Docker Certified Associate Training

Source: https://docs.docker.com

Introduction to Docker

Learning Objectives

By the end of this lesson, you will be able to:

- Describe Docker and its functionalities
- Explain Docker Architecture
- State the properties of Docker
- Demonstrate the installation of Docker Engine

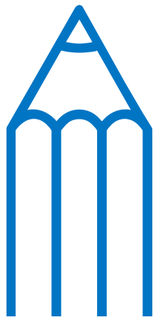
Docker: Overview

Introduction

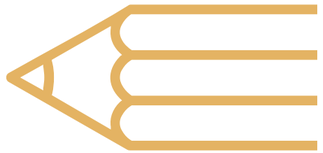
- Docker is a platform for developers and sysadmins to develop, ship, and run applications by using containers.
- Docker helps the user to quickly assemble applications from its components and eliminates the friction during code shipping.
- Docker aids the user to test and deploy the code into production.

Functionalities

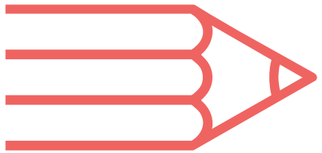
Test the application



Develop application and its supporting components using containers



Docker provides a platform for the user to:



Deploy the application into production environment, as a container or an orchestrated service

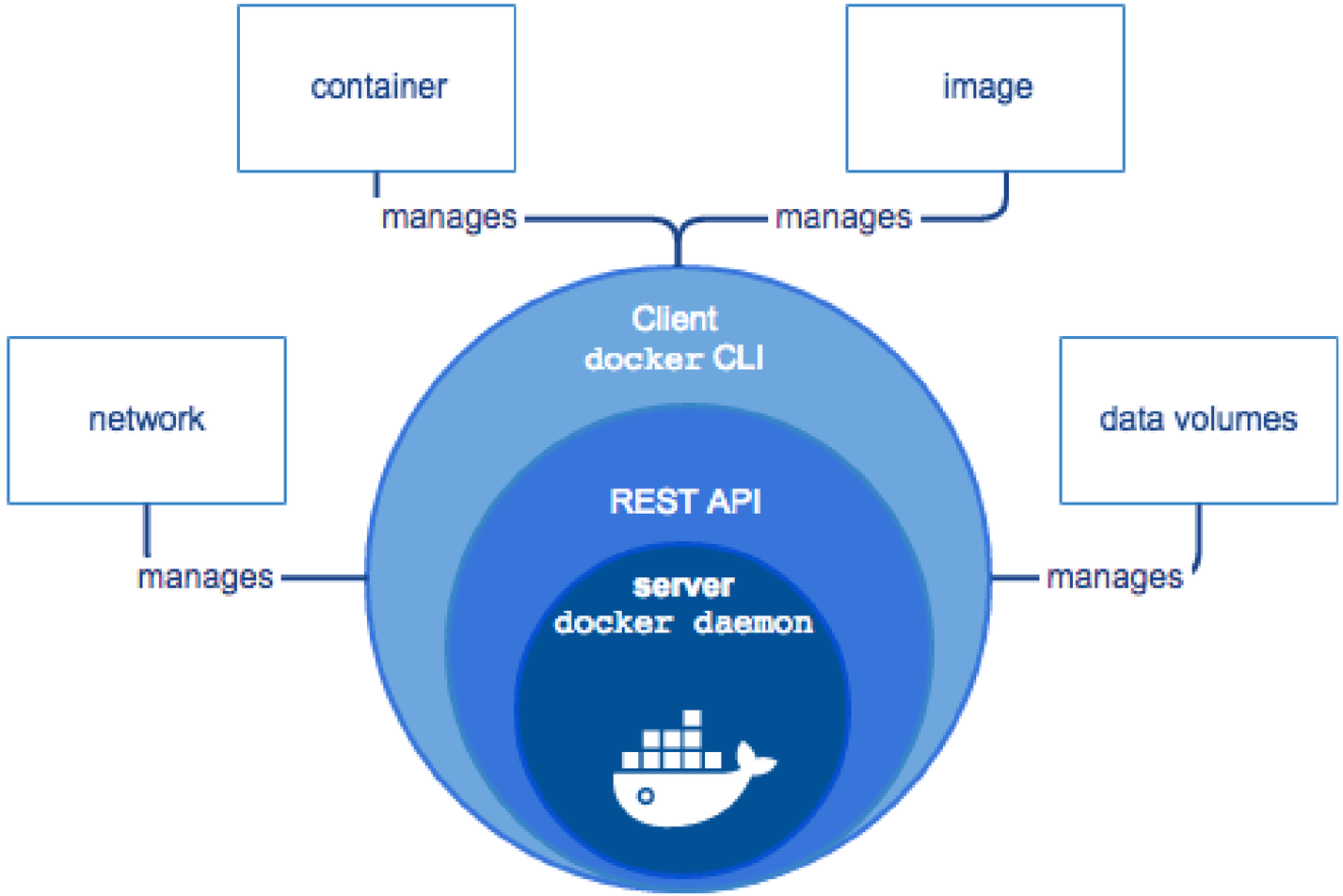
Docker Engine

Docker Engine

Docker Engine is a client-server application with these major components:

- Server: It is a type of long-running program called a daemon process (the dockerd command).
- REST API: It specifies the interfaces that programs can use to communicate with the daemon and instructs it on what to do next.
- CLI: It is a command line interface client that is used to write the docker commands.

