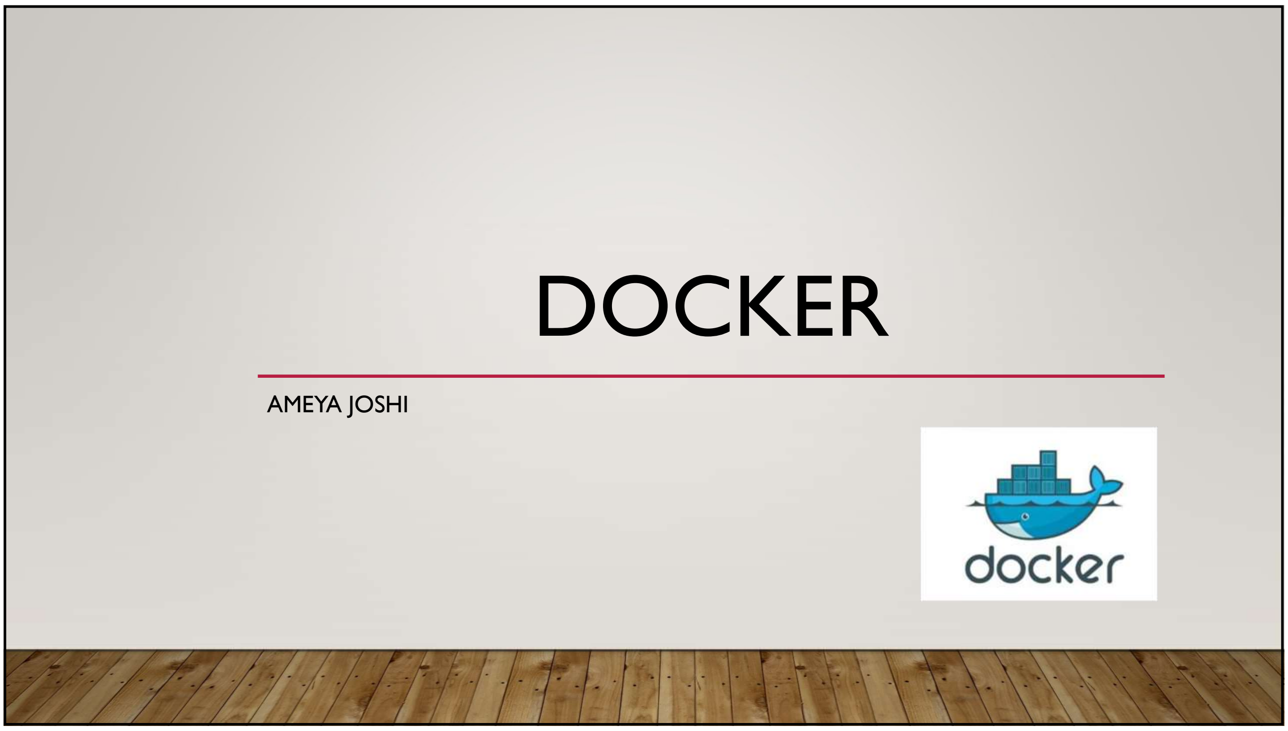
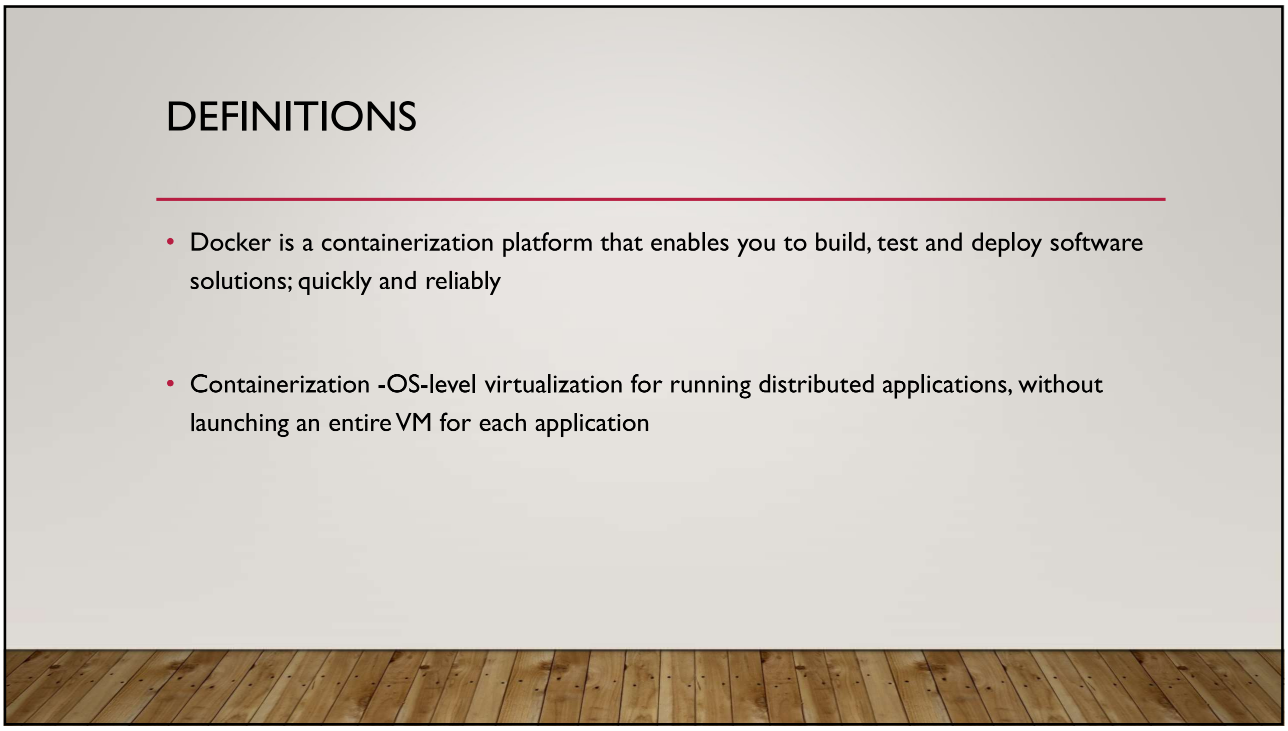


DOCKER

AMEYA JOSHI



DEFINITIONS

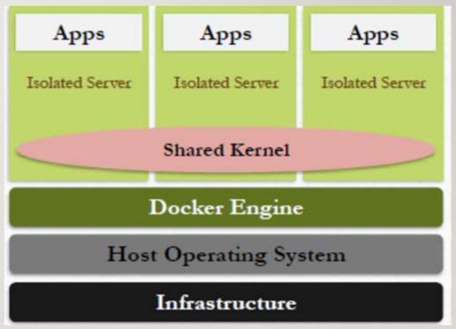
- Docker is a containerization platform that enables you to build, test and deploy software solutions; quickly and reliably
- Containerization -OS-level virtualization for running distributed applications, without launching an entire VM for each application

WHAT IS DOCKER

- A Docker host can run multiple containers, with a degree of isolation between the containers
- A Container is a standard unit which provides a single point of service
- Multiple containers when provided as a Service, create a scalable and reliable solution

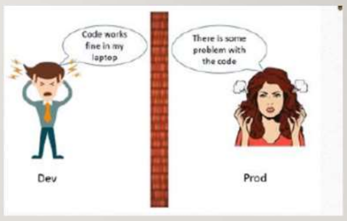
WHAT ARE CONTAINERS

- A running instance of a Docker Image
- Packages all the code and its dependencies
- OS level virtualization; containers share the operating system of the host
- Multiple containers run on a single control host and access a single kernel.



