



# SPRAWOZDANIE

PROGRAMOWANIE W CHMURZE OBLICZENIOWEJ

**IMIĘ I NAZWISKO:** Piotr Czajka
**NUMER LABORATORIUM** 3
**GRUPA:** 7.1.2
**Data wykonywania ćwiczenia:** 25.10.2018

# Spis treści

# 1 Cel laboratorium

Celem laboratorium było zapoznanie się z narzędziem docker-compose, cytując z oficjalnej strony, "Narzędzia do definiowania i uruchamiania wielokontenerowych aplikacji Docker"

# 2 Przebieg ćwiczenia

## 2.1 Zadanie pierwsze

Najpierw upewniam się, że docker-compose jest zainstalowany:

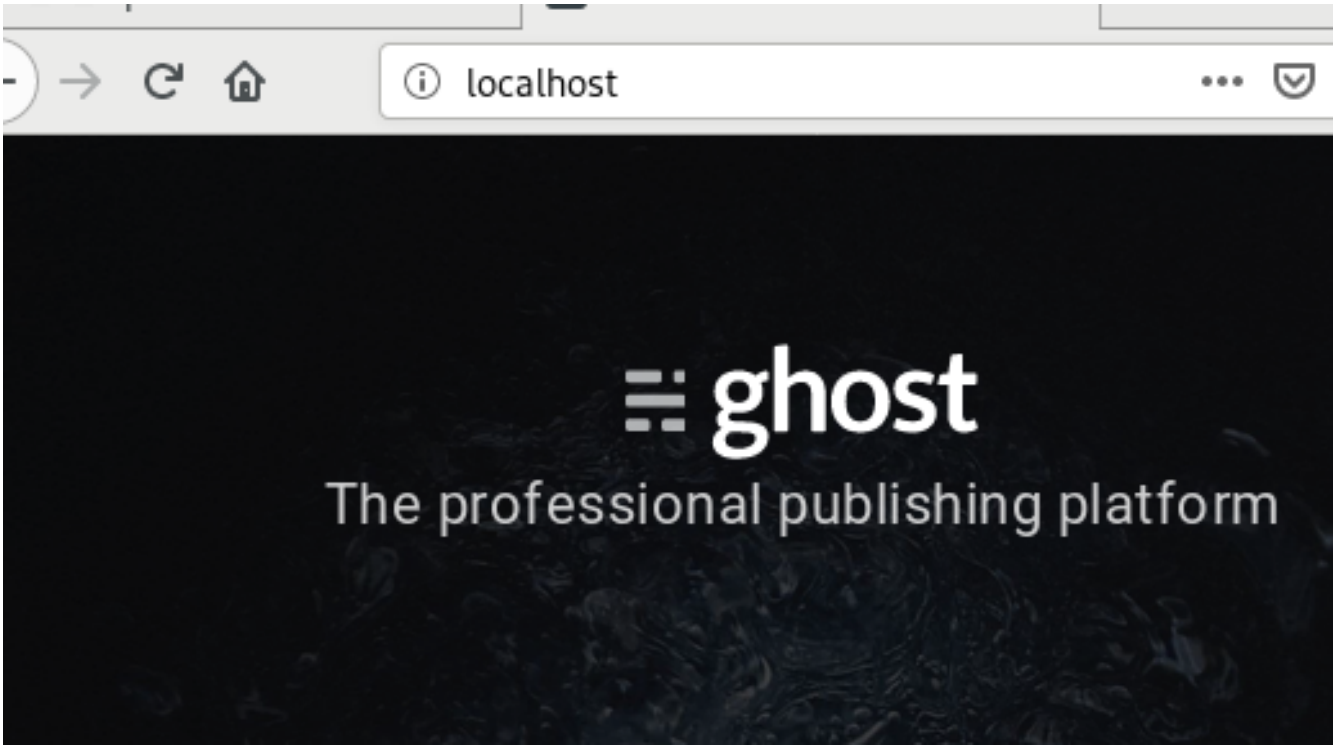
```
22:07:24 # root at Odysseus in [/home/ginkooo/polibudacode/programowanie_w_chmurze_obliczeniowej
/lab5]on git:master ✗
$ docker-compose version
docker-compose version 1.23.1, build unknown
docker-py version: 3.5.1
CPython version: 3.7.1
OpenSSL version: OpenSSL 1.1.1  11 Sep 2018
```

