Docker

Bevezető

Rostagni Csaba

2024. szeptember 7.

Tartalom

Hypervisor

Definition (Hypervisor)

A hypervisor – vagy magyarul hiperfelügyelő – olyan szoftver vagy hardver, ami virtuális számítógépek futtatását végzi.

Forrás:

- Hypervisor - hu.wikipedia.org

Host és Guest

Definition (Host)

A számítógépet, ami a hypervisort működteti hosztnak (kiszolgáló, virtualizációs szerver) nevezzük.

Definition (Guest)

A virtuális számítógépeket vendégnek (angolul guest) is nevezzük, ezek számára a hypervisor egy virtuális környezetet biztosít, amelyen a vendég operációs rendszer fut.

- Több ilyen virtuális számítógép osztozkodhat a hoszt gép erőforrásain.

Kernel

Definition (Kernel)

Rendszermag (angolul kernel): az operációs rendszer alapja (magja), amely felelős a hardver erőforrásainak kezeléséért (beleértve a memóriát és a processzort is).

Forrás:

- Rendszermag / Kernel - hu.wikipedia.org

Daemon

Definition (Daemon)

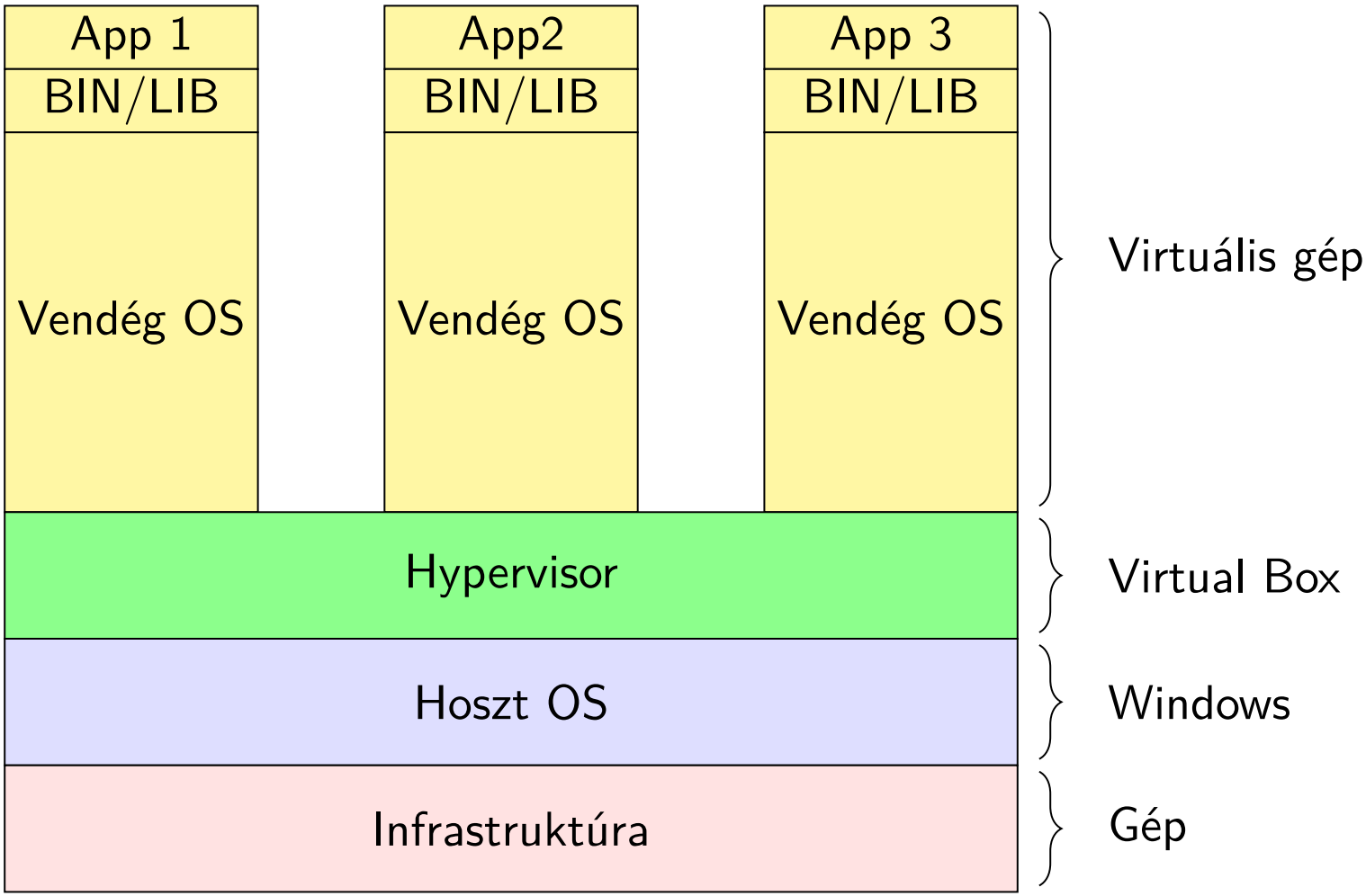
Olyan háttérben futó folyamat, ami valamilyen szolgáltatást nyújt, a felhasználó nem kommunikál vele közvetlenül.

