

Веб-сайты на PHP, использующие контейнеры Docker с PHP Apache и MySQL

18 июня 2021 г.

Темы: [Контейнеры](#)

[Контейнерная технология](#) развивается с каждым днем. Это технология, которая делает разработку приложений намного проще и быстрее. Она имеет чистую архитектуру, которая гарантирует, что службы приложений устойчиво используют ресурсы, разделяя приложение на более мелкие службы, называемые изображениями. Это позволяет настраивать каждую службу независимо, не влияя на работу других служб.

В этом случае [Docker](#) предоставляет файл [docker-compose](#), который позволяет вам настроить все среды вашего приложения и выполнить несколько команд для полной настройки вашего приложения более элегантным и быстрым способом.

Давайте рассмотрим случай запуска PHP-приложения. Вам придется установить все среды, необходимые для запуска PHP-скриптов. На вашем сервере / системе должен быть установлен сервер apache и, возможно, база данных MySQL. Затем настройте

С Docker все гораздо более управляемо. Docker позволяет вам настроить ваше приложение так, чтобы каждая служба работала как микросервис. Таким образом, вы устанавливаете единый файл YML, который изолирует все службы, необходимые вашему приложению для запуска. Файл настраивает для вас сервер PHP Apache и базу данных MySQL. Все, что вам нужно, это указать параметры, по которым вам нужно запускать ваше приложение.

Основное преимущество, которое предоставляет Containers, - это масштабируемая среда для запуска ваших прикладных сервисов. Это гарантирует, что конвейеры непрерывной интеграции и непрерывной доставки (CI / CD) совершенствуются во всей команде. Таким образом, вам нужно всего лишь предоставить общий доступ к этому YML-файлу каждому члену команды. Это установит все необходимые среды для всей команды, независимо от операционной системы, на которой они работают. Таким образом, члены команды могут синхронизировать свою работу, не нарушая код.

Это руководство покажет вам, как мы можем использовать среду разработки Docker для:

Настройте и запустите локальный экземпляр сервера PHP Apache.

Обслуживать динамический веб-сайт на основе PHP.

Настройка базы данных MySQL для запуска SQL-скриптов, извлечения записей и их печати на веб-сайте, управляемом PHP.

Предварительные требования

Убедитесь, что на вашем компьютере установлены [демоны Docker](#).

Базовые знания [PHP](#) и [SQL-запросов](#).

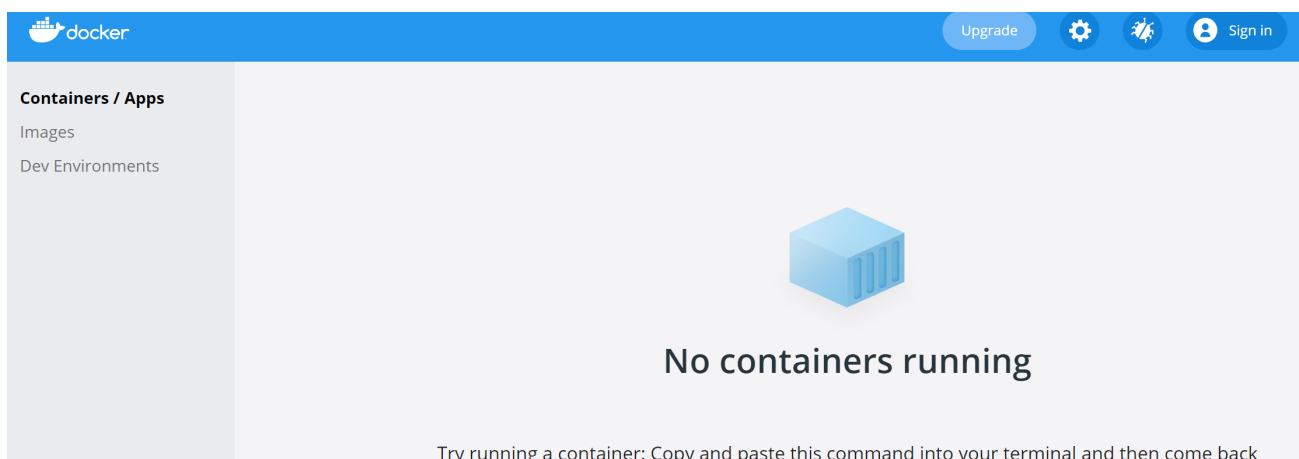
Фундаментальное понимание того, как [создавать и запускать образы Docker hub](#) из файла Docker.

