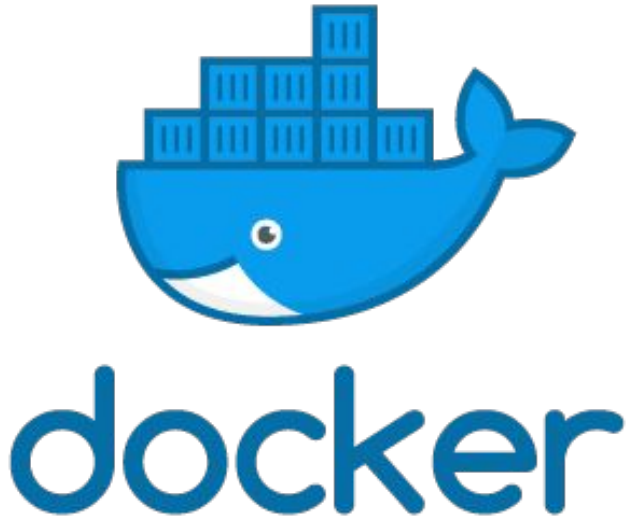

docker

What is Docker tool

Docker is an open source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.

- Containers simplify delivery of distributed applications, and have become increasingly popular as organizations shift to cloud-native development and hybrid multi cloud environments.
- Developers can create containers without Docker, but the platform makes it easier, simpler, and safer to build, deploy and manage containers. Docker is essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation through a single API.

How containers work, and why they're so popular ?

- Containers are made possible by process isolation and virtualization capabilities built into the Linux kernel. These capabilities - such as *control groups* (Cgroups) for allocating resources among processes, and *namespaces* for restricting a processes access or visibility into other resources or areas of the system - enable multiple application components to share the resources of a single instance of the host operating system in much the same way that a hypervisor enables multiple virtual machines (VMs) to share the CPU, memory and other resources of a single hardware server.
- container technology offers all the functionality and benefits of VMs - including application isolation, cost-effective scalability, and disposability - plus important additional advantages:

Why use Docker?

- Improved—and seamless—portability: While LXC containers often reference machine-specific configurations, Docker containers run without modification across any desktop, data center and cloud environment.
- Even lighter weight and more granular updates: With LXC, multiple processes can be combined within a single container. With Docker containers, only one process can run in each container.
  Automated container creation: Docker can automatically build a container based on application source code.
- Container versioning: Docker can track versions of a container image, roll back to previous versions, and trace who built a version and how. It can even upload only the deltas between an existing version and a new one.
- Container reuse: Existing containers can be used as *base images*—essentially like templates for building new containers.
- Shared container libraries: Developers can access an open-source registry containing thousands of user-contributed containers.

Docker tools and terms

DockerFile

Every Docker container starts with a simple text file containing instructions for how to build the Docker container image. *DockerFile* automates the process of Docker image creation. It's essentially a list of command-line interface (CLI) instructions that Docker Engine will run in order to assemble the image.

