



What is Docker ?

Docker is a tool designed to make it easier to deploy and run applications by using containers.

Most Common Problems for Developers :

The code works on developer system but the same code does not work on the {roduction Environment. Docker at a very basic level resolves this issue of an application working on one platform and not working on some other platform.

Containers allow a developer to package up an application with all of the parts it needs, Such as Libraries and other dependencies & ship it all out as one Package. Developer will package all the software and its components into a box which we call it as container & docker will take care of shipping this container to all the possible platforms. This is how we resolve the issue of an application working on one environment or a platform & not working on another.

Advantages of Docker :

Docker enables more efficient use of system resources


Instances of containerized apps use far less memory than virtual machines, they start up and stop more quickly, and they can be packed far more densely on their host hardware. All of this amounts to less spending on IT.


The cost savings will vary depending on what apps are in play and how resource-intensive they may be, but containers invariably work out as more efficient than VMs. It's also possible to save on costs of software licenses, because you need many fewer operating system instances to run the same workloads.

Docker enables faster software delivery cycles

Enterprise software must respond quickly to changing conditions. That means both easy scaling to meet demand and easy updating to add new features as the business requires.

Docker containers make it easy to put new versions of software, with new business features, into production quickly—and to quickly roll back to a previous version





if you need to. They also make it easier to implement strategies like blue/green deployments.

Docker enables application portability

Where you run an enterprise application matters—behind the firewall, for the sake of keeping things close by and secure; or out in a public cloud, for easy public access and high elasticity of resources. Because Docker containers encapsulate everything an application needs to run (and only those things), they allow applications to be shuttled easily between environments. Any host with the Docker runtime installed—be it a developer’s laptop or a public cloud instance—can run a Docker container.

Docker shines for microservices architecture

Lightweight, portable, and self-contained, Docker containers make it easier to build software along forward-thinking lines, so that you’re not trying to solve tomorrow’s problems with yesterday’s development methods.


One of the software patterns containers make easier is microservices, where applications are constituted from many loosely coupled components. By decomposing traditional, “monolithic” applications into separate services, microservices allow the different parts of a line-of-business app to be scaled, modified, and serviced separately—by separate teams and on separate timelines, if that suits the needs of the business.


Containers aren’t required to implement microservices, but they are perfectly suited to the microservices approach and to agile development processes generally.

Installing Docker on Centos :

#yum install -y yum-utils (Installing yum-utils package, This package is responsible for #yum-config-manager utility. Using this utility we can manage the repo's)

#yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo (Configuring Docker Repository)





#yum-config-manager --enable docker-ce-nightly (**Enabling Docker Repository**)

#yum install docker-ce docker-ce-cli containerd.io -y (**Installing Docker related packages**)

#which docker (**Verifying the docker installed location**)

#docker --version (**Displays Docker Version**)

#systemctl status docker (**Verifying Docker Service**)

#systemctl start docker (**Starting Docker Service**)

#systemctl enable docker (**Enabling Docker service to start automatically from the boot time**)

#systemctl status docker (**Ensure docker service is active and Enabled**)

#docker info (**Displays more detailed information about Docker host**)

