

Docker

Wojciech Szlosek





O czym pomówimy?



Docker - wstęp i podstawowe pojęcia

- Oprogramowanie open source działające jako "platforma dla programistów i administratorów do tworzenia, wdrażania i uruchamiania aplikacji rozproszonych"
- To obecnie niezbędne narzędzie dla większości programistów.
- Według badań Stack Overflow, Docker jest technologią, której znajomość jest najbardziej pożądana wśród programistów.
- Docker jest obecnie jednym z kilku narzędzi, które uruchamiają kontenery (narzędzia do konteneryzacji)...



Alternatywy

- ...inne narzędzia to m.in.:
- Containerd, Podman, BuildKit



Podman jako alternatywa - kompatybilność z Dockerem

- Jednym z założeń Podmana jest API kompatybilne z Dockerem. Dlatego też prawie wszystkie polecenia CLI (Command Line Interface) z Docker CLI są również dostępne w programie Podman. Poniższe przykładowe polecenia są równoważne:

```
docker run -<T0 SAM0> -p 8090:80 nginx
```

```
podman run -<T0 SAM0> -p 8090:80 nginx
```

Z przyzwyczajenia wielu użytkowników Podmana używa wręcz polecenia:

```
alias docker=podman
```

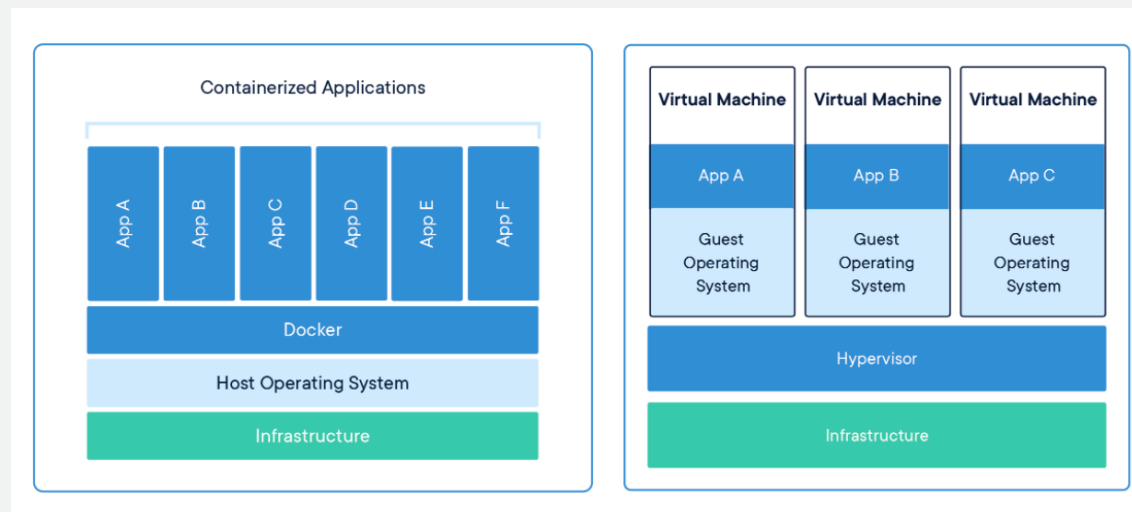
Główna różnica: architektura



Kontenery

Kontener jest to utworzone odseparowane środowisko dla danej aplikacji, zbudowane na podstawie obrazu. Zawierają niezbędne biblioteki i zależności. Kontenery są bardzo lekkie, pochłaniają minimalne zasoby – niezbędną do działania aplikacji. Wszystkie kontenery korzystają z jednego jądra (kernela) systemu hosta.

- Kontener \approx VM
- Izolacja procesów, niezależność kontenerów



Kontenery cd.

- Przeznaczeniem kontenera jest wykonywanie pojedynczego zadania (procesu). Kontener może znajdować się w jednym ze stanów:
- "created" – Kontener, który został stworzony, ale jeszcze nie uruchomiony.
- "restarting" – Kontener w trakcie ponownego uruchamiania swojego procesu.
- "up" – Kontener wykonujący swój proces.
- "paused" – Kontener, którego proces został wstrzymany.
- "exited" – Kontener, który zakończył wykonywanie swojego procesu.