Forrás:

- Démonok – szit.hu

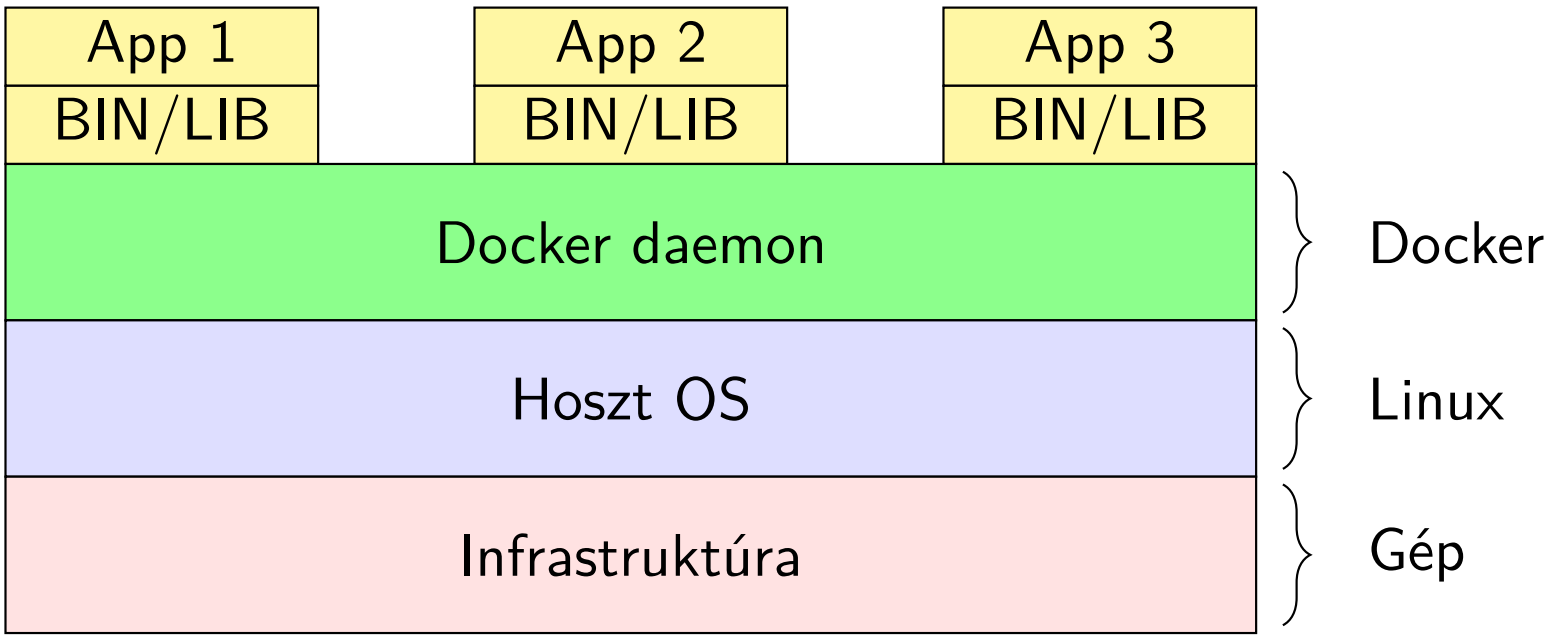
Tartalom

Virtuális gép felépítése



Virtuális gép (2-es típusú)

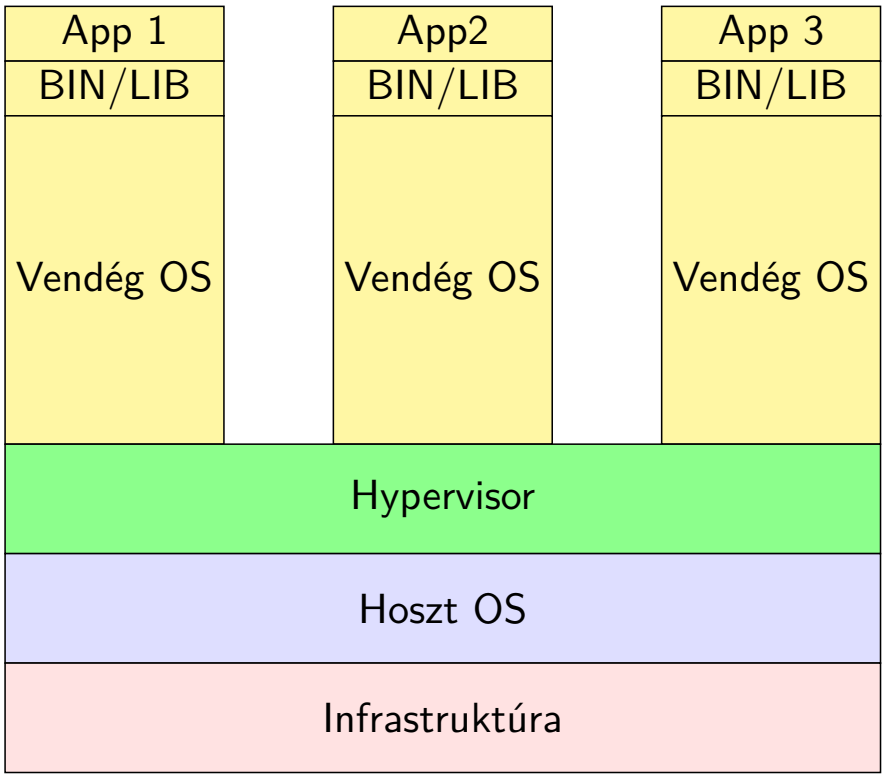
Docker (konténerizációs környezet) felépítése

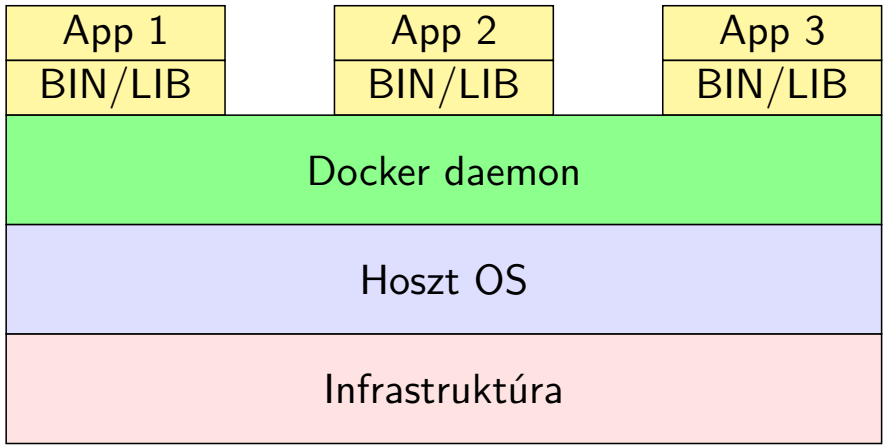


Docker

- A **Hoszt OS** lehet Linux, vagy Windows is, mi olyan konténereket használunk, amik **Linux** alapon működnek

