

- Контейнеризация
  - Урок 4. Dockerfile и слои
    - Задания
    - Решение
      - 1. Задача
      - 2. Задача

# Контейнеризация

---

## Урок 4. Dockerfile и слои

---

### Задания

1. Повторить то, что было на семинаре
2. \*\* Самостоятельно на основе голой ОС (alpine, ubuntu, debian) сделать образ MariaDB (создать Dockerfile) и подключиться к контейнеру пхпмайадмином.

### Решение

#### 1. Задача

Создаем *Dockerfile* в папке проекта *nginx*

```
vim Dockerfile
```

```
FROM ubuntu:22.10
RUN apt update && apt install nginx -y
EXPOSE 80
WORKDIR /var/www
CMD ["nginx", "-g", "daemon off;"]
```

Запускаем в папке проекта сборку образа *gb-nginx*

```
docker build -t gb-nginx .
```

Запускаем контейнер на основе образа *gb-nginx* с пробросом порта 80 контейнера на порт 8081 реальной машины

```
docker run -d -p 8081:80 gb-nginx
```

Проверку работы веб-сервера выполним с помощью утилиты *curl*

```
curl http://localhost:8081
```

```
root@fen:/mnt/backup/github/docker/04.Tasks/nginx# vim Dockerfile
root@fen:/mnt/backup/github/docker/04.Tasks/nginx# docker build -t gb-nginx .
[+] Building 0.6s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 158B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/ubuntu:22.10
=> [1/3] FROM docker.io/library/ubuntu:22.10@sha256:e322f4808315c387868a9135beeb11435b5b83130a8599fd7d0014452c34f489
=> CACHED [2/3] RUN apt update && apt install nginx -y
=> CACHED [3/3] WORKDIR /var/www
=> exporting to image
=> => exporting layers
=> => writing image sha256:901a192c33519b616c2ba6b3f4c923adf78119686b8bbd63ba22679506c6cab7
=> => naming to docker.io/library/gb-nginx
root@fen:/mnt/backup/github/docker/04.Tasks/nginx# docker run -d -p 8081:80 gb-nginx
b679891clee58348842fe448f863480b7222d9e0531f8058a0db0e6e2326f567
root@fen:/mnt/backup/github/docker/04.Tasks/nginx# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
b679891clee5   gb-nginx   "nginx -g 'daemon of..." 12 seconds ago Up 10 seconds   0.0.0.0:8081->80/tcp, :::8081->80/tcp   confident_knuth
root@fen:/mnt/backup/github/docker/04.Tasks/nginx# curl http://localhost:8081
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color:scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@fen:/mnt/backup/github/docker/04.Tasks/nginx#
```

Виден ответ веб-сервера с *html* страницей *nginx*

## 2. Задача

Для реализации проекта создам образ включающий сразу три службы:

- mariadb
- php-fpm
- nginx

На веб-сервере будет расположена система управления БД *phpmyadmin*

В папке проекта *maria* для веб-сервера создадим конфиг, включающий работу через *php-fpm*

```
vim nginx.conf
```

```
server {
    listen 80;
    server_name _;

    root /var/www/html/phpmyadmin;

    index index.php index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }

    location ~ \.php$ {
        include snippets/fastcgi-php.conf;
        fastcgi_pass unix:/run/php/php8.1-fpm.sock;
    }
}
```

В той же папке создадим конфиг для системы *phpmyadmin*

```
vim config.inc.php
```

```
<?php
// Настройки аутентификации
$cfg['Servers'][1]['auth_type'] = 'cookie';
$cfg['Servers'][1]['user'] = 'phpmyadmin';
$cfg['Servers'][1]['password'] = 'phpmyadmin';

// Настройки базы данных
$cfg['Servers'][1]['host'] = 'localhost';
$cfg['Servers'][1]['connect_type'] = 'tcp';
$cfg['Servers'][1]['compress'] = false;
$cfg['Servers'][1]['AllowNoPassword'] = false;

// Настройки безопасности
$cfg['blowfish_secret'] = 'secret'; // Секретная фраза для шифрования cookie
$cfg['UploadDir'] = '';
$cfg['SaveDir'] = '';
?>
```

