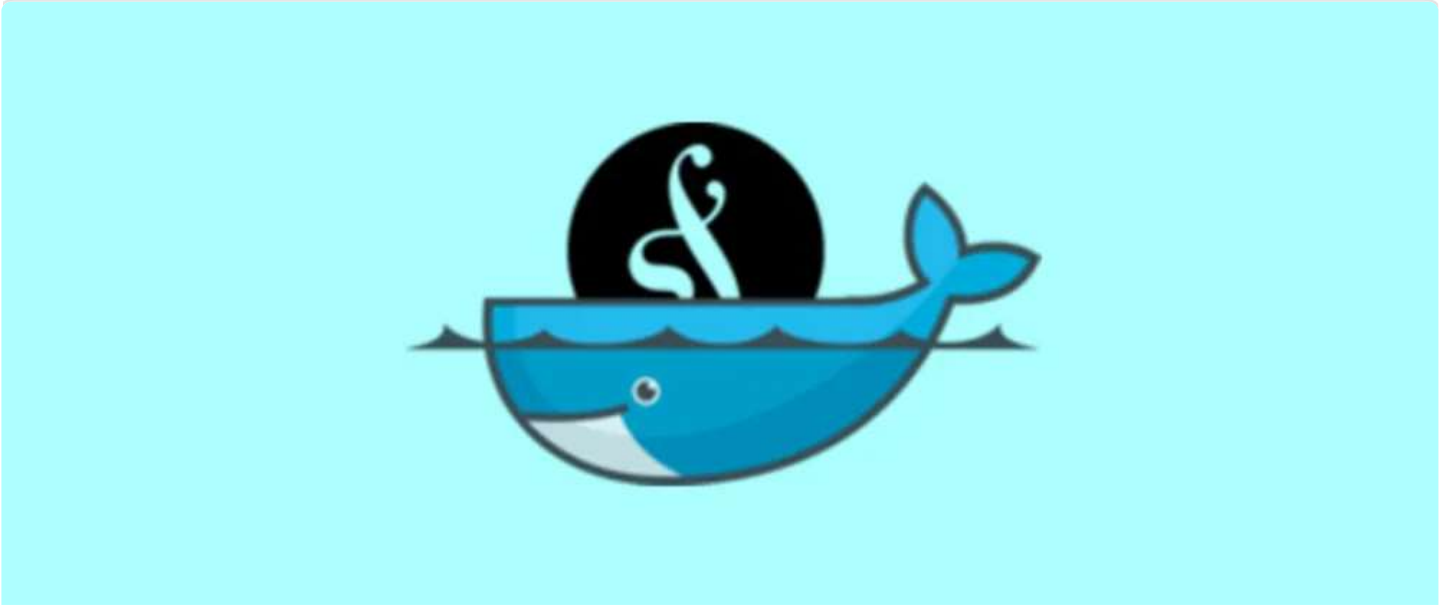


Symfony 5 development with Docker

#php #symfony #docker #container

Martin Pham 30 dic. 2019 · *Updated on 28 ene.* · 4 min read

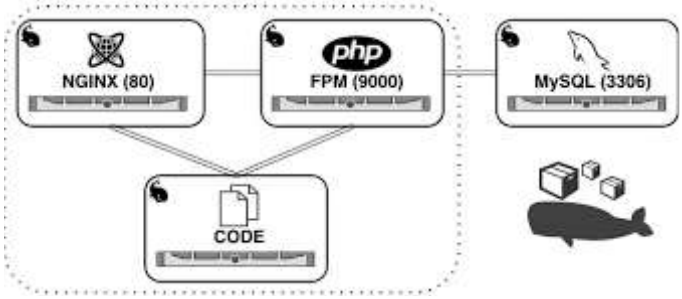
We were playing with Kubernetes last week, however the project was just a small PHP file with phpinfo() function call, no big deal.

Today my colleague asked me to guide him a bit on Docker, because he'd like to try it with a real world example: Developing a Symfony project. So let's take a look at this, it's quick, easy and fun!

Symfony uses Composer to manage its dependencies and scripts, namespaces,.. with a file named composer.json. Dependencies will be download to a directory called **vendor**.

Focus on development, we'd like to create a ready configured & isolated environment, so anyone can clone the repository and run the application easily. So we're gonna use 3 containers:

- MySQL, with mounted volume for data persistent
- PHP-FPM, with mounted volume for application's code
- NGINX, with mounted volumes for configurations, logs, and share mounted volume with PHP-FPM for application's assets



We will also need to use some environment variables for containers' parameters, like database credentials, application secret key,...

