Docker-контейнеры

План

- Создание docker host
- Создание своего образа
- Работа с Docker Hub

Проект microservices и проверка ДЗ

Создайте новую ветку в вашем microservices репозитории для выполнения данного ДЗ. Т.к. это второе задание, посвященное работе с Docker, то ветку назовите **docker-2**.

Проверка данного ДЗ будет производиться через Pull Request ветки с ДЗ.

После того, как один из преподавателей сделает approve пул реквеста, ветку с ДЗ можно смерджить.

Требования

- Установленный docker

- Создайте новый проект
  https://console.cloud.google.com/compute
- Назовите его **docker**
- Запомните ID вашего проекта

gcloud

- Установите GCloud SDK

- https://cloud.google.com/sdk/

gcloud init

```
$ gcloud init
You must log in to continue. Would you like to log in (Y/n)?  Y
Your browser has been opened to visit:
    https://accounts.google.com/o/oauth2/....
```

У вас должен запуститься браузер с выбором учетных записей Google. Выберите ту, к которой привязан GCE и разрешите доступ приложению

Если браузер не запустился, то запустите его самостоятельно и перейдите по ссылке выведенной в консоли

gcloud init

```
...

Pick cloud project to use:
 [1] docker-181710
 [2] Create a new project
Please enter numeric choice or text value (must exactly match list
item): 1

...

Do you want to configure Google Compute Engine
(https://cloud.google.com/compute) settings (Y/n)? n

...
```

Мы сконфигурировали gcloud

gcloud auth

```
$ gcloud auth application-default login
Your browser has been opened to visit:

    https://accounts.google.com/o/oauth2/....
```

У вас должен запуститься браузер с выбором учетных записей Google. Выберите ту, к которой привязан GCE и разрешите доступ приложению

Если браузер не запустился, то запустите его самостоятельно и перейдите по ссылке выведенной в консоли

gcloud auth

В итоге мы получили файл с аутентификационными данными.

Он будет использоваться docker-machine для работы с облаком.

Docker machine

- Mac OS и Windows
  идет в комплекте с docker (docker-machine -v)

- Linux
  https://docs.docker.com/machine/install-machine/

- **docker-machine** - встроенный в докер инструмент для создания хостов и установки на них docker engine. Имеет поддержку облаков и систем виртуализации (Virtualbox, GCP и др.)
- Команда создания - **docker-machine create <имя>**. Имен может быть много, переключение между ними через **eval $(docker-machine env <имя>)**. Переключение на локальный докер - **eval $(docker-machine env --unset).** Удаление - **docker-machine rm <имя>.**
- docker-machine создает хост для докер демона со указываемым образом в **--google-machine-image**, в ДЗ используется **ubuntu-16.04**. Образы которые используются для построения докер контейнеров к этому никак не относятся.
- Все докер команды, которые запускаются в той же консоли после **eval $(docker-machine env <имя>)** работают с удаленным докер демоном в GCP.

Docker machine

```
$ export GOOGLE_PROJECT=_ваш-проект_

$ docker-machine create --driver google \
    --google-machine-image https://www.googleapis.com/compute/v1/
projects/ubuntu-os-cloud/global/images/family/ubuntu-1604-lts \
    --google-machine-type n1-standard-1 \
    --google-zone europe-west1-b \
    docker-host
...
Docker is up and running!
To see how to connect your Docker Client to the Docker Engine
running on this virtual machine, run: docker-machine env docker-host
```

Регион можно поменять на другой, если этот блокируется

Docker machine

Проверяем, что наш Docker-хост успешно создан...

```
$ docker-machine ls
NAME           ACTIVE   DRIVER   STATE     URL                          SWARM   DOCKER
docker-host    -        google   Running   tcp://35.195.54.250:2376             v17.09.0-ce
```

```
$ eval $(docker-machine env docker-host)
```

... и начинаем с ним работу

Повторение практики из демо на лекции

Теперь когда у вас запущен докер хост в GCP, можете самостоятельно повторить демо из лекции посвященные:

- PID namespace (изоляция процессов)
- net namespace (изоляция сети)
- user namespaces (изоляция пользователей)

P.S. Для реализации Docker-in-Docker можно использовать этот образ. Дока по user namespace.