Docker Engine



Source: https://docs.docker.com/engine/docker-overview/#docker-engine

Docker Engine

Docker Engine supports the tasks and workflows involved to build, ship, and run container-based applications. The engine creates a daemon process on the server side that hosts volumes of files, containers, networks, and storage.

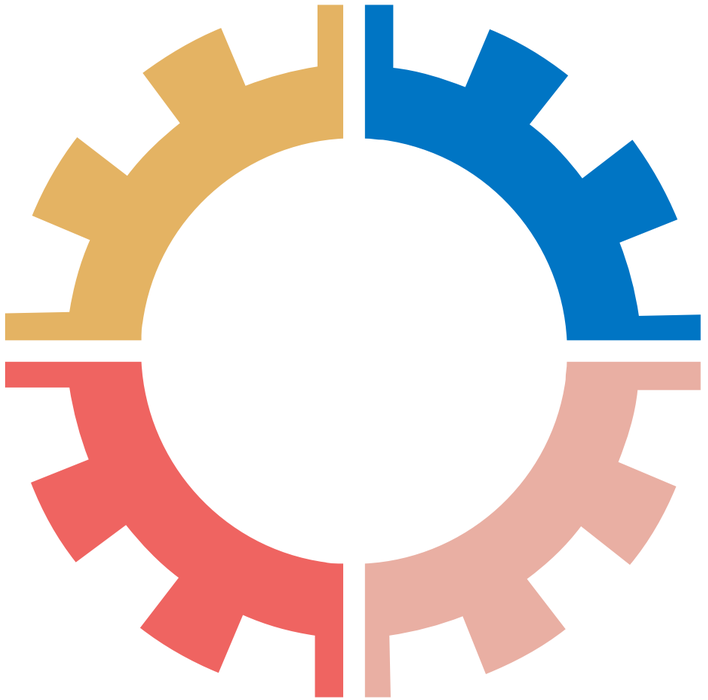
Docker consists of:

- The Docker Engine: It is a lightweight and powerful open source containerization technology combined with a workflow for building and containerizing your applications.
- Docker Hub: It is a Software as a Service (SAAS) for sharing and managing your application stacks.

Docker Properties

Docker Properties

Provides fast delivery of the applications



Is deployable and scalable

Has high density and runs more workloads

Aids in quick deployment for easy management

Install Docker

Installation of Docker

OS Requirements:

The user must have the 64-bit version of any one of the following Ubuntu versions:

- Disco 19.04
- Cosmic 18.10
- Bionic 18.04 (LTS)
- Xenial 16.04 (LTS)

Install, Upgrade, and Uninstall Docker CE for Linux

Problem Statement: You have been asked to install Docker Community Edition for Linux. Upgrade it if required and remove it later once your requirement is fulfilled.

Steps to Perform:

1. Install Docker CE from the Docker repository.
2. Install Docker CE from the *.deb* package file.
3. Upgrade Docker CE using the *update* command.
4. Uninstall Docker CE using the *purge* command.

ASSISTED PRACTICE

Configure Docker Daemon to Start on Boot

Problem Statement: Your manager has requested you to configure Docker daemon to start the Docker service on system boot to avoid manually starting the Docker service.

Steps to Perform:

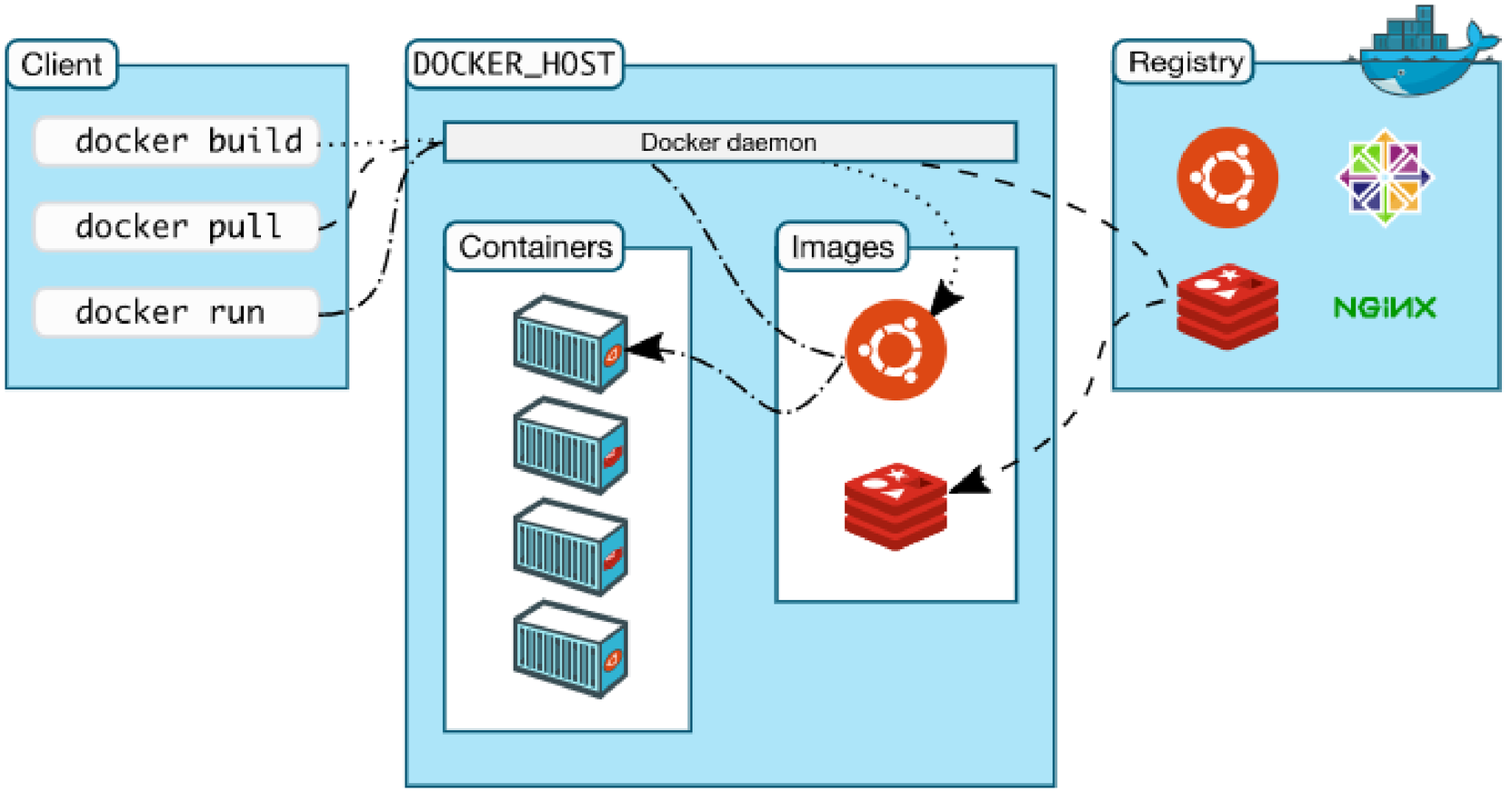
1. Use *systemd* command to start the Docker service on system boot for Ubuntu 16.04 and higher versions.
2. Use *systemctl* command to enable, disable, or reload Docker daemon on system boot.
3. Navigate to the *daemon.json* configuration file to configure the *daemon flags* and *environment variables*.

ASSISTED PRACTICE

Docker Architecture

Docker Architecture

Docker uses a client-server architecture. The docker client interacts with the Docker daemon that performs running, heavy lifting of building, and distribution of Docker containers.



Source: https://docs.docker.com/engine/docker-overview/#docker-architecture

Docker Architecture

Docker daemon

It accepts the Docker API requests and manages Docker objects, such as images, containers, networks, and volumes. A daemon can also communicate with other daemons to manage Docker services.

Docker client

It is the primary path for Docker users to interact with the Docker application.

Docker Architecture

Docker registries

It stores Docker images. A Docker registry can be of classified into two categories:

Local Registry: It helps the user to pull the image.

Docker Trusted Registry: It is a feature the Docker Enterprise that helps the user to pull the image and scan the image.

Docker objects

When the user uses Docker, in order to package the application and store it in isolated bundles, user creates and uses objects, such as images, containers, networks, volumes, plugins.

Images and Containers

Problem Statement: Your team lead wants you to containerize an existing application into docker images. In order to prepare you for that, first he wants you to get familiar with various aspects of Docker images and containers.

Steps to Perform:

1. Pull an image from Docker Hub.
2. Create a new container from image pulled.
3. Stop the running container.
4. Delete the container.
5. Remove the image.

ASSISTED PRACTICE

Key Takeaways

- Docker is a platform for developers and sysadmins to develop, ship, and run applications.
- Docker uses a client-server architecture.
- The docker client interacts with the Docker daemon that performs running, heavy lifting of building, and distribution of Docker containers.
- Docker Engine: Community is supported on x86_64 (or amd64), armhf, arm64, s390x (IBM Z), and ppc64le (IBM Power) architectures.