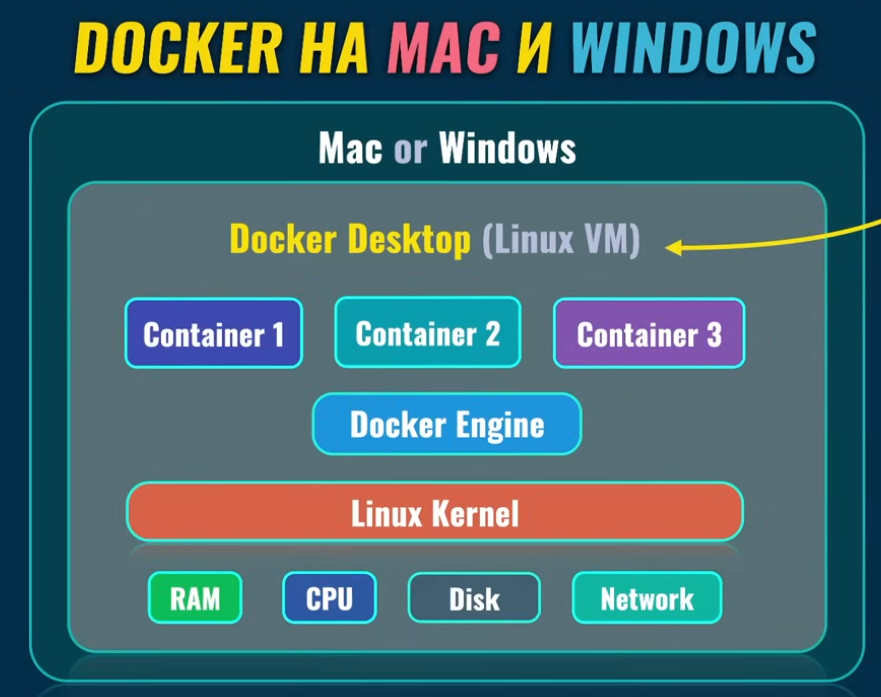
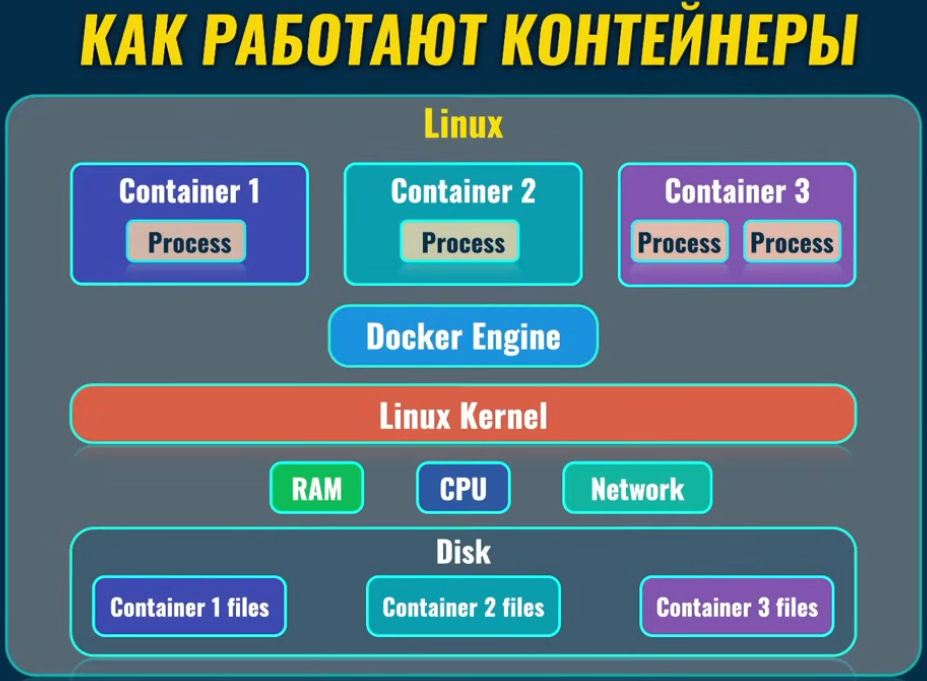
**Памятка DOCKER**

