Information

Очень крутая и все объясняющая статья - https://m.habr.com/ru/post/353238/ -
Контейнеры и контейнеризация -
https://tproger.ru/articles/containers-explained/?utm_source=tnull
Докер со всех сторон: https://habr.com/ru/company/ruvds/blog/441574/
Документация - https://docs.docker.com
Докерхаб - https://hub.docker.com/search?q=&type=image
Гитхаб - https://github.com/docker
Примеры докерфайлов - https://github.com/jessfraz/dockerfiles
Volumes: https://docs.docker.com/storage/volumes/
Настройка wordpress и mysql для 2х сайтов -
https://www.digitalocean.com/community/tutorials/how-to-set-up-multiple-wordpress-sites-on-a-single-ubuntu-vps

Check-list:

https://github.com/mashenkaPas/Docker-42/blob/master/docker-1%20Correction.pdf

Exercises

Installation

In General:

https://docs.docker.com/docker-for-mac/install/
info on our MAC: Apple () > «Об этом Mac»

In Project:

docker, docker-machine and virtualbox
On the local machine:

```
brew install docker docker-machine
```

Creation of the docker machine:
https://docs.docker.com/machine/drivers/virtualbox/
Make docker work again: brew install docker - update

Первоначально: VirtualBox Vm и .docker лежали на home
Потом перенесла:

```
mv -f ~/.docker ~/goinfre
cd ~
```

ln -s ~/goinfre/.docker ./.docker (если назвать иначе, то виртуалка не будет видеть машину)
То же самое можно сделать с Virtual Machine - но не обязательно во многих случаях
ln -s ~/goinfre/VirtualBox\ VM ./VM

Другие команды:

    docker-compose rm

Removes stopped service containers.

    docker container stop $(docker container ls -aq) - stops all the containers
    docker-machine stop Char Aiur

00_how_to_docker

1.Create a virtual machine with docker-machine using the virtualbox driver, and named Char.
Создайте виртуальную машину через команду docker-machine через драйвер virtualbox, которая будет называться Char

Создано: Creating client certificate: /Users/sschmele/.docker/machine/certs/cert.pem

```
          docker-machine ls
NAME   ACTIVE   DRIVER      STATE     URL                       SWARM   DOCKER
ERRORS
Char   *        virtualbox   Running   tcp://192.168.9*.1*0:2376          v18.09.9
```

Команда: docker-machine create --driver=virtualbox Char

2. Get the IP address of the Char virtual machine
Узнайте IP виртуальной машины Char

Ответ после команды:

```
          192.168.9*.1*0
```

Команда: docker-machine ip Char

3. Define the variables needed by your virtual machine Char in the general env of your terminal, so that you can run the docker ps command without errors. You have to fix all four environment variables with one command, and you are not allowed to use your shell's builtin to set these variables by hand
Выявите, какие переменные окружения виртуальной машины необходимо добавить к общим переменным окружениям терминала, чтобы работа с

машиной производилась без ошибок. Нельзя закреплять переменные окружения через builtin commands.

docker-machine env Char - окружение виртуальной машины

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.100:2376"
export DOCKER_CERT_PATH="/Users/sschmele/.docker/machine/machines/Char"
export DOCKER_MACHINE_NAME="Char"
```

Команда: eval $(docker-machine env Char) - фикс окружения

4. Get the hello-world container from the Docker Hub, where it’s available
Загрузите образ контейнера hello-world с Docker Hub

Ответ после команды:

```
Using default tag: latest
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest:
sha256:b8ba256769a0ac28dd126d584e0a2011cd2877f3f76e093a7ae560f2a5301c00
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

docker images

```
REPOSITORY        TAG           IMAGE ID         CREATED          SIZE
hello-world       latest        fce289e99eb9     8 months ago     1.84kB
```

Команда: docker pull hello-world

5. Launch the hello-world container, and make sure that it prints its welcome message, then leaves it. Запустите контейнер hello-world, он должен поприветствовать вас сообщением и закончить свою работу.

Сперва ищет контейнер на компе, если не находит, скачивает с Registry - Dockerhub - и запускает.
Ответ команды:

```
Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash (интерактивный запуск)

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

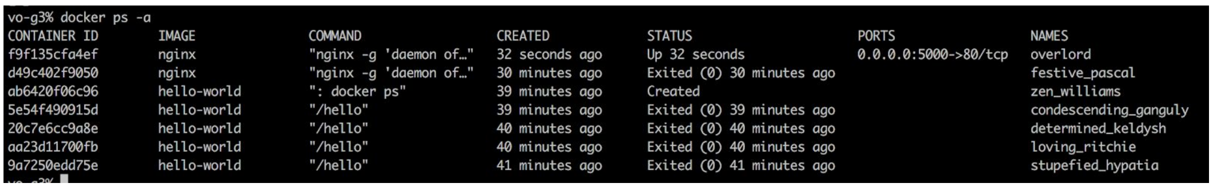
Команда: docker run hello-world

6. Launch an nginx container, available on Docker Hub, as a background task. It should be named overlord, be able to restart on its own, and have its 80 port attached to the 5000 port of Char. You can check that your container functions properly by visiting http://<ip-de-char>:5000 on your web browser
Запустите контейнер nginx в фоне, он должен называться overlord, уметь сам перезапускаться, работать через 80 порт внутри себя и через 5000 порт от довер-машины. Зайдите в браузер по IP адресу докер-машины и 5000 порту и убедитесь, что контейнер запущен и работает.

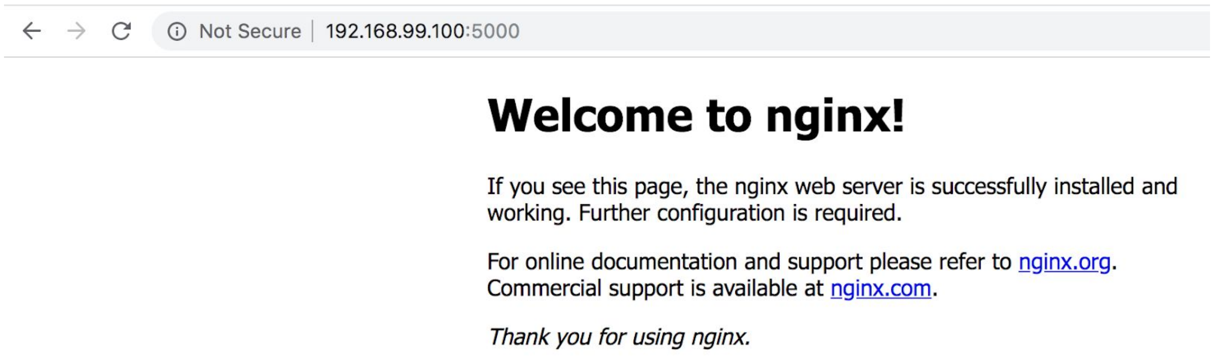
Overlord launch (nginx)

Дефолтный запуск контейнера по порту 8080 - без прописи через флаг -p
docker run -p 1234:8080 - на хосте доступ к порту докера через порт 1234
docker run -d - запуск контейнера в бэке

docker ps -a



```
vo-g3% docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED          STATUS                      PORTS                  NAMES
f9f135cfa4ef   nginx         "nginx -g 'daemon of…"   32 seconds ago   Up 32 seconds               0.0.0.0:5000->80/tcp   overlord
d49c402f9050   nginx         "nginx -g 'daemon of…"   30 minutes ago   Exited (0) 30 minutes ago                          festive_pascal
ab6420f06c96   hello-world   ": docker ps"            39 minutes ago   Created                                            zen_williams
5e54f490915d   hello-world   "/hello"                 39 minutes ago   Exited (0) 39 minutes ago                          condescending_ganguly
20c7e6cc9a8e   hello-world   "/hello"                 40 minutes ago   Exited (0) 40 minutes ago                          determined_keldysh
aa23d11700fb   hello-world   "/hello"                 40 minutes ago   Exited (0) 40 minutes ago                          loving_ritchie
9a7250edd75e   hello-world   "/hello"                 41 minutes ago   Exited (0) 41 minutes ago                          stupefied_hypatia
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

curl http://192.168.99.100:5000

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
   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>
192.168.99.1 - - [25/Sep/2019:15:06:15 +0000] "GET / HTTP/1.1" 200 612 "-"
"curl/7.54.0" "-"
```

Restart policy inspection

```
docker inspect -f "{{.HostConfig.RestartPolicy}}" overlord
{always 0}
```
Why?

Restart policies (--restart)

Use Docker’s `--restart` to specify a container’s *restart policy*. A restart policy controls whether the Docker daemon restarts a container after exit. Docker supports the following restart policies:

Policy	**Result**
`no`	Do not automatically restart the container when it exits. This is the default.
`on-failure[:max-retries]`	Restart only if the container exits with a non-zero exit status. Optionally, limit the number of restart retries the Docker daemon attempts.
`unless-stopped`	Restart the container unless it is explicitly stopped or Docker itself is stopped or restarted.
`always`	Always restart the container regardless of the exit status. When you specify always, the Docker daemon will try to restart the container indefinitely. The container will also always start on daemon startup, regardless of the current state of the container.

```
$ docker run --restart=always redis
```

This will run the `redis` container with a restart policy of **always** so that if the container exits, Docker will restart it.

Команда: docker run -d --name overlord --restart always -p 5000:80 nginx

7. Get the internal IP address of the overlord container without starting its shell and in one command. Узнайте IP адрес контейнера overlord, не заходя в него через интерактивный режим.

Не очень прикольная версия команды: docker inspect overlord | grep IPAddress | grep -o -E '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}' | awk 'NR == 1 {print}'

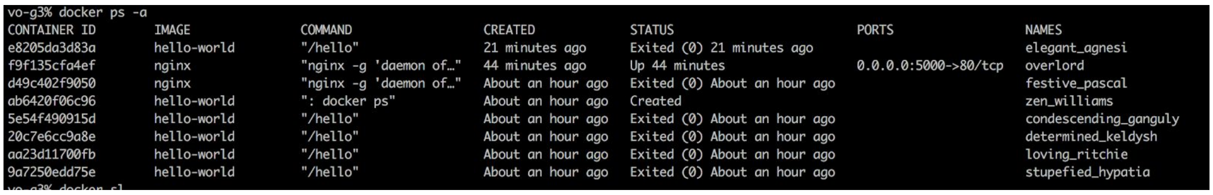
Ответ команды:

172.17.0.2

docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' overlord

8. Launch a shell from an alpine container, and make sure that you can interact directly with the container via your terminal, and that the container deletes itself once the shell's execution is done. Зайдите внутрь запущенного контейнера alpine (один из самых легких контейнеров в докере), убедитесь, что вы внутри и ваши команды выполняются (они особенные), выйдите и убедитесь, что контейнер удалился.

Удаляется из истории после запуска:



```
vo-g3% docker ps -a
CONTAINER ID   IMAGE         COMMAND                  CREATED             STATUS                        PORTS                  NAMES
e8205da3d83a   hello-world   "/hello"                 21 minutes ago      Exited (0) 21 minutes ago                            elegant_agnesi
f9f135cfa4ef   nginx         "nginx -g 'daemon of…"   44 minutes ago      Up 44 minutes                 0.0.0.0:5000->80/tcp   overlord
d49c402f9050   nginx         "nginx -g 'daemon of…"   About an hour ago   Exited (0) About an hour ago                         festive_pascal
ab6420f06c96   hello-world   ": docker ps"            About an hour ago   Created                                              zen_williams
5e54f490915d   hello-world   "/hello"                 About an hour ago   Exited (0) About an hour ago                         condescending_ganguly
20c7e6cc9a8e   hello-world   "/hello"                 About an hour ago   Exited (0) About an hour ago                         determined_keldysh
aa23d11700fb   hello-world   "/hello"                 About an hour ago   Exited (0) About an hour ago                         loving_ritchie
9a7250edd75e   hello-world   "/hello"                 About an hour ago   Exited (0) About an hour ago                         stupefied_hypatia
```

Команда: docker run -it --rm alpine

9. From the shell of a debian container, install via the container's package manager everything you need to compile C source code and push it onto a git repo (of course, make sure before that the package manager and the packages already in the container are updated). For this exercise, you should only specify the commands to be run directly in the container. Внутри контейнера debian установите все необходимое, чтобы скомпилировать и запустить код на Си. Укажите только команды, которыми вы пользовались внутри контейнера.

docker run -it --rm debian
Unable to find image 'debian:latest' locally
latest: Pulling from library/debian

```
4a56a430b2ba: Pull complete
Digest:
sha256:e25b64a9cf82c72080074d6b1bba7329cdd752d51574971fd37731ed164f3345
Status: Downloaded newer image for debian:latest
root@93abab17fc40:

        root@93abab17fc40:/# apt-get update
Get:1 http://cdn-fastly.deb.debian.org/debian buster InRelease [122 kB]
Get:2 http://security-cdn.debian.org/debian-security buster/updates InRelease [39.1 kB]
Get:4 http://security-cdn.debian.org/debian-security buster/updates/main amd64 Packages
[92.2 kB]
Get:3 http://cdn-fastly.deb.debian.org/debian buster-updates InRelease [49.3 kB]
Get:5 http://cdn-fastly.deb.debian.org/debian buster/main amd64 Packages [7899 kB]
Get:6 http://cdn-fastly.deb.debian.org/debian buster-updates/main amd64 Packages [5792
B]
Fetched 8206 kB in 4s (2317 kB/s)
Reading package lists... Done
        apt-get install git
        apt-get install gcc
        apt-get install vim
```