Teraz odpalam kontener z ghostem, aby później skopiować jego domyślną konfigurację na dysk i ją zmodyfikować:

```
22:21:36 # root at Odysseus in [/home/ginkooo/polibudacode/programowanie_w_chmurze_obliczeniowej
/lab5]on git:master ✗
$ docker run -d --name ghost -p 80:2368 ghost
Unable to find image 'ghost:latest' locally
latest: Pulling from library/ghost
a92a4af0fb9c: Pull complete
ca548e841944: Pull complete
459276e74726: Pull complete
cadbcc227129: Pull complete
d08ab9ec12a3: Pull complete
c51cc88369b0: Pull complete
828cc9bcf1a1: Pull complete
a956cbee7847: Pull complete
cdb20653067f: Pull complete
0790ab4da25e: Pull complete
Digest: sha256:11adb93c4ea109e38f95f050d6fb8c520fa10093cd0c1f7dc05b1e18d8e6af4b
Status: Downloaded newer image for ghost:latest
340e1eafcae7ccc86bcfe44f9b8a482ff6a10b8f7a29b270cb5bb28ed07dc959
22:28:52 # root at Odysseus in [/home/ginkooo/polibudacode/programowanie_w_chmurze_obliczeniowej
/lab5]on git:master ✗
$ docker cp -L ghost:/var/lib/ghost/config.production.json .
```

## 2.2 Zadanie drugie

Jak widać serwer działa, na razie korzysta z sqlite jako bazy danych:




Tak zmodyfikowałem config.production.json, aby korzystał z bazy mysql: (zmiana dotyczyła sekcji “database”)

```
{
  "url": "http://localhost:2368",
  "server": {
    "port": 2368,
    "host": "0.0.0.0"
  },
  "database": {
    "client": "mysql",
    "connection": {
            "host":     "mysql",
            "user":     "ghost",
            "password": "password",
            "database": "ghost",
            "charset":  "utf8"
    }
  },
  "mail": {
    "transport": "Direct"
  },
  "logging": {
    "transports": [
      "file",
      "stdout"
    ]
  },
  "process": "systemd",
  "paths": {
    "contentPath": "/var/lib/ghost/content"
  }
}
```

Utworzyłem taki oto Dockerfile, aby przy buildzie kopiował zmodyfikowany config do odpowiedniego katalogu:

```
FROM ghost
COPY ./config.production.json /var/lib/ghost/config.production.json
```

Teraz przyszła kolej na stworzenie odpowiedniego docker-compose:

```
version: '3'
services:
        mysql:
                image: mysql:5.7
                container_name: mysql
                ports:
                        - "3306"
                environment:
                        - MYSQL_ROOT_PASSWORD=root
                        - MYSQL_DATABASE=ghost
                        - MYSQL_USER=ghost
                        - MYSQL_PASSWORD=password
        ghost:
                build: .
                container_name: ghost
                depends_on:
                        - mysql
                ports:
                        - "80:2368"
                restart: on-failure
```

Mysql działa domyślnie na porcie 3306, więc otwieramy go, w sekcji enviroment konfigurujemy dane dostępowe do mysql. restart: on-failure ma za zadanie restartować serwer Ghosta, aż nie uda mu się połączyć z mysql, który startuje dość długo. Nie wystaraczy sama dyrektywa depends_on, ponieważ mysql włącza się, ale dopiero jakiś czas po włączeniu serwer jest dostępny, by się z nim łączyć.