135 Frequently Asked Docker Interview Questions and Answers

1. What is Docker?

Answer: Docker is an open-source container service designed to facilitate applications deployment inside the software containers. It will rely on Linux Kernel Features like namespaces and cgroups ensuring which resource isolation and application packaging along with its dependencies. Docker was licensed under Apache License 2.0 in the binary form and fully written in the Go programming language. It can support several operating systems like Linux, Cloud, Windows, and Mac OS and different platforms like ARM architecture and x86-64 windows platforms.

2. Why use Docker?

Answer: 1. A user can quickly build, ship, and run its applications.

2. A single operating system kernel will run all containers.

3. Docker container is more light-weight than the virtual machines.

4. A user will deploy Docker containers anywhere, on any physical and virtual machines and even on the cloud.

3. List the most used commands of Docker.

Answer: 1. ps lists the running containers.

2. dockerd launches Docker Daemon.

3. build is used to build an image from a DockerFile.

4. create is used to create a new image form container's changes.

5. pull is used to download a specific image or a repository.

6. run is used to run a container.

7. logs display the logs of a container.

8. rm removes one or more containers.

9.rmi removes one or more images.

10. stop is used to stop one or more container.

11. kill is used to kill all running containers.

4. Does the data get lost, if the Docker container exits?

Answer: No. any data which application will write to disk can get well preserved in its container until we will explicitly delete the container and the file system will persist even after the container halts.

5. What is Dockerfile and its use?

Answer: DockerFile is a text document which is used to assemble a Docker image. It is consist of a list of Docker commands and operating system commands for building an image. These commands will execute automatically in sequence in the Docker environment and create a new Docker image.

6. How Docker is advantageous over Hypervisors?

Answer: Docker is advantageous in following way

1. It is lightweight.

2. More efficient in terms of resources.

3. It uses very fewer resources and also the underlying host kernel rather than developing its hypervisor.

7. How to create a Docker container?

Answer: A Docker Container will be created by running any specific Docker image.

Use the following command to create a container .

docker run -t -i command name

If to verify that whether the container has created or whether that is running or not, use the following command as this command will lists out all the running Docker containers on the host along with their status.

docker ps -a

8. Can json be used instead of yaml for compose file?

Answer: Yes, json will be used instead of yaml for compose file.

9. How do I run multiple copies of Compose file on the same host?

Answer: Compose will use the project name which will allow to create unique identifiers for all of a project's containers and other resources. To run multiple copies of a project, set a custom project name using the -a command-line option or using COMPOSE_PROJECT_NAME environment variable.

10. What are the components of Docker Architecture?

Answer: Docker Client (docker) – it will enable a user for Docker interaction. It will communicate with more than one Docker Daemon. It can use Docker API and can send command (docker run) to Docker Daemon (dockerd) which carries them out.

Docker Daemon (dockerd) – It will give a complete environment to execute and run applications. It is consists of images, containers, volumes and responsible for all container-related actions. It can pull and create the container images as what the client requests. A Daemon will communicate with other daemons for its service management.

Docker registry -is versioning, storage, and distribution system for Docker images. It allows Docker users to pull images locally, and push new images to the registry. Docker Hub is a public registry instance of Docker used by default while installing Docker Engine.

Docker Image- is a lightweight, standalone, executable package of Docker stored in a Docker Registry. It can be used for creating a container. It will consist of everything required to run an application- code, a runtime, system libraries, system tools, environment variables, configuring files, and settings.

Docker Container– It is a standardized unit of software used for deploying a particular application or environment. It is launched by running an image. It can package up code and all of its dependencies therefore apps will run quickly and reliably from one computing environment to another.

11. On what platforms does Docker run?

Answer: Docker will run on various platforms as follows:

Linux

Ubuntu 12.04, 13.04 et al

Fedora 19/20+

RHEL 6.5+

CentOS 6+

Gentoo

ArchLinux

openSUSE 12.3+

CRUX 3.0+

Microsoft Windows

Windows Server 2016

Windows 10

Cloud

Amazon EC2

Google Compute Engine

Microsoft Azure

Rackspace

12. What is the purpose of Docker_Host?

Answer: It will contain container, images, and Docker daemon. It will offer a complete environment to execute and run application.