VM vs. Docker


App 1
BIN/LIB
Vendég OS
App2
BIN/LIB
Vendég OS
App 3
BIN/LIB
Vendég OS
Hypervisor
Hoszt OS
Infrastruktúra


App 1
BIN/LIB
App 2
BIN/LIB
App 3
BIN/LIB
Docker daemon
Hoszt OS
Infrastruktúra

VM vs. Docker

- Virtuális gép (VM)
  - Nagy méret (GB)
  - Lassú indulás
  - Robusztus
  - Izolált
- Docker
  - Kis méret (MB)
  - Gyors indulás
  - Könnyűsúlyú
  - Izolált

Tartalom

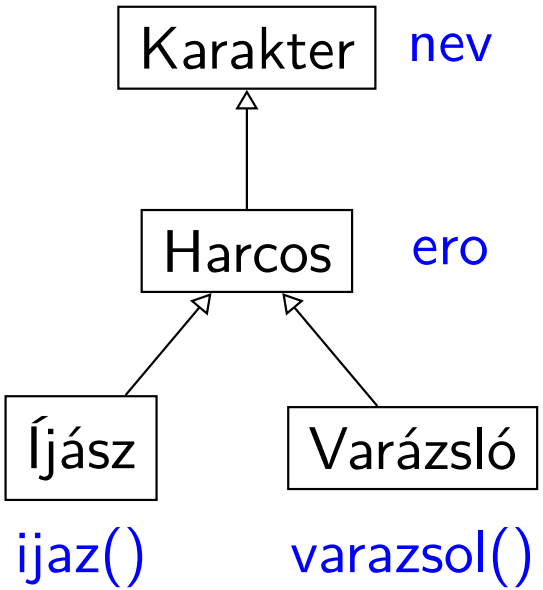
Tartalom

Docker image

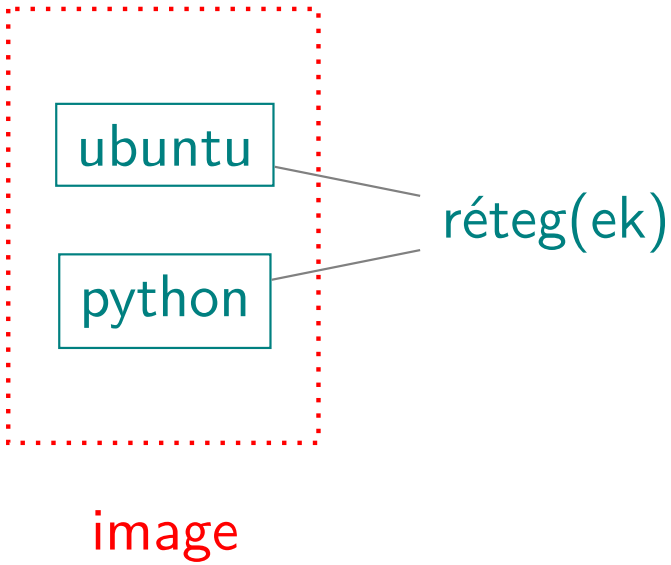
- A docker image rétegekből épül fel
- Ugyanarra a rétegre több különböző is épülhet (helytakarékosság)
- Tartalmazza
  - az applikáció futtatásához tartozó környezetet
  - a futtatandó programot vagy programkódot