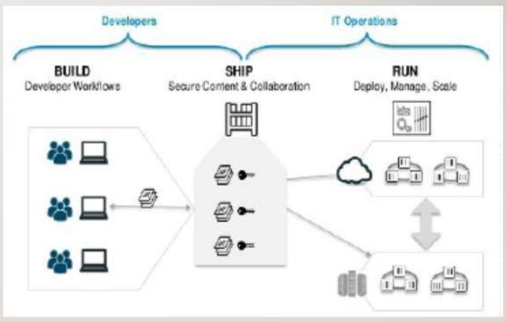
ADVANTAGES OF DOCKER

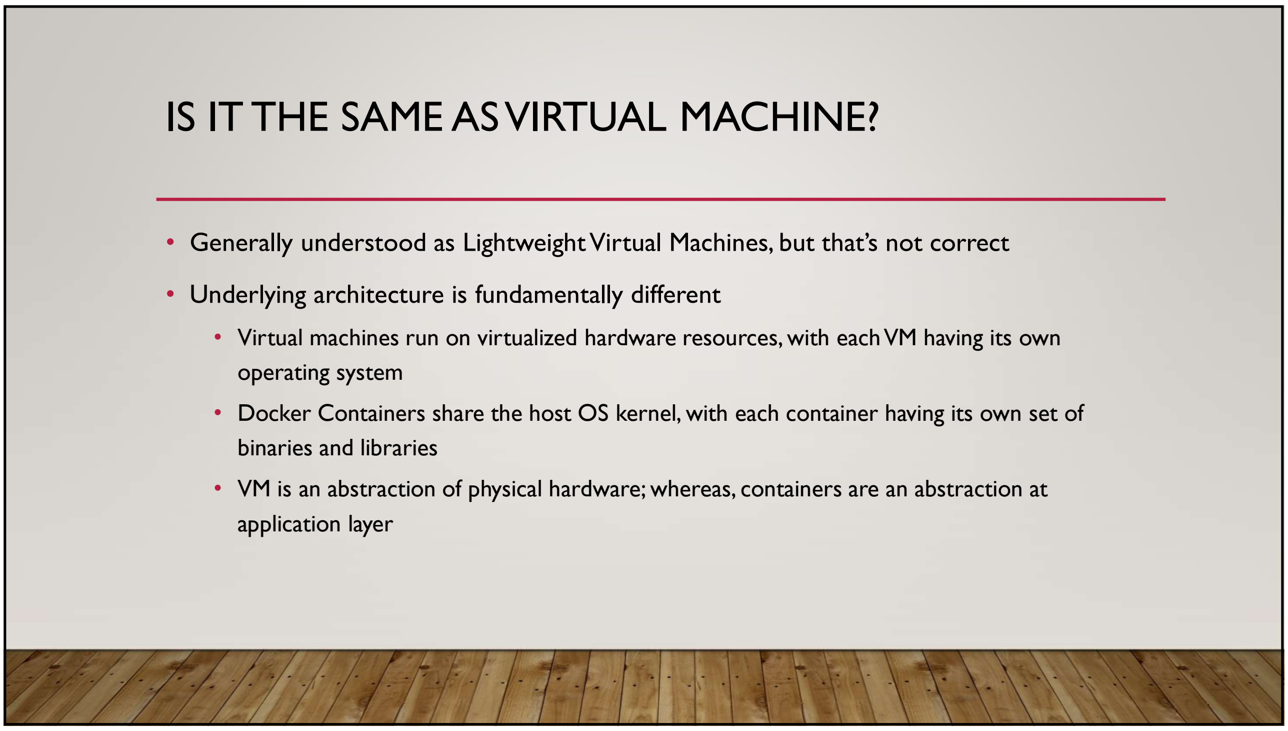
- Lightweight solution to virtual machines
- It is open source
- Convenient way to package up the application, including both the application layer and operating system dependencies


Code works fine in my laptop
There is some problem with the code
Dev
Prod

ADVANTAGES OF DOCKER

- Docker containers are easy to ship & deploy
- It is compatible with multi-could platforms.
- Integration with DevOps technologies
- Publish dockerized images on the internet, either publicly or privately, using Docker Cloud Registry

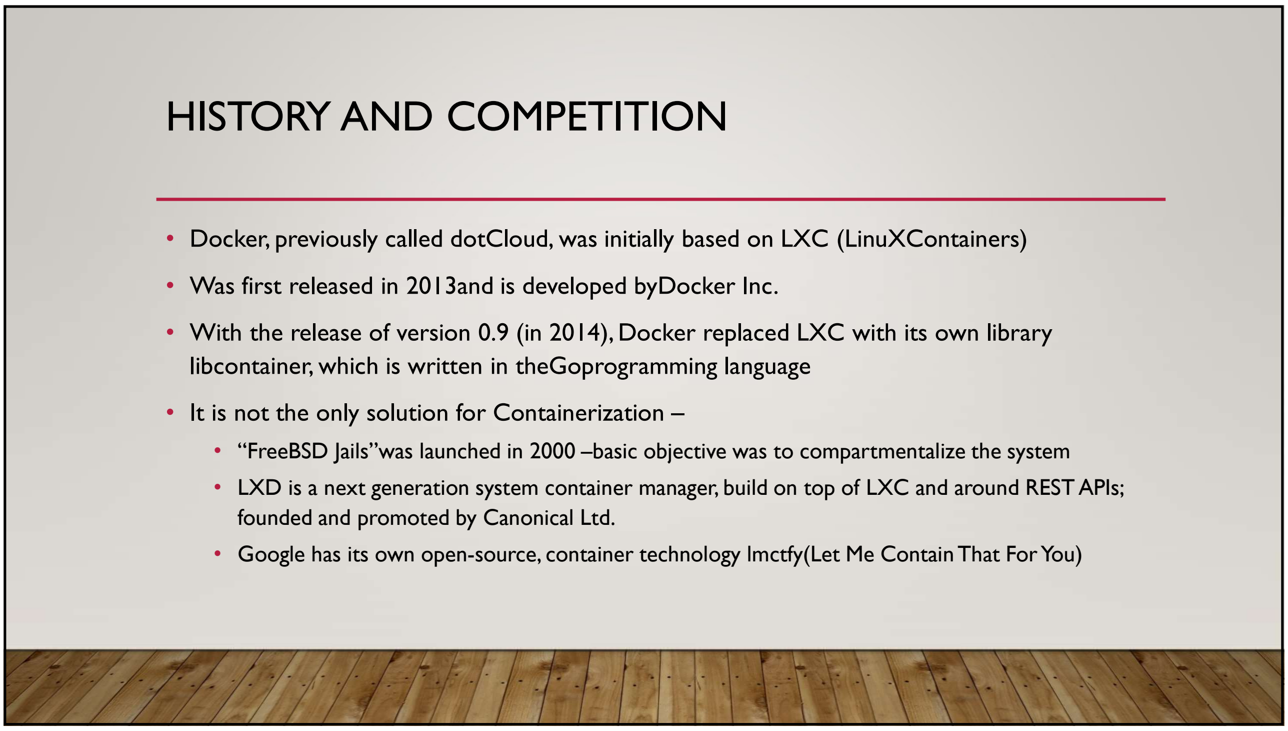




IS IT THE SAME AS VIRTUAL MACHINE?

