Docker

Bevezető

Rostagni Csaba

2024. szeptember 4.

Tartalom

Tartalom

Előkészületek

Vegyük az alábbi mappaszerkezetet és HTML fájlt

```
/
└── www
    └── index.html
```

index.html

```html
<!doctype html>
<html lang="hu">
<head>
    <meta charset="utf-8">
    <title>Hello World</title>
</head>
<body>
    <h1>Hello World</h1>
</body>
</html>
```

nginx webszerver indítása

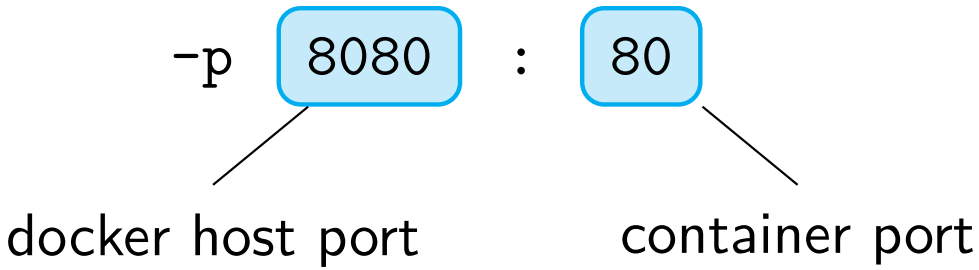
A projekt gyökérkönyvtárában állva kell kiadni az alábbi parancsot

Terminal

```
docker run \
        -d \
        --rm \
        --name hello-web \
        -p 8888:80 \
        -v "$(pwd)/www:/usr/share/nginx/html" \
        nginx
```

- `-d` – A háttérben fut, visszakapjuk a konzolt
- `--rm` – Miután befejezte a feladatát, törölje a webszervert
- `--name` – A létrejövő container neve legyen `hello-web`

publish



- A `-p` vagy `--publish` a konténeren kívülről is elérhetővé teszi a portot
- A külsö és a belső port lehet más, nem kell megegyeznie

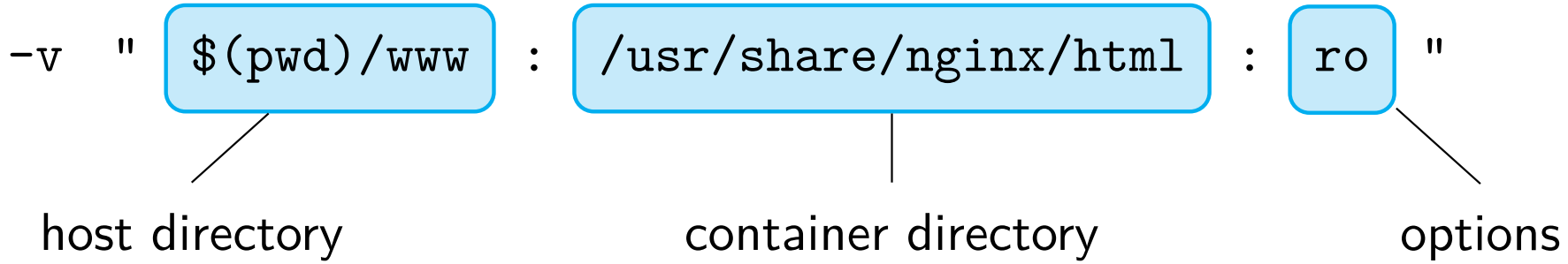
Volume

Terminal

```
-v "/home/neu/www:/usr/share/nginx/html"
```

- A `-v` vagy `--volume` fájlokat/mappákat csatol fel a konténerhez
- Abszolút hivatkozással kell megadni az elérhetőséget
- Az értéket célszerű idézőjelek között megadni, hogy a szóközt tartalmazó mappa nevek ne okozzanak gondot