Повторение практики из демо на лекции

И сравните сами вывод:

- docker run --rm -ti tehbilly/htop
- docker run --rm --pid host -ti tehbilly/htop

Можете добавить небольшое описание в README

Структура репозитория

Для дальнейшей работы нам потребуются четыре файла, их содержание вы найдете на следующих слайдах

- Dockerfile - текстовое описание нашего образа
- mongod.conf - подготовленный конфиг для mongodb
- db_config - содержит переменную окружения со ссылкой на mongodb
- start.sh - скрипт запуска приложения

Вся работа происходит в папке **docker-monolith**

mongod.conf

ссылка на gist

```
# Where and how to store data.
storage:
  dbPath: /var/lib/mongodb
  journal:
    enabled: true

  # where to write logging data.
  systemLog:
    destination: file
    logAppend: true
    path: /var/log/mongodb/mongod.log

  # network interfaces
  net:
    port: 27017
    bindIp: 127.0.0.1
```

start.sh

ссылка на gist

```
#!/bin/bash

/usr/bin/mongod --fork --logpath /var/log/mongod.log --config /
etc/mongodb.conf

source /reddit/db_config

cd /reddit && puma || exit
```

db_config

```
DATABASE_URL=127.0.0.1
```

Dockerfile

- Начнем создавать образ с приложением. За основу возьмем известный нам дистрибутив ubuntu версии 16.04

- Создадим файл "Dockerfile" и добавим в него строки:

```
$ vim Dockerfile
FROM ubuntu:16.04
```

- Вместо vim можете использовать ваш редактор

Dockerfile

- Для работы приложения нам нужны **mongo** и **ruby**.
- Обновим кеш репозитория и установим нужные пакеты
- Добавим в "Dockerfile" строки:

```
FROM ubuntu:16.04

RUN apt-get update
RUN apt-get install -y mongodb-server ruby-full ruby-dev build-essential git
RUN gem install bundler
```

Dockerfile

Скачаем наше приложение в контейнер:

```
FROM ubuntu:16.04

RUN apt-get update
RUN apt-get install -y mongodb-server ruby-full ruby-dev build-essential git
RUN gem install bundler
RUN git clone -b monolith https://github.com/express42/reddit.git
```

Dockerfile

Скопируем файлы конфигурации в контейнер:

```
FROM ubuntu:16.04
RUN apt-get update
RUN apt-get install -y mongodb-server ruby-full ruby-dev build-essential git
RUN gem install bundler
RUN git clone -b monolith https://github.com/express42/reddit.git

COPY mongod.conf /etc/mongod.conf
COPY db_config /reddit/db_config
COPY start.sh /start.sh
```

Dockerfile

Теперь нам нужно установить зависимости приложения и произвести настройку:

```
FROM ubuntu:16.04

RUN apt-get update
RUN apt-get install -y mongodb-server ruby-full ruby-dev build-essential git
RUN gem install bundler
RUN git clone -b monolith https://github.com/express42/reddit.git

COPY mongod.conf /etc/mongod.conf
COPY db_config /reddit/db_config
COPY start.sh /start.sh

RUN cd /reddit && bundle install
RUN chmod 0777 /start.sh
```

Dockerfile

Добавляем старт сервиса при старте контейнера:

ссылка на gist

```
FROM ubuntu:16.04

RUN apt-get update
RUN apt-get install -y mongodb-server ruby-full ruby-dev build-essential git
RUN gem install bundler
RUN git clone -b monolith https://github.com/express42/reddit.git

COPY mongod.conf /etc/mongod.conf
COPY db_config /reddit/db_config
COPY start.sh /start.sh

RUN cd /reddit && bundle install
RUN chmod 0777 /start.sh

CMD ["/start.sh"]
```

Сборка образа

Теперь мы готовы собрать свой образ

- Выполните команду:

пример лога

```
$ docker build -t reddit:latest .
Sending build context to Docker daemon  36.86kB
Step 1/12 : FROM ubuntu:16.04
16.04: Pulling from library/ubuntu
...
Successfully built ad21718d3eb5
Successfully tagged reddit:latest
```

- Точка в конце обязательна, она указывает на путь до Docker-контекста
- Флаг -t задает тег для собранного образа

Сборка образа

Посмотрим на все образы (в том числе промежуточные):

```
$ docker images -a
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
<none>              <none>              e0a3c979a0fb        2 minutes ago       687MB
<none>              <none>              0e07ff248072        2 minutes ago       687MB
reddit              latest              f7ff59db9594        2 minutes ago       687MB
<none>              <none>              96f7a69d48d2        2 minutes ago       687MB
<none>              <none>              a809efbf09a0        2 minutes ago       657MB
<none>              <none>              476d8d488857        2 minutes ago       657MB
<none>              <none>              b163309613e1        2 minutes ago       657MB
<none>              <none>              f7b02c42278f        2 minutes ago       657MB
<none>              <none>              746e7a1087d9        2 minutes ago       657MB
<none>              <none>              2d2469adc562        3 minutes ago       654MB
<none>              <none>              7ea520e2fdf1        6 minutes ago       161MB
ubuntu              16.04               2d696327ab2e        2 weeks ago         122MB
```

