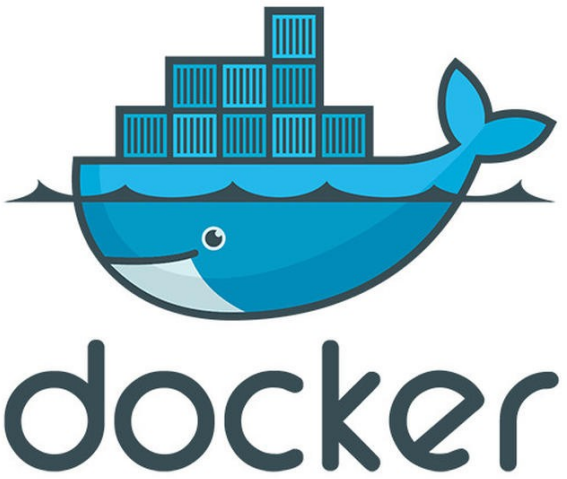
Основы Docker

Установка Docker

- Linux: установка из репозитория,
- инстукция - https://docs.docker.com/get-docker/
- OS X: установка dmg,
- инстукция - https://docs.docker.com/get-docker/
- Windows: установка exe,
- инстукция - https://docs.docker.com/get-docker/



https://www.docker.com/

Термины экоситемы Docker

Механизмы:

- Платформа Docker
- Движок Docker
- Клиент Docker
- Демон Docker
- Тома Docker
- Реестр Docker
- Репозиторий Docker

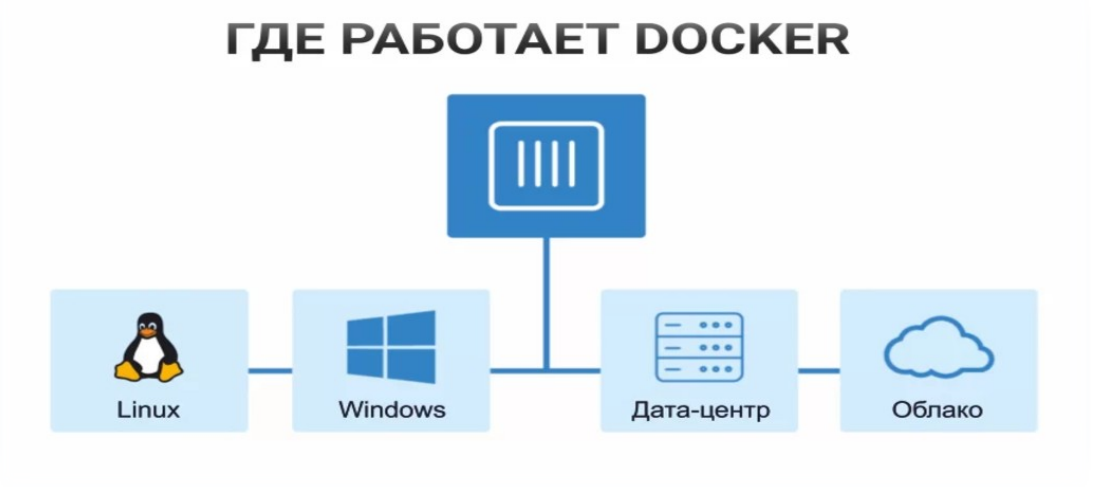
Масштабирование:

- Сеть Docker
- Docker Compose
- Docker Swarm
- Сервисы Docker

Платформа Docker

Платформа Docker (Docker Platform) — это программное обеспечение для автоматизации развертывания и управления приложениями в средах с поддержкой контейнеризации.