Volume



Terminal

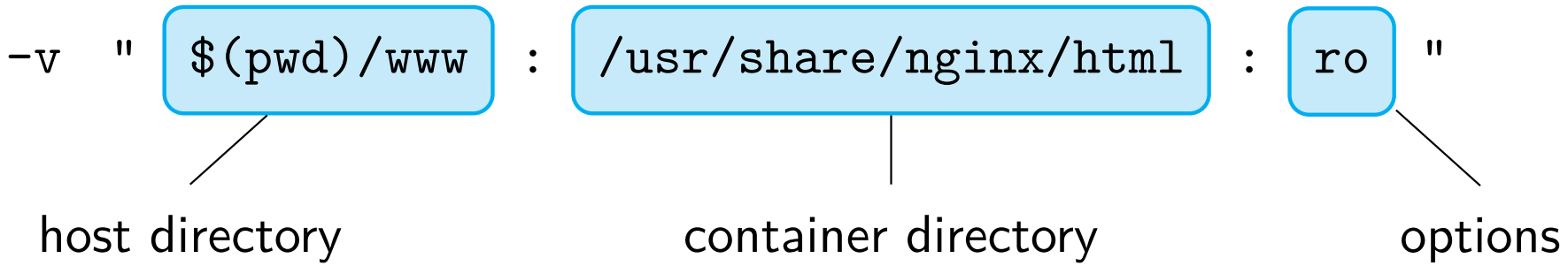
```
pwd
```

Eredmény

```
/home/neu/docker
```

- A `pwd` megadja az aktuális munkamappát (Print Working Directory)
- A `$()` parancsbehelyettesítést végez
  - Az aktuális helyi mappát (pl.: `/home/neu/www`) mappát csatolja fel
  - a konténer `/usr/share/nginx/html` mappájába

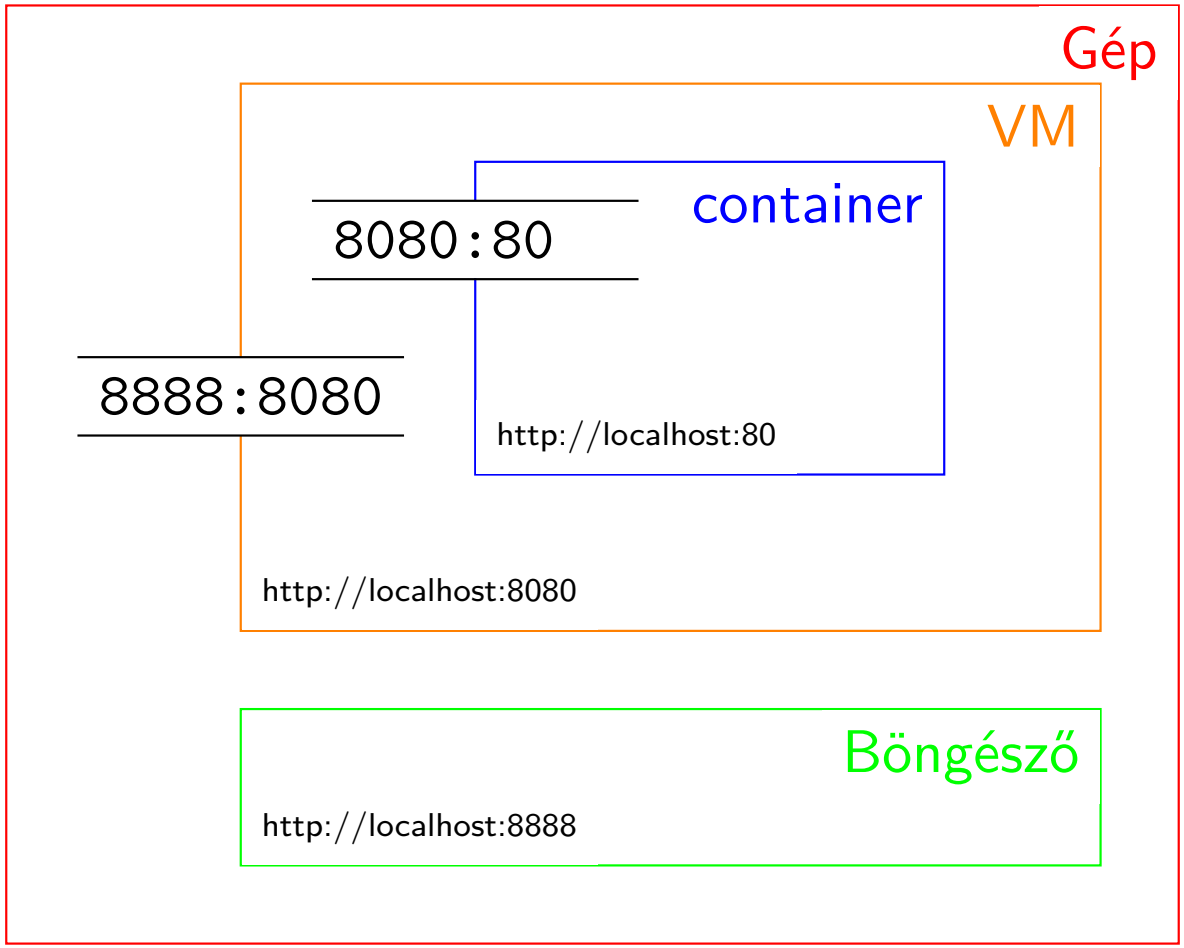
Volume



Opciók:

- `ro` - csak olvasható
- `rw` - írható és olvasható

Port és virtuális gép



Tartalom

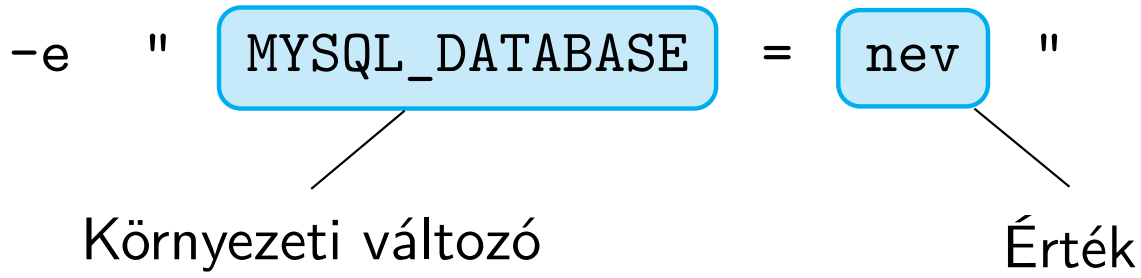
MySQL

Terminal

```
docker run \
        -d \
        --rm \
        --name my-mysql \
        -p "3306:3306" \
        -e "MYSQL_ROOT_PASSWORD=p_ssW0rd" \
        -e "MYSQL_DATABASE=hello_vilag" \
        mysql
```

- `MYSQL_ROOT_PASSWORD` – A root felhasználó jelszava
- `MYSQL_DATABASE` – Az adatbázis neve

Környezeti változók



- A `-e` vagy `--env` kapcsolókal lehet környezeti változókat beállítani
- Idézőjel nem kötelező, amíg valamelyik érték nem tartalmaz szóközt
- A egyenlőség jel előtt és után szóköz NEM szerepelhet
- A kulcsok nevét nagy betűvel írjuk általában
- Az image készítője dokumentálja, hogy miket használhatunk
- Több környezeti változó is átadható

Tartalom

Tartalom

Dockerfile példa: FROM

Dockerfile

Dockerfile

```
FROM alpine:3.20
ENTRYPOINT ["date", "-I"]
```

- A `FROM` után lehet meghatározni az alap image-et
- Az `ENTRYPOINT` határozza meg, a belépési pontot
  - Tömbszerűen célszerű meghatározni
  - A bemenő paraméterek lesznek a további elemek
  - Itt a `-I` miatt a `date` ISO formátumban adja meg az eredményt

Tartalom

Dockerfile példa: build

build

Terminal

```
docker build -t rcsnjszg/ma .
```

- A `-t` után lehet megadni a teljes nevét, a tag névvel együtt
- Az utolsó paraméter, hogy hol keresse a Dockerfile-t, a `.` az aktuális mappa

run

Terminal

```
docker run rcsnjszg/ma
```

- A build során, a `-t` által meghatározott névvel lehet futtatni
- Csak azon a gépen működik, ahol a build lefutott
- Eredményként megjeleníti a mai dátumot, és véget is ér a container pályafutása

Eredmény

```
2024-09-03
```

Dockerfile példa: build -f

hello.Dockerfile

hello.Dockerfile

```
FROM alpine:3.20
ENTRYPOINT ["date", "-I"]
```

build

Terminal

```
docker build -f hello.Dockerfile -t rcsnjszg/ma:v2 .
```

- Alapértelmezetten ahol állunk, abban a mappában keresi a `Dockerfile` nevű fájlt
- A `-f` után lehet megadni a fájl nevét, ha ettől eltérünk
- Az elenvezési séma legyen `xxx.Dockerfile`

Linkek:

- dockerfile filename – docs.docker.com

Tartalom

Dockerfile példa: WORKDIR ÉS COPY

Vegyük az alábbi mappaszerkezetet:

```
/
├── Dockerfile
└── program.py
```

Dockerfile

```
FROM python:3-alpine3.20
WORKDIR /app
COPY program.py hello.py
ENTRYPOINT ["python3","hello.py"]
```

- A `WORKDIR` beállítja a munkamappát az image-en belül
  - Ha nem létezik, akkor létrehozza
- A `COPY` másolásra alkalmas, többféleképpen is használható
  - A gépünkön lévő `program.py` fájlt másolja fel
  - Az image `WORKDIR`-ben meghatározott mappájába `hello.py` néven

Dockerfile példa: WORKDIR ÉS COPY

program.py

```
print("Hello World")
```

```
/
├── Dockerfile
└── program.py
```

build

Terminal

```
docker build -t rcsnjszg/hello:python .
```

- A namespace / szerző / regisztrált dockerhub fiók `rcsnjszg`
- A repository `hello`
- A tag `python`
- Egyelőre csak a gépen használható, ahol a build le lett futtatva

run

Terminal

```
docker run rcsnjszg/hello:python
```

Eredmény

```
Hello World
```

Dockerfile példa: .dockerignore

program.py
```
print("Hello World")
```

.dockerignore
```
.DS_Store
.git
```

```
/
├── .dockerignore
├── Dockerfile
└── program.py
```