We're gonna use Docker-Compose to put configurations and run all containers.

```
Project
│
├── docker-compose.yml
│
├── database/
│   ├── Dockerfile
│   └── data/
│
└── php-fpm/
│   └── Dockerfile
│
├── nginx/
│   ├── Dockerfile
│   └── nginx.conf
│
└── logs/
    └── nginx/
```

MySQL Database

Let's just create a MariaDB container

```
# docker/database/Dockerfile

FROM mariadb:latest
CMD ["mysqld"]
EXPOSE 3306
```

Explaination

- We use MariaDB official image
- Run mysqld to start the server
- Expose port 3306 for database connection

With PHP-FPM container, we'd like to install dependencies and run database migrations at start. So we need to install the PDO MySQL extension, then composer, and then Symfony migration script.

However, it could be a problem if we run the migration before the MySQL server is ready. We will need to handle this also.

With Docker-compose, we can specify a depends_on configuration to tell it wait for another container. But it doesn't mean Docker-compose will wait until the MySQL server is ready, it only waits until the MySQL container is up.

Fortunately, with the help from wait-for-it script, we can try to wait until the MySQL container's port 3306 is Open (Or you can even try to wait until you can connect to the MySQL using credentials).

```
# docker/php-fpm/Dockerfile

FROM php:fpm-alpine
COPY wait-for-it.sh /usr/bin/wait-for-it
RUN chmod +x /usr/bin/wait-for-it
RUN apk --update --no-cache add git
RUN docker-php-ext-install pdo_mysql
COPY --from=composer /usr/bin/composer /usr/bin/composer
WORKDIR /var/www
CMD composer install ; wait-for-it database:3306 -- bin/console doctrine:migrations:mig
EXPOSE 9000
```

Explaination

- We use PHP-FPM offical image
- Copy wait-for-it script into the container
- Allow execution for wait-for-it
- Add git for dependencies installation
- Install PHP PDO MySQL
- Take composer file from Composer official image
- Set working dir to /var/www
- Install dependencies, then wait until the MySQL container is Online to run migration script. Finally, run php-fpm to start the server
- Expose PHP-FPM port (9000)

Okey, this part is a bit complex, we're gonna create the NGINX configuration file, the PHP-FPM proxy, and a separated file for default NGINX site.

First the Dockerfile definition

```
# docker/nginx/Dockerfile

FROM nginx:alpine
WORKDIR /var/www
CMD ["nginx"]
EXPOSE 80
```

Explaination

- As above, we use NGINX official image
- Set working dir to /var/www, the same directory with PHP-FPM since we're gonna share this with a mounted volume
- Start nginx
- Expose the port 80 for web

Now, an NGINX server configuration

```
# docker/nginx/nginx.conf
user  nginx;
worker_processes  4;
daemon off;

error_log  /var/log/nginx/error.log warn;
pid        /var/run/nginx.pid;

events {
    worker_connections  1024;
}


http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;
    access_log  /var/log/nginx/access.log;
    sendfile        on;
    keepalive_timeout  65;

    include /etc/nginx/conf.d/*.conf;
    include /etc/nginx/sites-available/*.conf;
}
```

```
# docker/nginx/conf.d/default.conf

upstream php-upstream {
    server php-fpm:9000;
}
```

And an NGINX site's configuration

```
# docker/nginx/sites/default.conf

server {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;

    server_name localhost;
    root /var/www/public;
    index index.php index.html index.htm;

    location / {
         try_files $uri $uri/ /index.php$is_args$args;
    }

    location ~ \.php$ {
        try_files $uri /index.php =404;
        fastcgi_pass php-upstream;
        fastcgi_index index.php;
        fastcgi_buffers 16 16k;
        fastcgi_buffer_size 32k;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_read_timeout 600;
        include fastcgi_params;
    }

    location ~ /\.ht {
        deny all;
    }
}
```

Docker-Compose configuration

We have 3 container definitions, now we just need to setup a Docker-compose configuration to connect all togethers:

```
# docker/docker-compose.yml
version: '3'
```

```
    build:
      context: ./database
    environment:
      - MYSQL_DATABASE=${DATABASE_NAME}
      - MYSQL_USER=${DATABASE_USER}
      - MYSQL_PASSWORD=${DATABASE_PASSWORD}
      - MYSQL_ROOT_PASSWORD=${DATABASE_ROOT_PASSWORD}
    ports:
      - "3306:3306"
    volumes:
      - ./database/init.sql:/docker-entrypoint-initdb.d/init.sql
      - ./database/data:/var/lib/mysql

  php-fpm:
    build:
      context: ./php-fpm
    depends_on:
      - database
    environment:
      - APP_ENV=${APP_ENV}
      - APP_SECRET=${APP_SECRET}
      - DATABASE_URL=mysql://${DATABASE_USER}:${DATABASE_PASSWORD}@database:3306/${DATA
    volumes:
      - ../src:/var/www

  nginx:
    build:
      context: ./nginx
    volumes:
      - ../src:/var/www
      - ./nginx/nginx.conf:/etc/nginx/nginx.conf
      - ./nginx/sites/:/etc/nginx/sites-available
      - ./nginx/conf.d/:/etc/nginx/conf.d
      - ./logs:/var/log
    depends_on:
      - php-fpm
    ports:
      - "80:80"
```