```c
#include <unistd.h>

int main(void)
{
      write(1, "Hi\n", 3);
      return (0);
}
```

```
root@cb1c8c9c3532:/home# gcc compile.c
root@cb1c8c9c3532:/home# ./a.out
Hi
```

Команды: apt-get update
apt-get install git
apt-get install gcc
apt-get install vim

10. Create a volume named hatchery
Создайте разметку и назовите ее hatchery

Volume – это дисковое пространство между хостом и контейнером. Проще – это папка на вашей локальной машине примонтированная внутрь контейнера. Меняете тут меняется там, и наоборот, миракл.

Volume creation

Ответ команды:

```
hatchery
```

Команда: docker volume create --name hatchery

11. List all the Docker volumes created on the machine. Remember. VOLUMES. Выведите список всех разметок на машине.

https://docs.docker.com/storage/bind-mounts/
Volumes are stored in a part of the host filesystem which is managed by Docker (/var/lib/docker/volumes/ on Linux). Non-Docker processes should not modify this part of the filesystem. Volumes are the best way to persist data in Docker.
Volumes are the preferred mechanism for persisting data generated by and used by Docker containers.

Ответ команды:

```
DRIVER              VOLUME NAME
local               hatchery
```

Volume inspection

docker volume inspect hatchery

```
[
    {
        "CreatedAt": "2019-09-25T15:34:17Z",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/mnt/sda1/var/lib/docker/volumes/hatchery/_data",
        "Name": "hatchery",
        "Options": {},
        "Scope": "local"
    }
]
```

Команда: docker volume ls

12. Launch a mysql container as a background task. It should be able to restart on its own in case of error, and the root password of the database should be Kerrigan. You will also make sure that the database is stored in the hatchery volume, that the container directly creates a database named zerglings, and that the container itself is named spawning-pool.

Запустите контейнер mysql на фоне. Он должен перезагружаться и запускаться сам при возникновении ошибки, пароль для корневого юзера должен быть Kerrigan. Также нужно чтобы созданная база данных сохранялась в пространстве hatchery, что в контейнере создается база данных zerglings, а сам контейнер называется spawning-pool.

Spawning-pool

Ответ команды:

```
b58321a5c3eb76e00467f2b20634b7f209f61cd44159501e182b0d64013a3ab5
```

```
docker ps
CONTAINER ID       IMAGE              COMMAND                  CREATED            STATUS
PORTS                 NAMES
b58321a5c3eb       mysql              "docker-entrypoint.s…"   4 seconds ago      Up 4
seconds        3306/tcp, 33060/tcp    spawning-pool
```

```
docker exec spawning-pool env | grep MYSQL_ROOT_PASSWORD
MYSQL_ROOT_PASSWORD=Kerrigan
```

Environment overview in the container

```
docker exec spawning-pool env | grep MYSQL_DATABASE
MYSQL_DATABASE=zerglings
```

```
docker inspect -f "{{.HostConfig.RestartPolicy}}" spawning-pool
{on-failure 0}
```

```
docker inspect spawning-pool
```

```
"Mounts": [
    {
        "Type": "volume",
        "Name": "hathery",
        "Source": "/mnt/sda1/var/lib/docker/volumes/hathery/_data",
        "Destination": "/var/lib/mysql",
        "Driver": "local",
        "Mode": "z",
        "RW": true,
        "Propagation": ""
    }
],
```

Check of the data-base work in spawning-pool

```
        docker exec -it spawning-pool mysql -uroot -p
Enter password:
```

```
docker exec -it spawning-pool mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.17 MySQL Community Server - GPL

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| zerglings          |
+--------------------+
5 rows in set (0.03 sec)

mysql> exit
```

If you need to specify volume driver options, you must use `--mount`.

- **`-v`** or **`--volume`**: Consists of three fields, separated by colon characters (**`:`**). The fields must be in the correct order, and the meaning of each field is not immediately obvious.
    - In the case of named volumes, the first field is the name of the volume, and is unique on a given host machine. For anonymous volumes, the first field is omitted.
    - The second field is the path where the file or directory are mounted in the container.
    - The third field is optional, and is a comma-separated list of options, such as `ro`. These options are discussed below.
- **`--mount`**: Consists of multiple key-value pairs, separated by commas and each

consisting of a `<key>=<value>` tuple. The `--mount` syntax is more verbose than `-v` or `--volume`, but the order of the keys is not significant, and the value of the flag is easier to understand.

- The `type` of the mount, which can be `bind`, `volume`, or `tmpfs`. This topic discusses volumes, so the type is always `volume`.
- The `source` of the mount. For named volumes, this is the name of the volume. For anonymous volumes, this field is omitted. May be specified as `source` or `src`.
- The `destination` takes as its value the path where the file or directory is mounted in the container. May be specified as `destination`, `dst`, or `target`.
- The `readonly` option, if present, causes the bind mount to be mounted into the container as read-only.
- The `volume-opt` option, which can be specified more than once, takes a key-value pair consisting of the option name and its value.

Команда: docker run --volume=hatchery:/var/lib/mysql \
--restart=on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--name=spawning-pool \
-p 3306:3306 \ (дефолтные, но после многих ошибок решили прописать)
-d mysql:latest \
--default-authentication-plugin=mysql_native_password (только для версии mysql 18 и выше)

OR

docker run --mount source=hatchery,target=/var/lib/mysql \
--restart=on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--name=spawning-pool \
-p 3306:3306 \ (дефолтные, но после многих ошибок решили прописать)
-d mysql:latest \
--default-authentication-plugin=mysql_native_password (только для версии mysql 18 и выше)

13. Print the environment variables of the spawning-pool container in one command, to be sure that you have configured your container properly. Выведите переменные окружения контейнера spawning-pool одной командой.

docker exec spawning-pool env - внутри контейнера

docker inspect spawning-pool - снаружи контейнера из докер-машины

Команда: docker inspect spawning-pool

14. Launch a wordpress container as a background task, just for fun. The container should be named lair, its 80 port should be bound to the 8080 port of the virtual machine, and it should be able to use the spawning-pool container as a database service. You can try to access lair on your machine via a web browser, with the IP address of the virtual machine as a URL.

Congratulations, you just deployed a functional Wordpress website in two commands!

Запустите контейнер wordpress в фоне. Имя контейнера lair, 80 порт контейнера примонтирован к 8080 порту виртуалки, а также он должен использовать контейнер spawning-pool в качестве базы данных. Попробуйте зайти в контейнер через браузер.

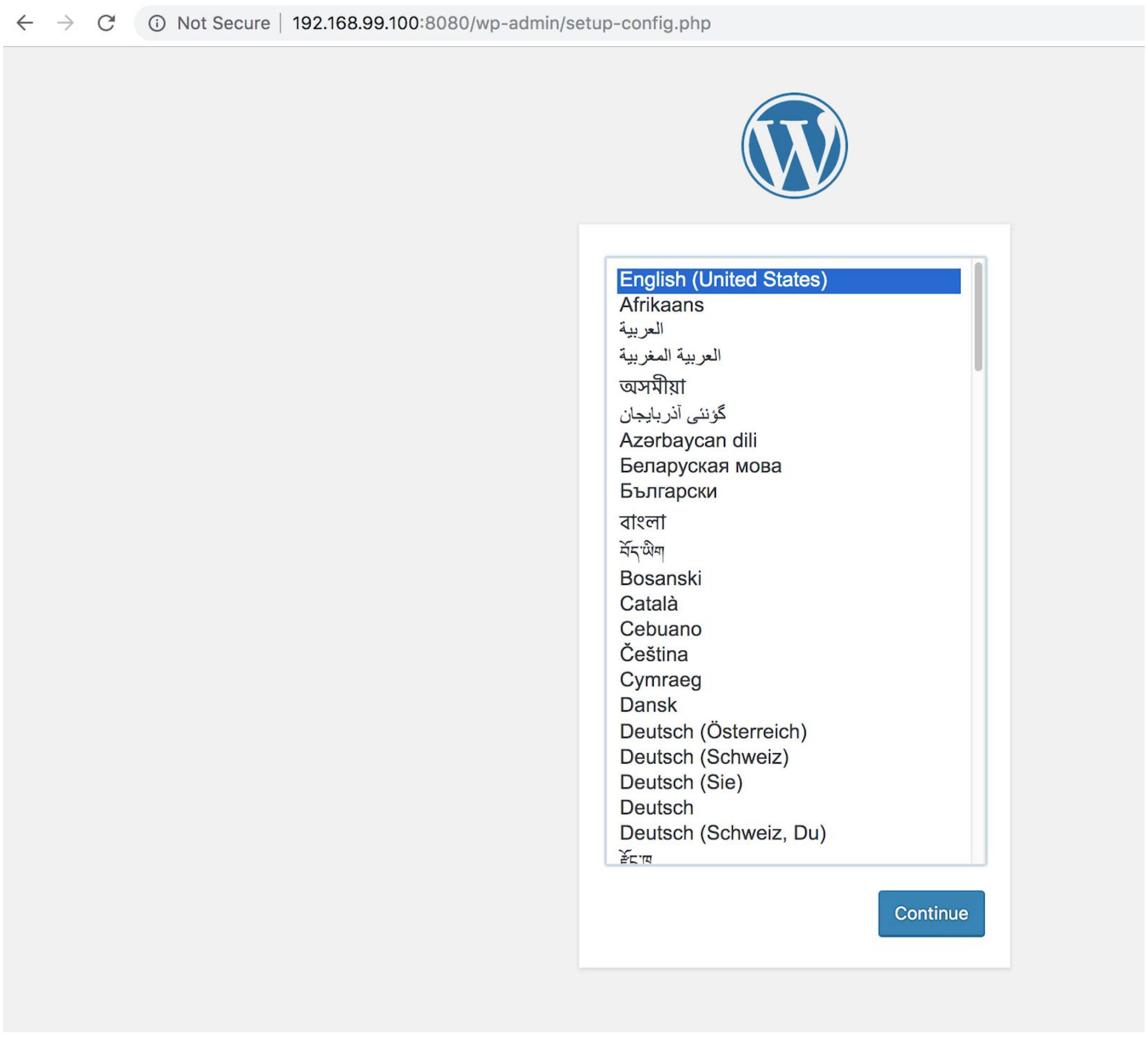
Wordpress

If you'd like to be able to access the instance from the host without the container's IP, standard port mappings can be used:

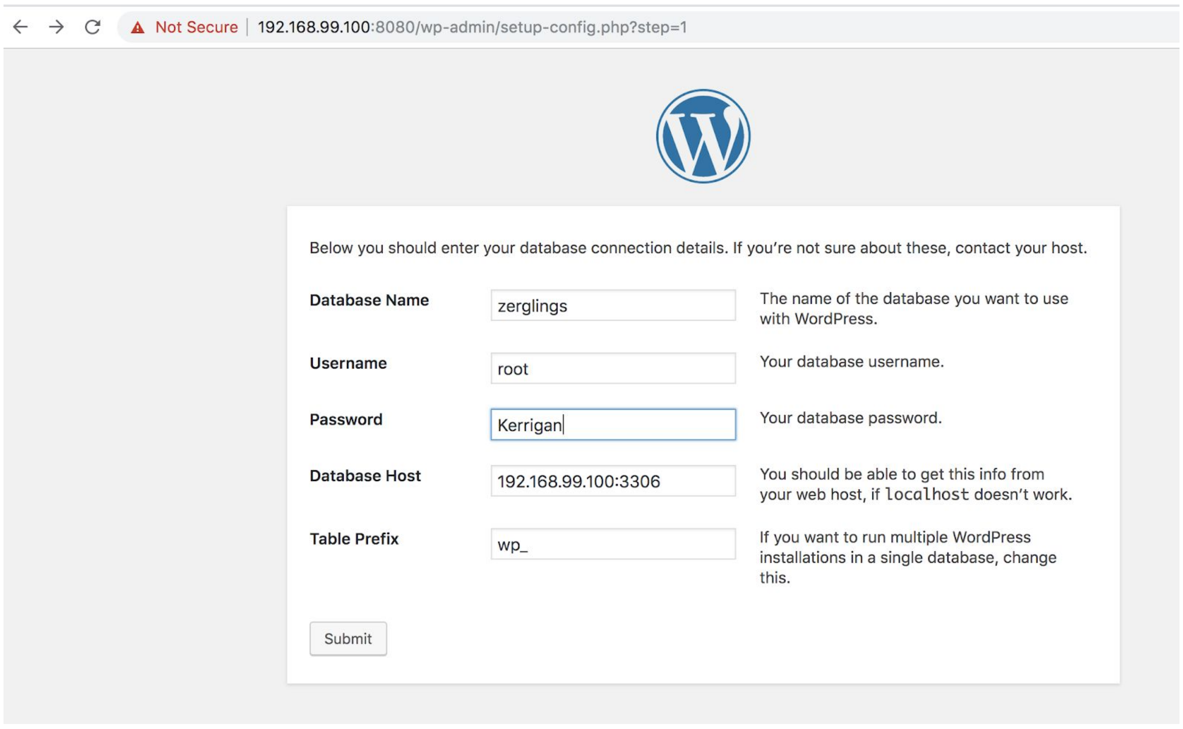
```
$ docker run --name some-wordpress -p 8080:80 -d wordpress
```

https://hub.docker.com/_/wordpress/

Результат:


Not Secure | 192.168.99.100:8080/wp-admin/setup-config.php

English (United States)
Afrikaans
العربية
العربية المغربية
অসমীয়া
گؤنئی آذربایجان
Azərbaycan dili
Беларуская мова
Български
বাংলা
བོད་ཡིག
Bosanski
Català
Cebuano
Čeština
Cymraeg
Dansk
Deutsch (Österreich)
Deutsch (Schweiz)
Deutsch (Sie)
Deutsch
Deutsch (Schweiz, Du)

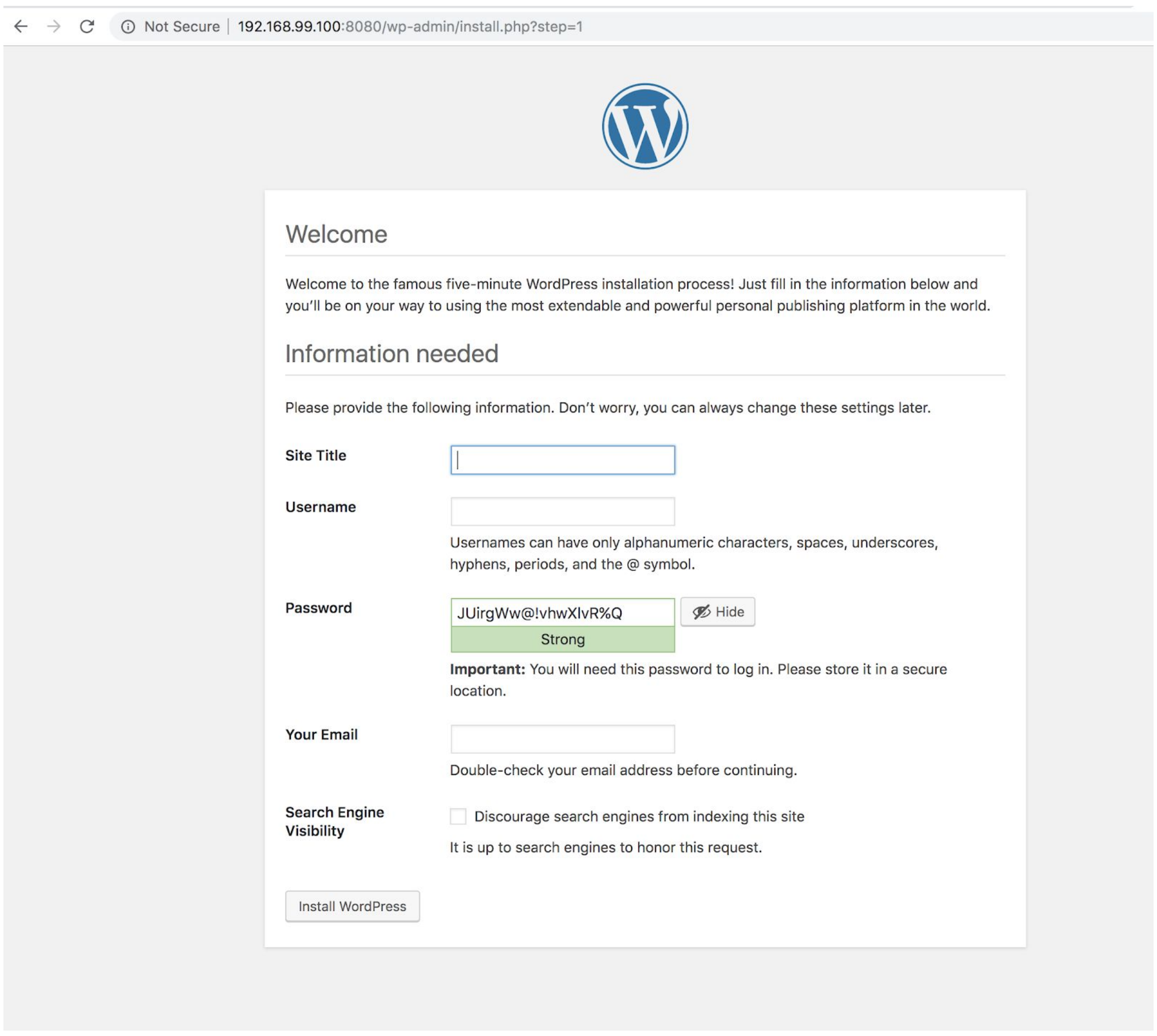


Not Secure | 192.168.99.100:8080/wp-admin/setup-config.php?step=1

Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	zerglings	The name of the database you want to use with WordPress.
Username	root	Your database username.
Password	Kerrigan	Your database password.
Database Host	192.168.99.100:3306	You should be able to get this info from your web host, if `localhost` doesn't work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

Submit



Not Secure | 192.168.99.100:8080/wp-admin/install.php?step=1

Welcome

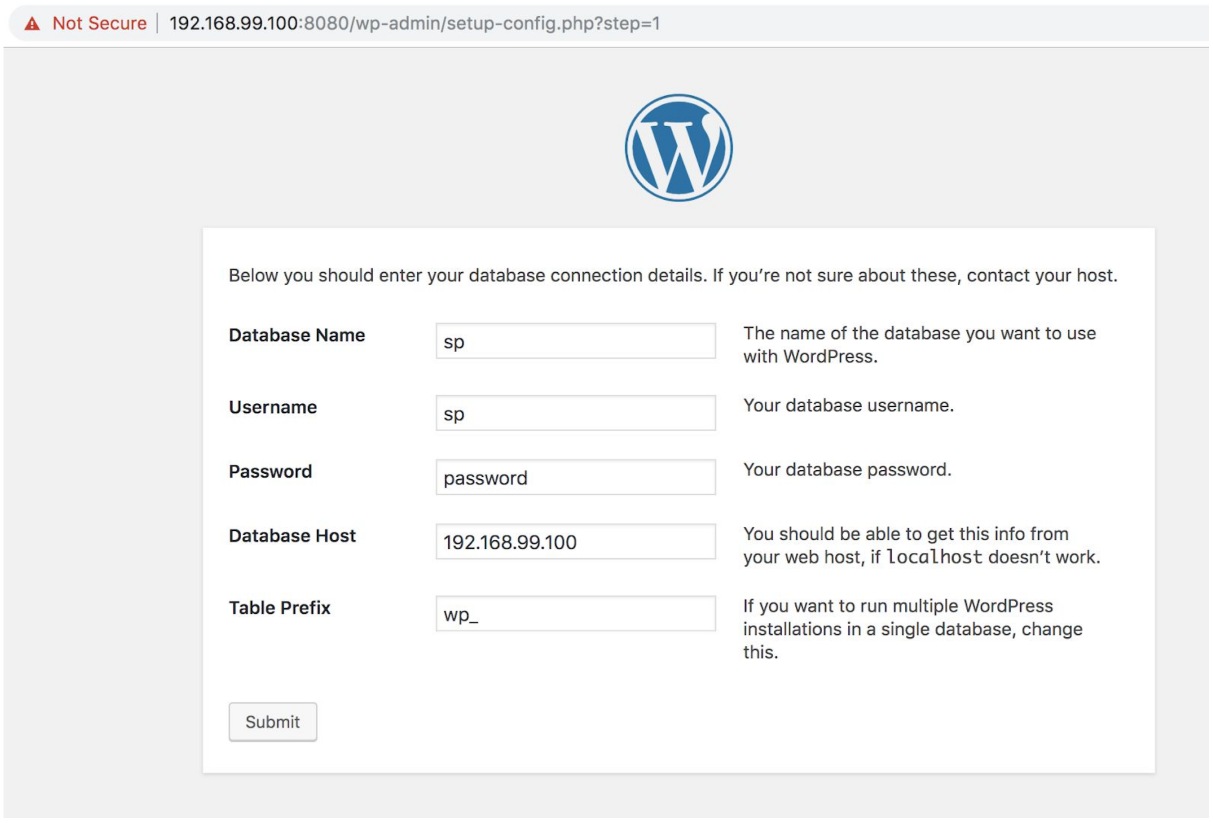
Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you'll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don't worry, you can always change these settings later.

Site Title	
Username	Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.
Password	JUirgWw@!vhwXlvR%Q Hide Strong **Important:** You will need this password to log in. Please store it in a secure location.
Your Email	Double-check your email address before continuing.
Search Engine Visibility	☐ Discourage search engines from indexing this site It is up to search engines to honor this request.

Install WordPress



Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	sp	The name of the database you want to use with WordPress.
Username	sp	Your database username.
Password	password	Your database password.
Database Host	192.168.99.100	You should be able to get this info from your web host, if `localhost` doesn't work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

Now, create a new web container and link it with your db container.