OOP hasonlat

OOP



docker image

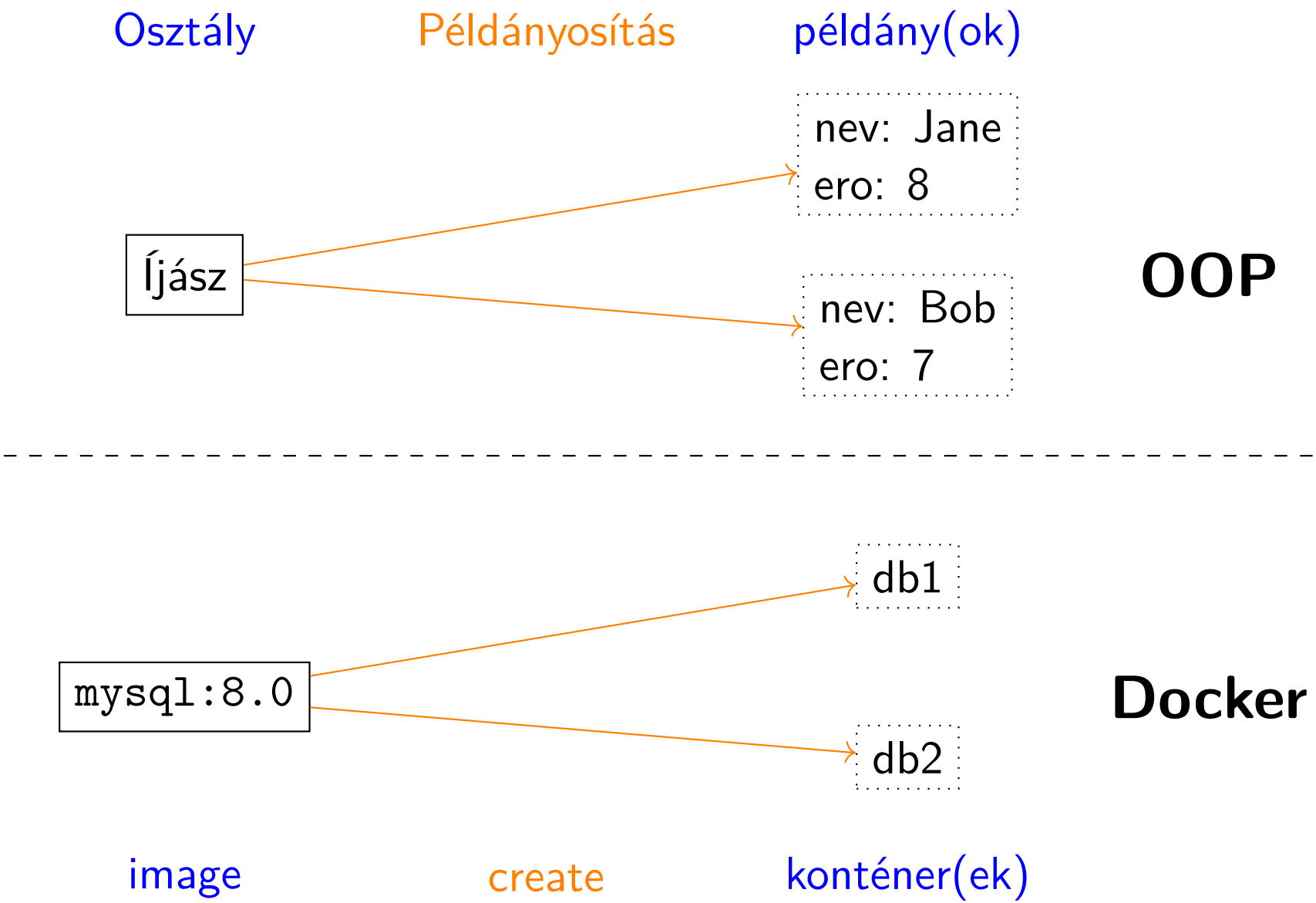


Tartalom

Docker container

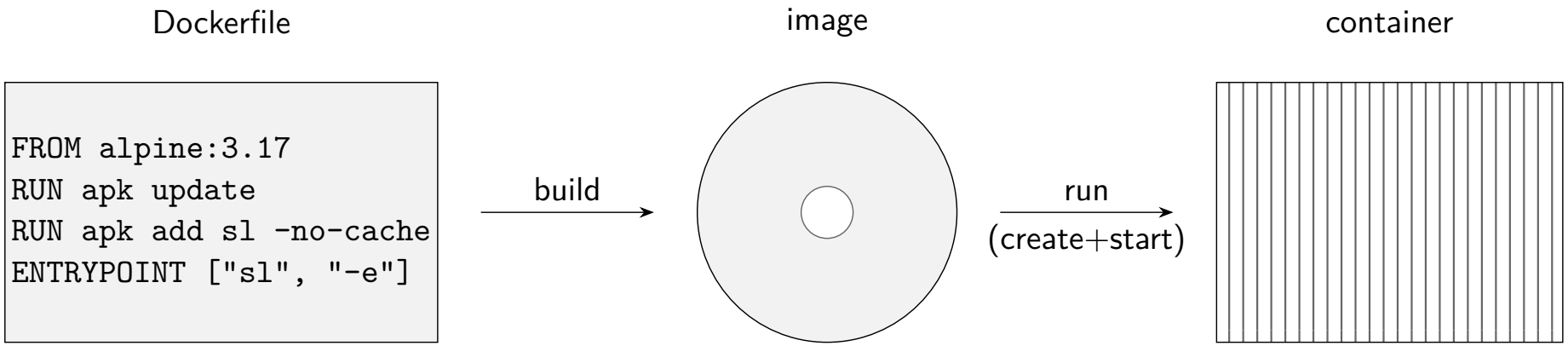
- Egy futó alkalmazás az előre beállított futtatókörnyezettel
- Az image-re alapszik
- Egy image-ből több containert is lehet létrehozni
- Tartalmazza az aktuális adatokat is