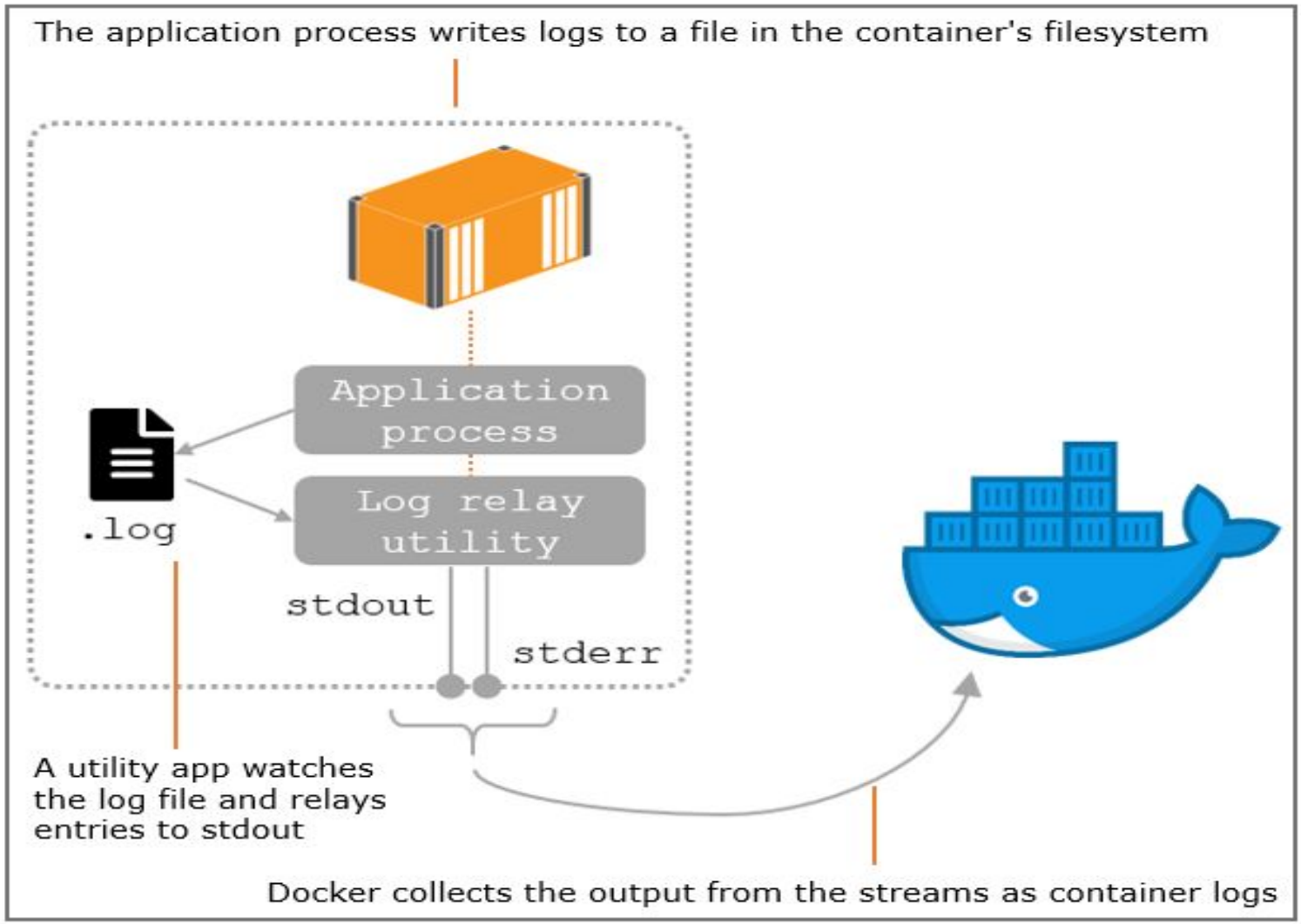
Docker images

Docker images contain executable application source code as well as all the tools, libraries, and dependencies that the application code needs to run as a container. When you run the Docker image, it becomes one instance (or multiple instances) of the container.

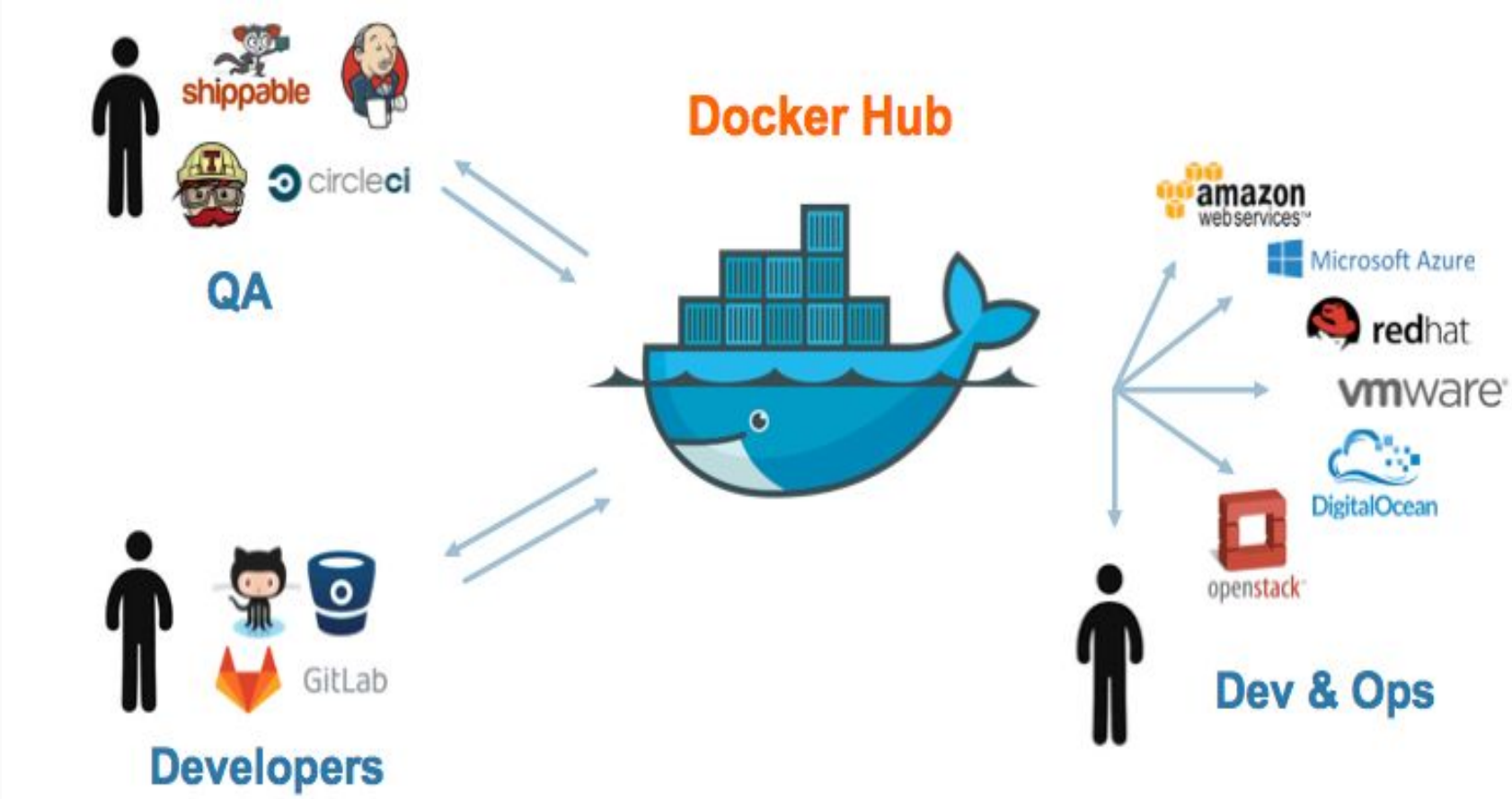
- It's possible to build a Docker image from scratch, but most developers pull them down from common repositories. Multiple Docker images can be created from a single base image, and they'll share the commonalities of their stack.