Поймите, как работают [контейнеры](#).

Базовые знания о том, как выполнять [команды Docker и docker-compose](#).

Настройка и тестирование, запущен ли Docker

После загрузки и установки Docker demon откройте движок Docker engine и убедитесь, что он запущен.



Компьютер:

```
docker version
```

При этом будут регистрироваться результаты, почти аналогичные этим журналам командной строки.

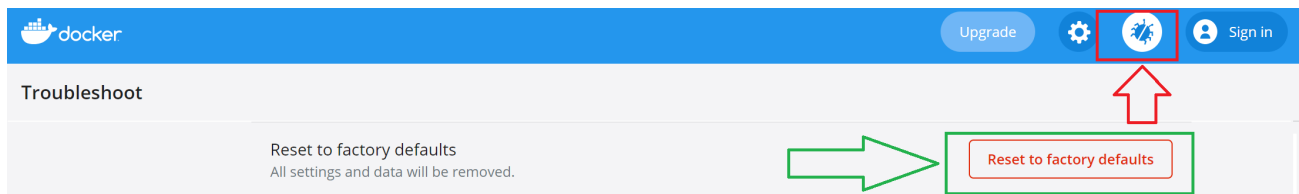
```
version: 20.10.6
API version: 1.41
Go version: go1.16.3
Git commit: 370c289
Built: Fri Apr 9 22:49:36 2021
OS/Arch: windows/amd64
Context: default
Experimental: true

Server: Docker Engine - Community
Engine:
  Version: 20.10.6
  API version: 1.41 (minimum version 1.12)
  Go version: go1.13.15
  Git commit: 8728dd2
  Built: Fri Apr 9 22:44:56 2021
  OS/Arch: linux/amd64
  Experimental: false
containerd:
  Version: 1.4.4
  GitCommit: 05f951a3781f4f2c1911b05e61c160e9c30eaa8e
runc:
  Version: 1.0.0-rc93
  GitCommit: 12644e614e25b05da6fd08a38ffa0cfe1903fdec
docker-init:
  Version: 0.19.0
  GitCommit: de40ad0
```

Если вы новичок в Docker, вы можете столкнуться с этой ошибкой Docker engine при выполнении приведенной выше команды.

```
error during connect: This error may indicate that the docker daemon is not running.: Get
"http://%2F%2F.%2Fpipe%2Fdocker_engine/v1.24/version": open //./pipe/docker_engine: The
system cannot find the file specified.
```

настройка не уместна.



Запустите `docker version` команду, и теперь она должна работать нормально. И если проблема не устраняется, выполните поиск и найдите необходимые способы правильной настройки вашего движка Docker.

Мы начинаем с нуля; убедитесь, что у вас нет контейнеров и изображений, запущенных в вашем движке Docker. Запустите `docker container ls`, чтобы проверить любой доступный контейнер. Чтобы удалить контейнер Docker, запустите `docker container rm <container's name>`. И убедитесь, что контейнер не запущен.

```
C:\Users\User>docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

Установка файла `docker-compose` YML

Docker-compose позволяет вам устанавливать параметры необходимых изображений, которые вы хотите запустить в своем приложении. В нашем случае мы будем использовать

Продолжайте и создайте папку проекта и `.yaml` файл внутри этой папки.

Например, `docker-compose.yaml`.

Чтобы настроить `docker-compose`, вам сначала нужно выбрать **версию** Docker, которую вы хотите использовать, сервисы, которые вы хотите предоставлять, и контейнеры, которые вы хотите запускать.