- Generally understood as Lightweight Virtual Machines, but that's not correct
- Underlying architecture is fundamentally different
  - Virtual machines run on virtualized hardware resources, with each VM having its own operating system
  - Docker Containers share the host OS kernel, with each container having its own set of binaries and libraries
  - VM is an abstraction of physical hardware; whereas, containers are an abstraction at application layer

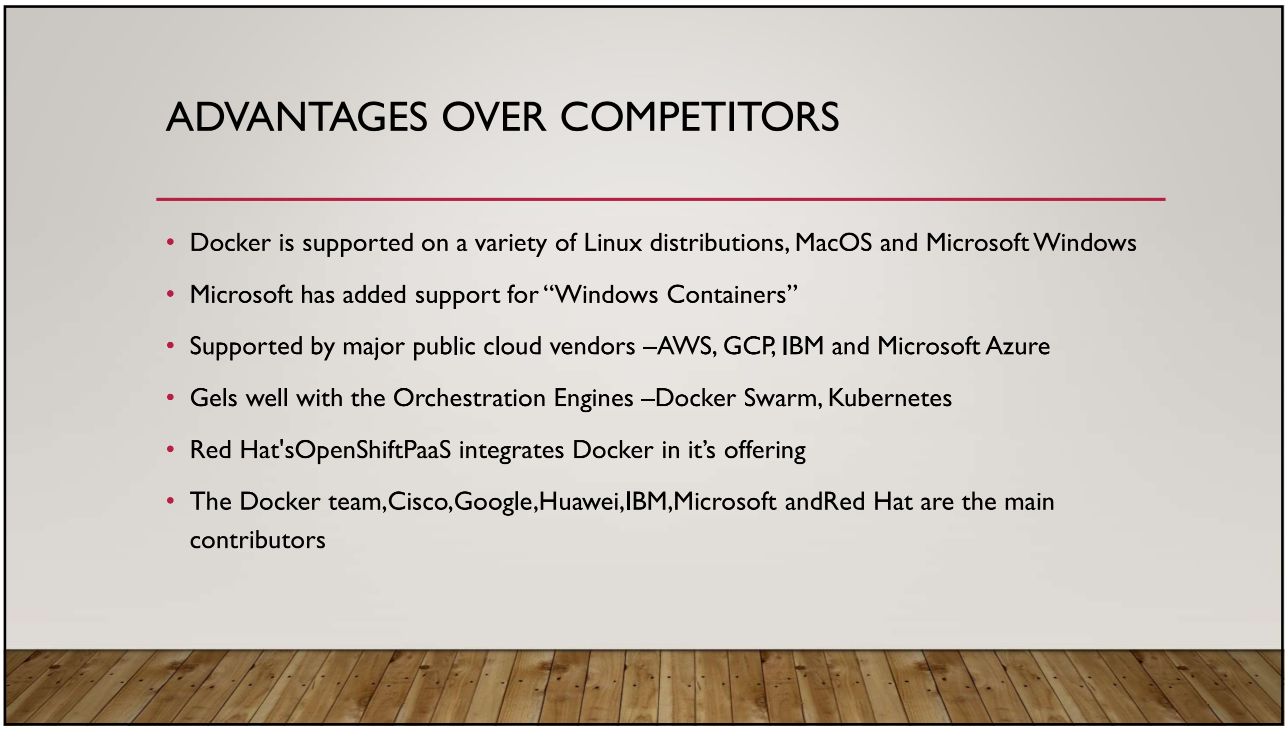
7



HISTORY AND COMPETITION

- Docker, previously called dotCloud, was initially based on LXC (LinuXContainers)
- Was first released in 2013and is developed byDocker Inc.
- With the release of version 0.9 (in 2014), Docker replaced LXC with its own library libcontainer, which is written in theGoprogramming language
- It is not the only solution for Containerization –
  - "FreeBSD Jails"was launched in 2000 –basic objective was to compartmentalize the system
  - LXD is a next generation system container manager, build on top of LXC and around REST APIs; founded and promoted by Canonical Ltd.
  - Google has its own open-source, container technology lmctfy(Let Me Contain That For You)

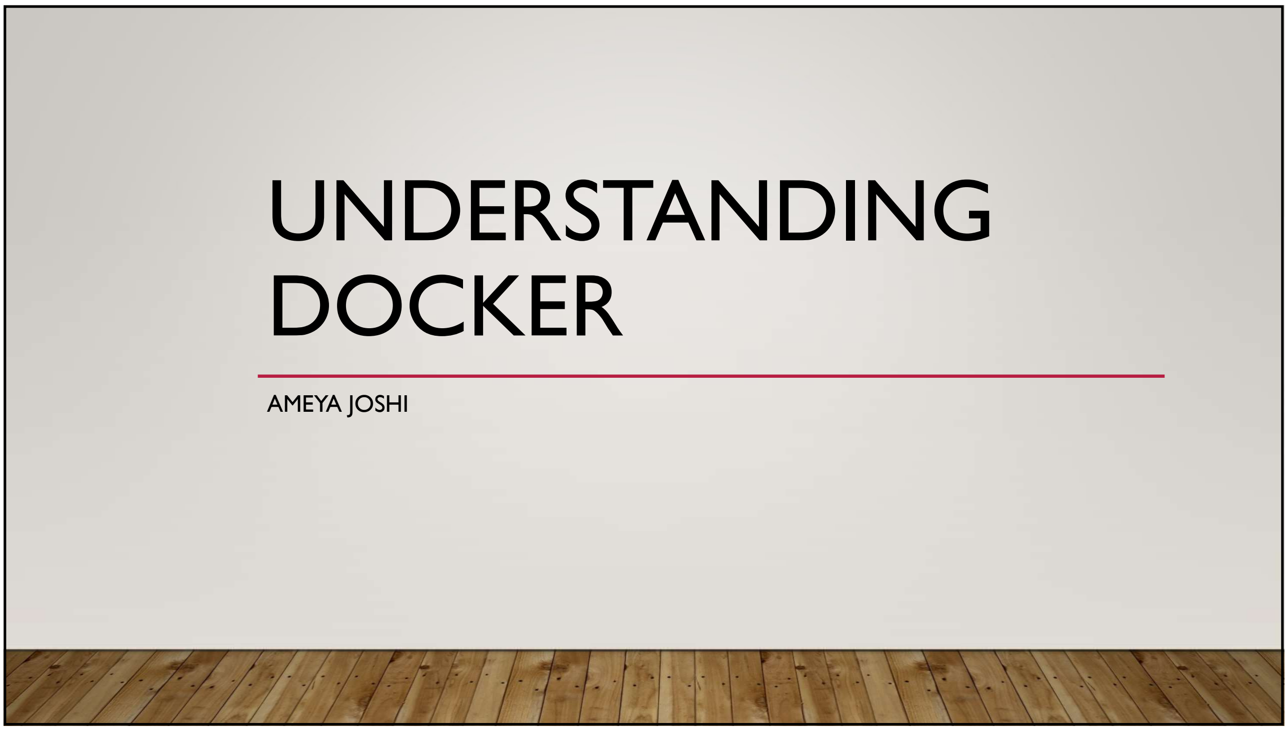
8



ADVANTAGES OVER COMPETITORS

- Docker is supported on a variety of Linux distributions, MacOS and Microsoft Windows
- Microsoft has added support for "Windows Containers"
- Supported by major public cloud vendors –AWS, GCP, IBM and Microsoft Azure
- Gels well with the Orchestration Engines –Docker Swarm, Kubernetes
- Red Hat'sOpenShiftPaaS integrates Docker in it's offering
- The Docker team,Cisco,Google,Huawei,IBM,Microsoft andRed Hat are the main contributors

