**What is Docker & Docker Container ?**

*What is Docker ?* – Docker is a containerization platform that packages your application and all its dependencies together in the form of a docker container to ensure that your application works seamlessly in any environment.

*What is Container ?* – Docker Container is a standardized unit which can be created on the fly to deploy a particular application or environment. It could be an Ubuntu container, CentOs container, etc. to full-fill the requirement from an operating system point of view. Also, it could be an application oriented container like CakePHP container or a Tomcat-Ubuntu container etc.

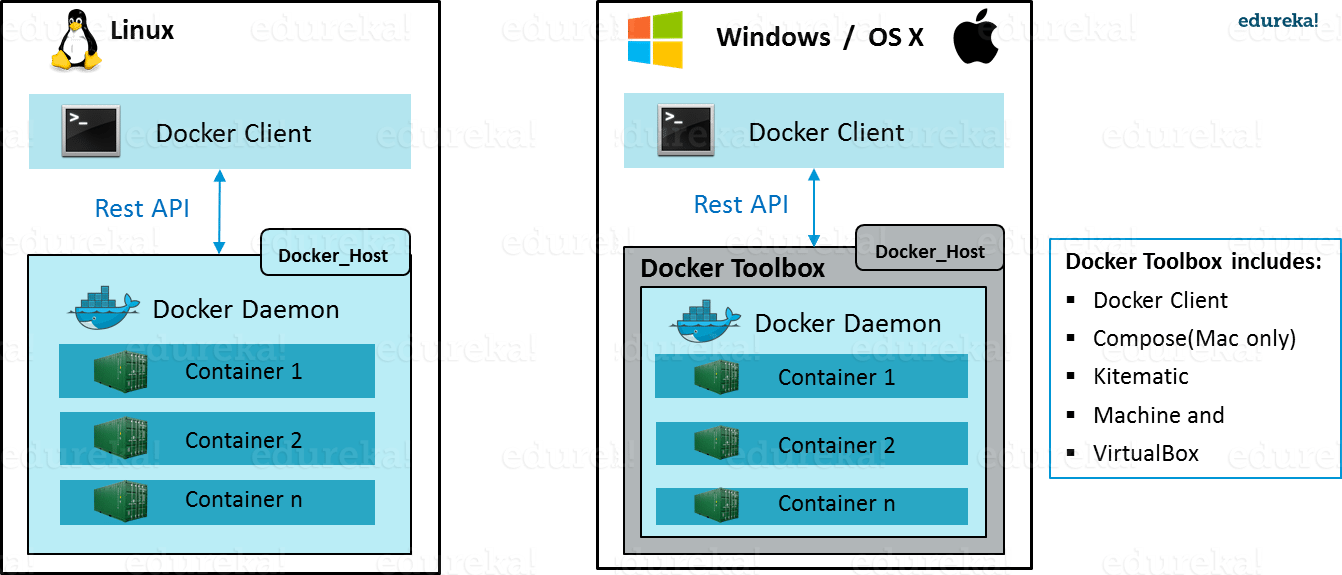
<https://www.edureka.co/blog/what-is-docker-container>

## ****What is Docker Engine?****

Now I will take you through Docker Engine which is the heart of the system.

Docker Engine is simply the application that is installed on your host machine. It works like a client-server application which uses:

* A **server** which is a type of long-running program called a daemon process
* A command line interface (CLI) **client**
* REST API is used for communication between the CLI client and Docker Daemon



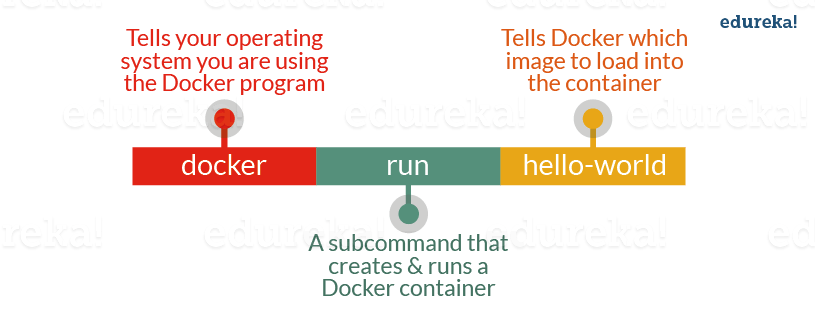
As per the above image, in a Linux Operating system, there is a client which can be accessed from the terminal and a Host which runs the Daemon. We build our images and run containers by passing commands from the CLI client to the Daemon.

However, in case of Windows/Mac there is an additional Toolbox component inside the Docker host. This Docker Toolbox is an installer to quickly and easily install and setup a Docker environment on your Windows/iOS. This Toolbox installs Docker Client, Machine, Compose (Mac only), Kitematic and VirtualBox.

Let’s now understand three important terms, i.e.**Docker Images**, **Docker Containers** and **Docker Registry**.

## ****What is Docker Image?****

Docker Image can be compared to a template which is used to create Docker Containers. They are the building blocks of a Docker Container. These Docker Images are created using the build command. These Read only templates are used for creating containers by using the run command. We will explore Docker commands in depth in the “Docker Commands blog”.