```
version: '3.8'
services:
  php-apache-environment:
    container_name:
```

Настройка и запуск локального экземпляра сервера PHP Apache

Чтобы настроить контейнер PHP Apache, вам необходимо указать следующие среды,

Имя контейнера - это просто случайное имя, которым вы хотели бы назвать свой PHP-контейнер.

Например `container_name: php-apache`.

используем данные image - php70-apache из Docker-хаб.

Том - это настроит ваш текущий рабочий `src` каталог для вашего кода / исходных файлов. Если бы вы запускали PHP-скрипт, этот файл должен был бы находиться в этом каталоге.

Такие как:

```
volumes:  
- ./php/src:/var/www/html/
```

Номера портов. Здесь определяются порты, с которых будет запускаться скрипт. Он настроит сопоставление порта сервера Apache порту вашего локального компьютера.

Например:

```
ports:  
- 8000:80
```

Это означает, что мы настраиваем сервер Apache для предоставления доступа к порту 80. Порт 8000 подключается к PHP-скриптам и выполняет их в браузере из контейнеров Docker.

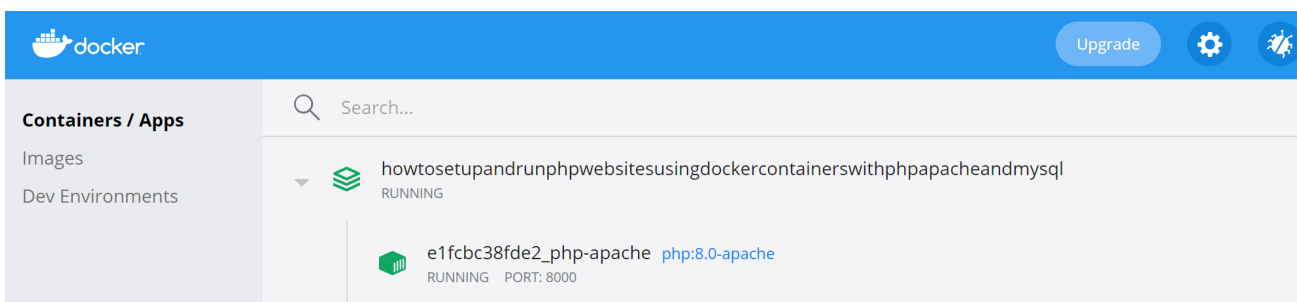
Вот как вы `docker-compose.yml` должны выглядеть.


```
php-apache-environment:  
  container_name: php-apache  
  image: php:8.0-apache  
  volumes:  
    - ./php/src:/var/www/html/  
  ports:  
    - 8000:80
```

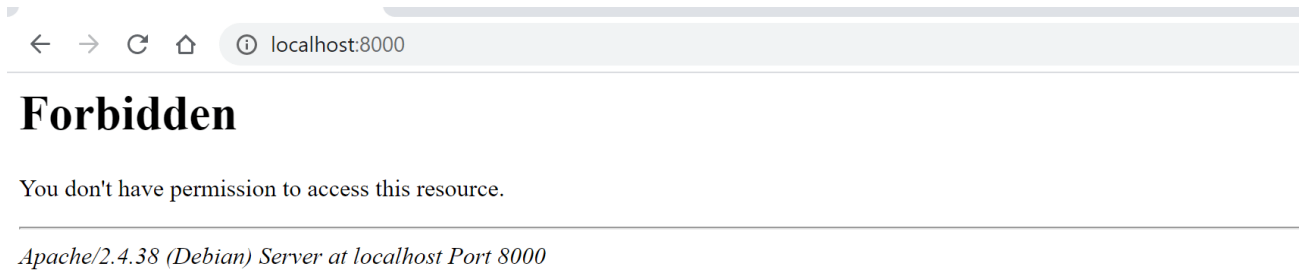
Давайте протестируем это. Продолжайте и запустите `docker-compose up`. Это позволит получить всю информацию, загрузить сервер Apache, создать образ и запустить контейнер.