Kilka słów o obrazach

- W dokumentacji Dockera często wymieniane jest określenie "container image". Podkreśla ono zależność - kontener powstaje na podstawie obrazu.
- Obraz kontenera to lekki, samodzielny, wykonywalny pakiet oprogramowania, który zawiera wszystko, co jest potrzebne do uruchomienia aplikacji (struktura plików + metadane i inne informacje):
 - - kod,
 - - środowisko wykonawcze,
 - - narzędzia systemowe,
 - - biblioteki systemowe,
 - - ustawienia.



Pora na przykład



```

drwxr-xr-x 1 root root 4096 Mar 11 10:18 etc
drwxr-xr-x 2 root root 4096 Apr 15 2020 home
lrwxrwxrwx 1 root root 7 Mar 2 18:55 lib -> usr/lib
lrwxrwxrwx 1 root root 9 Mar 2 18:55 lib32 -> usr/lib32
lrwxrwxrwx 1 root root 9 Mar 2 18:55 lib64 -> usr/lib64
lrwxrwxrwx 1 root root 10 Mar 2 18:55 libx32 -> usr/libx32
drwxr-xr-x 2 root root 4096 Mar 2 18:55 media
drwxr-xr-x 2 root root 4096 Mar 2 18:55 mnt
drwxr-xr-x 2 root root 4096 Mar 2 18:55 opt
dr-xr-xr-x 173 root root 0 Mar 11 10:18 proc
drwx----- 2 root root 4096 Mar 2 18:58 root
drwxr-xr-x 5 root root 4096 Mar 2 18:58 run
lrwxrwxrwx 1 root root 8 Mar 2 18:55 sbin -> usr/sbin
drwxr-xr-x 2 root root 4096 Mar 2 18:55 srv
dr-xr-xr-x 13 root root 0 Mar 11 10:18 sys
drwxrwxrwt 2 root root 4096 Mar 2 18:58 tmp
drwxr-xr-x 13 root root 4096 Mar 2 18:55 usr
drwxr-xr-x 11 root root 4096 Mar 2 18:58 var

```

```

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker run ubuntu uname -a
Linux 6bb2d34f6a4f 5.10.76-linuxkit #1 SMP Mon Nov 8 10:21:19 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker run debian uname -a

```

```

Unable to find image 'debian:latest' locally
latest: Pulling from library/debian
e4d61adff207: Already exists
Digest: sha256:10b622c6cf6daa0a295be74c0e412ed20e10f91ae4c6f3ce6ff0c9c04f77cbf6
Status: Downloaded newer image for debian:latest
Linux 4ed5085f91d 5.10.76-linuxkit #1 SMP Mon Nov 8 10:21:19 UTC 2021 x86_64 GNU/Linux
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker run -it ubuntu bash
root@8c7135cd7848:/# echo "tekst" > test.txt
root@8c7135cd7848:/# cat test.txt
tekst
root@8c7135cd7848:/# exit
exit