Запуск контейнера

- Отлично, теперь можно запустить наш контейнер командой:

```
$ docker run --name reddit -d --network=host reddit:latest
```

- Проверим результат:

```
$ docker-machine ls
NAME          ACTIVE DRIVER STATE   URL                       DOCKER
docker-host -        google Running tcp://_ваш_IP_:2376 v17.09.0-ce
```

- Откройте в браузере ссылку `http://_ваш_IP_адрес_:9292`

Ошибка

This site can’t be reached

35.195.54.250 took too long to respond.

Search Google for 195 250 9292

ERR_CONNECTION_TIMED_OUT

Настройка firewall'a

- Разрешим входящий TCP-трафик на порт 9292 выполнив команду:

```
$ gcloud compute firewall-rules create reddit-app \
    --allow tcp:9292 \
    --target-tags=docker-machine \
    --description="Allow PUMA connections" \
    --direction=INGRESS
```

- Попробуйте снова открыть ссылку (hint: должен открыться наш сервис) `http://_ваш_IP_адрес_:9292`

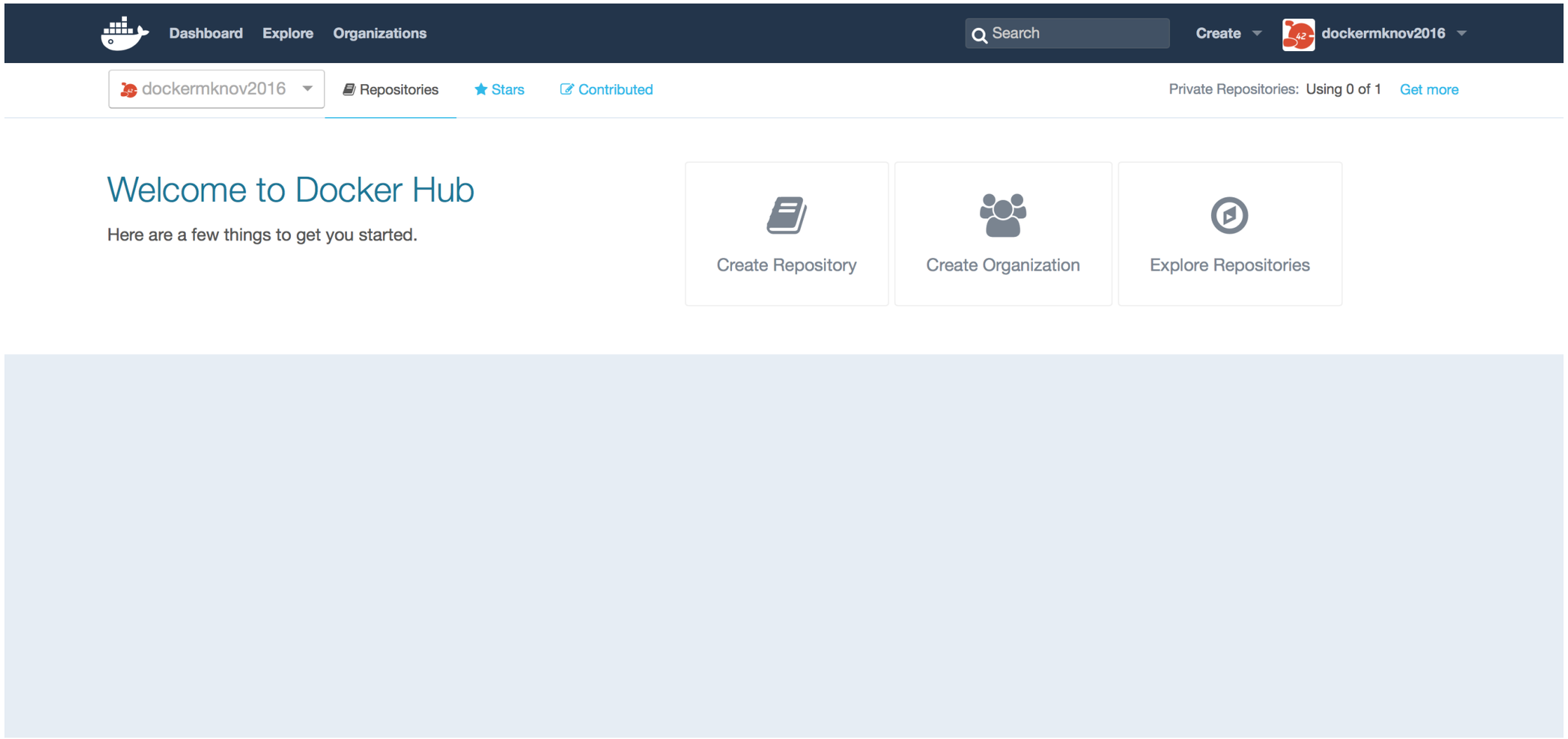
Docker hub: регистрация

Docker Hub - это облачный **registry** сервис от компании Docker. В него можно выгружать и загружать из него докер образы. Docker по умолчанию скачивает образы из докер хаба.

Зарегестрируемся в нем:

- Перейдите по ссылке https://hub.docker.com/
- Зарегистрируйте учетную запись, если у вас ее еще нет

Docker hub: регистрация



Docker hub: аутентификация

Аутентифицируемся на docker hub для продолжения работы:

```
$ docker login
Login with your Docker ID to push and pull images from Docker Hub.
If you don't have a Docker ID, head over to https://hub.docker.com to create one.

Username: your-login
Password:

Login Succeeded
```

Docker hub: push

Загрузим наш образ на docker hub для использования в будущем:

пример лога

```
$ docker tag reddit:latest <your-login>/otus-reddit:1.0

$ docker push <your-login>/otus-reddit:1.0
The push refers to a repository [docker.io/<your-login>/otus-reddit]
c6e5100de1e0: Pushed
...
a2022691bf95: Pushed
1.0: digest:
sha256:77c6070400a5b04f8db3f7c129a2c16084c2fcf186aa6b436c8d6f57e0014378 size:
3448
```


Проверка

Т.к. теперь наш образ есть в докер хабе, то мы можем запустить его не только в докер хосте в GCP, но и в вашем локальном докере или на другом хосте.

Выполним в другой консоли:

```
$ docker run --name reddit -d -p 9292:9292 <your-login>/otus-reddit:1.0
```

И проверим что в локальный докер скачался загруженный ранее образ и приложение работает.