```
Starting e1fcbc38fde2_php-apache ... done  
Attaching to e1fcbc38fde2_php-apache  
e1fcbc38fde2_php-apache | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using  
172.18.0.2. Set the 'ServerName' directive globally to suppress this message  
e1fcbc38fde2_php-apache | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using  
172.18.0.2. Set the 'ServerName' directive globally to suppress this message  
e1fcbc38fde2_php-apache | [Sat May 22 08:19:21.947726 2021] [mpm_prefork:notice] [pid 1] AH00163: Apache/2.4.38 (Debian)  
PHP/8.0.6 configured -- resuming normal operations  
e1fcbc38fde2_php-apache | [Sat May 22 08:19:21.947803 2021] [core:notice] [pid 1] AH00094: Command line: 'apache2 -D FORE  
GROUND'
```

Если вы откроете движок Docker desktop engine, контейнер должен быть запущен.



браузер, например, localhost:8000.



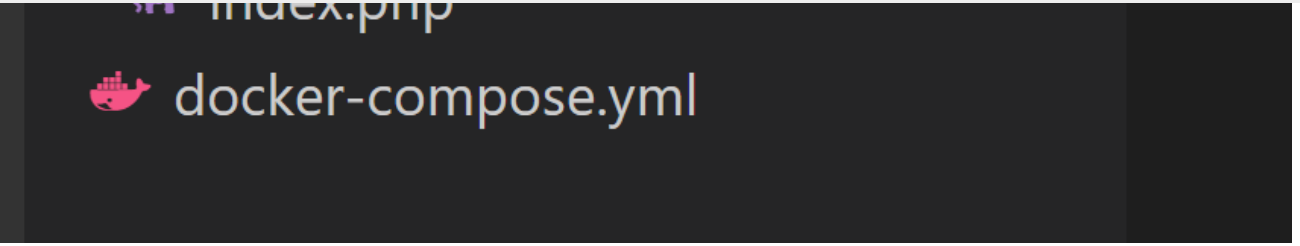
Это показывает, что контейнер настроен на запуск некоторого кода, управляемого PHP.

Обслуживать динамический веб-сайт на основе PHP

Теперь давайте выполним некоторый PHP-код и получим выходные данные в браузере. Вы будете выполнять скрипты из каталога, который вы определили в томах вашего docker-compose.

В данном случае мы используем `./php/src`.

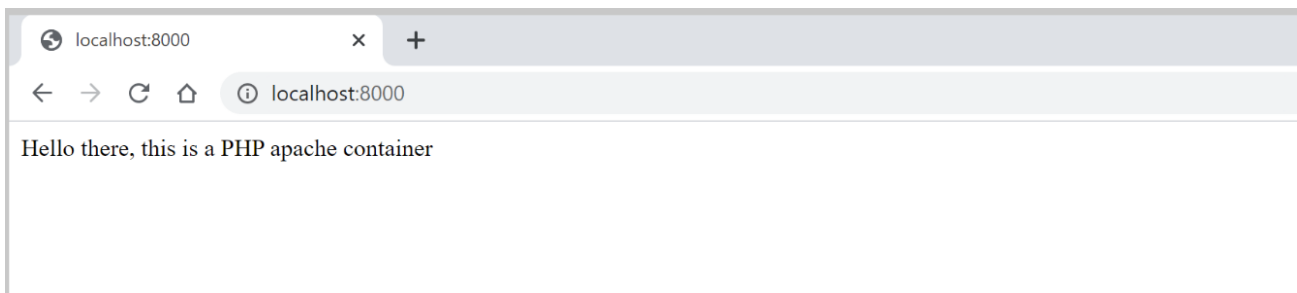
Перейдите в каталог вашего проекта → `./php/src`, создайте `index.php` файл и начните писать свои PHP-скрипты.



Простой `index.php` скрипт.

```
<?php
echo "Hello there, this is a PHP Apache container";
?>
```

Обновите свой браузер (`http://localhost:8000/`), и результаты работы этого простого веб-сайта на PHP Drive должны быть видны.



Поздравляем! Теперь у вас есть контейнерный веб-сайт на PHP.

Настройка контейнера базы данных MySQL

Давайте добавим службу MySQL в `docker-compose.yml` файл. Для настройки MySQL нам нужно настроить некоторую среду, такую как:

Аутентификация паролем. Чтобы использовать сервер MySQL и получить к нему доступ, вам необходимо установить среды аутентификации, которые позволят вам получить доступ к определенному серверу MySQL и его службам, таким как база данных. Мы будем использовать `MYSQL_USER: MYSQL_USER` и `MYSQL_PASSWORD: MYSQL_PASSWORD` для подключения к MySQL и доступа к `MYSQL_DATABASE: MYSQL_DATABASE`.

Политика перезапуска, установленная в `restart: always`. Это перезапускает службу всякий раз, когда изменяется какая-либо определенная конфигурация.

