**DOCKER(**[**https://dzone.com/articles/docker-command-cheatsheet?utm\_source=Top%205&utm\_medium=email&utm\_campaign=2017-01-06**](https://dzone.com/articles/docker-command-cheatsheet?utm_source=Top%205&utm_medium=email&utm_campaign=2017-01-06)**)**

1. What is Docker?

Docker is an open platform for developing, shipping, and running applications. Docker is designed to deliver your applications faster. With Docker you can separate your applications from your infrastructure AND treat your infrastructure like a managed application. Docker helps you ship code faster, test faster, deploy faster, and shorten the cycle between writing code and running code.

Docker provides a way to run almost any application securely isolated in a container.

Surrounding the container virtualization are tooling and a platform which can help you in several ways:

getting your applications (and supporting components) into Docker containers

distributing and shipping those containers to your teams for further development and testing

deploying those applications to your production environment, whether it be in a local data center or the Cloud.

1. What can I use Docker for?

Faster delivery of your applications

Deploying and scaling more easily

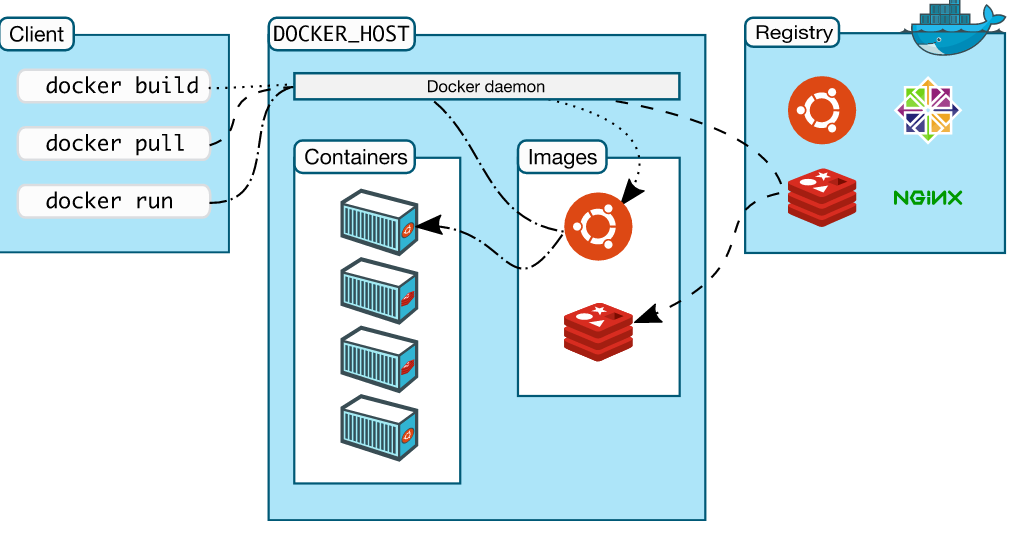
Achieving higher density and running more workloads

1. Docker has two major components:

* Docker: the open source container virtualization platform.
* [Docker Hub](https://hub.docker.com/): our Software-as-a-Service platform for sharing and managing Docker containers.

1. What is Docker’s architecture?

Docker uses a client-server architecture. The Dockerclient talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. Both the Docker client and the daemon canrun on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate via sockets or through a RESTful API.



1. Inside Docker

To understand Docker’s internals, you need to know about three components:

* Docker images: A Docker image is a read-only template. Images are used to create Docker containers. Docker provides a simple way to build new images or update existing images, or you can download Docker images that other people have already created. Docker images are the buildcomponent of Docker.
* Docker registries: Docker registries hold images. These are public or private stores from which you upload or download images. The public Docker registry is provided with the [Docker Hub](http://hub.docker.com/). It serves a huge collection of existing images for your use. These can be images you create yourself or you can use images that others have previously created. Docker registries are the distribution component of Docker
* Docker containers: Docker containers are similar to a directory. A Docker container holds everything that is needed for an application to run. Each container is created from a Docker image. Docker containers can be run, started, stopped, moved, and deleted. Each container is an isolated and secure application platform. Docker containers are the run component of Docker.

1. You can build Docker images that hold your applications.
2. You can create Docker containers from those Docker images to run your applications.
3. You can share those Docker images via [Docker Hub](https://hub.docker.com/) or your own registry.

Interactive container runs in the foreground.

Daemonized container runs in the background.

docker ps - Lists containers.

docker logs - Shows us the standard output of a container.

docker stop - Stops running containers.

Creating an docker container:

docker run -t -i --name myFirstContainer ubuntu /bin/bash

docker ps (list the running docker containers)

docker rm myFirstContainer (deleting container)

docker images (list the images)

docker build -t priya2711/test:snapshot . (here priya2711/test is my repository name from dokerhuband –t is tag)