9



UNDERSTANDING DOCKER

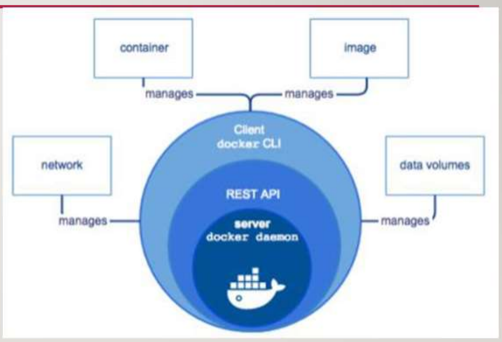
AMEYA JOSHI

10

DOCKER ARCHITECTURE

DOCKER ENGINE

- Engine gets installed on the Docker host
- ☐Client-server architecture based application which has the following components –
- ☐A daemon process (dockerd), which is a continuous running program
- ☐REST APIto communicate with the daemon process and provide instructions
- ☐Client (docker command), command line interface

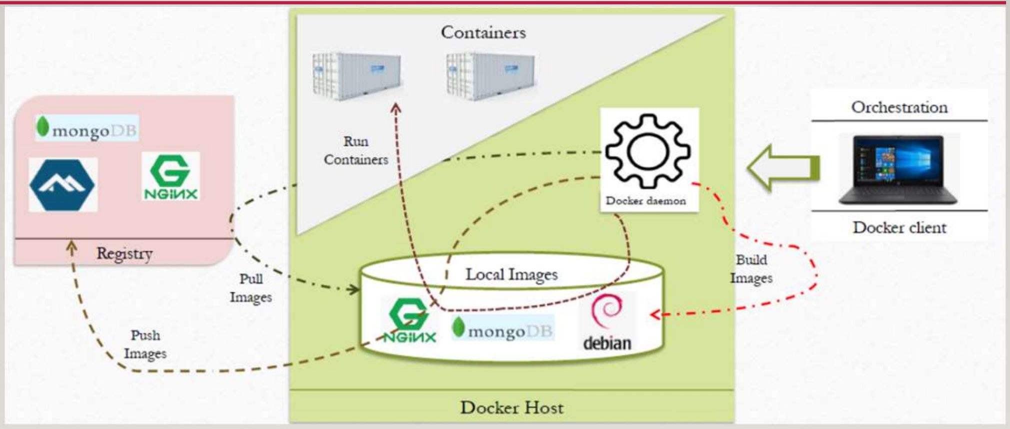


DOCKER OBJECTS

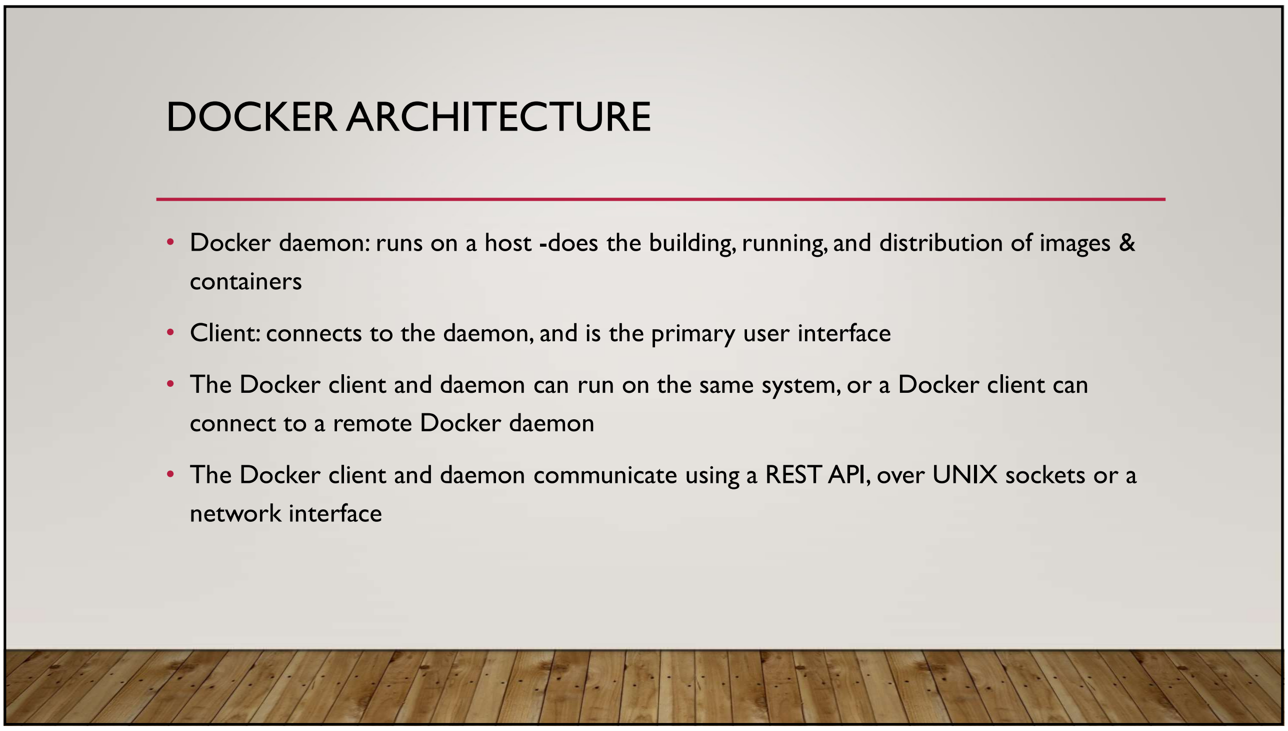
- Images: Read-only template with instructions for creating a Docker container
- •Container: Runnable instance of an image
- •Networks: Network interface to connect the container to external networks using the host machine’s network connection
- •Volumes: Mechanism for persisting data generated by and used by containers
- •Registry: Private or public registry of Docker images
- •Services: enables multi-host, multi-container deployment