```
db:
  container_name: db
  image: mysql
  restart: always
  environment:
    MYSQL_ROOT_PASSWORD: MYSQL_ROOT_PASSWORD
    MYSQL_DATABASE: MY_DATABASE
    MYSQL_USER: MYSQL_USER
    MYSQL_PASSWORD: MYSQL_PASSWORD
  ports:
    - "9906:3306"
```

Внутри каталога вашего проекта перейдите в /php папку, создайте файл Docker, назовите его Dockerfile и добавьте следующие конфигурации PHP.

```
FROM php:8.0-apache
RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli
RUN apt-get update && apt-get upgrade -y
```

Здесь мы создали пользовательский образ PHPApache и среду, в которой будет установлен mysqli, расширение PHP, которое подключит PHPApache к серверу MySQL.

Теперь нам нужно создать это пользовательское изображение внутри службы php-apache в docker-compose.yml файле. PHPApache также зависит от db службы для подключения к MySQL. Нам нужно настроить ее, указав depends_on: среду.

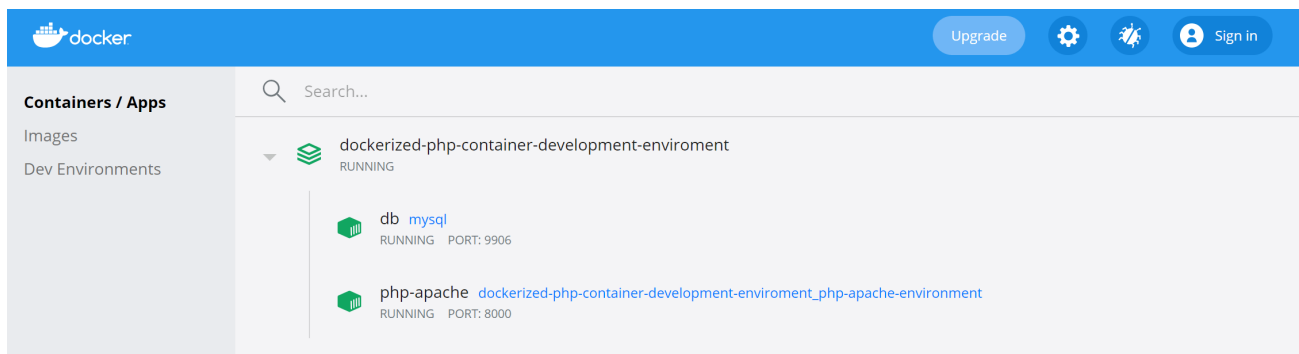
Вот как должен выглядеть ваш docker-compose.yml файл.

```
version: '3.8'
services:
  php-apache-environment:
    container_name: php-apache
    build:
      context: ./php
      dockerfile: Dockerfile
```

```
- ./php/src:/var/www/html/
ports:
  - 8000:80

db:
  container_name: db
  image: mysql
  restart: always
  environment:
    MYSQL_ROOT_PASSWORD: MYSQL_ROOT_PASSWORD
    MYSQL_DATABASE: MYSQL_DATABASE
    MYSQL_USER: MYSQL_USER
    MYSQL_PASSWORD: MYSQL_PASSWORD
  ports:
    - "9906:3306"
```

Запустите `docker-compose up`, чтобы извлечь и настроить среду MySQL. MySQL будет добавлен в контейнер.



Запуск SQL-запроса с использованием PHP-скриптов

Давайте проверим, работает ли контейнер должным образом. Перейдите к `index.php` файлу и следующему коду подключения PHP MySQL.

