar_messages.sql
    ports:
      - "8080:3306"
    networks:
      - net

networks:
  net:
    driver: bridge

volumes:
  db_data:
```

version: docker compose versiyonu belirtir. (3)

services: tanımlanan servisin başladığı bölümü belirtir. Örneğe göre burada app ve backend servisleri tanımlanmıştır.

app: app servisinin başladığını belirtir.

build: gerekli yapılandırmayı belirtir.

context: dosyanın bulunduğu dizini gösterir. (.)

dockerfile: dockerfile'ın ismini belirtir. (Dockerfile)

ports: konteynerın hangi portları kullanacağını belirtir. (80:80 app için, 8080:3306 db için.)

depends_on: bir servisin başlaması için gerekli olan diğer servisleri gösterir. (db)

networks: hangi docker ağında çalışacağını gösterir. (net)

db: db servisinin yapılandırılması.

image: kullanılacak olan docker image'ını belirtir. (mysql:latest)

environment: konteynerın ortam değişkenlerini ayarlar. [database(yavuzlar)/şifre(1)]

volumes: host makinadaki dizinlerin konteynera bağlanmasını sağlar.

DOCKERFILE

```
FROM php:7.4-apache

WORKDIR /var/www/html
COPY ./app .
RUN echo "ServerName localhost" >> /etc/apache2/apache2.conf
RUN apt-get update
RUN docker-php-ext-install pdo pdo_mysql
EXPOSE 80
```

FROM image:tag: php 7.4 sürümü ve apache web sunucusunu içeren bir docker imajı kullanılmaktadır.

WORKDIR: çalışma dizinini belirtir. (/var/www/html)

COPY: ana makinanın belirtilen path'inden dosya kopyalamamızı sağlar. (./app)

RUN: bu komut imaj üzerinde komut çalıştırmak için kullanılır.

- RUN echo "ServerName localhost" >> /etc/apache2/apache2.conf: belirtilen dizine stringi ekler.
- RUN apt-get update: paket listesini günceller.
- RUN docker-php-ext-install pdo pdo_mysql: pdo ve pdo_mysql uzantılarının indirilmesini sağlar.

EXPOSE <port>: konteynerın hangi portlarının açılacağını belirtir. (80)

GIT