****



Каждая версия образа может иметь несколько тегов

**Управление контейнерами**

**docker ps [-a] – список запущенных или всех (-a) контейнеров.**

docker **images** – список всех образов локально на компьютере.

docker **build** <путь к каталогу с Dockerfile | .> [-f <имя Dockerfile>] [-t <имя создаваемого образа : тег версии образа>]– создает образ по Dockerfile.

docker **run** – создает образ по Dockerfile и запускает контейнер из этого образа

[-d] [--rm] [-p] [--name <имя для внутреннего обращения>] <**имя образа**> [<имя команды> <имя параметра>]

docker run <имя образа>

docker run --rm <имя образа> - автоматическое удаление контейнера после остановки

docker run -it <имя образа> - запуск с подключением терминала к выводу запущенного процесса в контейнере

docker run -d <имя образа> - запускает контейнер в фоновом режиме

docker run –p <внешний порт>:<порт контейнера> <имя образа> - запускает контейнер с сопоставлением (mapping) портов

docker run –v <абсолютный путь к папке на компьютере>:<путь к папке в контейнере> <имя образа> - запускает контейнер с сопоставлением (mapping) папок (томов). Абсолютный путь может задаваться строкой ${PWD} – PWD это переменная ОС, в которой находится абсолютный путь к текущей папке

Команды интерактивной оболочки

hostname – выдает ID контейнера

hostname -i - выдает IP контейнера

docker **container inspect** <ID контейнера> - детальная информация о

контейнере, в том числе о mapping томов

docker **rm** <имя контейнера | ID> - удалить контейнер

docker **container prune** – удалить все контейнеры

docker **start** <имя контейнера | ID> - запустить существующий контейнер

docker **stop** <имя контейнера | ID> - остановить запущенный контейнер

(для остановки можно также остановить главный процесс в контейнере)

