|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №4**

по дисциплине «Технологии разработки программных приложений»

**Тема практической работы:** «Docker»

|  |  |  |
| --- | --- | --- |
| **Выполнил:**  Студент группыИКБО-42-23 | | Голев С.С. |
| **Проверил:** | Доцент кафедры МОСИТ,  кандидат технических наук, доцент Жматов Д.В. | |

# СОДЕРЖАНИЕ

[ХОД РАБОТЫ 3](#_Toc196731829)

[1 Образы 3](#_Toc196731830)

[2 Изоляция 3](#_Toc196731831)

[3 Работа с портами 4](#_Toc196731832)

[4 Именованные контейнеры, остановка и удаление 5](#_Toc196731833)

[5 Постоянное хранение данных 5](#_Toc196731834)

[7 Dockerfile 6](#_Toc196731835)

[ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ 8](#_Toc196731836)

[ВЫВОД 10](#_Toc196731837)

# ХОД РАБОТЫ

## 1 Образы

Загрузим последний образ Ubuntu и образ конкретной версии.

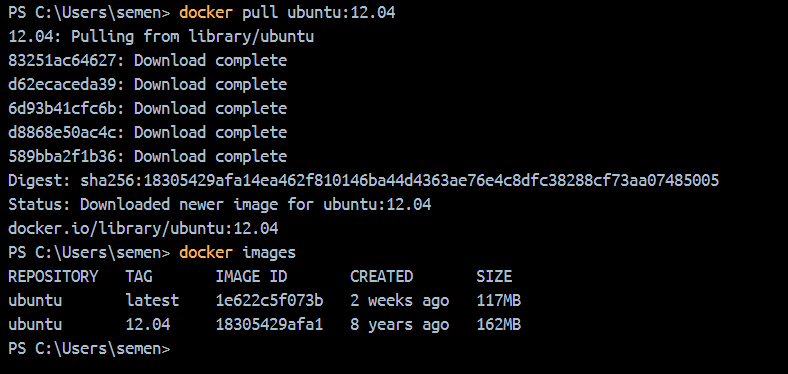


Рисунок 1 – Загрузка образов Ubuntu

Также проверим, имеются ли в системе контейнеры.

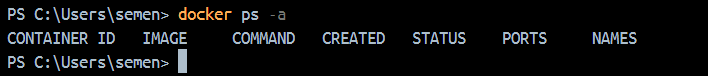


Рисунок 2 – Запрос списка контейнеров

## 2 Изоляция

Посмотрим информацию о хостовой системе, выполнив команду hostname.

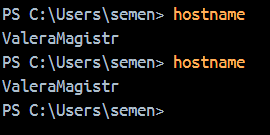


Рисунок 3 – Информация о системе

*Вопрос: одинаковый ли результат получился при разных запусках?*

Ответ: при запуске на одной машине, результат не будет меняться между запусками.

Попробуем выполнить то же самое в контейнерах.

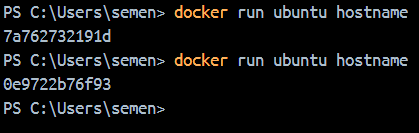


Рисунок 4 – Информация о системе в контейнерах

*Вопрос: одинаковый ли результат получился при разных запусках?*

Ответ: при вводе этой команды, для каждой команды, создаются новые контейнеры, которые изолированы друг от друга, каждый контейнер имеет собственный идентификатор.

## 3 Работа с портами

Загрузим образ python.

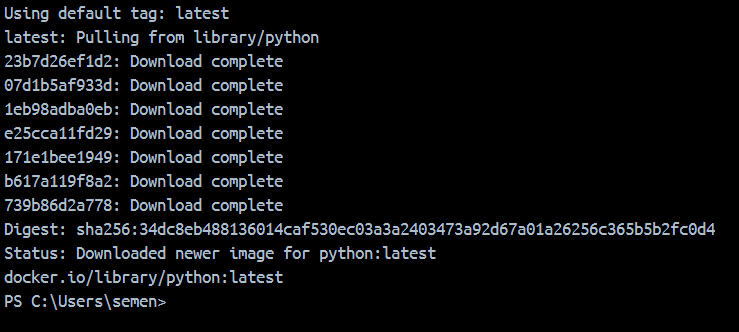


Рисунок 5 – Загрузка образа Python

Запустим встроенный в Python модуль веб-сервера из корня контейнера, чтобы отобразить содержание контейнера с портом 8000.