13. What is Docker Engine?

Answer: Docker Engine is a Client-Server application which is installed on the host machine. It will allows to develop, assemble, ship, and run applications anywhere. It can be available for Linux or Windows Server. Its major components are as follows:

1. Server is a long-running program which is called a Daemon process (docker)

2. REST API specifies interfaces which docker will use to communicate with Daemon and instruct it what to do.

3. CLI (Command Line Interface) – It will use the Docker REST API to manage and interact with the Daemon through its scripting commands.

14. What is the Lifecycle of Docker container?

Answer: The Lifecycle of Docker Container with CLI is as following:

1. Create a Container.

2. Run the created Container.

3. Pause the processes running inside the Container.

4. Unpause the processes running inside the Container.

5. Start the Container, if exists in a stopped state.

6. Stop the Container as well as the running processes.

7. Restart the Container as well as the running processes.

8. Kill the running Container.

9. Destroy the Container, only if it exists in a stopped state.

15. What is Kubernetes and Docker?

Answer: Docker is a platform and tool to build, distribute, and run Docker containers.

Kubernetes is a container orchestration system for Docker containers which is more extensive than Docker Swarm and will be meant to coordinate clusters of nodes at scale in production in an efficient manner.

16. When should I use Docker? When To Use Docker?

Answer: In following cases should I use Docker:

1. Use Docker as version control system for entire app's operating system.

2. Use Docker when want to distribute/collaborate on app's operating system with a team.

3. Use Docker to run code on laptop in the same environment as we have on your server (try the building tool).

17.What is the advantage of Docker?

Answer: The most important advantages to a Docker-based architecture is actually standardization. Docker will provide repeatable development, build, test, and production environments. Every team member can work on a production parity environment by Standardizing service infrastructure across the entire pipeline.

18. Why is Docker needed?

Answer: The Docker is required to ease the creation, deploy and the delivery of an application using the called Containers. A Docker Container has just the minimum set of operating system not a full operating system, a software required for the application to run and rely on the host Linux Kernel itself.

19. Is Kubernetes better than Docker?

Answer: Kubernetes and Dicker isn't an alternative of each other. Quite the contrary; Kubernetes will run without Docker and Docker will function without Kubernetes. Kubernetes will benefit greatly from Docker and vice versa. Docker is a standalone software which will be installed on any computer to run containerized applications.

20. What is the difference between Docker and Openshift?

Answer: The main difference between Docker and Openshift is that Docker as a project will focuse on the runtime container only, whereas OpenShift as a system will include both the runtime container along the REST API, coordination, and web interfaces for deploying and manage individual containers. A cartridge has similarity to a docker image.

21. What are the disadvantages of Docker?

Answer: Following are disadvantages with Docker:

1. Containers will not run at bare-metal speeds.

2. Containers consume resources more efficiently than virtual machines but still it is subject to performance overhead due to overlay networking, interfacing between containers and the host system and so on.

22. What is the most popular use of Docker?

Answer: The most common technologies running in Docker are:

1. NGINX: Docker is mostly used for deploying and run HTTP servers.

2. Redis: This popular key-value store is a regular feature a top the list of container images.

3. Postgres: The open source relational database is steadily increasing in popularity.

23. Should I use Docker for development?

Answer: The development environment is the similar as the production environment. We will deploy and it can "just work". If it is hard time to build something by build or compile, build it inside Docker.

24. Is Docker a VM?

Answer: In a virtual machine, valuable resources will be emulated for the guest OS and hypervisor, that will make it possible to run many instances of one or more operating systems in parallel on a single machine or host. Docker containers are executed with the Docker engine not with the hypervisor.

25. Why is Docker so popular?

Answer: Docker is popular because it has revolutionized development. Docker, and the containers it will make possible, has revolutionized the software industry and in five short years their popularity as a tool and platform has increased. The main reason is that containers will create vast economies of scale.

26. How do I download Docker?

Answer: Following way Docker can be downloaded:

1. Install Docker for Mac

2. run it

3. Double-click Docker.dmg for openning the installer, then drag Moby the whale to the Applications folder.

4. Double-click Docker.app in the Applications folder to start Docker.

5. Click the whale ( ) to get Preferences and other options.

6. Select About Docker to verify that you have the latest version.

27. Do we need Docker?

Answer: Docker will shine compared to virtual machines when it will come to performance because containers will share the host kernel and will not emulate a full operating system. Docker does impose performance costs. If we required to get the best possible performance out of server, we may required to avoid Docker.