```
$ docker run -d -P --name web --link db:db training/webapp python app.py
```

This links the new web container with the db container you created earlier. The --link flag takes the form:

```
--link <name or id>:alias
```

Where name is the name of the container we're linking to and alias is an alias for the link name. That alias is used shortly. The --link flag also takes the form:

```
--link <name or id>
```

In this case the alias matches the name. You could write the previous example as:

```
$ docker run -d -P --name web --link db training/webapp python app.py
```

Next, inspect your linked containers with docker inspect:

```
$ docker inspect -f "{{ .HostConfig.Links }}" web
```

```
[/db:/web/db]
```

You can see that the web container is now linked to the db container web/db. Which allows it to access information about the db container.

In our example, the recipient, web, can access information about the source db. To do this, Docker creates a secure tunnel between the containers that doesn't need to expose any ports externally on the container

Docker exposes connectivity information for the source container to the recipient container in two ways:

- Environment variables,
- Updating the /etc/hosts file.

РУБРИКА “ВЛАДА ОПЯТЬ ЗАПАРИЛАСЬ” - нормальным людям смотреть не стоит. Потому что получилось без этого

https://wordpress.org/support/article/creating-database-for-wordpress/
Настройка wordpress для его работы:
https://www.hostinger.ru/rukovodstva/kak-ispravit-establishing-a-database-connection-wordpress/#-2---wp-configphp

```
docker exec -it spawning-pool mysql -uroot -p - заходим в верхний слой контейнера
Enter password:

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.17 MySQL Community Server - GPL

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user 'db_user'@'localhost' identified by 'pass'; - создание юзера с паролем ‘pass’
Query OK, 0 rows affected (0.01 sec)

mysql> SHOW GRANTS FOR db_user@localhost; - показывает права, которые даны юзеру
+-----------------------------------------------------------------+
| Grants for db_user@localhost                                    |
+-----------------------------------------------------------------+
| GRANT USAGE ON *.* TO `db_user`@`localhost`                     |
| GRANT ALL PRIVILEGES ON `zergllings`.* TO `db_user`@`localhost` |
+-----------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql> GRANT ALL PRIVILEGES ON zerglings.* TO 'db_user'@'localhost'; - даем права юзеру
Query OK, 0 rows affected, 1 warning (0.00 sec)

mysql> show databases;
```

```
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| zerglings          |
+--------------------+
5 rows in set (0.00 sec)
```

mysql> FLUSH PRIVILEGES; - перезагрузка после выдачи прав

```
Query OK, 0 rows affected (0.00 sec)

mysql> exit
Bye
```

Для того, чтобы базу данных можно было запустить, нужен docker-compose, который нужно докачивать отдельно и к которому нет доступа на Маке. Но нам нужно иметь только “возможность” запуска по заданию

https://docs.docker.com/compose/wordpress/

```
su-d3% cat docker-compose.yml
version: '3.3'

services:
  db:
    image: mysql:8.0.17
    volumes:
      - db_data:/var/lib/mysql
    restart: on-failure
    environment:
      MYSQL_ROOT_PASSWORD: Kerrigan
      MYSQL_DATABASE: zerglings
      MYSQL_USER: root
      MYSQL_PASSWORD: Kerrigan

      wordpress:
        depends_on:
          - db
        image: wordpress:latest
        ports:
          - "8081:80"
        restart: no
        environment:
          WORDPRESS_DB_HOST:
          WORDPRESS_DB_USER:
          WORDPRESS_DB_PASSWORD:
          WORDPRESS_DB_NAME:
```