Docker containers

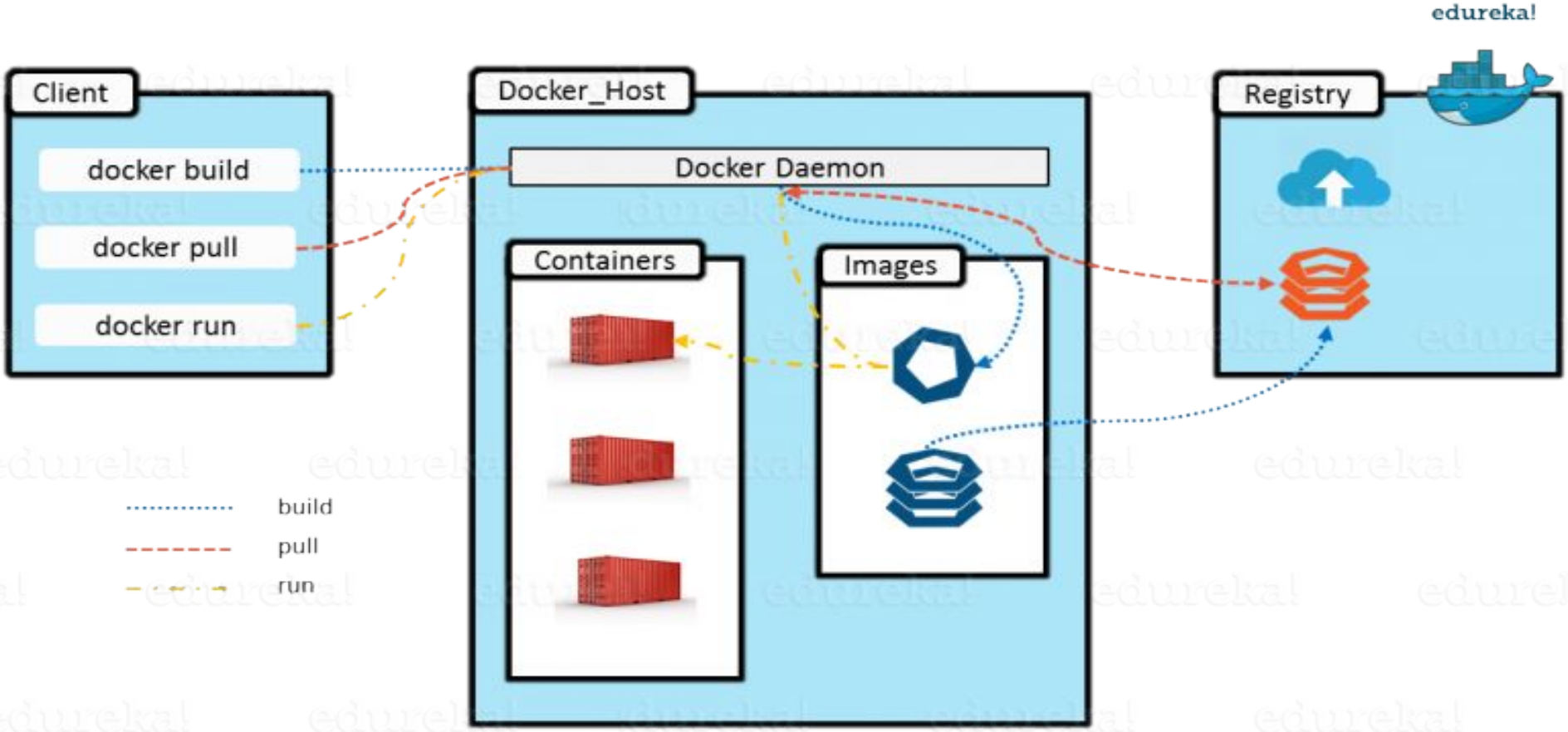
Docker containers are the live, running instances of Docker images. While Docker images are read-only files, containers are live, ephemeral, executable content. Users can interact with them, and administrators can adjust their settings and conditions using docker commands.



