

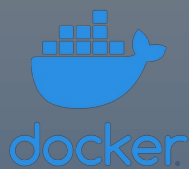
Docker – kick start

Krótkie wprowadzenie do technologii



docker®

Zespół: Alesia Filinkova, Diana Pelin, Weronika Maślana



Co to jest docker?

- To platforma do uruchamiania aplikacji w lekkich kontenerach.
- Kontener to mini-środowisko z aplikacją oraz jej zależnościami.
- Kontenery dają powtarzalne środowiska niezależne od systemu operacyjnego. To rozwiązuje problem „u mnie działa, a u ciebie nie”.



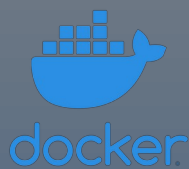
Docker vs maszyna wirtualna

Docker	Maszyny wirtualne
lekkie kontenery	pełne systemy operacyjne
Szybki start (w ms)	Wolniejszy start (w s. lub min.)
Mały rozmiar	Duży rozmiar
Współdzielone zasoby	Duplikacja zasobów



Najważniejsze pojęcia:

- Image (obraz) — gotowy przepis na kontener
- Container (kontener) — działająca instancja obrazu
- Dockerfile — instrukcje, jak zbudować obraz
- Docker Hub — repozytorium obrazów



Jak zainstalować Docker?

- Windows: Docker Desktop
- Linux: pakiet docker-ce (sudo apt-get install -y docker-ce)
- MacOS: Docker Desktop

Sprawdzamy działanie komendami:

```
docker --version
```

```
docker run hello-world
```



Pierwsze uruchomienie kontenera

Komenda:

```
docker run -it ubuntu bash
```

gdzie:

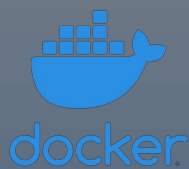
-it → tryb interaktywny

ubuntu → oficjalny obraz linuxa



Podstawowe komendy

Komenda	Opis
<code>docker pull <image>[:tag]</code>	Pobiera obraz
<code>docker images</code>	Wyświetla listę obrazów zapisanych lokalnie
<code>docker run [opcje] <image></code>	Uruchamia dany obraz
<code>docker ps</code>	Wyświetla działające kontenery
<code>docker ps -a</code>	Wyświetla wszystkie kontenery
<code>docker stop <nazwa_kontenera></code>	Zatrzymuje kontener
<code>docker rm <nazwa_kontenera></code>	Usuwa kontener



Tworzenie własnego Dockerfile

Prosty przykład

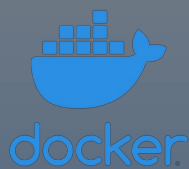
```
FROM openjdk:17
```

```
COPY app.jar app.jar
```

```
CMD ["java", "-jar", "app.jar"]
```

gdzie:

- Bazujemy na oficjalnym obrazie OpenJDK
- Kopiujemy nasz plik app.jar
- Uruchamiamy aplikację w utworzonym kontenerze



Budowanie obrazu

```
docker build -t <nazwa_obrazu> .
```

Uruchomienie obrazu:

```
docker run -p 8080:8080 myapp
```

gdzie:

-t <nazwa> → nazwa obrazu

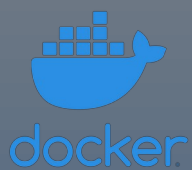
-p → mapowania portów z kontenera na hosta



Przykład uruchomienia bazy danych

```
docker run -d --name mydb -e  
POSTGRES_PASSWORD=secret -p  
5432:5432 postgres
```

To szybki sposób na uruchomienie bazy danych bez instalacji jej u siebie.



Docker Compose

To narzędzie do uruchamiania wielu kontenerów jako jeden system.
Przykładowy plik docker-compose.yml

```
services:
```

```
  db:
```

```
    image: postgres
```

```
    environment:
```

```
      POSTGRES_PASSWORD: secret
```

```
    ports:
```

```
      - "5432:5432"
```

```
  app:
```

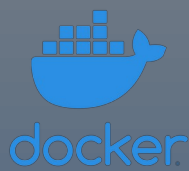
```
    build: .
```

```
    ports:
```

```
      - "8080:8080"
```

```
    depends_on:
```

```
      - db
```



Uruchomienie docker compose

`docker-compose build`

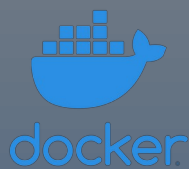
`docker-compose up`

Wtedy aplikacja i baza danych startują razem.
Umożliwia komunikację między kontenerami.



Gdzie warto używać dokera?

- Mikroserwisy
- Aplikacje webowe
- CI/CD (GitHub Actions, GitLab CI)
- Szybkie środowiska dev/test



Podsumowanie

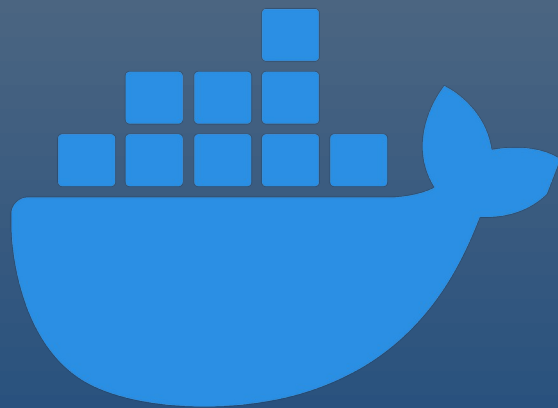
Docker = szybkie, lekkie kontenery

Dockerfile = przepis na dany kontener

Docker Compose = zapewnia komunikację między kontenerami, pozwala uruchamiać je razem

idealne narzędzie do pracy zespołowej, mikroservisów i aplikacji webowych

Dziękujemy za uwagę :)



docker®