```
volumes:
   db_data: {}
```

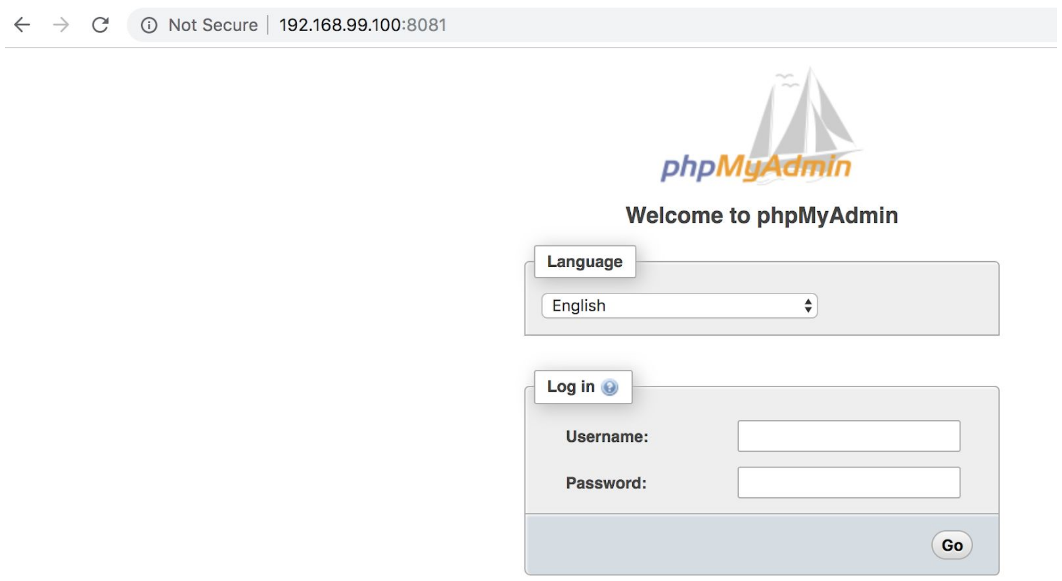
Команда: docker run -d --name lair -p 8080:80 --link spawning-pool:db wordpress

15. Launch a phpmyadmin container as a background task. It should be named roach-warden, its 80 port should be bound to the 8081 port of the virtual machine and it should be able to explore the database stored in the spawning-pool container.

Запустите контейнер phpadmin на фоне. Имя должно быть roach-warden, 80 порт примонтирован к 8081 порту машины и через него должно быть возможным просматривать базу контейнера spawning-pool.

Phpmyadmin

docker search phpmyadmin
docker pull phpmyadmin/phpmyadmin
Результат после root и пароля Kerrigan (вроде, уже не помню)



Not Secure | 192.168.99.100:8081

Welcome to phpMyAdmin

Language

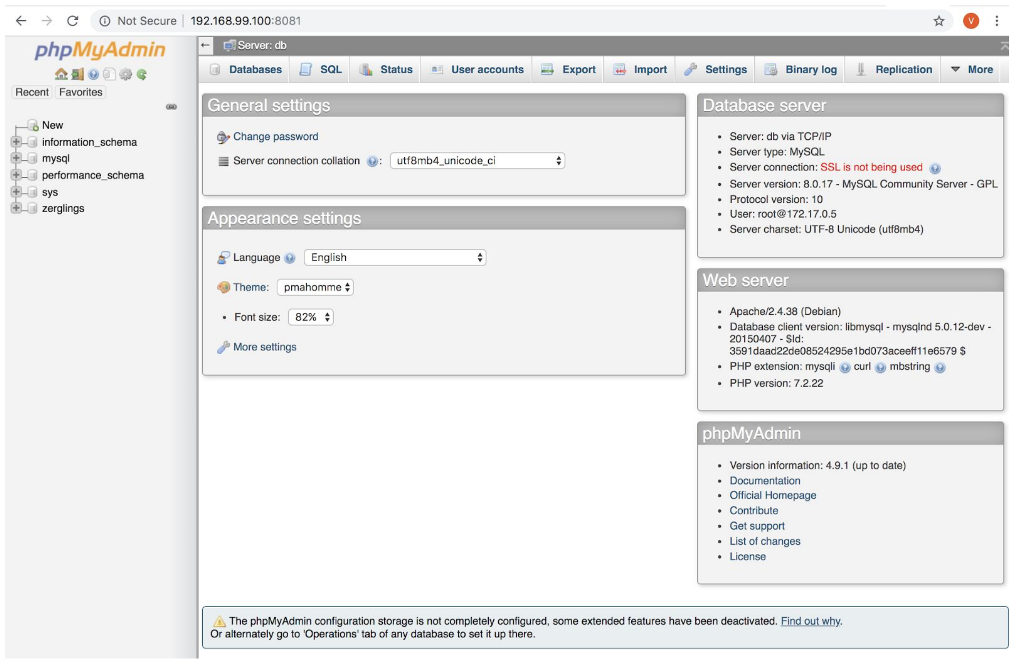
English

Log in

Username:

Password:

Go



Not Secure | 192.168.99.100:8081

Server: db

Databases SQL Status User accounts Export Import Settings Binary log Replication More

Recent Favorites

- New
- information_schema
- mysql
- performance_schema
- sys
- zerglings

General settings

Change password

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

- Font size: 82%

More settings

Database server

- Server: db via TCP/IP
- Server type: MySQL
- Server connection: SSL is not being used
- Server version: 8.0.17 - MySQL Community Server - GPL
- Protocol version: 10
- User: root@172.17.0.5
- Server charset: UTF-8 Unicode (utf8mb4)

Web server

- Apache/2.4.38 (Debian)
- Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - $Id: 3591daad22de08524295e1bd073aceeff11e6579 $
- PHP extension: mysqli curl mbstring
- PHP version: 7.2.22

phpMyAdmin

- Version information: 4.9.1 (up to date)
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

The phpMyAdmin configuration storage is not completely configured, some extended features have been deactivated. Find out why.
Or alternately go to 'Operations' tab of any database to set it up there.

Команда: docker run -d –name roach-warden -p 8081:80 –link spawning-pool:db phpmyadmin/phpmyadmin

16. Look up the spawning-pool container’s logs in real time without running its shell.
Просмотрите логи контейнера spawning-pool, не заходя в него в интерактивном режиме.

https://logdna.com/blog/docker-logs/

The docker logs --follow command will continue streaming the new output from the container’s STDOUT and STDERR.
Ctrl-C to leave

Команда: docker logs --follow spawning-pool

17. Display all the currently active containers on the Char virtual machine.
Выведите все запущеные на данный момент контейнеры на машине.

```
docker ps
```

18. Relaunch the overlord container. Перезапустите контейнер overlord.

```
su-d3% docker exec -it overlord /bin/sh -c "kill 1"
su-d3% docker inspect -f '{{.RestartCount}}' overlord
1
```

Команда: docker restart overlord

19. Launch a container name Abathur. It will be a Python container, 2-slim version, its /root folder will be bound to a HOME folder on your host, and its 3000 port will be bound to the 3000 port of your virtual machine.

You will personalize this container so that you can use the Flask micro-framework in its latest version. You will make sure that an html page displaying "Hello World" with <h1> tags can be served by Flask. You will test that your container is properly set up by accessing, via curl or a web browser, the IP address of your virtual machine on the 3000 port.

You will also list all the necessary commands in your repository.

Запустите контейнер Python версии 2-slim и назовите его Abathur, свяжите его рутовую директорию с домашней директорией на хосте, 3000 порт к 3000 порту машины.
Нужно сделать так, чтобы вы могли использовать фреймворк Flask (последнюю версию) - должна быть доступной html страница, которая будет показывать "Hello World" с тегом <h1> . Страницу должно быть видно через curl или браузер через IP машины и порт 3000.

Abathur

https://flask-russian-docs.readthedocs.io/ru/latest/quickstart.html
https://pythonhow.com/html-templates-in-flask/
https://prateekvjoshi.com/2016/03/08/how-to-create-a-web-server-in-python-using-flask/

docker run -di --name Abathur -v $HOME:/root -p 3000:3000 python:2-slim - это тот самый контейнер, который не запускается в фоновом режиме без i-флага
docker exec Abathur /bin/bash -c "pip install Flask" - скачиваем Flask (фреймворк для веб-приложений)
На хосте, потому что root привязан к домашней директории хоста:
mkdir ~/web

```
        mkdir ~/web/templates
        echo -e "<\!DOCTYPE html>\n<html>\n\t<body>\n\t\t<h1>Hello
world</h1>\n\t</body>\n</html>" > ~/web/templates/hello.html
        echo -e "from flask import Flask, render_template\napp =
Flask(__name__)\n\n@app.route('/')\ndef show():\n\treturn render_template('hello.html')\nif
__name__ == '__main__':\n\tapp.run(host='0.0.0.0',port=3000)" > ~/web/hi.py

        docker exec Abathur /bin/bash -c "python /root/web/hi.py"
        docker-machine ip Char
        rm -Rf ~/web
```

Flask будет искать шаблоны в папке templates. Поэтому, если ваше приложение выполнено в виде модуля, эта папка будет рядом с модулем.

Первый случай - модуль:

```
/application.py
/templates
    /hello.html
```

На хосте:

```
        vi ~/web/templates/hello.html
```

```
<!DOCTYPE html>
<html>
        <body>
                <h1>Hello world</h1>
        </body>
</html>
```

```
        vi ~/web/hi.py
```

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/')
def show():
        return render_template('hello.html')
if __name__ == '__main__':
        app.run(host='0.0.0.0',port=3000)
```

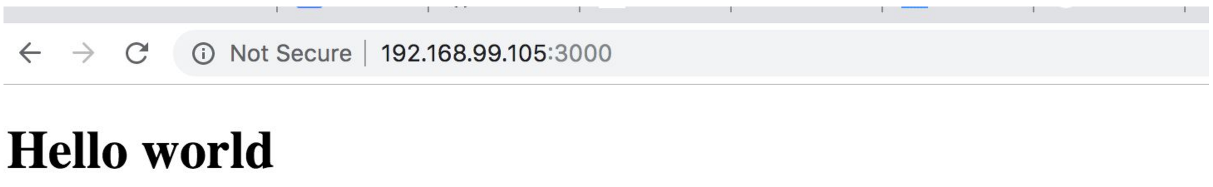
Итак, что же делает этот код?

1. Сначала мы импортировали класс Flask. Экземпляр этого класса и будет вашим WSGI-приложением. Для визуализации шаблона вы можете использовать метод render_template(). Всё, что вам необходимо - это указать имя шаблона, а также переменные в виде именованных аргументов, которые вы хотите передать движку обработки шаблонов.
2. Далее мы создаём экземпляр этого класса. Первый аргумент - это имя модуля или пакета приложения. Если вы используете единственный модуль (как в этом примере), вам следует использовать __name__, потому что в

зависимости от того, запущен ли код как приложение, или был импортирован как модуль, это имя будет разным ('__main__' или актуальное имя импортированного модуля соответственно). Это нужно, чтобы Flask знал, где искать шаблоны, статические файлы и прочее. Для дополнительной информации, смотрите документацию Flask.

3. Далее, мы используем декоратор route(), чтобы сказать Flask, какой из URL должен запускать нашу функцию.
4. And what we did is we imported the render_template method from the flask framework and then we passed an HTML file to that method. The method will generate a jinja2 template object out of that HTML and return it to the browser when the user visits associated URL.

```
docker exec Abathur /bin/bash -c "python /root/web/hi.py"
 * Serving Flask app "hi" (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 http://0.0.0.0:3000/ (Press CTRL+C to quit)
192.168.99.1 - - [28/Sep/2019 14:11:27] "GET / HTTP/1.1" 200 -
192.168.99.1 - - [28/Sep/2019 14:27:42] "GET / HTTP/1.1" 200 -
```



Hello world

```
Команды:
docker run -di --name Abathur -v $HOME:/root -p 3000:3000 python:2-slim
docker exec Abathur /bin/bash -c "pip install Flask"
mkdir ~/web
mkdir ~/web/templates
echo -e "<\!DOCTYPE html>\n<html>\n\t<body>\n\t\t<h1>Hello
world</h1>\n\t</body>\n</html>" > ~/web/templates/hello.html
echo -e "from flask import Flask, render_template\napp =
Flask(__name__)\n\n@app.route('/')\ndef show():\n\treturn render_template('hello.html')\nif
_name_ == '__main__':\n\tapp.run(host='0.0.0.0',port=3000)" > ~/web/hi.py
docker exec Abathur /bin/bash -c "python /root/web/hi.py"
```

20. Create a local swarm, the Char virtual machine should be its manager.
Создайте Сварм, в котором машина Char будет лидером.

Лучшее объяснение - https://docs.docker.com/engine/swarm/swarm-tutorial/create-swarm/
Docker: Orchestration of multi-container apps with Swarm and Compose - https://www.ionos.com/digitalguide/server/know-how/docker-orchestration-with-swarm-and-compose/
Сети Docker изнутри: связь между контейнерами в Docker Swarm и Overlay-сети - https://habr.com/ru/post/334004/
От создания докер-кластера до сервисов - https://habr.com/ru/company/southbridge/blog/310606/

Выписки из статей:

В случае отказа узла контейнеры которого были задействованы, swarm обнаружит, что желаемое состояние не совпадает с действительным и автоматически исправит ситуацию путем перераспределения контейнеров на другие доступные узлы.

Захват трафика в этом примере показал, что если ты видишь трафик между хостами, то увидишь и трафик внутри контейнеров, проходящий по overlay-сети. Именно поэтому в Docker есть опция шифрования. Можно запустить автоматическое IPSec-шифрование vxlan-туннелей, просто добавив --opt encrypted при создании сети.