Третьим файлом в папке проекта будет *Dockerfile*

```
vim Dockerfile
```

```
FROM ubuntu:22.04

# Обновляем пакеты и устанавливаем необходимые зависимости
RUN export DEBIAN_FRONTEND=noninteractive && \
    apt-get update && apt-get install -y \
    nginx php-fpm php-mysql php-mbstring php-xml php-zip php-curl \
    mariadb-server mariadb-client wget unzip && \
    rm -rf /var/lib/apt/lists/*

# Копируем конфигурационный файл nginx в контейнер
COPY nginx.conf /etc/nginx/sites-available/default

# Скачиваем и распаковываем архив с phpMyAdmin в папку /var/www/html
RUN wget https://files.phpmyadmin.net/phpMyAdmin/5.1.1/phpMyAdmin-5.1.1-all-languages.zip -O /tmp/phpmyadmin.zip && \
    unzip -q /tmp/phpmyadmin.zip -d /var/www/html && \
    mv /var/www/html/phpMyAdmin-5.1.1-all-languages /var/www/html/phpmyadmin && \
    rm /tmp/phpmyadmin.zip

# Копируем конфигурационный файл phpMyAdmin в контейнер
COPY config.inc.php /var/www/html/phpmyadmin/config.inc.php

# Создаем базу данных и пользователя для phpMyAdmin
RUN service mariadb start && \
    mysql -e "CREATE DATABASE gb;" && \
    mysql -e "CREATE USER 'geekbrains'@'localhost' IDENTIFIED BY 'test1234';" && \
    mysql -e "GRANT ALL PRIVILEGES ON gb.* TO 'geekbrains'@'localhost';" && \
    mysql -e "FLUSH PRIVILEGES;" && \
    service mariadb stop

# Открываем 80 порт для внешнего доступа
EXPOSE 80

# Запускаем сервисы nginx, php-fpm и mysql при старте контейнера
CMD service mariadb start && service php8.1-fpm start && nginx -g "daemon off;"
```

## Сборка образа *gb-maria*

```
docker build -t gb-maria .
```

Запуск контейнера на основе образа *gb-maria* с пробросом порта 80 контейнера на порт 8080 хостовой ОС

```
docker run -d -p 8080:80 gb-maria
```

Заходим с помощью браузера, подключившись к хостовому порту 8080

```

root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# ls
config.inc.php Dockerfile nginx.conf
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# docker build -t gb-maria .
[+] Building 1.2s (11/11) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 1.71kB
=> [internal] load metadata for docker.io/library/ubuntu:22.04
=> [1/6] FROM docker.io/library/ubuntu:22.04@sha256:aabed3296a3d45cede1dc866a24476c4d7e093aa806263c27ddaadbdc3c1054
=> [internal] load build context
=> => transferring context: 66B
=> CACHED [2/6] RUN export DEBIAN_FRONTEND=noninteractive && apt-get update && apt-get install -y nginx php-fpm php-mysql php-mbstring php-xml php-zip php-curl
=> CACHED [3/6] COPY nginx.conf /etc/nginx/sites-available/default
=> CACHED [4/6] RUN wget https://files.phpmyadmin.net/phpMyAdmin-5.1.1/phpMyAdmin-5.1.1-all-languages.zip -O /tmp/phpmyadmin.zip && unzip -q /tmp/phpmyadmin.zip -d /var/www
=> CACHED [5/6] COPY config.inc.php /var/www/html/phpmyadmin/config.inc.php
=> CACHED [6/6] RUN service mariadb start && mysql -e "CREATE DATABASE gb;" && mysql -e "CREATE USER 'geekbrains'@'localhost' IDENTIFIED BY 'test1234';" && mysql -e "GRANT
=> exporting to image
=> exporting layers
=> writing image sha256:dbc9cac8dd66012b34b25378a7f1f42015b138f7adc5d1a3ee4a7b3307fa1725
=> naming to docker.io/library/gb-maria
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# docker run -d -p 8080:80 gb-maria
a6d86332a81fb013d365c815a0347c3f3801232c2f257d8fdea62ecbc633b7e0
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED      STATUS      PORTS                               NAMES
a6d86332a81f   gb-maria   "/bin/sh -c 'service..." 5 minutes ago Up 5 minutes   0.0.0.0:8080->80/tcp   pedantic_meninsky
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# ip a | grep enp2s0
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    inet 192.168.1.8/24 brd 192.168.1.255 scope global noprefixroute enp2s0
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb# ss -tulpn | grep 8080
tcp        LISTEN    0          4096      0.0.0.0:*        users:((("docker-proxy",pid=98945,fd=4))
tcp        LISTEN    0          4096      [::]:8080       [::]:*        users:((("docker-proxy",pid=98952,fd=4))
root@fen:/mnt/backup/github/docker/04.Tasks/mariadb#

```

После успешной авторизации создаем таблицу *students* в базе *gb* и заносим данные.

The screenshot shows the phpMyAdmin 5.1.1 interface in a Chromium browser. The address bar shows the URL: 192.168.1.8:8080 / localhost / gb / students | phpMyAdmin 5.1.1 - Chromium. The browser's security indicator shows "Не защищено" (Not secure). The interface includes a sidebar with a tree view of databases (gb, Новая, students, Индексы, Столбцы, information\_schema) and a main panel. The main panel displays the "students" table structure and data. A yellow banner at the top indicates "Отображение строк 0 - 0 (1 всего, Запрос занял 0.0005 сек.)". Below this, the SQL query "SELECT \* FROM `students`" is shown. The table data is displayed with columns: id, name, surname, group\_id. One record is visible: id 1, name Ростислав, surname Ромашин, group\_id 4992. The interface also includes various action buttons like "Обзор", "Структура", "SQL", "Поиск", "Вставить", "Экспорт", "Импорт", "Операции", and "Триггеры".