```
// The MySQL service named in the docker-compose.yml.
$host = 'db';

// Database use name
$user = 'MYSQL_USER';

//database user password
$pass = 'MYSQL_PASSWORD';

// check the MySQL connection status
$conn = new mysqli($host, $user, $pass);
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
} else {
    echo "Connected to MySQL server successfully!";
}
?>
```

Сохраните файл и обновите свой `http://localhost:8000/` веб-адрес.



Connected to MySQL server successfully!

Бум. Вот и все. Среды PHP Apache и MySQL теперь настроены, и вы можете приступить к разработке своего приложения на базе PHP и взаимодействовать с сервером MySQL.

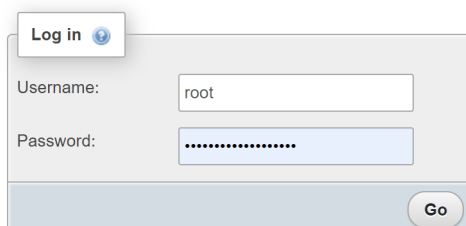
Теперь подключение к MySQL в порядке. Давайте посмотрим, как мы можем извлекать некоторые данные из базы данных MySQL и отображать их на веб-странице с помощью PHP-скриптов.

Предположим, что ваше приложение взаимодействует с базой данных; вероятно, вам нужен интерфейс для взаимодействия с вашими данными. Мы добавим службы phpMyAdmin, которые предоставят нам интерфейс для взаимодействия с базой данных MySQL.

Давайте добавим службу phpMyAdmin, как показано ниже.

```
phpmyadmin:
  image: phpmyadmin/phpmyadmin
  ports:
    - '8080:80'
  restart: always
  environment:
    PMA_HOST: db
  depends_on:
    - db
```

Откройте <http://localhost:8080/> в браузере для доступа к phpMyAdmin.



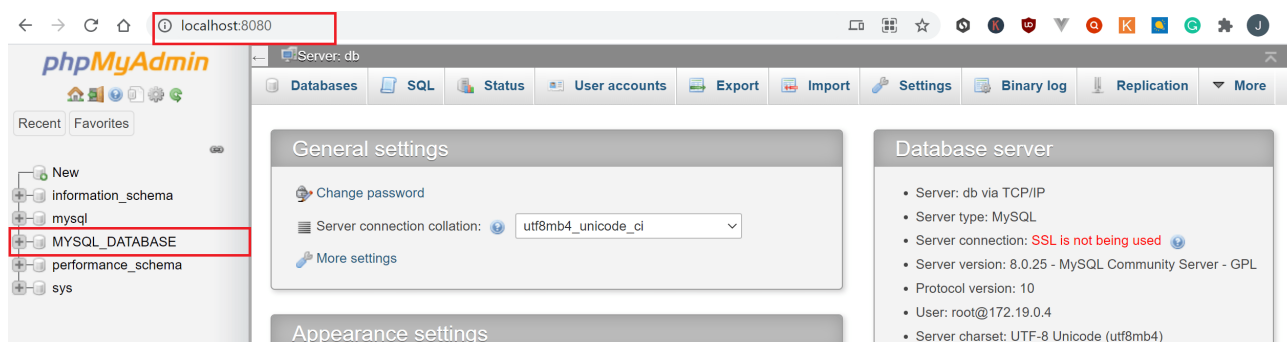
Log in ⓘ

Username:

Password:

Go

Для входа в панель Phpmysql используйте имя пользователя как root и пароль как MYSQL_ROOT_PASSWORD. Пароль уже был установлен в переменных среды MySQL (MYSQL_ROOT_PASSWORD: MYSQL_ROOT_PASSWORD)



Теперь вы можете видеть, что база данных, которую мы определили, уже установлена как MYSQL_DATABASE, и вы можете начать взаимодействовать с Phpmysql.

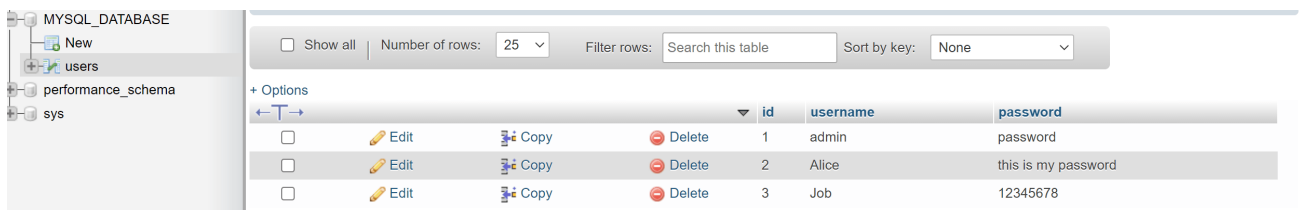
Извлекайте записи и распечатывайте их на веб-сайте, управляемом PHP

Создайте таблицу базы данных и заполните некоторые записи. Выберите базу данных и выполните следующий запрос.