Swarm

Ответ команды:

```
Swarm initialized: current node (kz6xyq00jlkbjrpq8lf1hpihd) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-1tozuup0b1kjc0wecs9vpb4l6vvlcgphjfk6r37nc40q6o1xaf-a576xbd983z56uawoaqtu31ki 192.168.99.100:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

```
docker node ls
ID                          HOSTNAME    STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
kz6xyq00jlkbjrpq8lf1hpihd *   Char        Ready     Active          Leader            18.09.9
```

Команда: docker swarm init --advertise-addr $(docker-machine ip Char)

21. Create another virtual machine with docker-machine using the virtualbox driver, and name it Aiur. Создайте еще одну машину через virtualbox и назовите ее Aiur

Команда: docker-machine create --driver=virtualbox Aiur

22. Turn Aiur into a slave of the local swarm in which Char is leader (the command to take control of Aiur is not requested). Введите машину Aiur в сварм Char и сделайте ее работником.

docker swarm join-token: https://docs.docker.com/engine/reference/commandline/swarm_join-token/

docker swarm join: https://docs.docker.com/engine/reference/commandline/swarm_join/

```
        docker-machine ssh Aiur
   ( '>')
  /) TC (\   Core is distributed with ABSOLUTELY NO WARRANTY.
 (/-_--_-\)           www.tinycorelinux.net

        docker swarm join --token
SWMTKN-1-6bxj44pxct97swd9sqvset75m7q2em7t802vpwgq8zefmkk474-aih3l5xpa0s82rim
7yz9bx8qw 192.168.99.102:2377
This node joined a swarm as a worker.
        docker@Aiur:~$ exit
logout
su-e3% docker node ls
ID                          HOSTNAME        STATUS          AVAILABILITY      MANAGER
STATUS      ENGINE VERSION
b8hgltz7zqze2a4594e41f12b     Aiur            Ready           Active
18.09.9
assleasa880otbv8vcb53o16z *   Char            Ready           Active            Leader
18.09.9
```

После этого также вышла и снова зашла в Swarm:

```
        docker-machine ssh Aiur "docker swarm leave"
Node left the swarm.
        docker node ls
ID                          HOSTNAME        STATUS          AVAILABILITY      MANAGER
STATUS      ENGINE VERSION
b8hgltz7zqze2a4594e41f12b     Aiur            Down            Active
18.09.9
assleasa880otbv8vcb53o16z *   Char            Ready           Active            Leader
18.09.9

docker-machine ssh Aiur "docker swarm join --token $(docker swarm join-token -q worker)
```

```
$(docker-machine ip Char):2377"
This node joined a swarm as a worker.
su-e3% docker node ls
ID                          HOSTNAME         STATUS          AVAILABILITY      MANAGER
STATUS      ENGINE VERSION
b8hgltz7zqze2a4594e41f12b     Aiur             Down            Active
18.09.9
qmmadiadb9s51rlqygg77mre6     Aiur             Ready           Active
18.09.9
assleasa880otbv8vcb53o16z *   Char             Ready           Active            Leader
18.09.9
```

Команда: docker-machine ssh Aiur "docker swarm join --token $(docker swarm join-token -q worker) $(docker-machine ip Char):2377"

23. Create an overlay-type internal network that you will name overmind. Создайте внутреннюю оверлейную сеть для сварма и назовите ее overmind.

Network in Swarm

Сети Docker изнутри: как Docker использует iptables и интерфейсы Linux- https://habr.com/en/post/333874/

Оверлейная сеть (от англ. Overlay Network) — общий случай логической сети, создаваемой поверх другой сети. Узлы оверлейной сети могут быть связаны либо физическим соединением, либо логическим, для которого в основной сети существуют один или несколько соответствующих маршрутов из физических соединений. Примерами оверлеев являются сети VPN и одноранговые сети, которые работают на основе интернета и представляют собой «надстройки» над классическими сетевыми протоколами, предоставляя широкие возможности, изначально не предусмотренные разработчиками основных протоколов.

```
        docker network ls
NETWORK ID        NAME             DRIVER           SCOPE
fb5376887047      bridge           bridge           local
f4eb4c5f7be2      docker_gwbridge  bridge           local
2b4c40bf6f9b      host             host             local
fekniu1i7okq      ingress          overlay          swarm
2cd554fc02b0      none             null             local
```

Ответ команды:

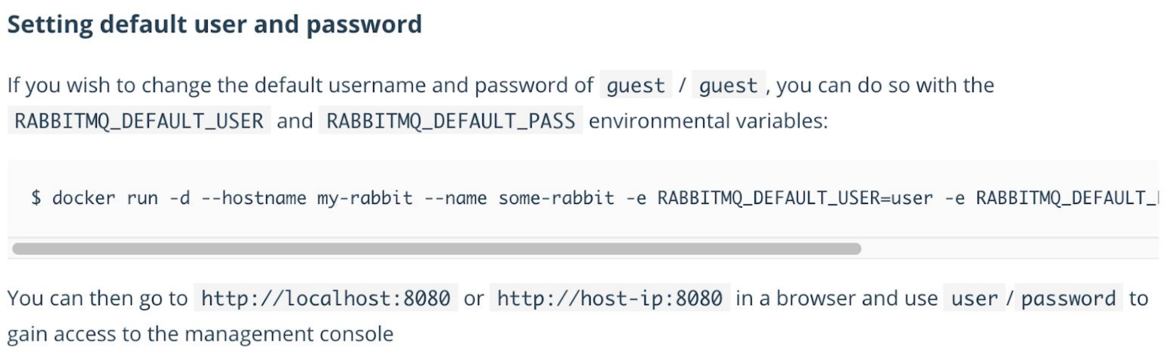
```
a9x2dyxbnsy7ns2ltabwgqabu
        docker network ls
NETWORK ID        NAME             DRIVER           SCOPE
fb5376887047      bridge           bridge           local
f4eb4c5f7be2      docker_gwbridge  bridge           local
2b4c40bf6f9b      host             host             local
```

```
fekniu1i7okq      ingress          overlay          swarm
2cd554fc02b0       none             null            local
a9x2dyxbnsy7       overmind          overlay          swarm
```

Команда: docker network create -d overlay –internal overmind

24. Launch a rabbitmq SERVICE that will be named orbital-command. You should define a specific user and password for the RabbitMQ service, they can be whatever you want. This service will be on the overmind network. Запустите сервис rabbitmq и назовите его orbital-command. Нужно определить юзера и пароль для этого сервиса. Этот сервис будет выполняться в рамках сети overmind.

Не работает без юзера и пароля.



Setting default user and password

If you wish to change the default username and password of `guest` / `guest`, you can do so with the `RABBITMQ_DEFAULT_USER` and `RABBITMQ_DEFAULT_PASS` environmental variables:

```
$ docker run -d --hostname my-rabbit --name some-rabbit -e RABBITMQ_DEFAULT_USER=user -e RABBITMQ_DEFAULT_
```

You can then go to `http://localhost:8080` or `http://host-ip:8080` in a browser and use `user` / `password` to gain access to the management console

Ответ команды:

```
hrw24ng9ifav8zunggfc5s04k
su-e3% docker service ls
ID              NAME              MODE              REPLICAS          IMAGE             PORTS
hrw24ng9ifav       orbital-command     replicated          1/1              rabbitmq:latest
```

```
          docker service ps orbital-command (смотрим таски сервиса)
ID              NAME              IMAGE             NODE              DESIRED STATE
CURRENT STATE          ERROR             PORTS
m6slz3ha6sb9       orbital-command.1   rabbitmq:latest    Char              Running
Running 6 minutes ago
          docker service inspect -f"{{.Spec.TaskTemplate.ContainerSpec}}" orbital-command
{rabbitmq:latest@sha256:fd7bd829f35c112cf99548f6d9a2f49936ff093b8a01f4acaeab4a2d1
5698996 map[] [] []  [RABBITMQ_DEFAULT_USER=sschmele
RABBITMQ_DEFAULT_PASS=pass]   [] <nil> 0xc0004bea3a  false false false [] 10s <nil> []
0xc0002f2500 [] [] default map[]}
```

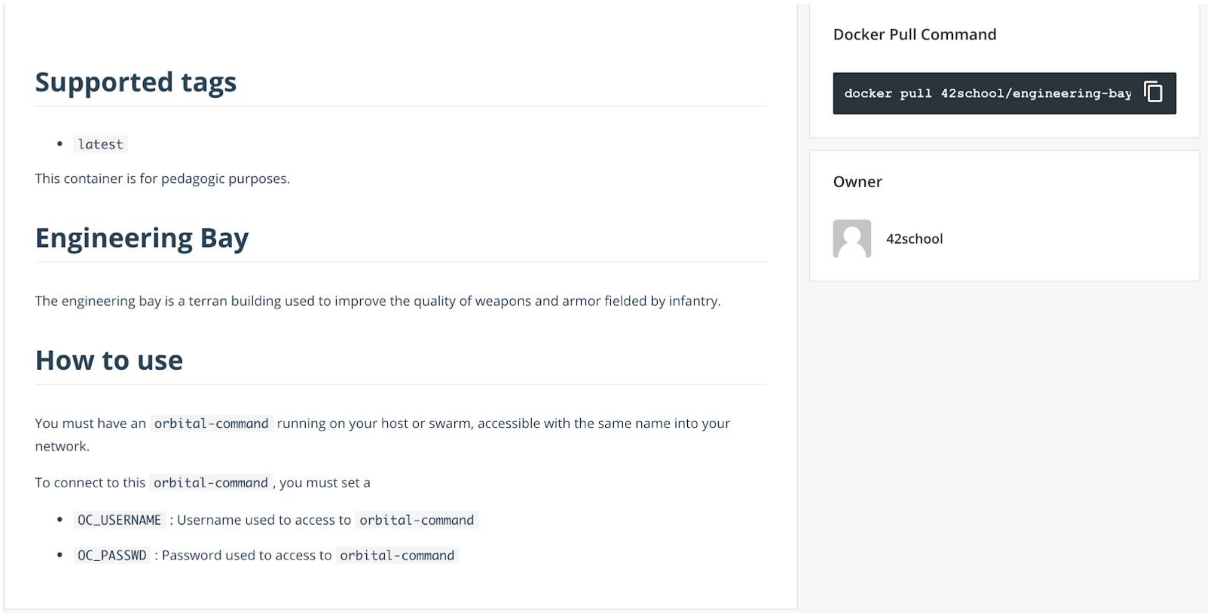
```
Команда: docker service create \
–name orbital-command \
-e RABBITMQ_DEFAULT_USER=sschmele \
-e RABBITMQ_DEFAULT_PASS=pass \
-d rabbitmq
```

25. List all the services of the local swarm.

docker service ls

26. Launch a 42school/engineering-bay service in two replicas and make sure that the service works properly (see the documentation provided at hub.docker.com). This service will be named engineering-bay and will be on the overmind network. Запустите в 2-ом экземпляре сервис 42school/engineering-bay , который должен называться engineering-bay и работать в рамках сети overmind.

Не работает без юзера и пароля.
Очень круто следить за работой через lazy docker - вроде этот https://github.com/jesseduffield/lazydocker
Описание с докер хаба:



Supported tags

- latest

This container is for pedagogic purposes.

Engineering Bay

The engineering bay is a terran building used to improve the quality of weapons and armor fielded by infantry.

How to use

You must have an `orbital-command` running on your host or swarm, accessible with the same name into your network.

To connect to this `orbital-command` , you must set a

- `OC_USERNAME` : Username used to access to `orbital-command`
- `OC_PASSWD` : Password used to access to `orbital-command`

Docker Pull Command

```
docker pull 42school/engineering-bay
```

Owner

42school

docker service ps engineering-bay

```
ID              NAME                  IMAGE                              NODE          DESIRED STATE
CURRENT STATE          ERROR                    PORTS
xjzanmo5ncyt      engineering-bay.1      42school/engineering-bay:latest   Char
Ready          Ready 4 seconds ago
inc2utg17tww      \_ engineering-bay.1   42school/engineering-bay:latest   Char
Shutdown         Failed 4 seconds ago   "task: non-zero exit (1)"
aisrsmp71wg1       \_ engineering-bay.1   42school/engineering-bay:latest   Char
Shutdown         Failed 10 seconds ago   "task: non-zero exit (1)"
oesjgvh25f9g      \_ engineering-bay.1   42school/engineering-bay:latest   Char
Shutdown         Failed 16 seconds ago   "task: non-zero exit (1)"
zlsiw0iuhe50      \_ engineering-bay.1   42school/engineering-bay:latest   Char
Shutdown         Failed 22 seconds ago   "task: non-zero exit (1)"
j1l0mjwe5nua      engineering-bay.2      42school/engineering-bay:latest   Char
Ready          Ready 4 seconds ago
```

```
my0tdjj4cekf        \_ engineering-bay.2   42school/engineering-bay:latest   Char
Shutdown            Failed 4 seconds ago    "task: non-zero exit (1)"
oayo8o6tpcsl        \_ engineering-bay.2   42school/engineering-bay:latest   Char
Shutdown            Failed 10 seconds ago   "task: non-zero exit (1)"
wmw9palpkfvz        \_ engineering-bay.2   42school/engineering-bay:latest   Char
Shutdown            Failed 16 seconds ago   "task: non-zero exit (1)"
6et5j5robb99        \_ engineering-bay.2   42school/engineering-bay:latest   Char
Shutdown            Failed 22 seconds ago   "task: non-zero exit (1)"
```

```
docker service inspect -f"{{.Spec.TaskTemplate.ContainerSpec}}"engineering-bay
```

Так как функционал swarm распределяет сервисы между нодами кластера (машинами), то в docker ps отдельного кластера сервисы могут то появляться, то исчезать:

```
vo-k6% docker ps
CONTAINER ID       IMAGE            COMMAND                CREATED           STATUS
PORTS                        NAMES
a2a02e49c972       nginx            "nginx -g 'daemon of…"   54 minutes ago      Up About
an hour    0.0.0.0:5000->80/tcp              overlord
288dbd92041e       rabbitmq:latest     "docker-entrypoint.s…"   About an hour ago   Up
About an hour    4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.m4wxl0ktkvptsf2yt440mdpit
vo-k6% docker ps
CONTAINER ID       IMAGE            COMMAND                CREATED           STATUS
PORTS                        NAMES
a2a02e49c972       nginx            "nginx -g 'daemon of…"   54 minutes ago      Up About
an hour    0.0.0.0:5000->80/tcp              overlord
288dbd92041e       rabbitmq:latest     "docker-entrypoint.s…"   About an hour ago   Up
About an hour    4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.m4wxl0ktkvptsf2yt440mdpit
vo-k6% docker ps
CONTAINER ID       IMAGE            COMMAND                CREATED           STATUS
PORTS                        NAMES
a2a02e49c972       nginx            "nginx -g 'daemon of…"   54 minutes ago      Up About
an hour    0.0.0.0:5000->80/tcp              overlord
288dbd92041e       rabbitmq:latest     "docker-entrypoint.s…"   About an hour ago   Up
About an hour    4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.m4wxl0ktkvptsf2yt440mdpit
vo-k6% docker ps
CONTAINER ID       IMAGE            COMMAND                CREATED           STATUS
PORTS                        NAMES
a2a02e49c972       nginx            "nginx -g 'daemon of…"   54 minutes ago      Up About
an hour    0.0.0.0:5000->80/tcp              overlord
288dbd92041e       rabbitmq:latest     "docker-entrypoint.s…"   About an hour ago   Up
About an hour    4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.m4wxl0ktkvptsf2yt440mdpit
vo-k6% docker ps
CONTAINER ID       IMAGE                       COMMAND               CREATED
```

```
STATUS              PORTS                           NAMES
b67d35dcafa3       42school/engineering-bay:latest   "/bin/sh -c 'python …"   5 seconds
ago       Up Less than a second
engineering-bay.1.r2lqan5p96b1rv49um1npk4dv
a2a02e49c972       nginx                          "nginx -g 'daemon of…"   55 minutes ago
Up About an hour       0.0.0.0:5000->80/tcp               overlord
288dbd92041e       rabbitmq:latest               "docker-entrypoint.s…"   About an hour ago
Up About an hour       4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.m4wxl0ktkvptsf2yt440mdpit
```

```
Команда: docker service create -d \
--name engineering-bay \
--network overmind --replicas=2 \
-e OC_USERNAME=sschmele \
-e OC_PASSWD=pass \
42school/engineering-bay:latest
```

27. Get the real-time logs of one of the tasks of the engineering-bay service. Выведите логи выполнения заданий сервиса engineering-bay

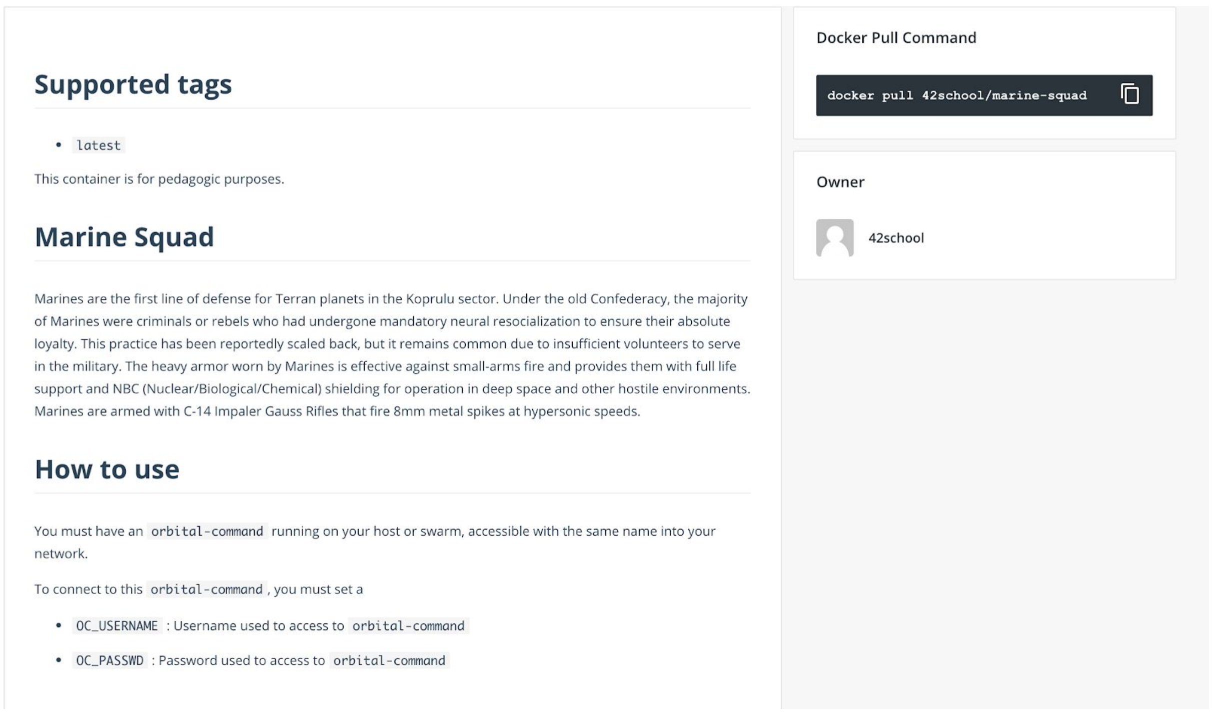
Ответ команды:

```
engineering-bay.2.ohcvz7esceo0@Char    | Traceback (most recent call last):
engineering-bay.2.ohcvz7esceo0@Char    |   File "zergrush.py", line 12, in <module>
engineering-bay.2.ohcvz7esceo0@Char    |     connection =
pika.BlockingConnection(params)
engineering-bay.2.ohcvz7esceo0@Char    |   File
"/usr/local/lib/python2.7/site-packages/pika/adapters/blocking_connection.py", line 339, in
__init__
engineering-bay.2.ohcvz7esceo0@Char    |     self._process_io_for_connection_setup()
engineering-bay.2.ohcvz7esceo0@Char    |   File
"/usr/local/lib/python2.7/site-packages/pika/adapters/blocking_connection.py", line 374, in
_process_io_for_connection_setup
engineering-bay.2.ohcvz7esceo0@Char    |     self._open_error_result.is_ready)
engineering-bay.2.ohcvz7esceo0@Char    |   File
"/usr/local/lib/python2.7/site-packages/pika/adapters/blocking_connection.py", line 395, in
_flush_output
engineering-bay.2.ohcvz7esceo0@Char    |     raise exceptions.ConnectionClosed()
engineering-bay.2.ohcvz7esceo0@Char    | pika.exceptions.ConnectionClosed
```

```
Команда: docker service logs -f $(docker service ps engineering-bay -f
"name=engineering-bay.2" -q | awk 'NR == 1 {print}')
```

28. ... Damn it, a group of zergs is attacking orbital-command, and shutting down the engineering-bay service won't help at all... You must send a troup of Marines to eliminate

the intruders. Launch a 42school/marine-squad in two replicas, and make sure that the service works properly (see the documentation provided at hub.docker.com). This service will be named... marines and will be on the overmind network. Нужно запустить сервис 42school/marine-squad в 2-ом экземпляре под названием marines и в рамках сети overmind.



Supported tags

- `latest`

This container is for pedagogic purposes.

Marine Squad

Marines are the first line of defense for Terran planets in the Koprulu sector. Under the old Confederacy, the majority of Marines were criminals or rebels who had undergone mandatory neural resocialization to ensure their absolute loyalty. This practice has been reportedly scaled back, but it remains common due to insufficient volunteers to serve in the military. The heavy armor worn by Marines is effective against small-arms fire and provides them with full life support and NBC (Nuclear/Biological/Chemical) shielding for operation in deep space and other hostile environments. Marines are armed with C-14 Impaler Gauss Rifles that fire 8mm metal spikes at hypersonic speeds.

How to use

You must have an `orbital-command` running on your host or swarm, accessible with the same name into your network.

To connect to this `orbital-command` , you must set a

- `OC_USERNAME` : Username used to access to `orbital-command`
- `OC_PASSWD` : Password used to access to `orbital-command`

Docker Pull Command

```
docker pull 42school/marine-squad
```

Owner

42school

Ответ команды:

```
bs82dpc69182ob9oao088sqcw

                 docker service ls
ID           NAME          MODE          REPLICAS       IMAGE
PORTS
5bmu49uckscq     engineering-bay    replicated      2/2
42school/engineering-bay:latest
fb6c1hjl9gnl     marines        replicated      1/2         42school/marine-squad:latest
x38aq3j1mg9s     orbital-command    replicated      1/1         rabbitmq:latest
vo-k6% docker service ls
ID           NAME          MODE          REPLICAS       IMAGE
PORTS
5bmu49uckscq     engineering-bay    replicated      2/2
42school/engineering-bay:latest
fb6c1hjl9gnl     marines        replicated      0/2         42school/marine-squad:latest
x38aq3j1mg9s     orbital-command    replicated      1/1         rabbitmq:latest
                 docker service ls
ID           NAME          MODE          REPLICAS       IMAGE
PORTS
```

```
5bmu49uckscq       engineering-bay     replicated         0/2
42school/engineering-bay:latest
fb6c1hjl9gnl       marines           replicated         1/2              42school/marine-squad:latest
x38aq3j1mg9s       orbital-command     replicated         1/1              rabbitmq:latest
vo-k6% docker service ls
ID                 NAME              MODE              REPLICAS           IMAGE
PORTS
5bmu49uckscq       engineering-bay     replicated         0/2
42school/engineering-bay:latest
fb6c1hjl9gnl       marines           replicated         2/2              42school/marine-squad:latest
x38aq3j1mg9s       orbital-command     replicated         1/1              rabbitmq:latest

                          docker service ps marines
ID                 NAME              IMAGE                       NODE              DESIRED STATE
CURRENT STATE          ERROR                      PORTS
oemezaow4udd       marines.1          42school/marine-squad:latest   Aiur              Running
Running 4 seconds ago
b6jh5xpdsbhw        \_ marines.1       42school/marine-squad:latest   Char
Shutdown           Failed 9 seconds ago    "task: non-zero exit (1)"
rmo0un1a3j7o        \_ marines.1       42school/marine-squad:latest   Aiur              Shutdown
Failed 20 seconds ago   "task: non-zero exit (1)"
5r4rncrwoi02        \_ marines.1       42school/marine-squad:latest   Aiur              Shutdown
Failed 31 seconds ago   "task: non-zero exit (1)"
kfwdnhroqw18        \_ marines.1       42school/marine-squad:latest   Aiur              Shutdown
Failed 42 seconds ago   "task: non-zero exit (1)"
phs1f7t6p76p       marines.2          42school/marine-squad:latest   Aiur              Ready
Ready 2 seconds ago
f993ovff30m7        \_ marines.2       42school/marine-squad:latest   Char              Shutdown
Failed 2 seconds ago   "task: non-zero exit (1)"
l8imxpwkn6g8        \_ marines.2       42school/marine-squad:latest   Aiur              Shutdown
Failed 13 seconds ago   "task: non-zero exit (1)"
o5a3urd451mz        \_ marines.2       42school/marine-squad:latest   Char
Shutdown           Failed 24 seconds ago   "task: non-zero exit (1)"
odousgaddt4q        \_ marines.2       42school/marine-squad:latest   Char
Shutdown           Failed 34 seconds ago   "task: non-zero exit (1)"
```

```
Команда: docker service create -d \
--name marines \
--network overmind --replicas=2 \
-e OC_USERNAME=sschmele \
-e OC_PASSWD=word \
42school/marine-squad:latest
```

29. Display all the tasks of the marines service.

Команда: docker service ps marines

30. Increase the number of copies of the marines service up to twenty, because there's never enough Marines to eliminate Zergs. (Remember to take a look at the tasks and logs of the service, you'll see, it's fun.) Увеличьте количество копий сервиса marines до 20, чтобы они одолели зоргов.

Ответ команды:
marines

```
Команда: docker service update –replicas=20 -d marines
```

Swarm выводы

Вывод из первой части про swarm:
Контейнеры, запущенные на разных машинах не видны друг другу, вне зависимости от статуса в swarm
Сервисы - это “контейнеры” для всего кластера - собрания машин с распределением на менеджеров и исполнителей
Например:

1) У меня есть машина Char-менеджер, на которой больше нет запущенных контейнеров
2) У меня есть вторая машина Aiur-работник, на которой больше нет запущенных контейнеров
3) Я запускаю сервис с 1 репликой на Char и расширяю его до 12 через update

```
                              eval $(docker-machine env Char) - на машине Char
                              docker ps - контейнеров нет
CONTAINER ID      IMAGE          COMMAND             CREATED          STATUS
PORTS                          NAMES

                              docker service create \
                              --name orbital-command \
                             -e RABBITMQ_DEFAULT_USER=sschmele \
                             -e RABBITMQ_DEFAULT_PASS=pass \
                             -d rabbitmq
p4x2aysmjx1c6wbmd7g6jco7n

                             docker ps
CONTAINER ID      IMAGE          COMMAND             CREATED          STATUS
PORTS                          NAMES
9041882fed34      rabbitmq:latest     "docker-entrypoint.s…"   5 seconds ago      Up 4
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.xduv0sp7i5pzk366y2xba22ny

                            docker service update --replicas=12 -d orbital-command
orbital-command
                            docker ps
```