And a sample environment variables:

```
# docker/.env

DATABASE_NAME=symfony
DATABASE_USER=appuser
DATABASE_PASSWORD=apppassword
```

```
APP_ENV=dev
APP_SECRET=24e17c47430bd2044a61c131c1cf6990
```

Symfony

Let's proceed to the Symfony installation:

```
$ symfony new src
```

Play time

Everything is setup correctly! Let's play with our containers!

```
$ docker-compose up
```

When those containers are ready, you can start to open http://localhost, you will see a Symfony 5 welcome screen. Everything works perfectly, have fun!

I've create a repository for all the files we talked above https://gitlab.com/martinpham/symfony-5-docker

Discussion

Subscribe

Add to the discussion

saggzz • Mar 29

For those which rather want to use postgresql instead of mariadb.

```
# docker/database/Dockerfile
FROM postgres:latest

ENV POSTGRES_USER <your user>
ENV POSTGRES_PASSWORD <your password>
EXPOSE 3306
```

And I had to add

```
RUN apk add --no-cache bash
```

in php-fpm dockerfile (due to this warning: env: can't execute 'bash': No such file or

29 3 35

Dan Seely • Feb 28

Great writeup! This is going to help me out a ton.

After reading through the article and looking at your example repo, I think the directory structure at the top of this article would look more like this (i.e. dedicated `docker` and `src` directories):

```
Project
│
├── docker
├─── docker-compose.yml
│
├─── database/
│   ├─── Dockerfile
│   └─── data/
│
└─── php-fpm/
│   └─── Dockerfile
│
├─── nginx/
│   ├─── Dockerfile
│   └─── nginx.conf
│
└─── logs/
│   └─── nginx/
├── src
├─── <Symfony app>
```

2

abelardoit • Oct 8

Hi there!
I am a beginner developer. I downloaded your project from your repo for learning purposes.
I use Maker Bundle to automatically create an user.
When I run the command (docker and mysql are already running):
php bin/console make:migration

this error is shown several times in a row:
SQLSTATE[HY000] [2002] php_network_getaddresses: getaddrinfo failed: Temporary failure in name resolution

Could be possible that this bundle is unable to know where my database is running?
How could I solve this issue?

29 3 35

1

fadilxcoder • Oct 26

You should connect to you docker terminal to run the above command... It should be something like `docker exec -it docker_container_name ash`. Your docker_container_name should be listed in `docker ps -a`

1

samikerb • Sep 21

Hello,

thank you for this amazing post,

i have an issue with login, once i submit to login i got this error :

php-fpm_1 | [21-Sep-2020 16:45:16] WARNING: [pool www] child 38 exited on signal 9 (SIGKILL) after 318.447129 seconds from start
php-fpm_1 | [21-Sep-2020 16:45:16] NOTICE: [pool www] child 50 started

any idea ?

1

samikerb • Sep 21

2020/09/21 16:48:50 [error] 29#29: *37 recv() failed (104: Connection reset by peer) while reading response header from upstream, client: 192.168.0.1, server: localhost, request: "POST /login HTTP/1.1", upstream: "fastcgi://192.168.0.3:9000", host: "localhost:1609", referrer: "localhost:1609/login"

1

Arthur Guillerm • Feb 4

Excellent article !

2

Martin Pham • Feb 4

29 3 35

1

CyrilCharlier • Jul 13

Great tutorial !
Just one thing, remember to add "wait-for-it.sh" in "php-fpm" directory ;)

1

eiosca • Jan 17

Thk's a lot for this article, simple and functional. +1

1

Galgaldas • Aug 15

If you prefer apache over nginx and you need phpmyadmin, mysql, email dockerized services, I would recommend checking this repository - github.com/kasteckis/symfony-docke...

1

bkrnetic • May 21

Great article!

Would be great if I was able to enter container bash by running

```
docker-compose exec nginx bash
```

Is it possible to update Dockerfile to enable that?

1

Nawa Augustine • Oct 21

You can try

docker exec -it nginx sh

1

29 3 35

Wonderful post! I was wondering how one would deploy something like this on AWS/Azure?

1

Code of Conduct • Report abuse

Martin Pham

Just another boring developer - https://mph.am

Follow

WORK
CTO at Fornace

LOCATION
Italy

JOINED
23 dic. 2019

More from Martin Pham

Try PHP 8.0 beta features with Docker / Kubernetes

#php #beta #docker #kubernetes

Having fun with Kubernetes - Chapter 4

#kubernetes #docker #php #linux

Having fun with Kubernetes - Chapter 3

#kubernetes #docker #deployment #loadbalancer

29 3 35