```

```

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker ps -a

```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
8c7135cd7848	ubuntu	"bash"	44 seconds ago	Exited (0) 7 seconds ago		hardcore_robinson

```
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED             STATUS              PORTS          NAMES
8c7135cd7848   ubuntu    "bash"                  44 seconds ago     Exited (0) 7 seconds ago           hardcore_robinson
4ed50085f91d   debian    "uname -a"              About a minute ago Exited (0) About a minute ago       exciting_payne
6bb2d34f6a4f   ubuntu    "uname -a"              About a minute ago Exited (0) About a minute ago       practical_jemison
7f8e83468e72   ubuntu    "ls -l"                 2 minutes ago      Exited (0) 2 minutes ago           jovial_mendel

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker run ubuntu cat test.txt
cat: test.txt: No such file or directory

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED             STATUS              PORTS          NAMES
6cc53beb6bc9   ubuntu    "cat test.txt"          19 seconds ago     Exited (1) 19 seconds ago           cranky_poincare
8c7135cd7848   ubuntu    "bash"                  About a minute ago Exited (0) 51 seconds ago           hardcore_robinson
4ed50085f91d   debian    "uname -a"              2 minutes ago      Exited (0) 2 minutes ago           exciting_payne
6bb2d34f6a4f   ubuntu    "uname -a"              2 minutes ago      Exited (0) 2 minutes ago           practical_jemison
7f8e83468e72   ubuntu    "ls -l"                 2 minutes ago      Exited (0) 2 minutes ago           jovial_mendel

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker start 6cc
6cc

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker exec 6cc cat test.txt
Error response from daemon: Container 6cc53beb6bc971df40f50fae8c6b55a0cfcfcfd6aa6fd44f0df607020fd5c918e is not running

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker start 6cc
6cc
Error response from daemon: page not found
Error: failed to start containers: 6cc

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker start 6cc
6cc

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker exec cranky_poincare cat test.txt
Error response from daemon: Container 6cc53beb6bc971df40f50fae8c6b55a0cfcfcfd6aa6fd44f0df607020fd5c918e is not running

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker start cranky_poincare
cranky_poincare

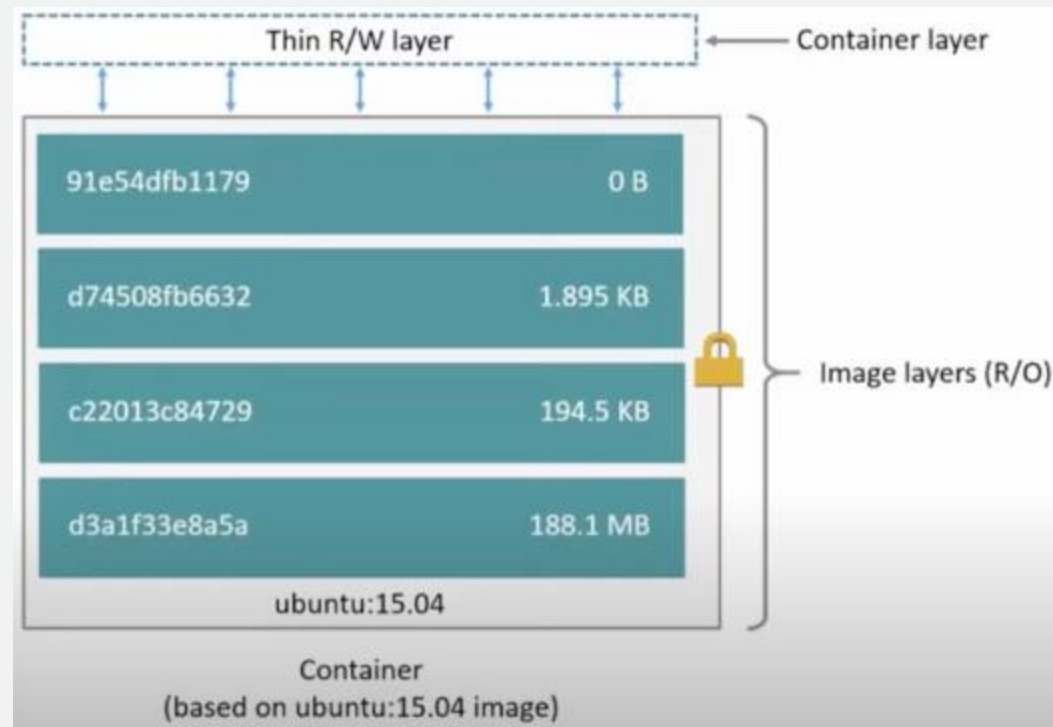
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker exec cranky_poincare cat test.txt
Error response from daemon: Container 6cc53beb6bc971df40f50fae8c6b55a0cfcfcfd6aa6fd44f0df607020fd5c918e is not running

wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker start 8c7
8c7

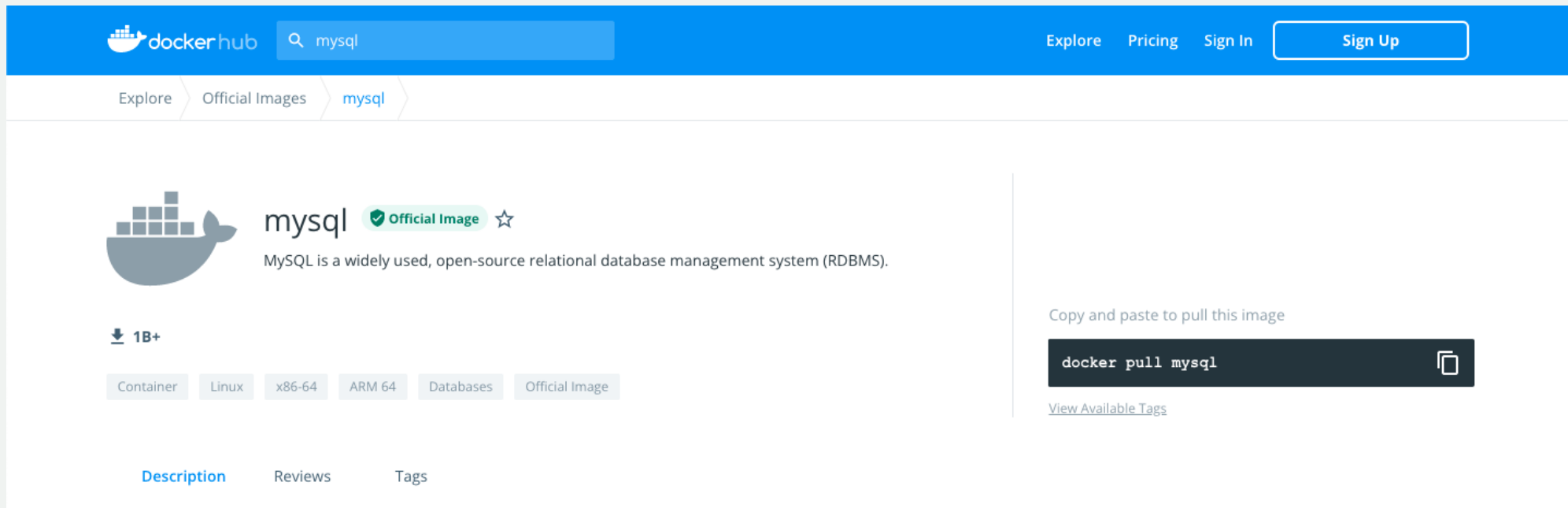
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker exec 8c7 cat test.txt
test.txt: No such file or directory

tekst
```