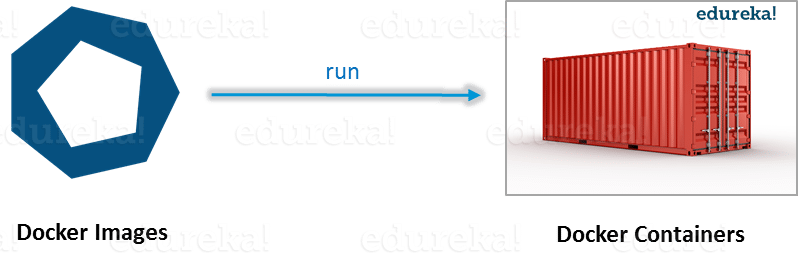


Docker lets people (or companies) create and share software through Docker images. Also, you don’t have to worry about whether your computer can run the software in a Docker image — a Docker container can always run it.

I can either use a ready-made docker image from docker-hub or create a new image as per my requirement. In the Docker Commands blog we will see how to create your own image.

## ****What is Docker Container?****

Containers are the ready applications created from Docker Images. Or you can say they are running instances of the Images and they hold the entire package needed to run the application. This happens to be the ultimate utility of the technology.



## ****What is Docker Registry?****

Finally, Docker Registry is where the Docker Images are stored. The Registry can be either a user’s local repository or a public repository like a Docker Hub allowing multiple users to collaborate in building an application. Even with multiple teams within the same organization can exchange or share containers by uploading them to the Docker Hub, which is a cloud repository similar to GitHub.

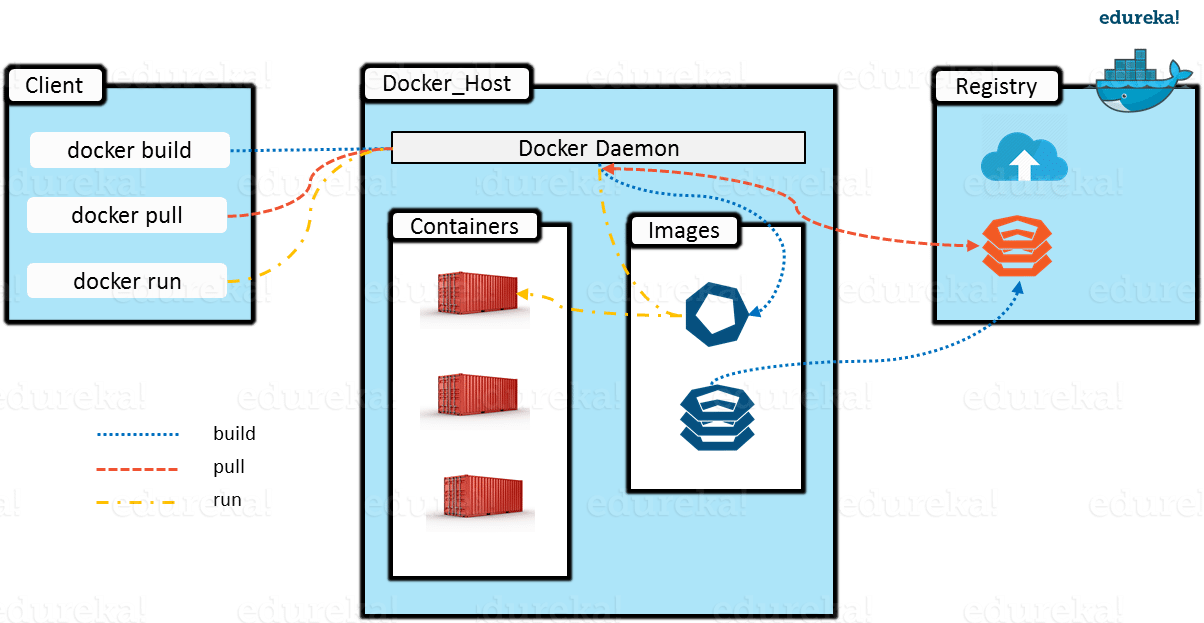
## ****What is Docker Architecture?****

Docker Architecture includes a Docker client – used to trigger Docker commands, a Docker Host – running the Docker Daemon and a Docker Registry – storing Docker Images. The Docker Daemon running within Docker Host is responsible for the images and containers.

[[](https://www.edureka.co/docker-training)](https://www.edureka.co/docker-training" \t "_blank)

### [Docker Certified Associate Training](https://www.edureka.co/docker-training" \t "_blank)

[Weekday / Weekend BatchesSee Batch Details](https://www.edureka.co/docker-training" \t "_blank)



* To build a Docker Image, we can use the CLI (client) to issue a build command to the Docker Daemon (running on Docker\_Host). The Daemon will then build an image based on our inputs and save it in the Registry, which can be either Docker hub or a local repository
* If we do not want to create an image, then we can just pull an image from the Docker hub, which would have been built by a different user
* Finally, if we have to create a running instance of my Docker image, we can issue a run command from the CLI, which will create a Container.

The aforementioned is a simple functionality of this technology!

I hope you enjoyed this blog. Now you are ready to get hands on experience with this. I will come up withthe third blog on Docker Commands.