Példányosítás és a create parancs példa



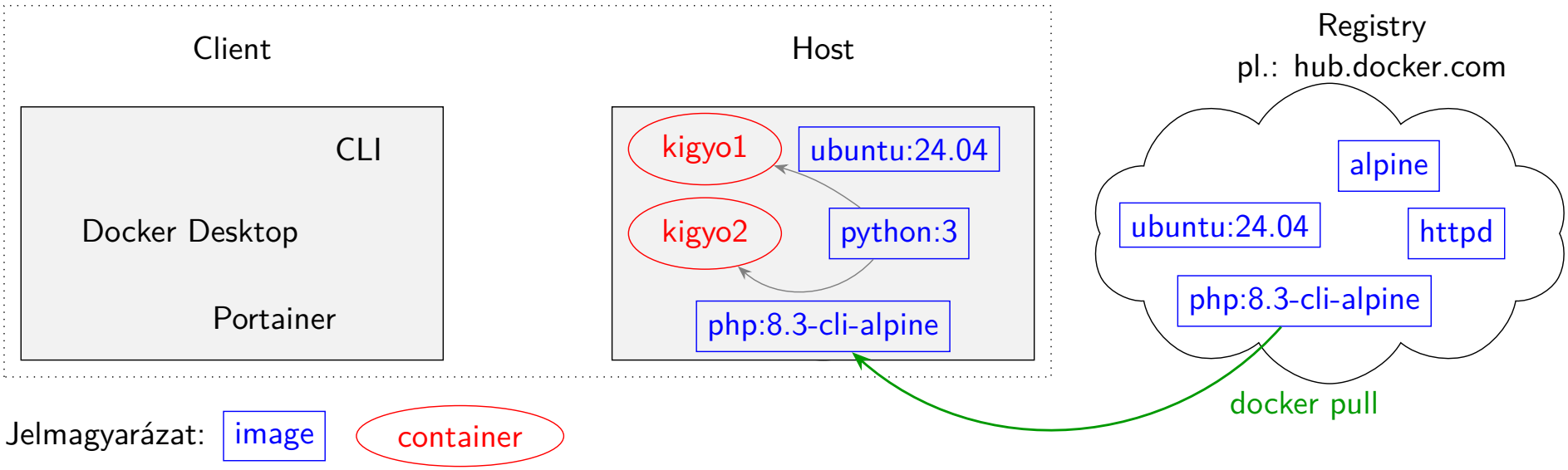
Tartalom

Dockerfile, image és container



Tartalom

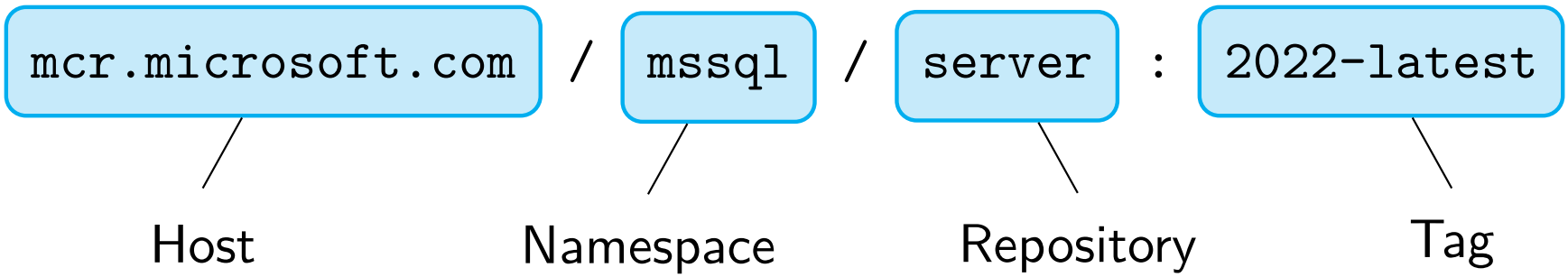
Architektúra: client, host és registry



- A Host gépen található a docker daemon, és az általa használta image-ek és container-ek
- Szerver-Kliens architektúrát használ, a kliens REST API-n keresztül kommunikál a Docker daemon-nal (szerver)
- A szerver és a kliens futhat ugyanazon a (virtuális) gépen, vagy külön
- A registry egy tároló, ahol előre elkészített image-ek vannak

Tartalom

Image nevek

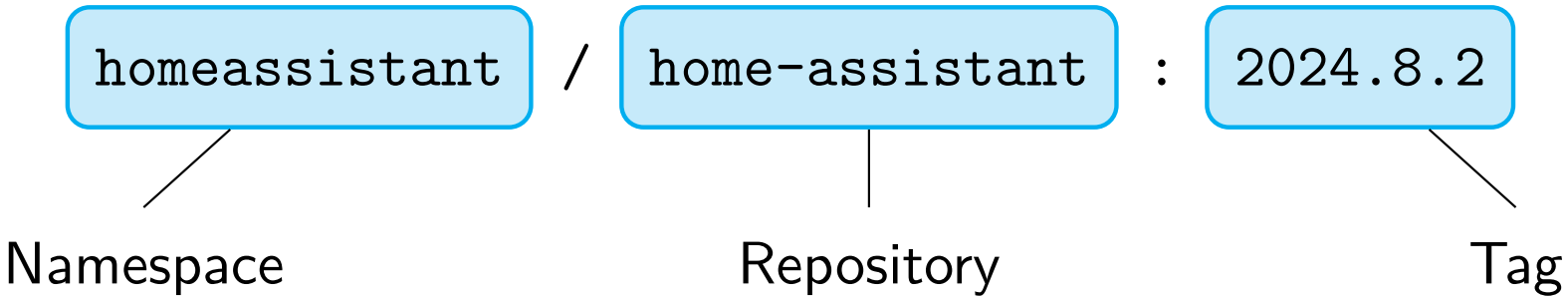


- **Host** – A szerver, ahol az image megtalálható, alapértelmezetten a dockerhubon keresi (`docker.io`)
- **Namespace** – A felhasználó vagy gyártó neve, alapértelmezetten `library` lesz
- **Repository** – A kódtár neve (általában maga a program)
- **Tag** – Egy megadott verzió, alapértelmezetten `latest` , de valójában nem biztos, hogy ez a legfrissebb

Linkek:

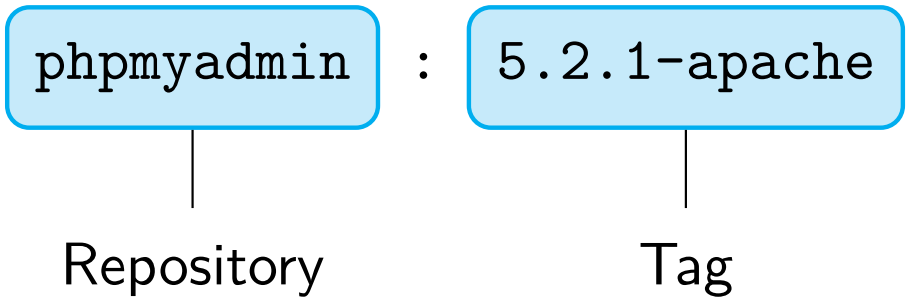
- tag – docs.docker.com

Image nevek



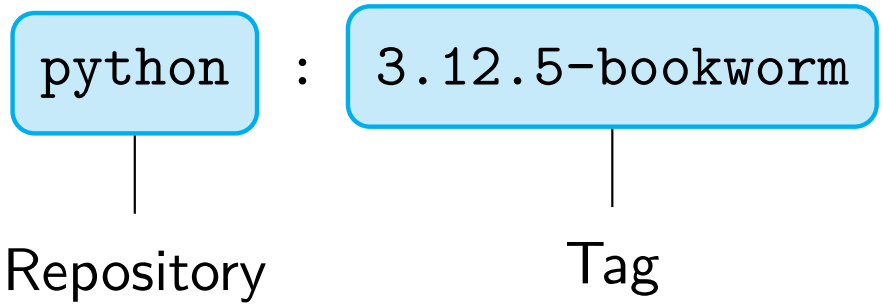
- A host nincs meghatározva, így a Docker Hub-ról szerezhető be
- A namespace **homeassistant**, több repository is tartozik hozzá
- Az applikáció neve **home-assistant**
- A kiválasztott verzió **2024.8.2**