Obrazy cd.




Docker Hub



The screenshot shows the Docker Hub interface for the MySQL image. At the top is a blue navigation bar with the Docker Hub logo, a search bar containing 'mysql', and links for 'Explore', 'Pricing', 'Sign In', and a 'Sign Up' button. Below the navigation bar is a breadcrumb trail: 'Explore' > 'Official Images' > 'mysql'. The main content area features the MySQL logo, the name 'mysql', a green 'Official Image' badge, and a star icon. A description states: 'MySQL is a widely used, open-source relational database management system (RDBMS)'. Below this, it shows '18+' downloads. A row of tags includes 'Container', 'Linux', 'x86-64', 'ARM 64', 'Databases', and 'Official Image'. At the bottom are tabs for 'Description', 'Reviews', and 'Tags'. On the right side, there is a section titled 'Copy and paste to pull this image' with a dark box containing the command 'docker pull mysql' and a copy icon. Below this is a link to 'View Available Tags'.

dockerhub Explore Pricing Sign In Sign Up

Explore > Official Images > mysql

 **mysql** Official Image ☆

MySQL is a widely used, open-source relational database management system (RDBMS).

↓ 18+


Container Linux x86-64 ARM 64 Databases Official Image

Description Reviews Tags

Copy and paste to pull this image

```
docker pull mysql
```

[View Available Tags](#)

 `docker push repoName`

`docker tag oldName newName`

Dockerfile

Plik **Dockerfile** to zbiór instrukcji, gdzie zwykle każda z instrukcji dodaje kolejną warstwę do finalnego obrazu. Z reguły w pierwszej warstwie obrazu znajdują się pliki systemowe. Kolejne polecenia takie jak RUN, COPY, ADD tworzą następne warstwy, by finalnie stworzyć kompletny obraz naszej aplikacji.