```
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                          NAMES
85227d47766c        rabbitmq:latest     "docker-entrypoint.s…"   18 seconds ago      Up 17
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.3.l8oix4k4zjn6s3xut92b8e1ru
494904af4d63        rabbitmq:latest     "docker-entrypoint.s…"   18 seconds ago      Up 17
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.2.juq9spj9whxdht8pvx5wlvqlo
20439f41c3ff        rabbitmq:latest     "docker-entrypoint.s…"   18 seconds ago      Up 17
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.11.mvziq6439n3qy3xhwi51rpln4
585ffb4f9bd5        rabbitmq:latest     "docker-entrypoint.s…"   18 seconds ago      Up 17
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.5.jtp6fby4h5e31ebrpkqtnbpn0
8f733572de65        rabbitmq:latest     "docker-entrypoint.s…"   18 seconds ago      Up 17
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.9.oqgk8jqif8zrwv2aa4gyevci3
9041882fed34        rabbitmq:latest     "docker-entrypoint.s…"   2 minutes ago       Up 2
minutes        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.1.xduv0sp7i5pzk366y2xba22ny
                              eval $(docker-machine env Aiur) - меняю машины
                              docker ps - до запуска сервиса контейнеров не было
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                          NAMES
f9af65c047ba        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 27
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.12.zbmxontjfct80305c09p0s4xf
d73c45b9782d        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 27
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.7.y68arxkfl21g7s4jjv1b3trnd
bb996c265afb        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 27
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.10.lakyk1xra03bk983pe0w50axw
8946291f3cda        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 27
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.6.ihelgzc4bol4vlp2znhybw6vu
6ddfcc2cf783        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 27
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.4.pbzw5dc8w8le78g6e8w1b80xm
ddcde7967625        rabbitmq:latest     "docker-entrypoint.s…"   29 seconds ago      Up 28
seconds        4369/tcp, 5671-5672/tcp, 25672/tcp
orbital-command.8.owmzuaoveugk3gpdewlwc8aui
```

ВАЖНО! Удалять сервис можно только от менеджера

31. Force quit and delete all the services on the local swarm, in one command. Удалите все сервисы сварма (одной командой)

Ответ команды:
```
bs82dpc69182
ipd4ssbpvkg2
```

Команда: docker service rm $(docker service ls -q)

32. Force quit and delete all the containers (whatever their status), in one command. Завершите работу и удалите все контейнеры одной командой

docker container rm $(docker ps -q) - если попытаться сделать rm
```
Error response from daemon: You cannot remove a running container
a2a02e49c97280c074207fe6d2a7b30564e1d0afa6b41890ad84c629c5bcbee7. Stop the
container before attempting removal or force remove
```

docker container kill $(docker ps -q)

Команда: docker rm --force $(docker ps -a -q)

33. Delete all the container images stored on the Char virtual machine, in one command as well. Удалите все образы на машине Char одной командой

Ответ команды:
```
Untagged:
rabbitmq@sha256:5927e7c2bb4caf4f2a478a5ebcf9d7a8caebf9f95d554e61d29e0190f8582ff
4
Deleted:
sha256:b9e17734a1b22bdf1ee00e89df68297cd4257935c3eefff99e9cd5a9c00b84ce
Deleted:
sha256:0deb83ab13def44db772bb6899be5142cfb57c25c3e06f9206a21698d3a7d131
Deleted:
sha256:fdb42dd5cb5cf4bcbb6b6e4c6fc859725520e895b99f58e8c361922e5ca75b82
Deleted:
sha256:ac4b212ec4d2ab82e642861f84f3773e66e0d617ffdba59cd5a161355e135419
Deleted:
sha256:a2935992da82e29d01bb2587680ba054f2f3c1d695b13d928ce7cbdbd79d4940
Deleted:
sha256:7a8be7a318b0601bab8d183132c65e94090ac54b37ff60cc94ba49653feb060c
Deleted:
sha256:4abf2272224deebb1bd133b0ca30a45f579c4677c93c5b48ac35d5e72ad6c974
Deleted:
sha256:bd416bed302bc2f061a2f6848a565483a5f265932d2d4fa287ef511b7d1151c8
Deleted:
sha256:5308e2e4a70bd4344383b8de54f8a52b62c41afb5caa16310326debd1499b748
```

```
Deleted:
sha256:dab02287e04c8b8207210b90b4056bd865fcfab91469f39a1654075f550c5592
Deleted:
sha256:a1aa3da2a80a775df55e880b094a1a8de19b919435ad0c71c29a0983d64e65db
Untagged:
42school/engineering-bay@sha256:5bc69a7b7ad5c5de54cffcfae09e94960ee66437ae0f7f2
d270bde8e49559a62
Deleted: sha256:faa22fd5eedd1421ac84b89b728645f052914c560aae2cfd9431ff781fe5833a
Deleted:
sha256:ea1c5846bc2910f1dab95d4a9ca3d40ff90cbb62749cacdb9746683781f84191
Deleted:
sha256:db7ce7fc70649b8d53344fd4f696f830c53cdd1645b4ca8ad5b2852297e90235
Deleted:
sha256:01e37fc3133816108efbffcee5c9648b934f4eb20ebc2c7b67a68976413105a3
Deleted:
sha256:318da86d7b0a95efdbe3269ceacd7c438aa4701e72092da60dde4b0237d430e0
Deleted: sha256:7cbcbac42c44c6c38559e5df3a494f44987333c8023a40fec48df2fce1fc146b
```

Команда: docker rmi $(docker images -a -q)

34. Delete the Aiur virtual machine without using rm -rf. Удалите машину Aiur без использования rm -rf (то есть из директории докера)

Команда: docker-machine rm -y Aiur

Bloopers

Начало работы с docker-machine

Если вылезает ошибка: docker: Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?.
See 'docker run --help'.

Во-первых, нужна установленная виртуалка, а также .docker в домашней директории - это демон докера, через который он работает
Во-вторых, докер машина, причем запущенная: docker-machine start [NAME] (лежат в папке ~/goinfre/.docker/machine/machines (после переноса)
Для как такового “запуска” демона нужно закрепить среду запуска:
eval $(docker-machine env [docker-machine name])
        env | grep DOCKER (на хосте через первый терминал)

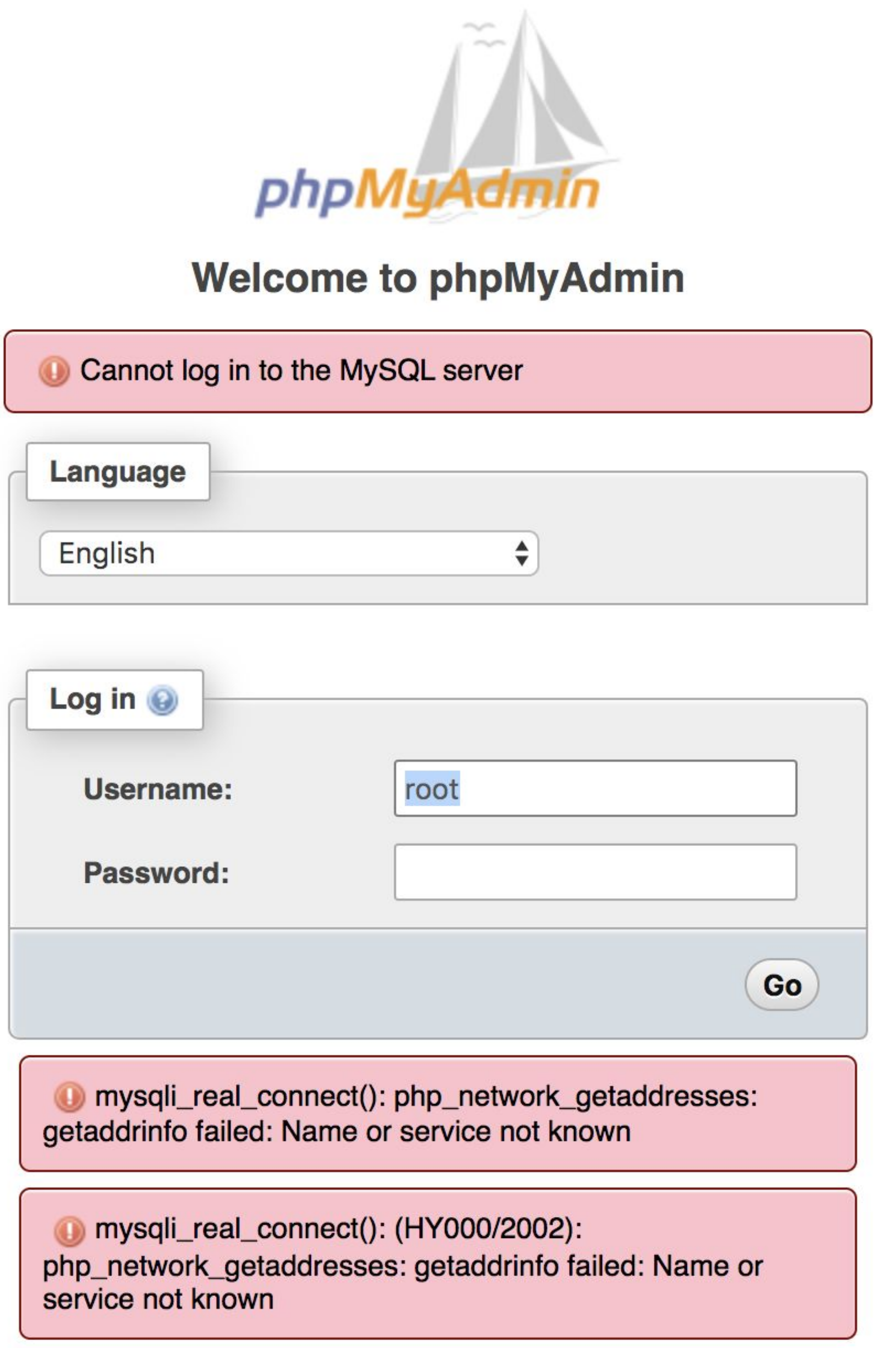
```
DOCKER_TLS_VERIFY=1
DOCKER_HOST=tcp://192.168.9*.1*0:2376
DOCKER_CERT_PATH=/Users/sschmele/.docker/machine/machines/Char
DOCKER_MACHINE_NAME=Char
```

Закрепление среды нужно делать в каждом терминале, где будет производиться работа с докером.

Wordpress and PHPadmin

Что будет происходить, если неправильно настроить mysql сервер и привязать контейнер

Сперва задание 15:
Ошибка и как мы с ней боролись:



Welcome to phpMyAdmin

Cannot log in to the MySQL server

Language: English

Log in

Username: root

Password:

Go

mysqli_real_connect(): php_network_getaddresses: getaddrinfo failed: Name or service not known

mysqli_real_connect(): (HY000/2002): php_network_getaddresses: getaddrinfo failed: Name or service not known

Первая версия 12 задания была:

```
docker run -d --name spawning-pool \
--restart on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--mount source=hatchery,target=/var/lib/mysql \
mysql
```

Вторая:

```
docker run -d --name spawning-pool \
--restart on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--mount source=hatchery,target=/var/lib/mysql \
mysql \
--default-authentication-plugin=mysql_native_password
```

Третья:

```
docker run --volume=hatchery:/var/lib/mysql \
--restart=on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--name=spawning-pool \
-d mysql:latest \
--default-authentication-plugin=mysql_native_password
```

Четвертая и финальная с портами:

```
docker run --volume=hatchery:/var/lib/mysql \
--restart=on-failure \
-e MYSQL_ROOT_PASSWORD=Kerrigan \
-e MYSQL_DATABASE=zerglings \
--name=spawning-pool \
-p 3306:3306 \
-d mysql:latest \
--default-authentication-plugin=mysql_native_password
```

И тут оно стало работать и на других версиях.
Плагин необходим для доступа к аутентификации в восьмой версии mysql - раньше так прописывать было не надо.

Порты нужны точно. Mount или Volume не влияют

И ТУТ МЫ ПОНЯЛИ
Первая версия 15-го задания:

```
docker run -d --name roach-warden -p 8081:80 --link spawning-pool:sp
phpmyadmin/phpmyadmin
```

Вторая и конечная:

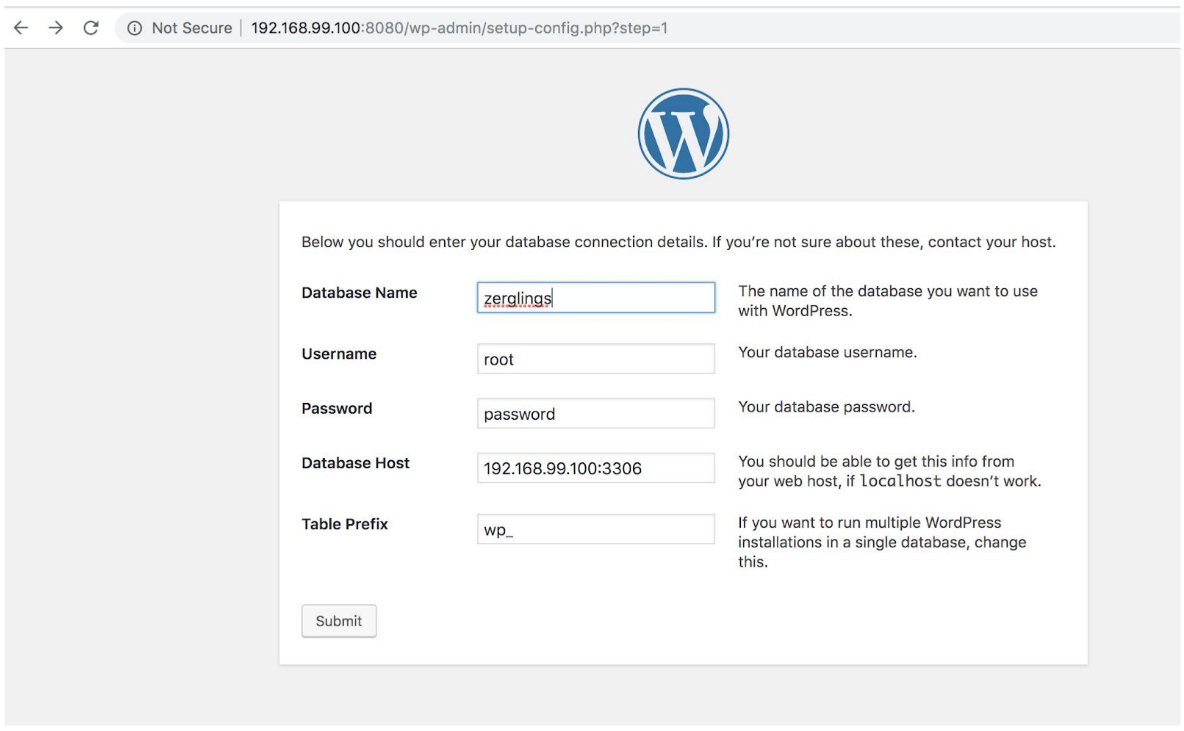
```
docker run -d --name roach-warden -p 8081:80 --link spawning-pool:db
phpmyadmin/phpmyadmin
```

ВЛИЯЕТ DB. Это какой-то алиас. Видимо.

Так вот. Проблема опять в db и в настройке на странице Wordpress

БЫЛО:

```
docker run -d --name lair -p 8080:80 --link spawning-pool:sp wordpress
```

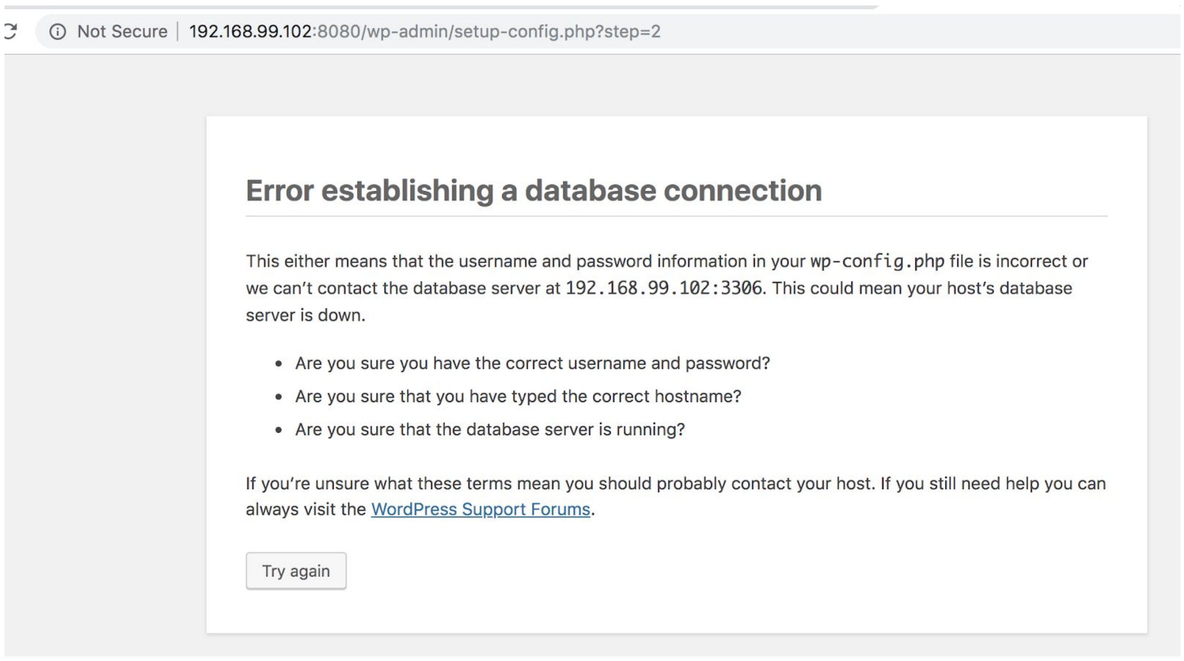

Not Secure | 192.168.99.100:8080/wp-admin/setup-config.php?step=1

Below you should enter your database connection details. If you're not sure about these, contact your host.

Database Name	zerglings	The name of the database you want to use with WordPress.
Username	root	Your database username.
Password	password	Your database password.
Database Host	192.168.99.100:3306	You should be able to get this info from your web host, if `localhost` doesn't work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.

Раньше я также писала в графах:
Database Name: spawning-pool / spawning-pool/zerglings
Username: root
Password Kerrigan
Database Host: localhost / localhost:3306 / 192.168.99.100

Теперь непонятно, все ли они точно были неправильны. Теперь подключается всегда.


Not Secure | 192.168.99.102:8080/wp-admin/setup-config.php?step=2

Error establishing a database connection

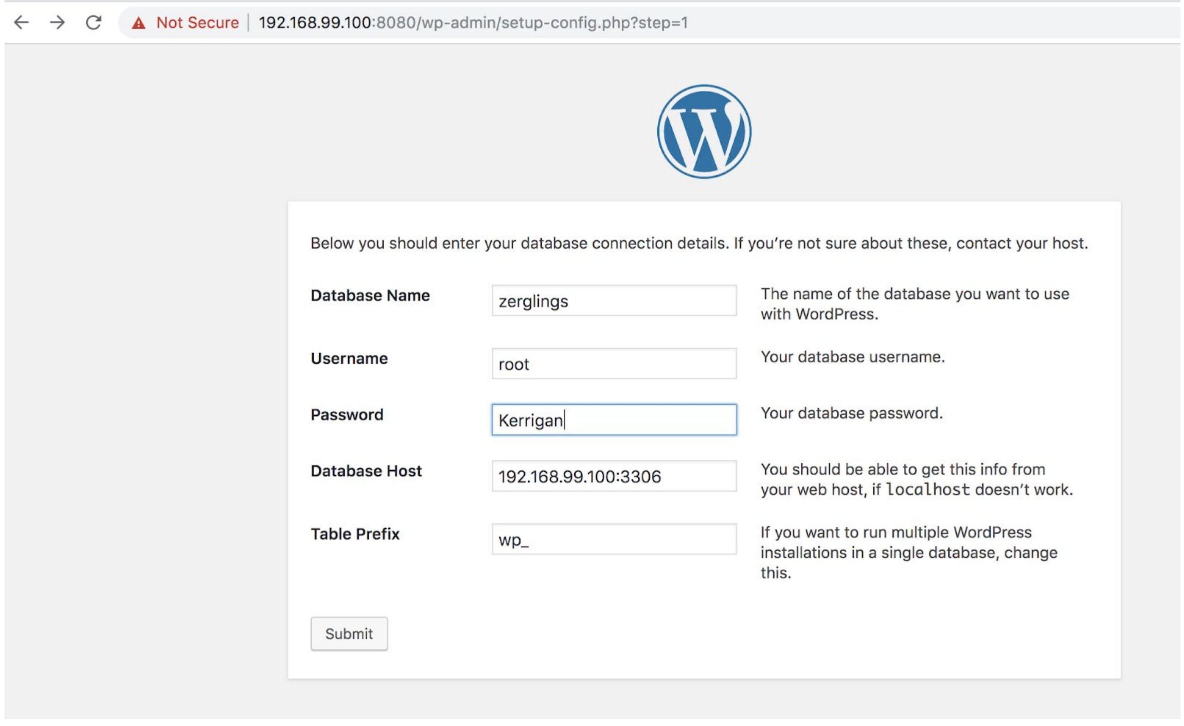
This either means that the username and password information in your `wp-config.php` file is incorrect or we can't contact the database server at `192.168.99.102:3306`. This could mean your host's database server is down.

- Are you sure you have the correct username and password?
- Are you sure that you have typed the correct hostname?
- Are you sure that the database server is running?

If you're unsure what these terms mean you should probably contact your host. If you still need help you can always visit the WordPress Support Forums.

СТАЛО:

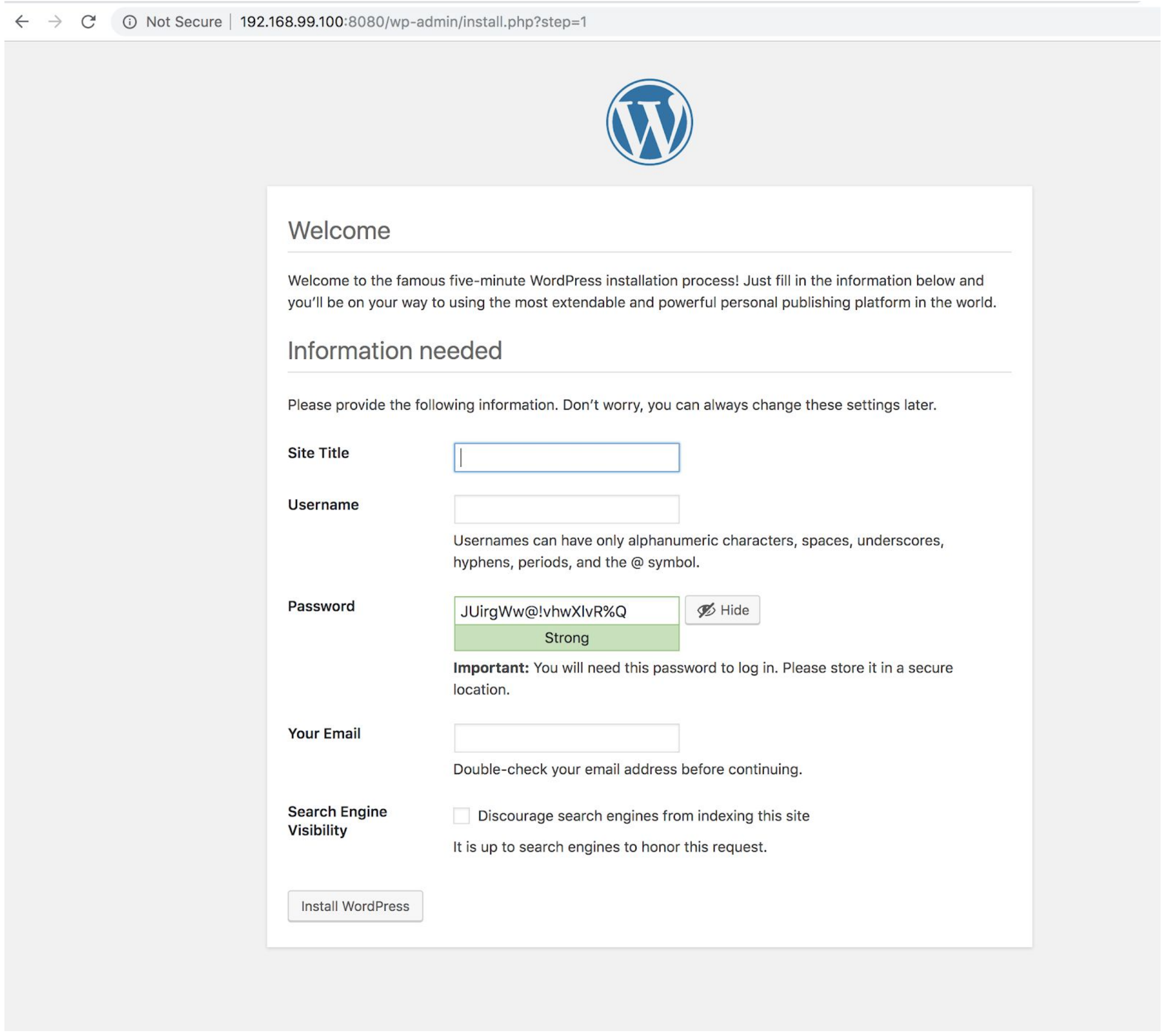
```
docker run -d --name lair -p 8080:80 --link spawning-pool:db wordpress
```



Not Secure | 192.168.99.100:8080/wp-admin/setup-config.php?step=1

Below you should enter your database connection details. If you’re not sure about these, contact your host.

Database Name	zerglings	The name of the database you want to use with WordPress.
Username	root	Your database username.
Password	Kerrigan	Your database password.
Database Host	192.168.99.100:3306	You should be able to get this info from your web host, if `localhost` doesn’t work.
Table Prefix	wp_	If you want to run multiple WordPress installations in a single database, change this.



Not Secure | 192.168.99.100:8080/wp-admin/install.php?step=1

Welcome

Welcome to the famous five-minute WordPress installation process! Just fill in the information below and you’ll be on your way to using the most extendable and powerful personal publishing platform in the world.

Information needed

Please provide the following information. Don’t worry, you can always change these settings later.

Site Title

Username

Usernames can have only alphanumeric characters, spaces, underscores, hyphens, periods, and the @ symbol.

Password JUirgWw@!vhwXlvR%Q Strong

Important: You will need this password to log in. Please store it in a secure location.

Your Email

Double-check your email address before continuing.

Search Engine Visibility ☐ Discourage search engines from indexing this site

It is up to search engines to honor this request.

Новый докер

То, что теперь нам доступно через Managed Software Center - не очень мне понравилось.

Работает через виртуалку, долго запускается, а если что-то пошло не так и упало - прощайтесь с машиной. Если через Virtual Box можно просто вылетить из машины, перезапустить Virtual Box и все заново настроить, то вот эта штука работает каким-то образом глубже. Возможно, я ее просто не прочувствовала. Но из-за нее тормозит все, а у меня еще и жесточайше заполнялась память.