Image nevek



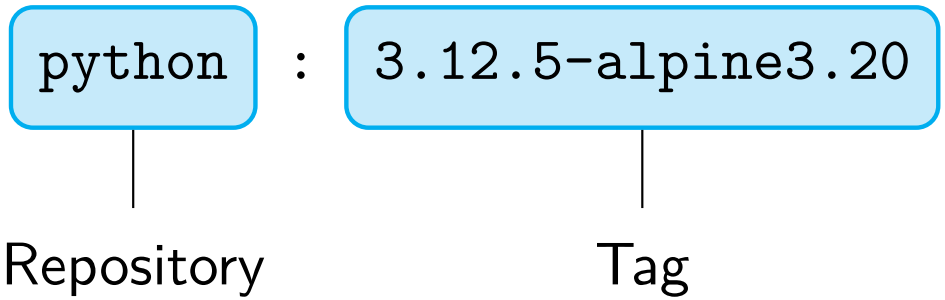
- A host nincs meghatározva, így a Docker Hub-ról szerezhető be
- A namespace elhagyható, ugyanis ez már egy hivatalos image, az alapértelmezett `library` értéket használja
- Az applikáció neve **phpmyadmin**
- A kiválasztott tag **5.2.1-apache**
  - A PHPMyAdmin verziója 5.2.1
  - Háromféle kiadás létezik, ez az, ami mögött egy **apache** szerver működik, és minden ami szükséges egybe van csomagolva

Image nevek



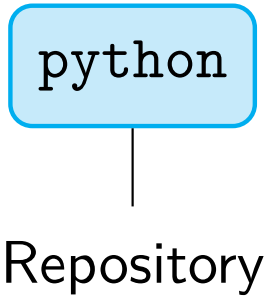
- A host nincs meghatározva, így a Docker Hub-ról szerezhető be
- A namespace elhagyható, ugyanis ez már egy hivatalos image, az alapértelmezett `library` értéket használja
- Az applikáció neve **python**
- A kiválasztott tag **3.12.5-bookworm**
  - A Python verziója 3.12.5
  - Ez az a verzió, aminek az alapja a **debian:bookworm** image

Image nevek



- A host nincs meghatározva, így a Docker Hub-ról szerezhető be
- A namespace elhagyható, ugyanis ez már egy hivatalos image, az alapértelmezett `library` értéket használja
- Az applikáció neve **python**
- A kiválasztott tag **3.12.5-alpine3.20**
  - A Python verziója 3.12.5
  - Ez az a verzió, aminek az alapja a **alpine:3.20** image

Image nevek



- A host nincs meghatározva, így a Docker Hub-ról szerezhető be
- A namespace elhagyható, ugyanis ez már egy hivatalos image, az alapértelmezett `library` értéket használja
- Az applikáció neve **python**
- Nincsen tag meghatározva, így ennek értéke `latest` lesz
  - Ez jelenleg a `3.12.5-bookworm` verzióval egyezik meg
  - Előny
    - Nincs meghatározott verzió, egyszerű frissítést
  - Hátrány
    - Valójában nem mindig az ezzel ellátott a legfrissebb
    - Előfordulhat, hogy az újabb verzióval nem kompatibilis a kódunk

Tartalom

Tartalom

docker create

```
docker create [OPTIONS] IMAGE [COMMAND] [ARG...]
```

- Letölti az image fájlt, ha az nem létezik
- Létrehoz egy konténert a megadott imageből, de nem indítja el
- Valójában a `docker create` csak egy alias (lásd dokumentáció)

Linkek:

- docker create – docs.docker.com

docker create első futtatás

Terminal

```
docker create --name ubuntu1 ubuntu:24.04
```

- A `--name ubuntu1` egy elhagyható opció
- A felhasznált image "teljes neve": `ubuntu:24.04`

Eredmény

```
Unable to find image 'ubuntu:24.04' locally
24.04: Pulling from library/ubuntu
31e907dcc94a: Pull complete
Digest: sha256:8a37d68f4f73ebf3d4efafbcf66379bf3728902a8038616808f04e34a9ab63ee
Status: Downloaded newer image for ubuntu:24.04
7537b0f26dbeb4325774903cc941bcc4d8c531b6ec3512db49d1f02daa60eedf
```

- Mivel az image nincs a gépen, így azt letölti
- Alapértelmezetten a Docker Hub-on keresi

Linkek:

- Ubuntu – hub.docker.com

docker create későbbi futtatás

D1

Terminal

```
docker create --name ubuntu2 ubuntu:24.04
```

Eredmény

```
f73e788e8c5e7add0a8f23abdda7e11ebafa5e1fa7990c23bbcf6a2a68187829
```

- Mivel az image már megtalálható a gépen így nem tölti le, hanem rögtön létrehozza a konténert

Tartalom

docker create példa eredménye

D2

Terminal

```
docker ps -a
```

Eredmény

```
CONTAINER ID   IMAGE          COMMAND   CREATED              STATUS    PORTS   NAMES
f73e788e8c5e   ubuntu:24.04   "bash"    About a minute ago   Created           ubuntu2
3b88ac096a0d   ubuntu:24.04   "bash"    5 minutes ago        Created           ubuntu1
```

- Létrejött a két konténer
- A `STATUS` mind a két esetben `Created`

Terminal

```
docker ps -a --no-trunc
```

- A teljes ID kinyeréséhez a `--no-trunc` opció szükséges
  - `f73e788e8c5e7add0a8f23abdda7e11ebafa5e1fa7990c23bbcf6a2a68187829`
  - `3b88ac096a0d0e4f3ceb6507abf968043f6868751af47c10e9ffe6616ece39bf`

Linkek:

- docker ps – docs.docker.com

Tartalom

Mi a konténer azonosítója?