13

DOCKER ARCHITECTURE

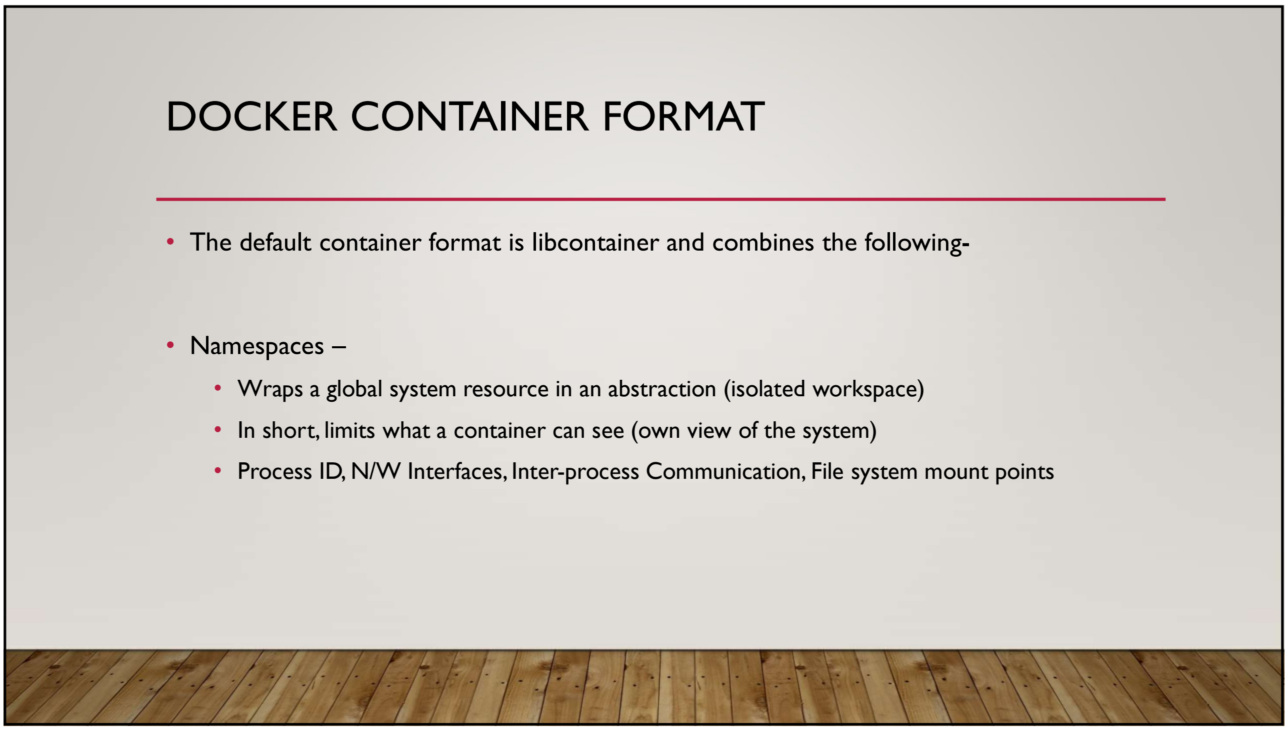


14



DOCKER ARCHITECTURE

- Docker daemon: runs on a host -does the building, running, and distribution of images & containers
- Client: connects to the daemon, and is the primary user interface
- The Docker client and daemon can run on the same system, or a Docker client can connect to a remote Docker daemon
- The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface



DOCKER CONTAINER FORMAT

- The default container format is libcontainer and combines the following-
- Namespaces –
  - Wraps a global system resource in an abstraction (isolated workspace)
  - In short, limits what a container can see (own view of the system)
  - Process ID, N/W Interfaces, Inter-process Communication, File system mount points

DOCKER CONTAINER FORMAT

- Control groups (cgroups) –
  - Share available hardware resources to containers and optionally enforce limits and constraints
  - In short, limits how much a container can use (metering & limiting)
  - Memory, CPU, block I/O, network
- Union file systems (UnionFS) –
  - Operate by creating layers, making them very lightweight and fast
  - Union mounting is a way of combining multiple directories into one that appears to contain their combined contents

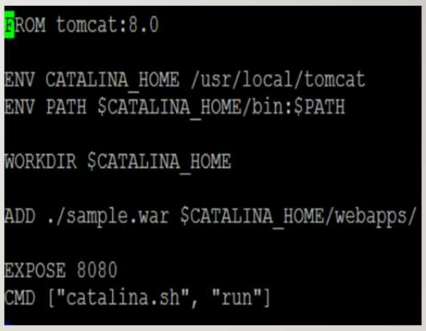
IMAGES AND CONTAINERS

IMAGES

- Read-only template with instructions for running a Docker container
- Images are the 'build' or 'packaging' part of Docker's life cycle
- Can be considered to be the 'source code' for the containers
- Highly portable and can be shared, stored, and updated
- You can create your own images or you can use those created by others and published in a registry
- One image can be based on another image, with some additional customization

USING IMAGES

- Creating images –
  - From Dockerfile providing step by step instructions
  - Using docker commit command
- Distributing images –
  - Using the default Docker Hub registry
  - You can create a public or private registry
  - Save images to archive (tar files) and share

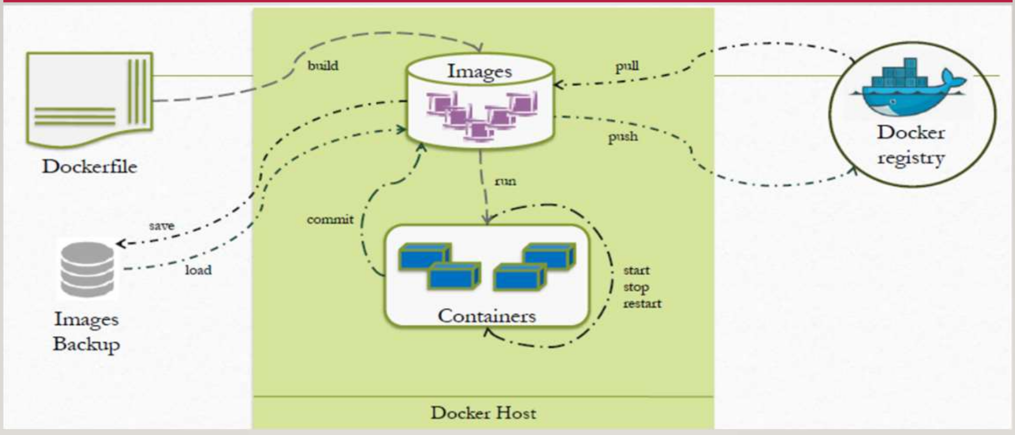


```
FROM tomcat:8.0

ENV CATALINA_HOME /usr/local/tomcat
ENV PATH $CATALINA_HOME/bin:$PATH

WORKDIR $CATALINA_HOME

ADD ./sample.war $CATALINA_HOME/webapps/

EXPOSE 8080
CMD ["catalina.sh", "run"]
```

CONTAINERS

- Containers are the 'running' or 'execution' aspect of Docker's life cycle
- Contains one or more running processesin a self-containedenvironment
- Wraps up an application into its own isolated box
- An application in its container, has no knowledge of any other applications or processes that exist outside of its box
- A running container can't touch the original image itself nor the filesystem of the host
- Changes made in a container are preserved in that container itself and don't affect the original image