Dockerfile
```
FROM python:3-alpine3.20
WORKDIR /app
COPY . .
ENTRYPOINT ["python3","program.py"]
```

- A `COPY . .` mindent felmásol a gépről a WORKDIR mappába, kivéve a `.dockerignore` fájlban meghatározottak
- Az ENTRYPOINT-ban is a fájlrendszerben lévő `program.py`-nek kell szerepelnie

Linkek:

- dockerignore – shisho.dev

Tartalom

RUN

- A `RUN` a buildelés során hajt végre utasításokat
- Minden egyes RUN parancs új réteget hoz létre
- Két formában is megadható
  - **Shell formátum** – pont úgy, ahogy a shell-ben `apt install`
  - **Exec formátum** – tömbszerűen: `["apt", "install"]`
- A RUN esetében a Shell formátum a gyakoribb

Linkek:

- dockerfile RUN – docs.docker.com

Dockerfile példa: RUN

Dockerfile

```
FROM ubuntu:24.04

RUN apt update
RUN apt install ncal

ENTRYPOINT ["ncal", "-b", "-M"]
```

- Az első `RUN` a csomaglista frissítésekor új réteget hoz létre
- A második `RUN` az ncal csomag telepítésekor új réteget hoz létre
- A rétegek cachben tárolódnak el
- A build után futtatható az új image, ami egy naptárt jelenít meg a konzolon

Dockerfile példa: RUN

Dockerfile

```
FROM ubuntu:24.04

RUN apt update && apt install ncal

ENTRYPOINT ["ncal", "-b", "-m"]
```

- Túl sok réteg kerülendő
- A két RUN összevonható, így kettő helyett egy réteg keletkezik
- Az összevonás után a debuggolás nehezebbé válik
- A build után futtatható az új image, ami egy naptárt jelenít meg a konzolon

Tartalom

Dockerfile példa: CMD önállóan

Dockerfile

```
FROM ubuntu:24.04
RUN apt update && apt install ncal
CMD ["ncal", "-b", "-M"]
```

- A végén nem ENTRYPOINT, hanem CMD szerepel

Terminal

```
docker build -t rcsnjszg/calendar:v2 .
```

Terminal

```
docker run rcsnjszg/calendar:v2
```

Eredmény

```
   September 2024
Mo Tu We Th Fr Sa Su
                   1
2  3  4  5  6  7  8
9  10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

- Felparaméterezi az `ncal` -t
  - `-b` – vízszintes elrendezés
  - `-M` – hétfővel kezdje a hetet

Tartalom

Dockerfile példa: CMD és ENTRYPOINT

Dockerfile

```
FROM ubuntu:24.04
RUN apt update && apt install ncal
ENTRYPOINT ["ncal"]
CMD ["-b", "-M"]
```

- A program az ENTRYPOINT-ban, a paraméterezése a CMD-ben

Terminal

```
docker build -t rcsnjszg/calendar:v3 .
```

Terminal

```
docker run rcsnjszg/calendar:v3
```

Eredmény

```
   September 2024
Mo Tu We Th Fr Sa Su
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

- Felparaméterezi az `ncal` -t
  - `-b` – vízszintes elrendezés
  - `-M` – hétfővel kezdje a hetet

Dockerfile példa: CMD és ENTRYPOINT

Dockerfile

```
FROM ubuntu:24.04
RUN apt update && apt install ncal
ENTRYPOINT ["ncal"]
CMD ["-b", "-M"]
```

Terminal

```
docker run rcsnjszg/calendar:v3 12 2028
```

Eredmény

```
December 2028
Su      3 10 17 24 31
Mo      4 11 18 25
Tu      5 12 19 26
We      6 13 20 27
Th      7 14 21 28
Fr   1  8 15 22 29
Sa   2  9 16 23 30
```

- Felparaméterezi az `ncal` -t
  - `12` – hónap
  - `2028` – év
- Megváltozott a formátum
- Vasárnappal kezdődik
- Felülírta a CMD-t, a `run` utasítás vége

Dockerfile példa: CMD és ENTRYPOINT

Dockerfile

```
FROM ubuntu:24.04
RUN apt update && apt install ncal
ENTRYPOINT ["ncal", "-b"]
CMD ["-M"]
```

- A `-b` az ENTRYPOINT-ban szerepel a CMD helyett

Terminal

```
docker build -t rcsnjszg/calendar:v4 .
```

Terminal

```
docker run rcsnjszg/calendar:v4 09 2035
```

Eredmény

```
September 2035
Su Mo Tu We Th Fr Sa
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
```

- Az ENTRYPOINT-ban meghatározott paraméter (vízszintes formátum) megmaradt
- A CMD-ben megatározott paraméter (hétfő az első nap) felülíródott