Рисунок 6 – Запуск веб-сервера

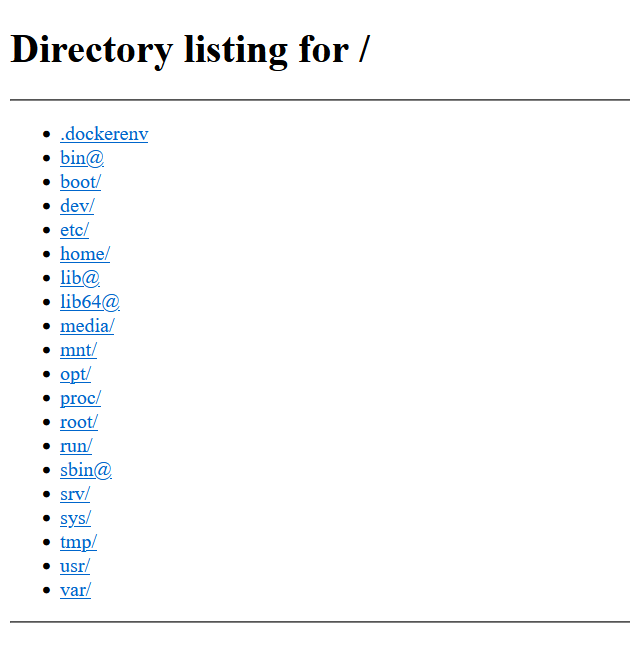


Рисунок 7 – Содержимое корневой директории в контейнере

## 4 Именованные контейнеры, остановка и удаление

Запустим контейнер из прошлого раздела в фоновом режиме.

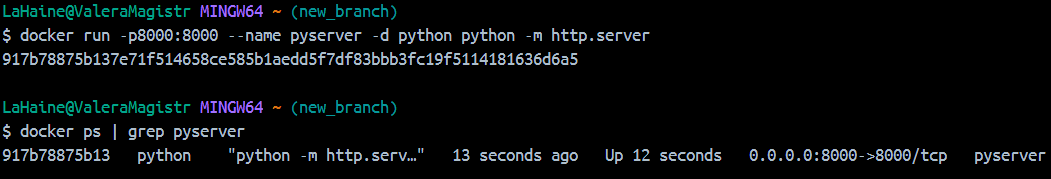


Рисунок 8 – Запуск контейнера и проверка

## 5 Постоянное хранение данных

Запустите контейнер, в котором веб-сервер будет отдавать содержимое директории /mnt.

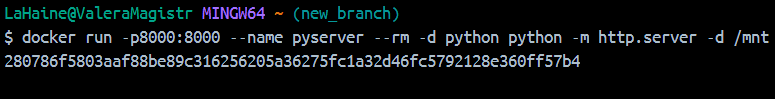


Рисунок 9 – Запуск контейнера

*Вопрос: что значат остальные флаги запуска? Где здесь команда, которая выполнится в контейнере?*

Ответ: **-p8000:8000** выбор портов, левый – порт на хостовой машине, правый – порт внутри контейнера. **--name pyserver** – явное указание имени контейнера. **–name** – автоматическое удаление контейнера после завершения работы. **-d** – запуск контейнера в фоновом режиме.

Попадём в запущенный контейнер используя оболочку bash и создадим текстовый файл в /mnt.

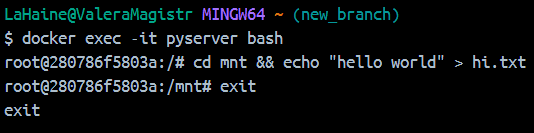


Рисунок 10 – Создание файла внутри контейнера



Рисунок 11 – Содержимое директории в контейнере

## 7 **Dockerfile**

Напишем Dockerfile, соберём образ и произведём запуск на хостовой машине.

|  |
| --- |
| FROM ubuntu:latest  RUN apt update \  && apt install -y python3 fortune \  && cd /usr/bin \  && ln -s python3 python  RUN /usr/games/fortune > /mnt/greeting-while-building.txt  ADD ./data /mnt/data  EXPOSE 80  CMD ["python" , "-m" , "http.server" , "-d" , "/mnt/" , "80"] |