docker **kill** <имя контейнера | ID> - завершить контейнер, если процесс не

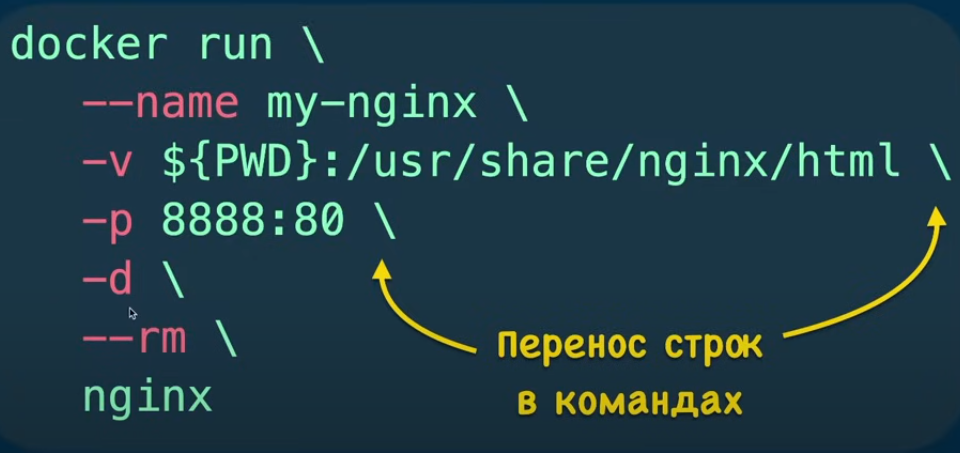
отвечает

docker **exec** [-it] <имя контейнера | ID> <имя процесса> - запустить какой-

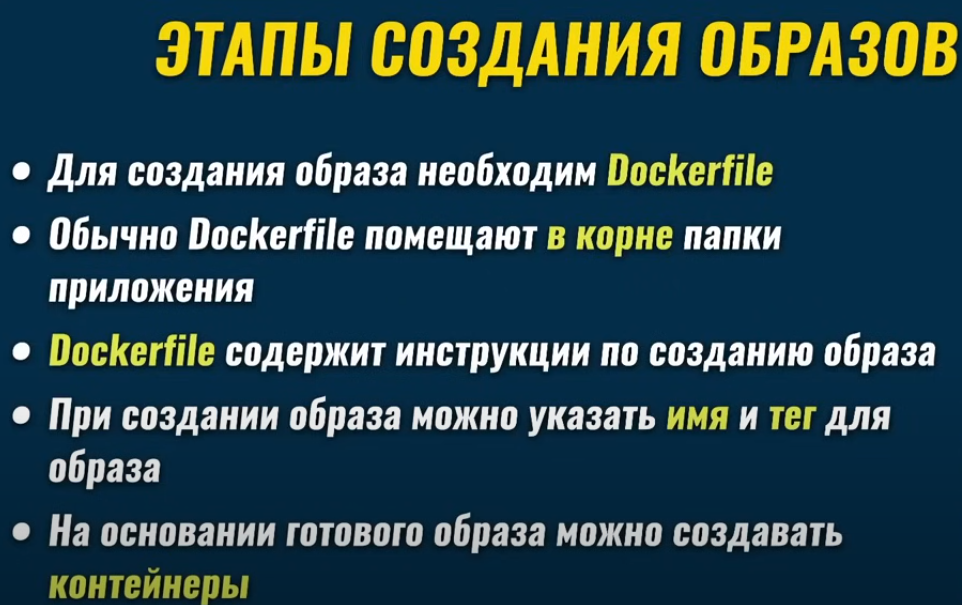
либо процесс внутри контейнера

docker **attach** <имя контейнера | ID> - подключить процесс (основной?) в контейнере к терминалу

docker **logs** <имя контейнера | ID> <имя процесса> - выводит историю вывода процесса в контейнере; необходимо когда запуск идет в фоновом режиме (параметр -d)



Создание образов

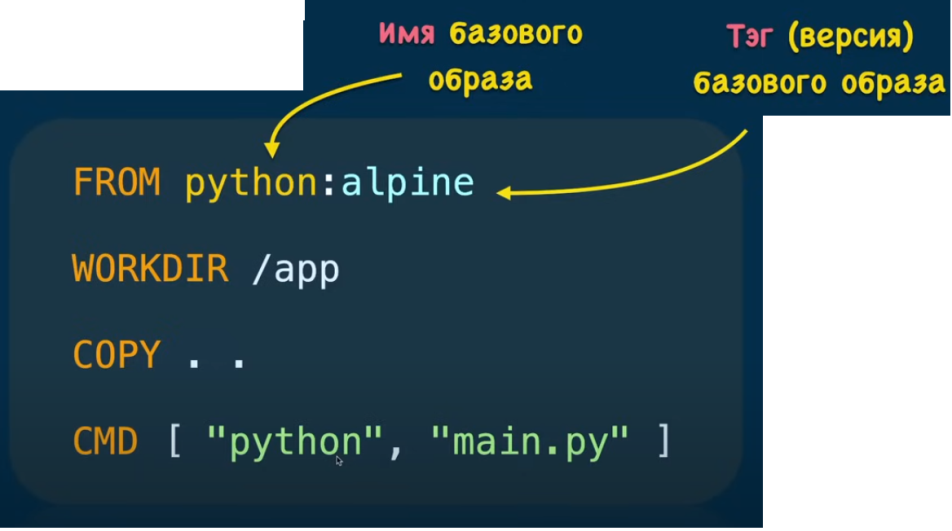
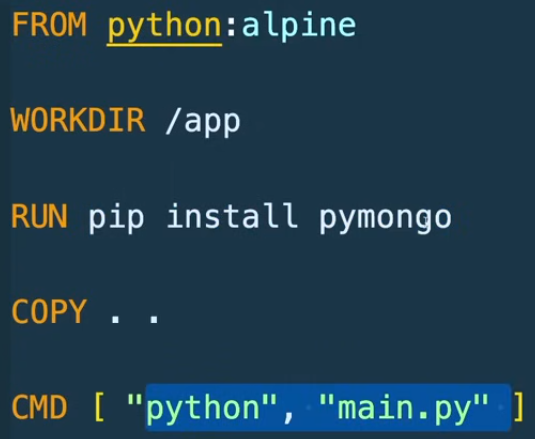