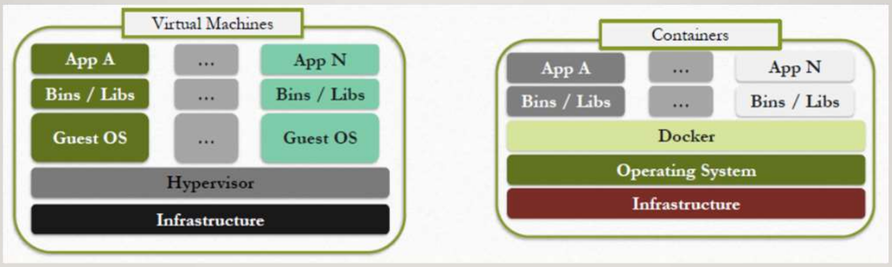
CONTAINER -FLOWS

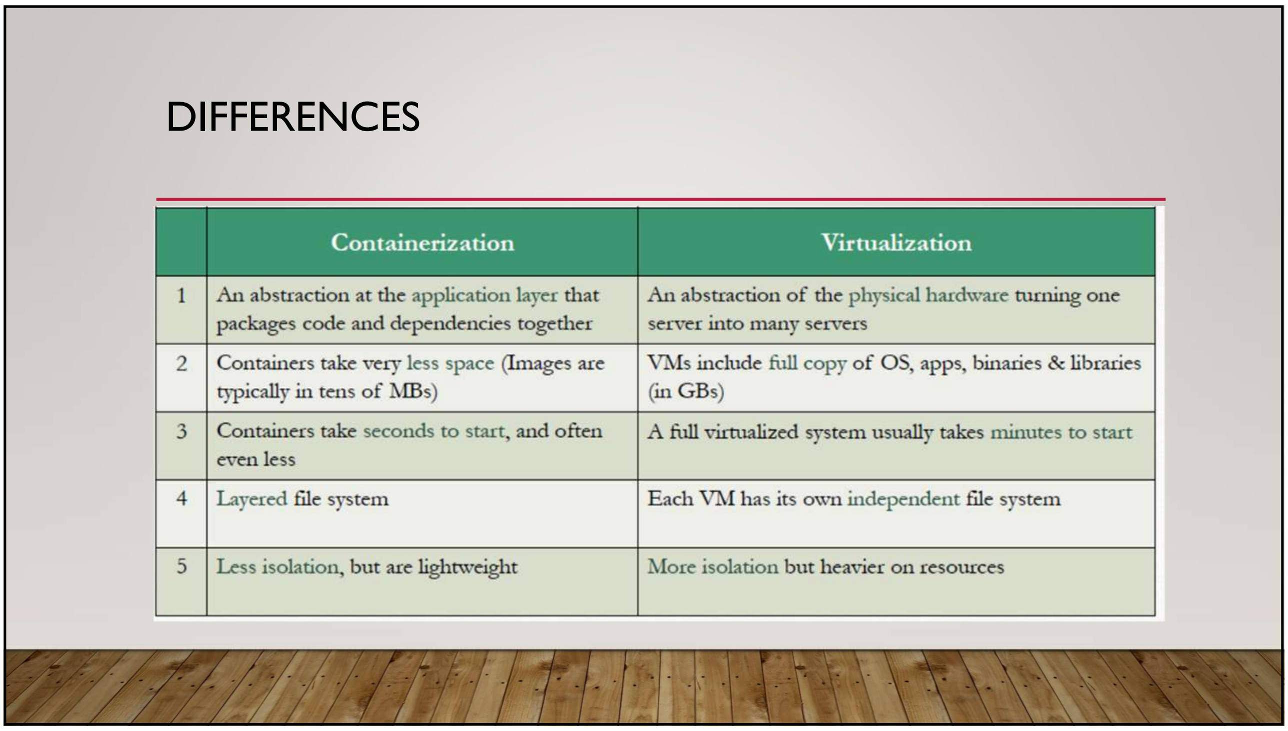


CONTAINERIZATION AND VIRTUALIZATION DIFFERENCES

CONTAINERS ARE NOT VMS

- Docker containers are executed with the Docker engine rather than the hypervisor
- Each VM has a full OS with its own kernel, memory management and virtual device drivers
- Docker Containers are able to share a host kernel and share application libraries

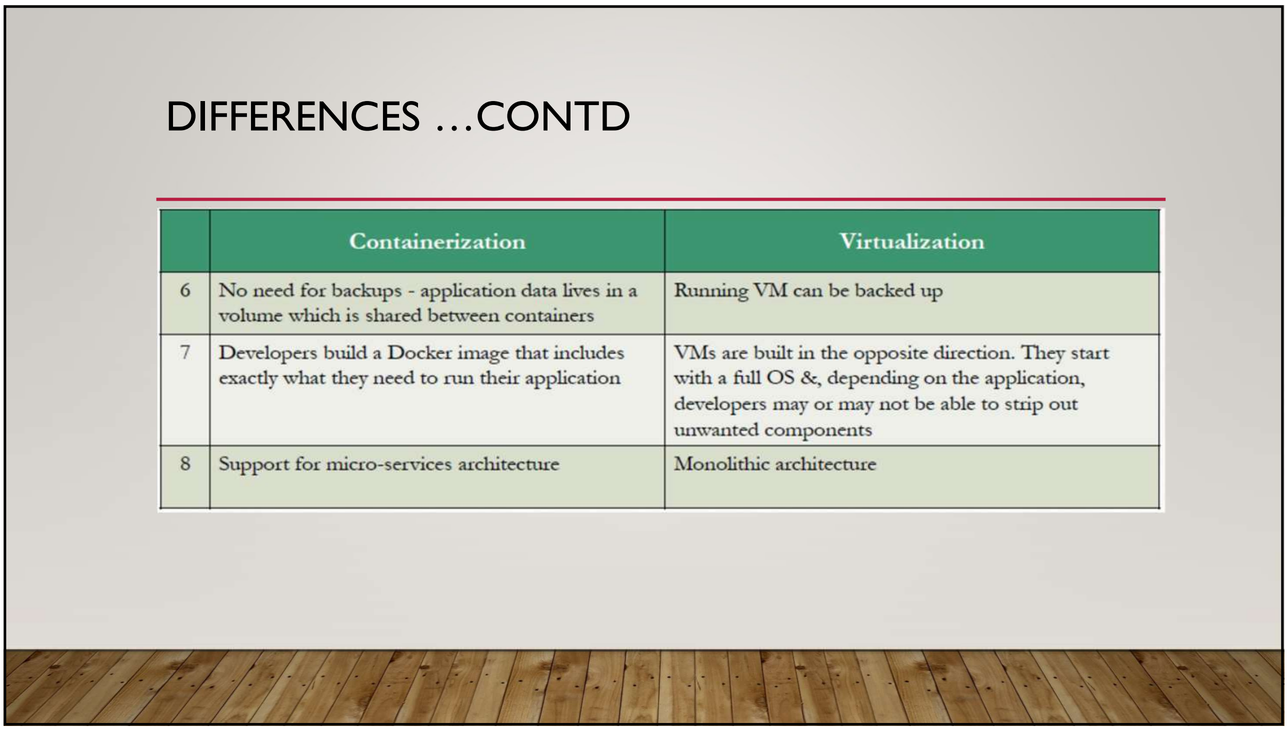




DIFFERENCES

	Containerization	Virtualization
1	An abstraction at the application layer that packages code and dependencies together	An abstraction of the physical hardware turning one server into many servers
2	Containers take very less space (Images are typically in tens of MBs)	VMs include full copy of OS, apps, binaries & libraries (in GBs)
3	Containers take seconds to start, and often even less	A full virtualized system usually takes minutes to start
4	Layered file system	Each VM has its own independent file system
5	Less isolation, but are lightweight	More isolation but heavier on resources