Листинг 1 – Содержимое Dockerfile

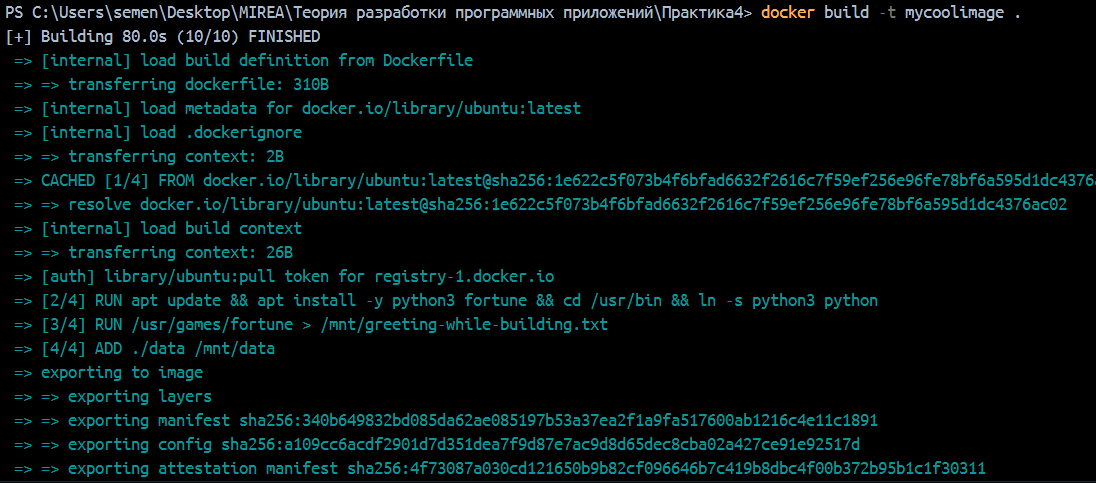


Рисунок 12 – Сборка образа

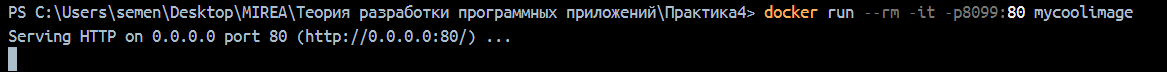


Рисунок 13 – Запуск

# ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Напишем Dockerfile, соберём образ и запустим контейнер согласно варианту.

Необходимо использовать базовый образ ubuntu:20.10, примонтировать файл data/student.txt как /mnt/files/student.txt в контейнере. Установить пакет: figlet.

|  |
| --- |
| FROM ubuntu:20.10  RUN sed -i 's/archive.ubuntu.com/old-releases.ubuntu.com/g' \  /etc/apt/sources.list &&  RUN apt update && \  apt install -y python3 figlet  EXPOSE 8802  CMD ["python3", "-m", "http.server", "-d", "/mnt/files", "8802"] |

Листинг 2 – Содержимое Dockerfile

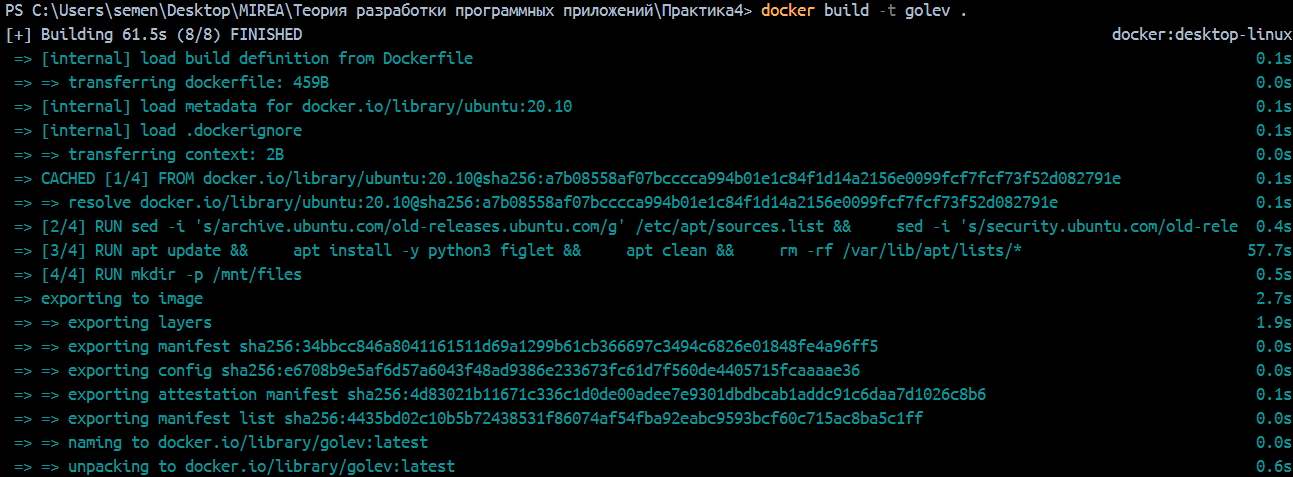


Рисунок 14 – Сборка образа



Рисунок 15 – Запуск контейнера

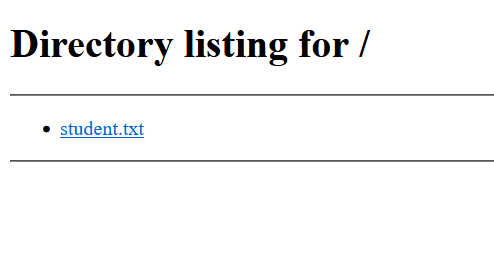


Рисунок 16 – Содержимое директории /mnt/files

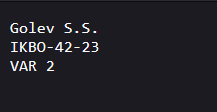


Рисунок 17 – Содержимое файла в директории /mnt/files

# ВЫВОД

В ходе выполнения практической работы были изучены основные принципы работы с Docker и его компонентами.

Изучена работа с образами и контейнерами: загрузка образов, запуск контейнеров, просмотр списка контейнеров и образов. Были освоены команды docker pull, docker run, docker ps и другие.

Особое внимание уделено изоляции контейнеров, передаче переменных окружения, монтированию директорий и файлов с хоста в контейнер, а также постоянному хранению данных с помощью томов.

Собран собственный образ с использованием Dockerfile, в котором установлены необходимые пакеты и настроен запуск встроенного веб-сервера Python. Изучены принципы проброса портов и работы с внутренними и внешними портами контейнера.

В результате сформировано практическое понимание создания, настройки и управления окружениями на базе Docker.