Przykładowy plik Dockerfile

```
FROM ubuntu
```

```
COPY t1.txt .
```

```
RUN apt-get update
```

```
RUN apt-get install --yes vim
```



```
wojciechszlosek@MacBook-Air-Wojciech ~ % cd Desktop/DockerPr
wojciechszlosek@MacBook-Air-Wojciech DockerPr % echo "ttt" > t1.txt
wojciechszlosek@MacBook-Air-Wojciech DockerPr % vim Dockerfile
wojciechszlosek@MacBook-Air-Wojciech DockerPr % ls
Dockerfile      t1.txt
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker build .
[+] Building 19.7s (9/9) FINISHED
=> [internal] load build definition from Dockerfile                                0.1s
=> => transferring dockerfile: 119B                                              0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest                 0.0s
=> [internal] load build context                                                 0.1s
=> => transferring context: 37B                                                 0.0s
=> [1/4] FROM docker.io/library/ubuntu:nagical_hamilton ubuntu                 0.0s
=> [2/4] COPY t1.txt .                                                         0.1s
=> [3/4] RUN apt-get update                                                     7.0s
=> [4/4] RUN apt-get install --yes vim                                         10.8s
=> exporting to image                                                         1.5s
=> => exporting layers                                                         1.5s
=> => writing image sha256:949a0bad22a40d42945c945539e25371421647bb976f0 0.0s
```

```
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker run 949a cat t1.txt
ttt /usr/share/man/ja/man1/vim.1.gz (of link group editor)
```




```
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker build --tag mojanazwa .
```

```
[+] Building 0.4s (9/9) FINISHED
```

```
=> [internal] load build definition from Dockerfile
```

```
=> => transferring dockerfile: 36B
```

```
=> [internal] load .dockerignore
```

```
=> => transferring context: 2B
```

```
=> [internal] load metadata for docker.io/library/ubuntu:latest
```

```
=> [1/4] FROM docker.io/library/ubuntu
```

```
=> [internal] load build context
```

```
=> => transferring context: 27B
```

```
=> CACHED [2/4] COPY t1.txt .
```

```
=> CACHED [3/4] RUN apt-get update
```

```
=> CACHED [4/4] RUN apt-get install --yes vim
```

```
=> exporting to image
```

```
=> => exporting layers
```

```
=> writing image sha256:949a0bad22a40d42945c945539e25371421647bb976f02f1d98945ae064feb77
```

```
=> => naming to docker.io/library/mojanazwa
```

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

```
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker tag mojanazwa:latest mojanazwa:v1.0
```

```
wojciechszlosek@MacBook-Air-Wojciech DockerPr % docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
mojanazwa	latest	949a0bad22a4	5 minutes ago	175MB
mojanazwa	v1.0	949a0bad22a4	5 minutes ago	175MB



Konteneryzacja aplikacji konsolowej i webowej

```
wojciechszlosek@MacBook-Air-Wojciech pythonapp % vim req.txt
wojciechszlosek@MacBook-Air-Wojciech pythonapp % vim app.py
wojciechszlosek@MacBook-Air-Wojciech pythonapp % vim Dockerfile
wojciechszlosek@MacBook-Air-Wojciech pythonapp % ls
Dockerfile  app.py  req.txt
wojciechszlosek@MacBook-Air-Wojciech pythonapp % cat req
cat: req: No such file or directory
wojciechszlosek@MacBook-Air-Wojciech pythonapp % cat req.txt
Flask
wojciechszlosek@MacBook-Air-Wojciech pythonapp % cat app.py
from flask import Flask

app = Flask(__name__)

@app.route("/")
def hi():
    return "hello, flask"

if __name__ == '__main__':
    print(f"hello, world! (in console)")

wojciechszlosek@MacBook-Air-Wojciech pythonapp % cat Dockerfile
FROM python:3.8

COPY req.txt .

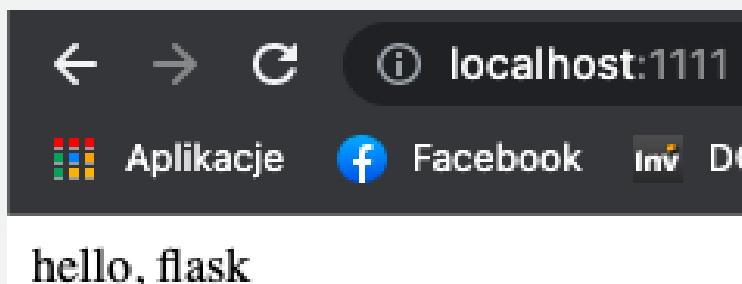
RUN pip install -r req.txt

COPY app.py .