D3

Terminal

```
docker ps -a
```

Eredmény

```
CONTAINER ID   IMAGE          COMMAND   CREATED              STATUS    PORTS   NAMES
f73e788e8c5e   ubuntu:24.04   "bash"    About a minute ago   Created           ubuntu2
3b88ac096a0d   ubuntu:24.04   "bash"    5 minutes ago        Created           ubuntu1
```

Terminal

```
docker inspect --format="{{.Id}}" ubuntu2
```

Eredmény

```
f73e788e8c5e7add0a8f23abdda7e11ebafa5e1fa7990c23bbcf6a2a68187829
```

- A `--format` szabja meg mely adatok kerüljenek megjelenítésre
- A példában csak az `Id` kerül megjelenítésre

Linkek:

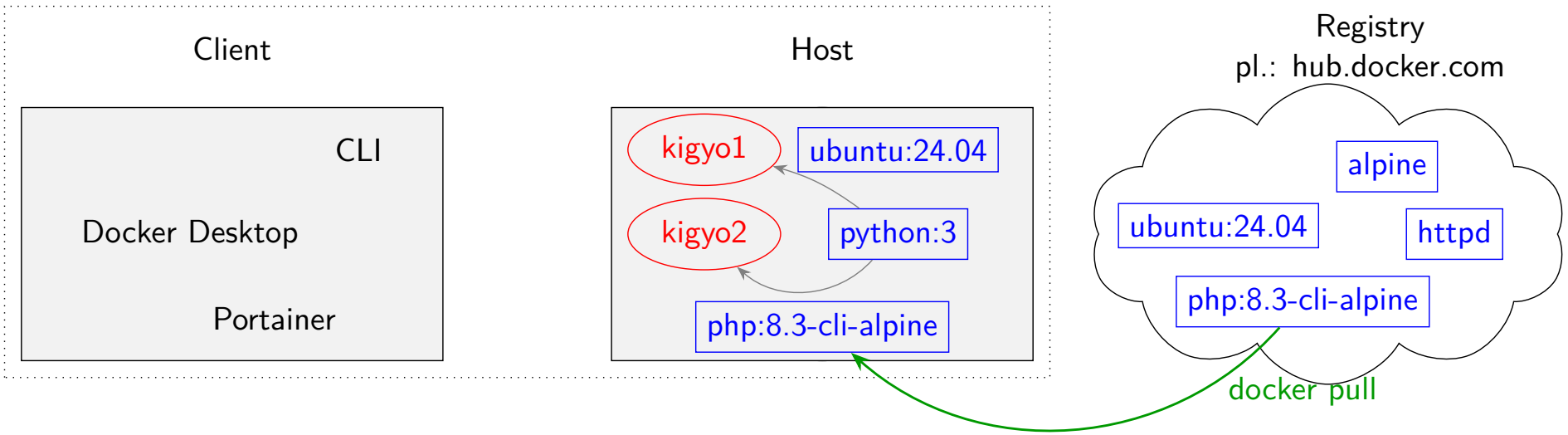
- docker inspect – docs.docker.com
- formatting (GO) – docs.docker.com

Tartalom

docker pull


```
docker pull [OPTIONS] NAME[:TAG|@DIGEST]
```

- A `docker pull` letölti a registryből az image-t



Linkek:

- docker pull – docs.docker.com

Tartalom

docker start

```
docker start [OPTIONS] CONTAINER [CONTAINER...]
```

- Elindít egy létező konténert, ami `Created` vagy `Stopped` állapotban van

Linkek:

- docker start – docs.docker.com

docker start példa

D5

Terminal

```
docker start ubuntu1
```

Eredmény

```
ubuntu1
```

- Az indítás után, ha minden rendben ment a konténer nevét írja ki

Terminal

```
docker ps -a --format="table {{.ID}}\t {{.Image}}\t{{.Status}}\t{{.Names}}"
```

Eredmény

```
CONTAINER ID   IMAGE          STATUS                     NAMES
f73e788e8c5e   ubuntu:24.04   Created                    ubuntu2
3b88ac096a0d   ubuntu:24.04   Exited (0) 2 minutes ago   ubuntu1
```

- Amint elindult ki is lépett `0` állapotkóddal, azaz rendben lefutott
- `Exited` állapotba került
- Miért? A konténernek nem volt mit csinálnia, így gyorsan végzett a feladatával

Tartalom

docker run

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

- A `docker create` és a `docker start` parancsokat együttesen helyettesíti

Linkek:

- docker run – docs.docker.com

docker run példa

D6

```
docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Terminal

```
docker run --rm --name szamologep ubuntu:24.04 expr 3 + 7
```

- OPTIONS
  - `--rm`: a konténer törlésre kerül, miután véget ért a futása
  - `--name`: a konténer neve, jelen esetben `szamologep` lesz
- IMAGE
  - `ubuntu:24.04`
- COMMAND
  - A konténeren belül az `expr` parancsot futtatja
- ARGS: `3`, `+` és `7`

Eredmény

```
10
```

docker run interaktív példa

Terminal

```
docker run -i -t --rm --name ubuntu3 ubuntu:24.04 /bin/bash
```

- OPTIONS
  - `-i`, vagy `--interactive`: Interaktív mód, azaz a konténer megkapja a terminálban kiadott parancsokat
  - `-t`, vagy `--tty`: Lefoglal egy pszeudó TTY-t (Teletypewritert)
  - A `--rm` hatására eltávolítja kilépéskor a konténert
  - `--name`: a konténer neve, jelen esetben `ubuntu3` lesz
- IMAGE – `ubuntu:24.04`
- COMMAND – `/bin/bash`

Infó

Az `-i` és `-t` együttesen használandó!

Eredmény