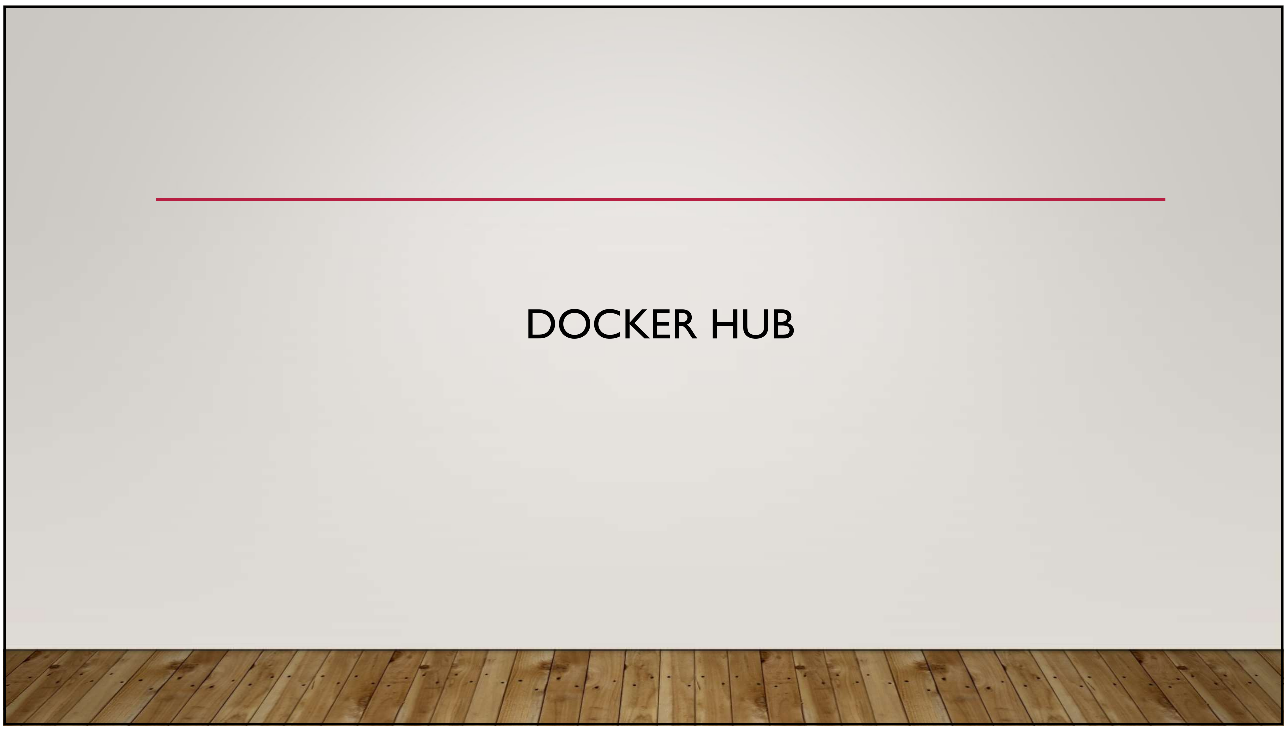
25



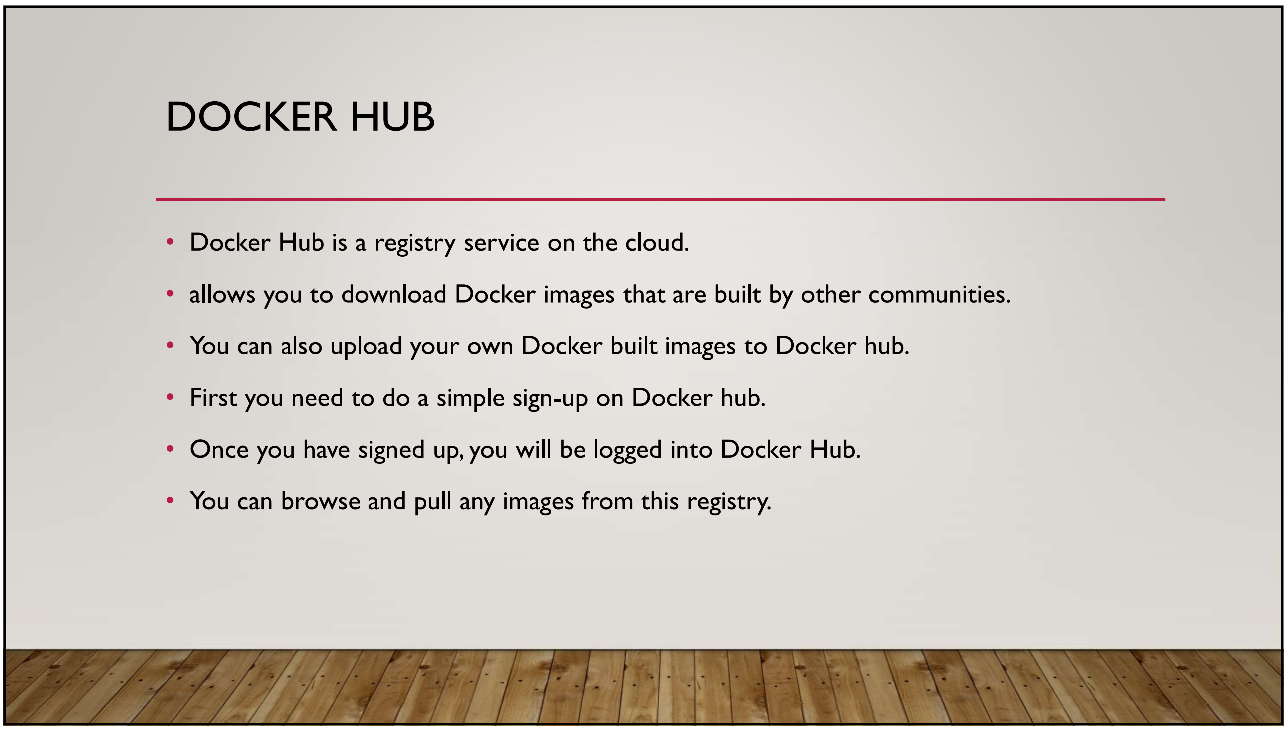
DIFFERENCES …CONTD

	Containerization	Virtualization
6	No need for backups - application data lives in a volume which is shared between containers	Running VM can be backed up
7	Developers build a Docker image that includes exactly what they need to run their application	VMs are built in the opposite direction. They start with a full OS &, depending on the application, developers may or may not be able to strip out unwanted components
8	Support for micro-services architecture	Monolithic architecture

26



DOCKER HUB



DOCKER HUB

- Docker Hub is a registry service on the cloud.
- allows you to download Docker images that are built by other communities.
- You can also upload your own Docker built images to Docker hub.
- First you need to do a simple sign-up on Docker hub.
- Once you have signed up, you will be logged into Docker Hub.
- You can browse and pull any images from this registry.