Для создания образа необходимо создать файл с инструкциями - Dockerfile

Каждая инструкция в Dockerfile описывает очередной слой образа

При создании собственного образа необходимо всегда указывать

**базовый** образ

Dockerfile

From <имя базового образа>:<тег>

WORKDIR <путь к рабочей директории> - если ее нет, то будет создана в образе; она же становится текущей после этой команды при обработке образа.

RUN <команда> - выполняется команда в процессе создания контейнера (как если бы набирать их в консоли)

**pip** – встроенный в пайтон инсталятор пакетов

COPY . . – точки означают текущую директорию (скопировать все файлы из текущей директории на компьютере в рабочий каталог в образе)

CMD [<”команда”>, <”параметры”>] – команда которую нужно выполнить в контейнере при его запуске

alpine – на DockerHab означает наименьшую версию

docker build [-f] <путь к каталогу с Dockerfile> [-t <имя образа>[:<имя тега>]]

будет искаться файл с именем Dockerfile (имя по умолчанию)

если указать точку, то файл с именем Dockerfile будет

искаться в текущей директории

-f – если файл имеет имя отличное от Dockerfile, то используя эту опцию можно указать егыо имя

Если не указать имя тега, то будет назначено latest

Docker compose

Используется YML файл с декларативным описанием действий

Стандартное название файла - **docker-compose.yml**

docker-compose up [--build] – создает контейнеры (сервисы) прописанные в файле, а также создает сеть между ними; обращение к сервесу по этой сети возможен по имени сервиса заданного в ф файле yml.

Параметр --build указывает docker-compose пересоздать

образы по Dockerfile, даже если он уже есть. Это требуется,

когда мы поменяли, например файл с кодом приложения

docker-compose down – останавливает и удаляет все контейнеры (сервисы) прописанные в файле, в том числе и сеть