```
root@bae4c0ab048f:/#
```

- Az ubuntu promptja megjelenik, ami tartalmazza a konténer ID-t

docker run interaktív példa

bae4c0ab048f

```
expr 2 + 2
```

Eredmény

```
4
```

bae4c0ab048f

```
exit
```

- Az `exit` utasítással lehet kilépni a bashből, így a konténer élete is véget ér
- A `--rm` miatt a kilépés után meg is szűnik a a konténer

docker run interaktív példa 2

D7

Terminal

```
docker run -it --rm ubuntu
```

- OPTIONS
  - A `-it` az a `-i` és `-t` összevonása, de lehetne `-ti` is
  - hatására a konténer standard kimenetre írt szövege megjelenik, továbbá a terminálba írt utasításokat is megkapja a konténer
  - A `--rm` hatására eltávolítja kilépéskor a konténert
- IMAGE
  - ubuntu

Eredmény

```
root@f7fdede22552:/#
```

- A `--name` elhagyható, ha nem szükséges hivatkozni rá
- A tag elhagyásával az `ubuntu:latest` image lesz az alap
- Alapértelmezetten a `bash` indul el, de ez image függő
- Ez egy másik container, így a hash is megváltozott (f7fdede22552)

Tartalom

docker stop

```
docker stop [OPTIONS] CONTAINER [CONTAINER...]
```

- A konténer fő folyamatát állítja le `SIGTERM`, majd a türelmi idő lejárta után `SIGKILL` segítségével
- A konténer állapota `Stopped` lesz
- Egyszerre több konténert is le lehet állítani

Linkek:

- docker stop – docs.docker.com
- sigterm vs. sigkill – linuxhandbook.com

docker stop példa: leállítás név alapján

Terminal

```
docker run --name kigyo -it python:3
```

- Elindít egy interaktív módban futó konténert `kigyo` néven

7daed29f5e07

```
>>> exit()
```

- Az `exit()` függvénnyel lép ki a python 3 interaktív felülete

Terminal

```
docker stop kigyo
```

- Az utasítást egy másik terminál ablakban lehet kiadni!
- A konténer a nevére hivatkozva leállítható

docker stop példa: leállítás ID alapján

Terminal

```
docker run --name kigyo2 -it python:3
```

- Elindít egy interaktív módban futó konténert `kigyo2` néven

Terminal

```
docker inspect --format="{{.Id}}" kigyo2
```

- A `kigyo2` nevű konténer teljes azonosítójának lekérdezése

Eredmény

```
7daed29f5e078113696cf3bff335bbb122e5f1e2ad2d3b77c4ad0392a9531b1e
```

Terminal

```
docker stop 7daed29f5e078113696cf3bff335bbb122e5f1e2ad2d3b77c4ad0392a9531b1e
```

- A konténer ID megadásával is leállítható

docker stop példa: leállítás ID részlet alapján

Terminal

```
docker run -d httpd
```

- Elindít egy, a konzolról lecsatlakoztatott konténert, ami a háttérben tovább fut
- A `-d` vagy `--detach` hatására a háttérben fut a folyamat

Eredmény

```
8eb7130627a5937e3ada73a4524f9e4e24718eaa3c9baadc2eea9aff74d6e778
```

- A konténer indításakor megjeleníti az ID-t

Terminal

```
docker stop 8eb
```

- Elegendő az első néhány karaktert megadni az azonosítóból, amennyiben az egyedi

Eredmény

```
8eb
```

docker stop példa: azonos kezdetű azonosítók

Hiba!

```
docker stop 3b
```

Eredmény

```
Error response from daemon:
Cannot kill container: 3b:
Multiple IDs found with provided prefix: 3b
```

- A rendszeren több olyan konténer is található, melyeknek az azonosítójának az első két karaktere: `3b`

docker stop példa: több image leállítása

Terminal

```
docker run --rm -d nginx;
docker run --rm -d --name web2 httpd;
```

Eredmény

```
52e6413b43e924b2abe90fbff36c4aff0f2755ddfb9aa41479bc874e63b8a870
b4bfdee10b47dbbcbc70807d848263336e35af95ec58677d61d32478aa1cb7dc
```

Terminal

```
docker stop 52e6413b web2
```

Eredmény

```
52e6413b
web2
```

- Az `nginx` konténer az azonosítójának a részletével lett leállítva
- A `httpd` az neve által lett leállítva
- Amikor leáll ugyanezeket az értékeket írja ki

Tartalom

docker kill

```
docker kill [OPTIONS] CONTAINER [CONTAINER...]
```

- A konténer fő folyamatát "öli meg" `SIGKILL` segítségével
- A konténer állapota `Stopped` lesz
- Egyszerre több konténerre is kiadható

Linkek:

- docker kill – docs.docker.com
- sigterm vs. sigkill – linuxhandbook.com

Tartalom

docker rm

```
docker rm [OPTIONS] CONTAINER [CONTAINER...]
```

- Alapesetben leállított konténereket távolítja el
- A `-f` vagy `--force` opcióval futó konténereket is eltávolítja
- A `-v` vagy `--volume` opcióval a névtelen felcsatolt köteteket is törli
- Egyszerre több konténerre is kiadható

Linkek:

- docker rm – docs.docker.com

docker rm példa

D8

Terminal

```
docker run -d --name web nginx;
```

Terminal

```
docker rm web;
```

Eredmény

```
Error response from daemon:
You cannot remove a running container ....
Stop the container before attempting removal or force remove
```

Terminal

```
docker rm -f web;
```

- A `-f` opció kikényszeríti a konténer törlését