Docker Hub


shippable
circleci
QA
Docker Hub
amazon web services
Microsoft Azure
redhat
vmware
DigitalOcean
openstack
GitLab
Developers
Dev & Ops

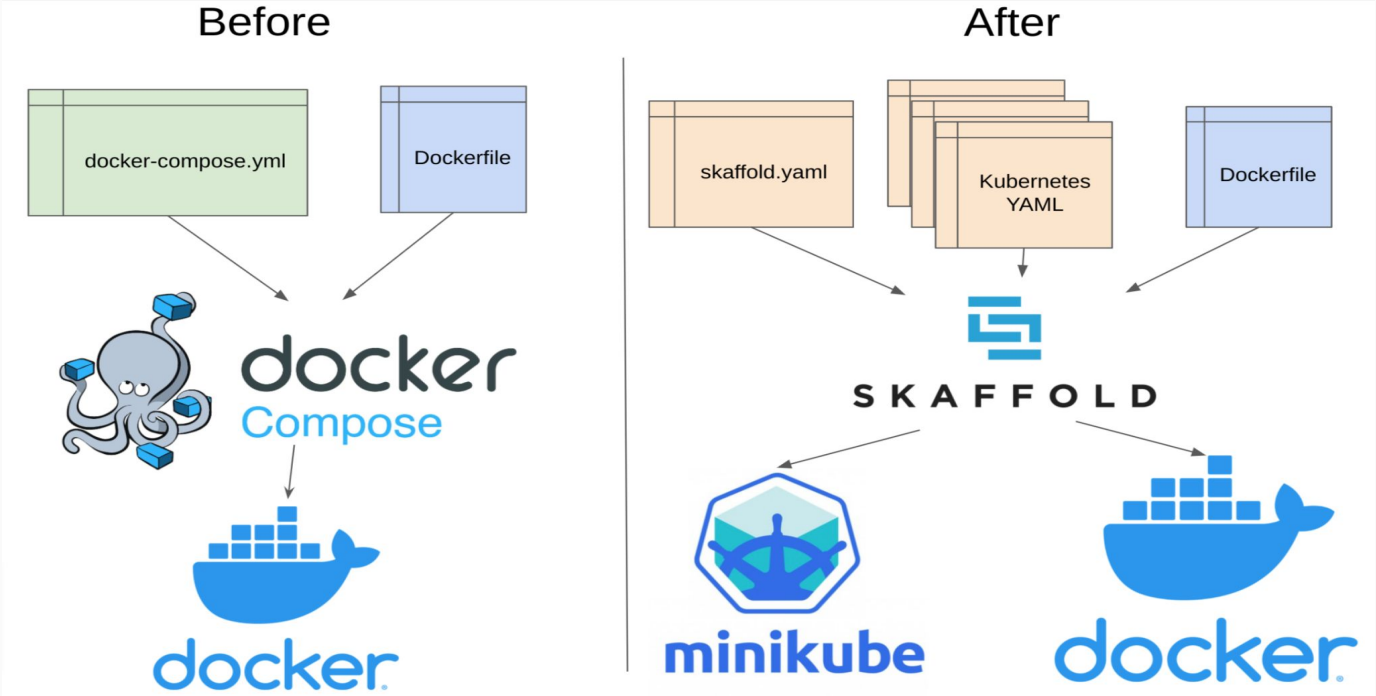
Docker daemon



Docker deployment and orchestration

If you’re running only a few containers, it’s fairly simple to manage your application within Docker Engine, the industry de facto runtime. But if your deployment comprises thousands of containers and hundreds of services, it’s nearly impossible to manage that workflow without the help of these purpose-built tools.

Docker Compose



If you're building an application out of processes in multiple containers that all reside on the same host, you can use Docker Compose to manage the application's architecture. Docker Compose creates a YAML file that specifies which services are included in the application and can deploy and run containers with a single command.

Docker Swarm