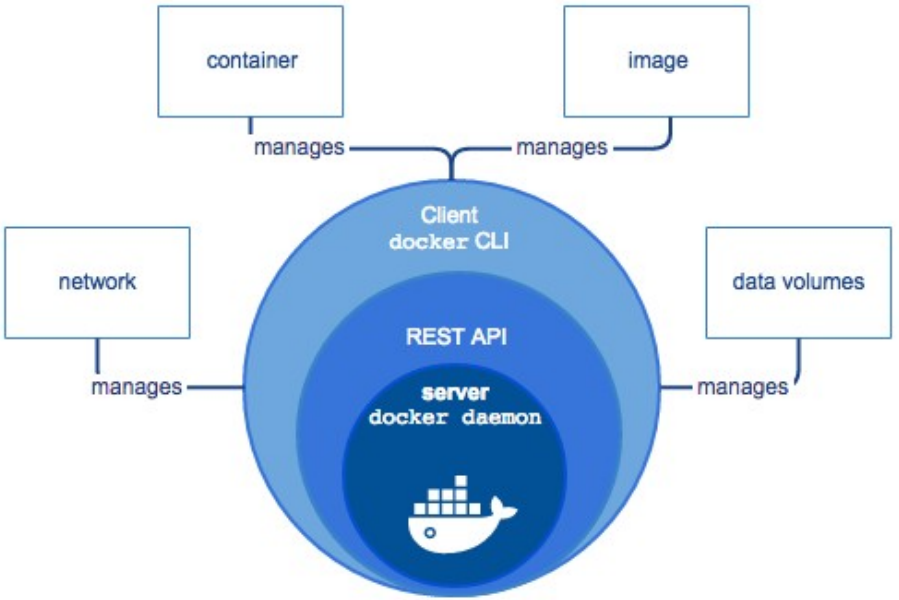
Платформа Docker позволяет «упаковать» приложение со всем его окружением и зависимостями в контейнер. Как результат, системы, основанные на контейнерах, легко масштабировать, так как контейнеры можно переносить и воспроизводить.



Движок Docker

Движок Docker (Docker Engine) — это клиент-серверное приложение. Компания Docker разделила движок Docker на два продукта.

- Docker Community Edition (CE) — это бесплатное ПО, основанное на open source инструментах.
- Docker Enterprise — это платная версия системы, дающая пользователям дополнительные возможности в области поддержки систем, управления ими и безопасности.