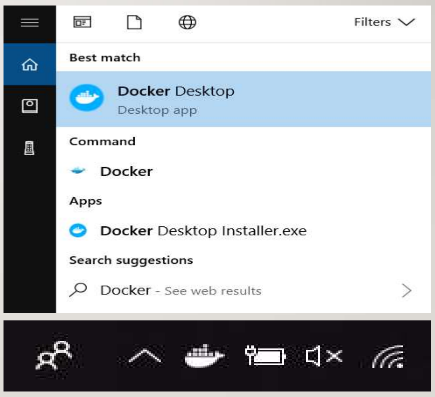
DOCKER ON WINDOWS
DOCKER DESKTOP

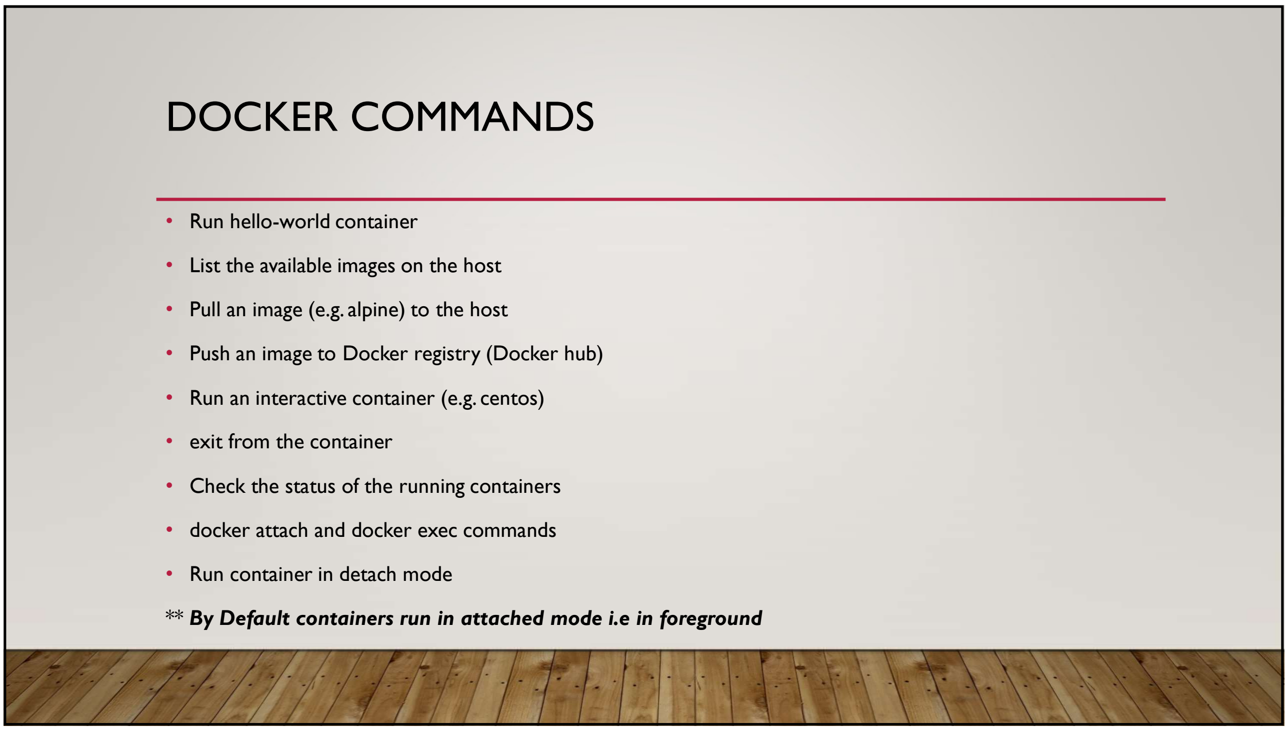
- download Docker for Windows from – https://docs.docker.com/docker-for-windows/
- System Requirements
- Windows 10 64-bit: Pro, Enterprise, or Education (Build 16299 or later).
- Hyper-V and Containers Windows features must be enabled.
- The following hardware prerequisites are required to successfully run Client Hyper-V on Windows 10:
  - 64 bit processor with Second Level Address Translation (SLAT)
  - 4GB system RAM
  - BIOS-level hardware virtualization support must be enabled in the BIOS settings

- Install Docker Desktop on Windows
  - Double-click Docker Desktop Installer.exe to run the installer.
  - If you haven't already downloaded the installer (Docker Desktop Installer.exe), you can get it from Docker Hub. It typically downloads to your Downloads folder, or you can run it from the recent downloads bar at the bottom of your web browser.
  - When prompted, ensure the Enable Hyper-V Windows Features option is selected on the Configuration page.
  - Follow the instructions on the installation wizard to authorize the installer and proceed with the install.
  - When the installation is successful, click Close to complete the installation process.
  - If your admin account is different to your user account, you must add the user to the docker-users group. Run Computer Management as an administrator and navigate to Local Users and Groups > Groups > docker-users. Right-click to add the user to the group. Log out and log back in for the changes to take effect.

- Start Docker Desktop
  - Docker Desktop does not start automatically after installation.
  - To start Docker Desktop, search for Docker, and select **Docker Desktop** in the search results.

  - A steady whale icon in status bar means Docker is up and ready

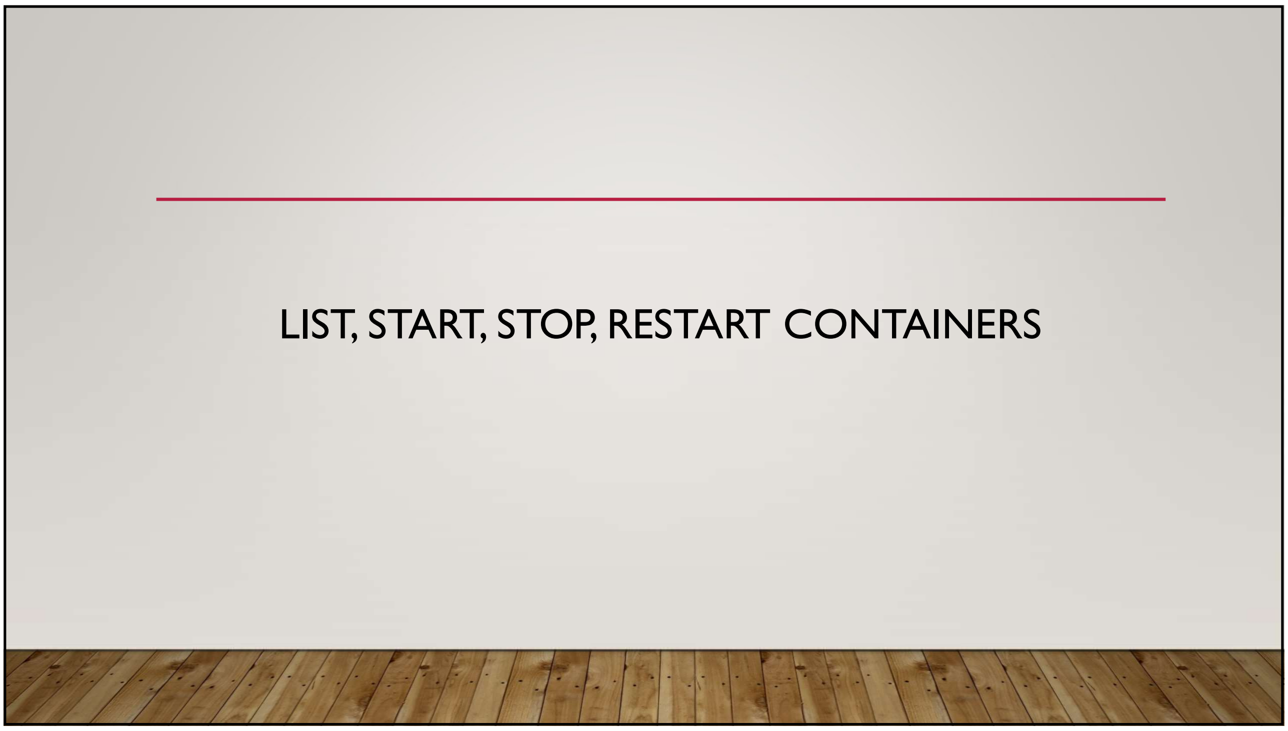




DOCKER COMMANDS

- Run hello-world container
- List the available images on the host
- Pull an image (e.g. alpine) to the host
- Push an image to Docker registry (Docker hub)
- Run an interactive container (e.g. centos)
- exit from the container
- Check the status of the running containers
- docker attach and docker exec commands
- Run container in detach mode

** ***By Default containers run in attached mode i.e in foreground***



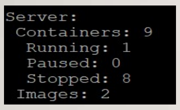
LIST, START, STOP, RESTART CONTAINERS

- List the running containers on the host
  - docker ps
  - docker ps -a
- Stop a running container
  - docker stop <container id> OR <container name>
- Start a stopped container
  - docker start <container id> OR <container name>
- Restart a stopped container
  - docker restart <container id> OR <container name>

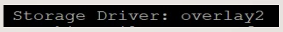
35

INFORMATION ABOUT DOCKER INSTALLATION

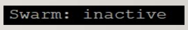
- Run command – docker info



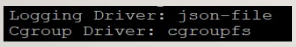
```
Server:
 Containers: 9
  Running: 1
  Paused: 0
  Stopped: 8
 Images: 2
```



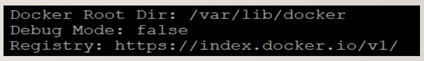
```
Storage Driver: overlay2
```



```
Swarm: inactive
```



```
Logging Driver: json-file
Cgroup Driver: cgroupfs
```



```
Docker Root Dir: /var/lib/docker
Debug Mode: false
Registry: https://index.docker.io/v1/
```

36