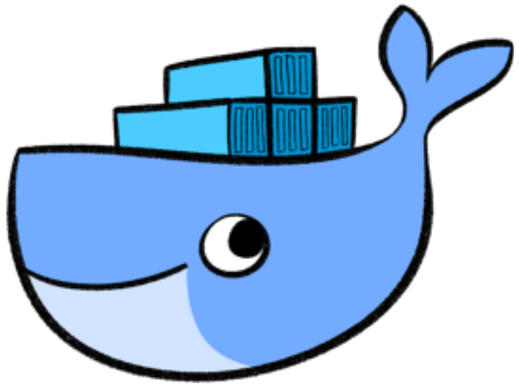
# Docker

sd@freematiq.com
https://github.com/freematiq/php

# Docker

Инструмент, который помогает упаковать и запустить  приложение в контейнере, со всеми зависимостями



# Виртуализация или контейнер

Разные цели (зависит от модели)
Цельный образ с OS
Можно держать несколько на хосте
Раздельное пространство и полная изоляция

Контейнер
Работает в операционной системе
Нет необходимости ставить ПО
Установка с помощью конф. файлов

# Контейнер

*Image* (образ) - основа контейнера, базовый набор файлов. Список образов можно увидеть с помощью **docker images**

*Container* (контейнер) - Создаются на основе образа и запускают само приложение. Список запущенных контейнеров можно увидеть с помощью команды **`docker ps`**.

Volume (том) – общая папка. Тома инициализируются при создании контейнера и предназначены для сохранения данных, независимо от жизненного цикла контейнера.

# Установка

https://www.docker.com/

docker-ce - бесплатная версия

Создан для linux на основе cgroup, но портирован на Windows и macOS

При установке в linux пользователя нужно добавить в группу docker

# docker

Клиент серверное приложение
sudo systemctl status docker

# Hello world

docker run hello-world

`docker ps -a` список контейнеров

# Cheatsheet

- docker search ubuntu - поиск образа
- docker pull ubuntu - стянуть убунту из hub
- docker images - посмотреть набор образов
- docker run ubuntu - запустить контейнер
- docker run -it ubuntu - запустить в интерактивном режиме

# Custom images

- **Официальные образы** — это образы, которые официально поддерживаются командой Docker. Обычно в их названии одно слово: `ubuntu`, `hello-world` — базовые образы.
- **Пользовательские образы** — образы, созданные пользователями. Они построены на базовых образах и называются по формату `user/image-name`.

Когда запускается образ - можно его модифицировать как обычную виртуальную машину, и все изменения сохраняться только для этого контейнера.

docker rm - удалить контейнер

docker commit -m "What did you do to the image" -a "Author Name" container-id repository/new_image_name - сохранить контейнер

# Dockerfile

Используется для сборки своих образов

docker build -t ph4n70m/demo_image .

docker push ph4n70m/demo

## Команды для сборки

- FROM – задать базовый образ
- RUN – выполнить команду в контейнере
- ENV – задать переменную среды
- WORKDIR – установить рабочий каталог
- VOLUME – создать точку монтирования для тома
- CMD – установить исполняемый файл для контейнера
- ADD - добавить файл в контейнер

# Dockerfile

```
FROM ubuntu:latest
RUN apt-get update
RUN apt-get install --no-install-recommends --no-
install-suggests -y curl
ENV SITE_URL https://ya.ru/
WORKDIR /data
VOLUME /data
CMD sh -c "curl -L $SITE_URL > /data/results.txt”
```

## Сбор и запуск

```
docker build . -t site-curl

sudo docker history site-curl

docker run --rm -v $(pwd)/volume:/data/:rw site-curl
docker run --rm -e SITE_URL=https://vk.com -v $(pwd)/
volume:/data/:rw site-curl
```

# docker-compose

Связь между контейнерами docker-compose.yml

Хорошая архитектура - когда каждый контейнер содержит свой набор ПО

docker-compose up - для запуска контейнера

Чтобы открывался пример в сайте - надо прописать в hosts файл (в разной ОС он разный)
127.0.0.1 php-docker.local

# Домашнее задание

Запустить пример в docker