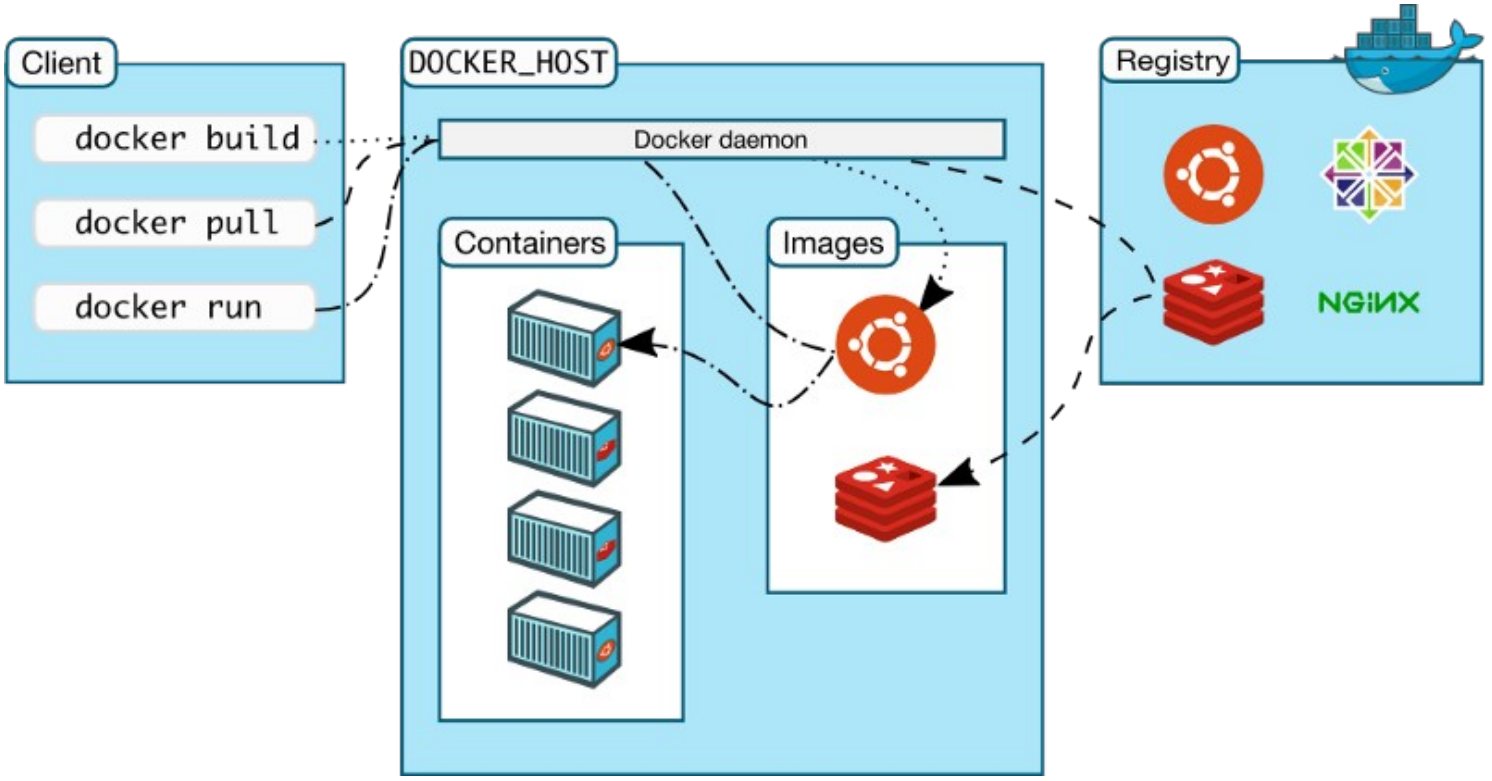
Демон и клиент Docker

- **Docker-демон (Docker-daemon)** — сервер Docker, входящий в состав программных средств Docker. Демон управляет Docker-объектами (сети, хранилища, образы и контейнеры). Демон также может связываться с другими демонами для управления сервисами Docker.

- **Docker-клиент (Docker-client / CLI)** — интерфейс взаимодействия пользователя с Docker-демоном. Клиент и Демон — важнейшие компоненты «движка» Докера (Docker Engine). Клиент Docker может взаимодействовать с несколькими демонами.

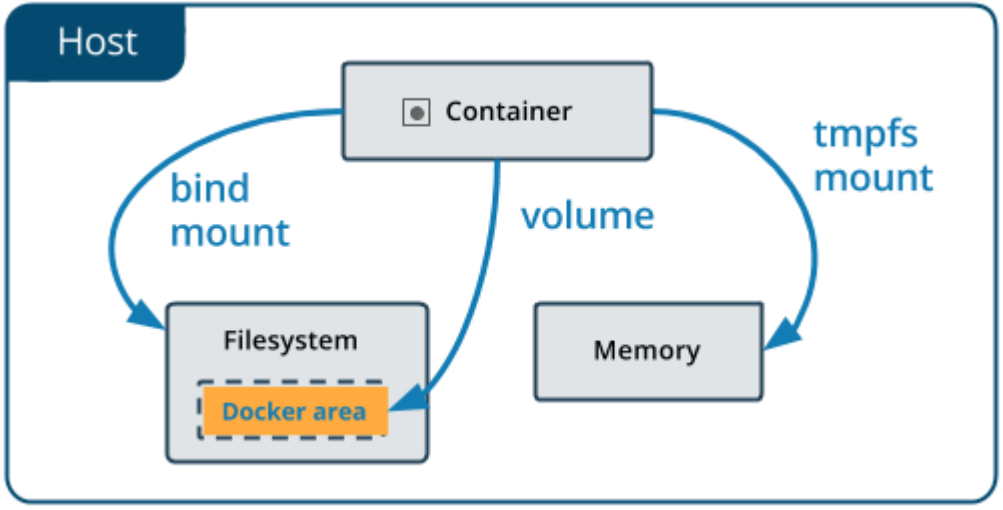
- **REST API** — механизм, отвечающий за организацию взаимодействия Docker-клиента и Docker-демона.

Демон и клиент Docker



Тома Docker

- **Тома Docker (Docker Volumes)** представляют собой наиболее предпочтительный механизм постоянного хранения данных, потребляемых или производимых приложениями.



Реестр Docker

- **Реестр Docker (Docker Registry)** — представляет собой удалённую платформу, используемую для хранения образов Docker. В ходе работы с Docker образы отправляют в реестр и загружают из него. Подобный реестр может быть организован тем, кто пользуется Docker. Кроме того, поставщики облачных услуг могут поддерживать и собственные реестры. Например, это касается AWS и Google Cloud.

- **Хаб Docker (Docker Hub)** — это бесплатный публичный реестр образов Docker. Кроме того, именно этот реестр используется при работе с Docker по умолчанию.