```

    id int not null auto_increment,
    username text not null,
    password text not null,
    primary key (id)
);
insert into `users` (username, password) values
    ("admin","password"),
    ("Alice","this is my password"),
    ("Job","12345678");

```



	id	username	password
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	admin	password
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	Alice	this is my password
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	Job	12345678

Продолжайте и напишите SQL-запрос select с помощью PHP.

```

<?php
//These are the defined authentication environment in the db se

// The MySQL service named in the docker-compose.yml.
$host = 'db';

// Database use name
$user = 'MYSQL_USER';

//database user password
$pass = 'MYSQL_PASSWORD';

// database name
$mydatabase = 'MYSQL_DATABASE';

```

```
// select query
$sql = 'SELECT * FROM users';

if ($result = $conn->query($sql)) {
    while ($data = $result->fetch_object()) {
        $users[] = $data;
    }
}

foreach ($users as $user) {
    echo "<br>";
    echo $user->username . " " . $user->password;
    echo "<br>";
}
?>
```

Обновите <http://localhost:8000/>, чтобы просмотреть результаты.

admin password

Alice this is my password

Job 12345678

Мы использовали операцию чтения только для демонстрации этого. Продолжайте и попробуйте другие операции CRUD с докеризованным PHP-приложением.

Заключение

Docker - это впечатляющая контейнерная технология с множеством завораживающих преимуществ. Если вы еще не начали изучать Docker, ознакомьтесь с этим [руководством](#) и узнайте, как начать работу с Docker.

Не стесняйтесь ознакомиться с приведенными ниже руководствами и узнать больше о Docker и контейнерах.

Счастливого кодирования!

[Понимание концепций Docker](#)

[Начало работы с Docker](#)

[Краткая история контейнерной технологии](#)

Материалы для рецензирования: [Srishilesh P S](#)



144

26 Comments

Sort By Best ▼

The EngEd community is subject to Section's [moderation policy](#).

Write your comment...

[LOGIN](#) [SIGNUP](#)

ScottTrakker 1 year ago

Thank you for this excellent tutorial!

Two things I would like to add:

1. Use 'docker-compose up -d'. This way it starts detached from the terminal.
2. Make the database persistent by adding the code below. volumes:
 - dbdata:/var/lib/mysql

Instead of docker I used Podman (with podman-compose) and that worked fine!

[Reply](#) [Share](#)

 1  0

Ricardo Lima 1 year ago

Very much helpful, Thanks

[Reply](#) [Share](#)

 1  0

Fridtha 3 months ago

Joseph, you are officially THE MAN! This saved me DAYS of more work on something I had already been beating my head against for days before. Thank you. Thank you. Thank you.

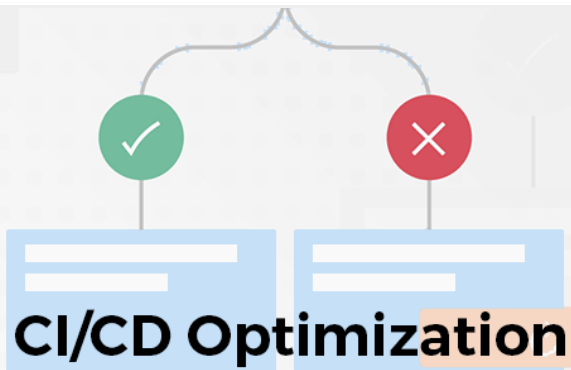
[Reply](#) [Share](#)

 0  0

khw 12 months ago

I usually don't register on a site to leave back, now I do (did)

GITHUB Actions



Containers

Continuous Integration and Deployment Pipelines with Flask, Docker and Github Actions

[Read More](#)

Following command to rebuild docker
docker-compose up --build



Containers, API

DevOps Pipeline Automation with Google Cloud Build and Triggers

[Read More](#)

Connection failed: Access denied for user
'axsodb'@'172.18.0.4' (using password: YES)



Containers

Dockerizing an ASP.NET Core Web API App and SQL Server

[Read More](#)

up --build

[Reply](#) [Share](#)

 0  0

ScottTrakker 1 year ago

I try to run Laravel (PHP framework) with the above setup but when I run 'php artisan migrate' I get the error 'Class "PDO" not found'.

This is docker-compose file that I use:

```
version: 3.4
services:
  php-apache-environment:
    container_name: php-apache
    build:
      context: .
      dockerfile: Dockerfile
    depends_on:
      - db
      - mysql
    volumes:
      - ./www:/var/www/html
    ports:
      - 8000:80
    db:
      container_name: db
      image: mysql
      restart: always
      environment:
        MYSQL_ROOT_PASSWORD: zqw36vuahnrk
        MYSQL_DATABASE: laravel-mvc
        MYSQL_USER: laravel-user
        MYSQL_PASSWORD: dw3kngbtd773
```

- 8080:80 restart: always environment:
PMA_HOST: db depends_on:
- db

With this Dockerfile:

```
FROM php:8.1-apache
RUN docker-php-ext-install pdo pdo_mysql
RUN docker-php-ext-install mysqli && docker-php-
```

Show More

Reply Share

 0  0

Art 1 year ago

try

```
RUN docker-php-ext-install pdo_mysql &&
docker-php-ext-enable pdo_mysql
```

Reply Share

 0  0

Scott Trakker 1 year ago

Thanks, but I already fixed it!

This is now my Dockerfile:

```
FROM php:8.1-apache
RUN apt-get update
RUN apt-get install -y git libzip-dev zip
unzip npm
RUN docker-php-ext-install pdo pdo_mysql
zip
RUN a2enmod rewrite
RUN curl -sS
https://getcomposer.org/installer | php -
- --install-dir=/usr/local/bin --
filename=composer
```

And this is my docker-compose.yml file:

```
version: 3.4
services:
  webserver:
    container_name: webserver
    build:
      context: .
      dockerfile: Dockerfile
```



```
database:
  container_name: database
  image: mysql
  restart: always
  environment:
    MYSQL_ROOT_PASSWORD: [root
password]
    MYSQL_DATABASE: [database name]
    MYSQL_USER: [database user]
    MYSQL_PASSWORD: [password user]
  ports:
    - 9906:3306
  volumes:
    - dbdata:/var/lib/mysql
phpmyadmin:
  container_name: phpmyadmin
  image: phpmyadmin/phpmyadmin
  ports:
```

[Show More](#)[Reply](#) [Share](#) 0  0

r

roughi 1 year ago

so good, thanks! you explained very nicely how to build up the docker-compose file, helped me a lot!

[Reply](#) [Share](#) 0  0

Felipe 1 year ago

For those who are having issue trying to show de data from database, there is a little mistake on tutorial. On line **\$mydatabase = 'MYSQL_DATABASE';** the correct is **\$mydatabase = 'MY_DATABASE';**

Thanks Joseph, really helpful!!

[Reply](#) [Share](#) 0  0

sonali 1 year ago

very helpful article. thanks a lot.

[Reply](#) [Share](#) 0  0

H

Harischandra Matara Kanka 1 year ago

This is and Excellent Work. Very short, Simple, To the Point, Clearly Explained, and NO Errors. Great!

[Reply](#) [Share](#) 0  0



Биография автора EngEd



Джозеф Чеге

Джозеф Чеге - студент бакалавриата по бизнес-информационным технологиям, студент 4 курса Технологического университета Дедана Кимати. Джозеф свободно владеет разработкой мобильных приложений для Android и очень увлечен серверной разработкой.

[Просмотреть полный профиль автора](#) →

Company

About

Careers

Legals

Resources

Blog

Content Library

Engineering Education

Support




Docs

Community Slack

Help & Support

[Contact Us](#)

Section supports many open source projects including:

 [varnish cache
logo](#) [cloud native
computing foundation
logo](#) [the linux
foundation logo](#) [k8s edge
logo](#)