Формат YML

Структуры (типы) данных

Список

<имя списка>

- <значение 1>

- <значение n>

Словарь

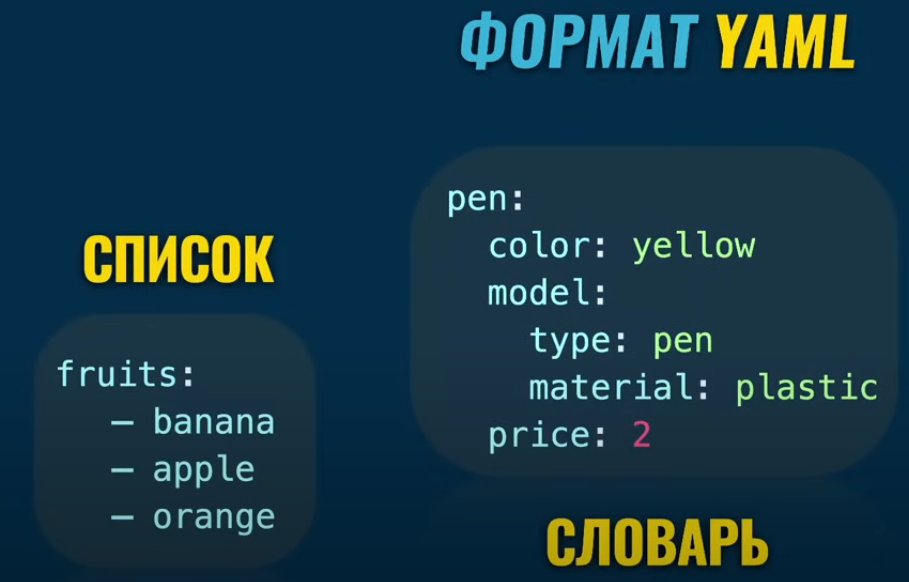
<Имя словаря>

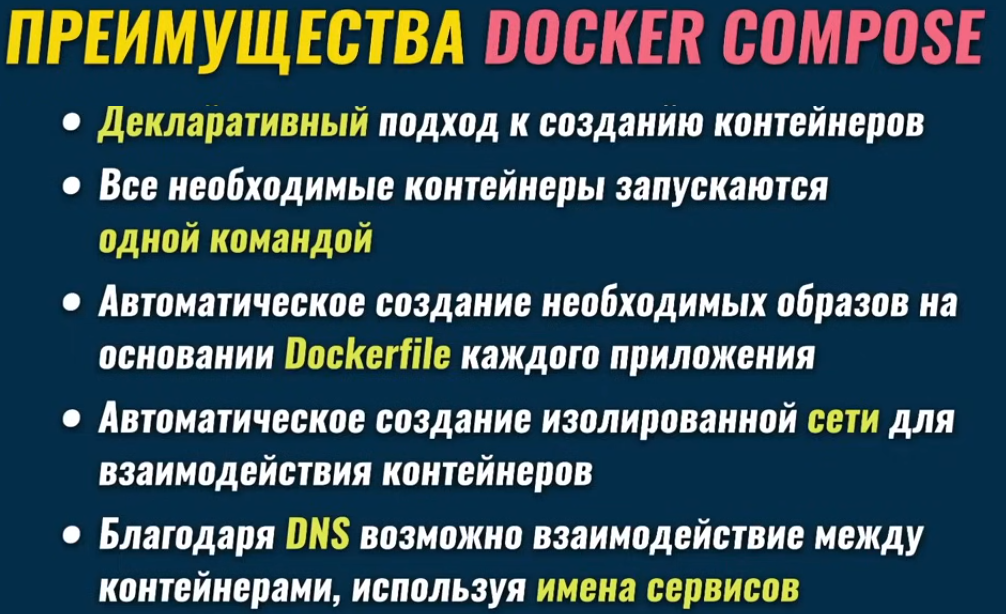
<имя ключа>:<значение>

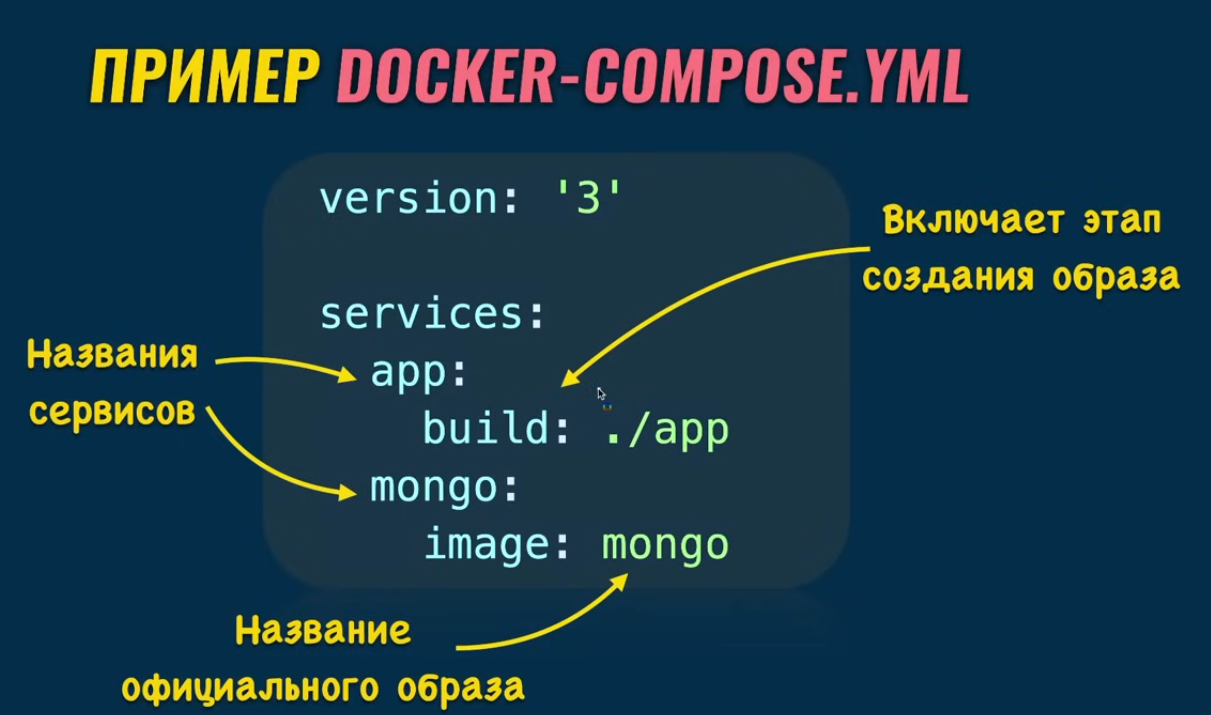
Значения могут быть вложенными структурами.