CMD FLASK_APP=app python -m flask run --host=0.0.0.0
```

```
wojciechszlosek@MacBook-Air-Wojciech pythonapp % docker build --tag pyappconsole .
[+] Building 2.7s (10/10) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> == transferring dockerfile: 137B 0.0s
=> [internal] load .dockerignore 0.0s
=> == transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3.8 2.2s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/4] FROM docker.io/library/python:3.8@sha256:53cb5152064a7e7b485ad42704ea63c515 0.0s
=> [internal] load build context 0.0s
=> == transferring context: 241B 0.0s
=> CACHED [2/4] COPY req.txt . 0.0s
=> CACHED [3/4] RUN pip install -r req.txt 0.0s
=> [4/4] COPY app.py . 0.1s
=> exporting to image 0.1s
=> == exporting layers 0.1s
=> == writing image sha256:d1c07e54e51cb59f7367c7746c0f6166b25cf8b3f289bf64fa2895b27 0.0s
=> == naming to docker.io/library/pyappconsole 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
wojciechszlosek@MacBook-Air-Wojciech pythonapp % docker run pyappconsole
hello, world! (in console)
```



```
wojciechszlosek@MacBook-Air-Wojciech pythonapp % vim Dockerfile
wojciechszlosek@MacBook-Air-Wojciech pythonapp % cat Dockerfile
FROM python:3.8

COPY req.txt .

RUN pip install -r req.txt

COPY app.py .

CMD FLASK_APP=app python -m flask run --host=0.0.0.0

wojciechszlosek@MacBook-Air-Wojciech pythonapp % docker build --tag pyappweb .
[+] Building 1.2s (9/9) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> == transferring dockerfile: 174B 0.0s
=> [internal] load .dockerignore 0.0s
=> == transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3.8 0.8s
=> [internal] load build context 0.0s
=> == transferring context: 54B 0.0s
=> [1/4] FROM docker.io/library/python:3.8@sha256:53cb5152064a7e7b485ad42704ea63c515 0.0s
=> CACHED [2/4] COPY req.txt . 0.0s
=> CACHED [3/4] RUN pip install -r req.txt 0.0s
=> CACHED [4/4] COPY app.py . 0.0s
=> exporting to image 0.1s
=> == exporting layers 0.0s
=> == writing image sha256:2f9991980cc89a69233cd61bbd41b29883002139535ebc65b70059d1d 0.0s
=> == naming to docker.io/library/pyappweb 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
wojciechszlosek@MacBook-Air-Wojciech pythonapp % docker run --publish 1111:5000 pyappweb
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
  * Running on http://172.17.0.2:5000/ (Press CTRL+C to quit)
172.17.0.1 - - [10/Mar/2022 14:23:01] "GET / HTTP/1.1" 200 -
```



Kompendium najważniejszych komend

- `docker run`
- `docker create`
- `docker start`
- `docker ps -a`
- `docker images`
- `docker rm`
- `docker system prune`



Najważniejsze komendy cd.

- `docker logs`
- `docker stop`
- `docker kill`
- `docker exec`
- `docker build`



Bibliografia

- <https://www.docker.com/> (w tym grafika)
- <https://www.ibm.com/pl-pl/cloud/learn/docker>
- <https://sii.pl/blog/docker-dla-programistow-co-to-jest/>
- <https://frontstack.pl/docker-komendy/>
- <https://docs.docker.com/get-started/overview/>
- <https://www.youtube.com/playlist?list=PLkcy-k498-V5AmftzfqipMF2LFqSHK5n>



A blue cartoon whale is shown from the side, facing left. On its back, there is a keyboard layout with two rows of keys. Each key is represented by a blue square with three vertical white lines. The whale has a small white eye with a blue pupil and a simple curved line for a mouth. A large yellow rectangular box is superimposed over the whale's body, containing the text "Dzięki za uwagę!".

Dzięki za uwagę!