Еще проверка

Дополнительно можете с помощью следующих команд изучить логи контейнера, зайти в выполняемый контейнер, посмотреть список процессов, вызвать остановку контейнера, запустить его повторно, остановить и удалить, запустить контейнер без запуска приложения и посмотреть процессы:

- `docker logs reddit -f`
- `docker exec -it reddit bash`
  - `ps aux`
  - `killall5 1`
- `docker start reddit`
- `docker stop reddit && docker rm reddit`
- `docker run --name reddit --rm -it <your-login>/otus-reddit:1.0 bash`
  - `ps aux`
  - `exit`

И еще проверка

И с помощью следующих команд можно посмотреть подробную информацию о образе, вывести только определенный фрагмент информации, запустить приложение и добавить/удалить папки и посмотреть дифф, проверить что после остановки и удаления контейнера никаких изменений не останется:

- `docker inspect <your-login>/otus-reddit:1.0`
- `docker inspect <your-login>/otus-reddit:1.0 -f '{{.ContainerConfig.Cmd}}'`
- `docker run --name reddit -d -p 9292:9292 <your-login>/otus-reddit:1.0`
- `docker exec -it reddit bash`
    - `mkdir /test1234`
    - `touch /test1234/testfile`
    - `rmdir /opt`
    - `exit`
- `docker diff reddit`
- `docker stop reddit && docker rm reddit`
- `docker run --name reddit --rm -it <your-login>/otus-reddit:1.0 bash`
    - `ls /`

Задание со *

Теперь, когда есть готовый образ с приложением, можно автоматизировать поднятие нескольких инстансов в GCP, установку на них докера и запуск там образа <**your-login**>/otus-reddit:1.0

Нужно реализовать в виде прототипа в директории **/docker-monolith/infra/**

- Поднятие инстансов с помощью Terraform, их количество задается переменной;
- Несколько плейбуков Ansible с использованием динамического инвентори для установки докера и запуска там образа приложения;
- Шаблон пакера, который делает образ с уже установленным Docker;

Проверка ДЗ

- Результаты вашей работы находятся в ветке **docker-2** вашего microservices репозитория.
- В README внесите описание того, что сделано.
- Создайте Pull Request к ветке master (**описание PR нужно заполнять**);
- В ревьюверы можно никого не добавлять;
- Добавьте "Labels" **docker** и **docker-2** к вашему Pull Request;

- После того, как один из преподавателей сделает approve пул реквеста, ветку с ДЗ можно смерджить и закрыть PR.