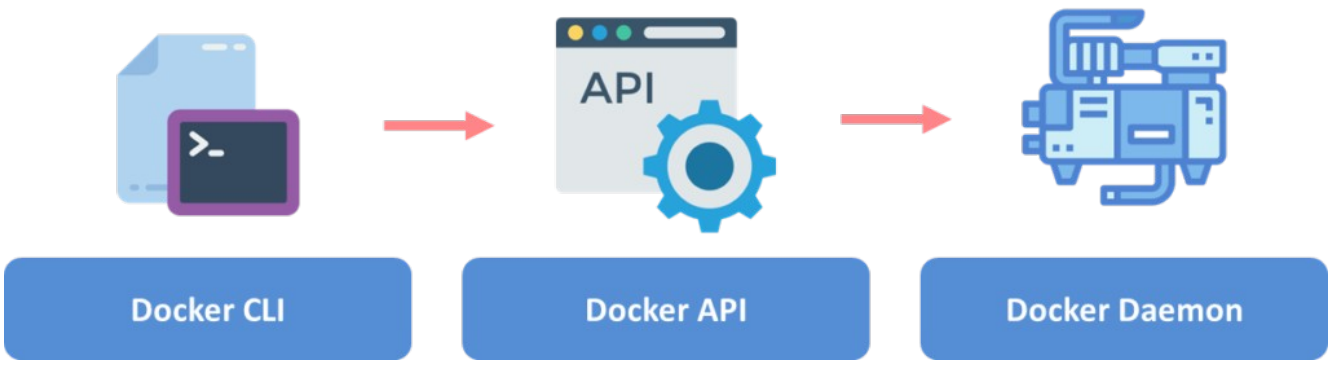
- **Репозиторием Docker (Docker Repository)** называют набор образов Docker, обладающих одинаковыми именами и разными тегами. Теги это идентификаторы образов.

Масштабирование

- **Сетевая подсистема Docker** — среда, которая позволяет организовывать взаимодействие контейнеров.
- **Docker Compose** — технология, упрощающая работу с многоконтейнерными приложениями.
- **Docker Swarm** — средство для управления развёртыванием контейнеров.
- **Сервисы Docker** — контейнеры в продакшне.

Команды Docker CLI

- Команды интерфейса командной строки (CLI) Docker начинаются с ключевого слова *docker*. Далее идет указание на что именно будет направлена некая команда, а потом следует сама команда. Например: *docker container stop*
- Если команда направлена на конкретный образ или контейнер, то в ней используется имя или идентификатор такого образа или контейнера.



Команды управления образами

Общая схема: $ docker image <COMMAND> Список некоторых команд:

- build — сборка образа
- push — отправить образ в registry
- pull — скачать образ из registry
- ls — вывод списка образов
- history — вывод сведений о слоях образа
- inspect — вывод подробной информации об образе
- rm — удаление образа

Команды управления контейнерами

Общая схема: $ docker container <COMMAND> Список некоторых команд:

- create - создание контейнера из образа
- start - запуск существующего контейнера
- run - создание контейнера и его запуск
- stop - остановка работающего контейнера
- rm — удаление остановленного контейнера
- ls — вывод списка контейнеров
- inspect — вывод информации о контейнере

Базовые команды

Общая схема: $ docker <COMMAND> Список некоторых команд:

- ps/ps -a — показать запущенные/все контейнеры
- images — показать локальные образы
- search — поиск образа в registry
- pull — скачать образ из registry
- build — собрать образ
- run — запуск контейнера
- rm/rmi — удалить контейнер/образ
- start/stop/restart — запуск/остановка/перезапуск контейнера

Запуск контейнеров Docker

$ docker start <CONTAINER>

Запуск существующего контейнера, который был уже загружен или создан.

$ docker stop <CONTAINER>

Остановка запущенного контейнера.

$ docker stop $(docker container ls -aq)

Остановка всех запущенных контейнеров по списки ID этих контейнеров

$ docker exec -ti <CONTAINER> [COMMAND]

Запуск команды в работающем контейнере

Запуск через RUN

$ docker run -ti - image <IMAGE> <CONTAINER> [COMMAND]

Существует чёткое различие между *run* и *start*. По сути *run* делает две вещи: создаёт новый контейнер образа и выполняет этот контейнер.

$ docker run -ti --rm - image <IMAGE> <CONTAINER> [COMMAND]

Это команда, предназначенная для единовременного создания и запуска контейнера. После выполнения этой команды, созданный контейнер удаляется.

$ docker run -d <IMAGE>

Запустить контейнер в фоновом режиме.

$ docker run -d --name <MYNAME> <IMAGE>

Задать имя контейнеру и запустить в фоновом режиме.