Ключ **services** – словарь для определения сервисов

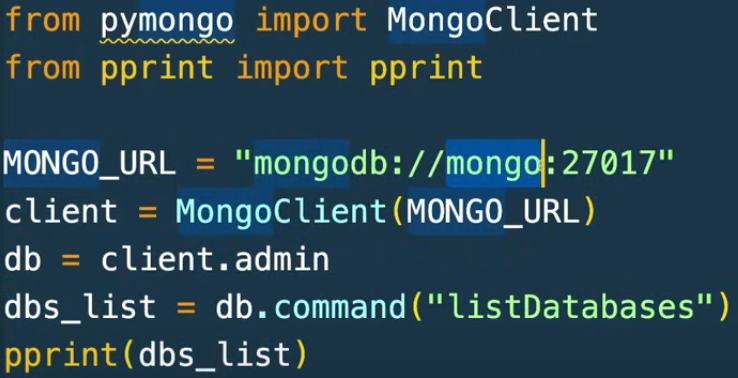
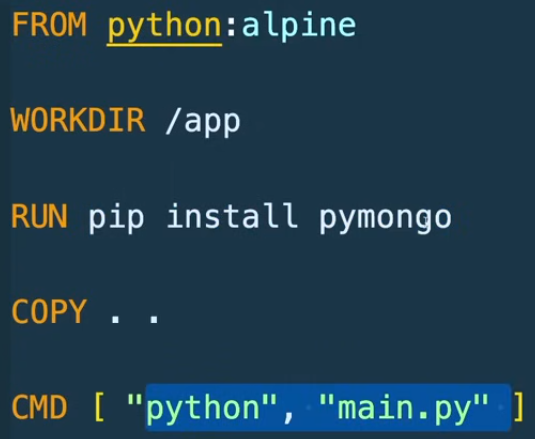
(контейнеров) и способов их создания.







Сквозной пример





Примеры команд

Docker

docker run --rm -it --name my\_app <image\_id> - создание контейнера с указанным именем, его запуск и автоматическое удалением после завершения процесса в нем

docker start -i <container\_name | container\_id> - запуск контейнера с подключением вывода к терминалу

docker run -v D:\ test\_docker\temp:/temp --rm -it --name my\_app myimage:v1

docker run -v C:\temp\_os\os.txt:/temp --rm -it --name my\_app myimage:v1

Dockerfile

FROM python:alpine3.12

WORKDIR /app

COPY . .

VOLUME C:\temp\_os\os.txt:/temp #mapping портов

CMD ["python", "main.py"]