Если вы видите что-то необычное, просто сообщите мне.

Dockerfile, ???????? docker image.

Dockerfile

Представим, приложение уже работает на вашей машине, но еще не имеет образа. Для того, чтобы получить docker образ, нам необходимо описать его.

???????? ???????????? ??? ?????? ??????

В папке с проектом(мы рассматриваем node проект) создадим файлы:

```
touch Dockerfile
```

Для того, чтобы в докер не попапи ненужные файлы - рядом создадим `dockerignore` и впишем в него всё, что нам не нужно.

```
touch .dockerignore
```

Пример Dockerfile:

```
#Выберем базовый образ из которого мы будем создавать наш проект
FROM node:10-apline
# Укажем папку внутри докера, которая будет являться домашней при выполнении различных команд
WORKDIR /usr/src/app
# Скопируем из папки с проектом все файлы по маске package*.json в папку WORKDIR
COPY package*.json ./
```

```
# Запускаем установку пакетов npm
RUN npm install
# Копируем всё что осталось в папке с проектом в папку WORKDIR
COPY . .
# Указываем порт, через который будет доступен наш проект
EXPOSE 3000
# Указываем команду которая будет работать в тот момент, когда запустится контейнер
CMD["npm", "start"]
```

?????? ?????? ????????????? ??? ???????? ??????

Ключ	Назначение
FROM	Указываем на основе которого из образов будем создавать новый образ
MAINTAINER	Указываем автора созданного образа
RUN	Запукаем команды в внутри контейнера необхдимые для работы образа
CMD	Команда которая будет выполнена при запуске контейнера(может быть только одна)
EXPOSE	Говорит докеру, что контейнер слушает на определенном порту
ENV	Указываем с каким переменным окружением необходимо создавать контейнер
ADD	Копирует файлы, папки, URL и добавляет их в файловую систему образа, может распаковать архив
COPY	Копирует файлы, папки из и по указанному пути
ENTRYPOINT	Позволяет задавать команду запуска контейнера, при этом при старте указывать ключи запуска этого контейнера, переопределяет CMD
USER	Указывает имя пользователя под которым будет происходить выполнение команд RUN, CMD, ENTRYPOINT внутри контейнера
WORKDIR	Указываем директорию внутри которой будет происходить выполнение команд RUN, CMD, ENTRYPOINT, COPY, ADD
ARG	Определяем переменные, которые могут быть переданы билдеру при запуске `docker build` используя ключ `--build-arg <varname>=<value>`

?????? ??????

Создаем билд с помощью команды:

```
docker build . -t firstimage
```

Ожидаем заверешения команды. Для этого может потребоваться много времени.

```
В случае, если билд завершился неудачно, смотрим что за ошибка и правим Dockerfile
```

После завершения: мы можем посмотреть на наш `firstimage` в списке, который можно получить с помощью команды:

```
docker image
```

?????? ?????? ???????

Чтобы запустить наш проект - необходимо указать несколько параметров:

```
docker run -p 80:3000 firstimage
```

-p - говорит о том, что мы мапируем(прим. ред. mapping - сопоставление) системный 80 порт в порт(помните внутри контейнера использовали EXPOSE 3000) внутри контейнера.

Заходим на http://localhost и убеждаемся, что наше приложение доступно внутри контейнера.

Revision #4
Created 2021-09-22 11:53:26 UTC by gasick
Updated 2023-11-08 16:10:47 UTC by ivan.serov37@gmail.com