28. What can you do with Docker?

Answer: We are just some of the use cases which will provide a consistent environment at low overhead with the enabling technology of Docker.

1. Simplifying Configuration.

2. Code Pipeline Management.

3. Developer Productivity.

4. App Isolation.

5. Server Consolidation.

6. Debugging Capabilities.

7. Multi-tenancy.

29. What are Docker images?

Answer: A Docker image is a file which is comprised of multiple layers, used for executing code in a Docker container. An image will be essentially built from the instructions for a complete and executable version of an application, which will relies on the host OS kernel.

30. Is Docker a Microservice?

Answer: Docker will Benefits for Microservices. Docker, as a containerization tool, will be often compared to virtual machines. Virtual machines are introduced to optimize the use of computing resources. We will run several VMs on a single server and deploy each application instance on a separate virtual machine.

31. How much does Docker cost?

Answer: If we required to run Docker in production, however, the company will encourage users to sign up for a subscription package for an enterprise version of the platform. Docker will offer three enterprise editions of its software. Pricing which is start at $750 per node per year.

32.What is docker in AWS?

Answer: Docker is a software platform which will allow to build, test, and deploy applications quickly. Running Docker on AWS will provide developers and admins a highly reliable, low-cost way for building, ship, and run distributed applications at any scale.

33. Is Kubernetes free?

Answer: Pure open source Kubernetes is free and it can be downloaded from its repository on GitHub. Administrators should build and deploy the Kubernetes release to a local system or cluster or to a system or cluster in a public cloud, such as AWS, Google Cloud Platform (GCP) or Microsoft Azure.

34. Difference between virtualization and containerization?

Answer: Containers will provide an isolated environment to run the application. The entire user space will be explicitly dedicated to the application. Any changes which are made inside the container will never reflected on the host or even other containers running on the same host. Containers will be an abstraction of the application layer. Each container has a different application.

In virtualization, hypervisors will provide an entire virtual machine to the guest including Kernal. Virtual machines will be an abstraction of the hardware layer. Each VM is a physical machine.

35. Explain how you can clone a Git repository via Jenkins?

Answer: To clone a Git repository via Jenkins, we have to enter the e-mail and user name for Jenkins system. To do that we have to switch into job directory and execute the “git config” command.

36. What is a Docker Container and its advantages?

Answer: Docker containers will include the application and all of its dependencies. It will share the kernel with other containers, running as isolated processes in user space on the host operating system. Docker containers will not required any specific infrastructure, they will run on any infrastructure, and in any cloud. Docker containers will be basically runtime instances of Docker images.

Following are major advantage of using Docker Container:-

1. It will offer an efficient and easy initial set up.

2. It will allow to describe application lifecycle in detail.

3. Simple configuration and interacts with Docker Compose.

4. Documentation will provide every bit of information.

37. Explain Docker Architecture?

Answer: Docker Architecture will consist of a Docker Engine that is a client-server application:-

1. A server which is a type of long-running program which is called a daemon process (the docker command).

2. A REST API which will specify interfaces that programs will use to talk the daemon and instruct it what to do.

3. A command-line interface (CLI) client (the docker command) The CLI will use the Docker REST API to control or interact with Docker daemon applications use the underlying API and CLI.

38. What is Docker Hub?

Answer: Docker hub is a cloud-based registry that will help us to link code repositories. It will allows us to build, test, store image in the Docker cloud. We can deploy the image to host with the help of the Docker hub.

39. What are the important features of Docker?

Answer: Following are the essential features of Docker:

1. Easy Modeling

2. version Control

3. Placement/Affinity

4. Application Agility

5. Developer Productivity

6. Operational Efficiencies

40. What are the main drawbacks of Docker?

Answer: Following are the disadvantages of Docker which we should keep in mind

1. It will not provide a storage option.

2. Offer a poor monitoring option.

3. No automatic rescheduling of inactive Nodes.

4. Complicated automatic horizontal scaling set up.

41. Tell us something about Docker Compose.

Answer: Docker Compose is a YAML file which will contain details about the service, network, and volumes to set up the Docker application. Therefore, we will use Docker compose for creating separate containers, host them and get them to communicate with other containers.

42. What is Docker Swarm?

Answer: Docker Swarm is native clustering for Docker. It will turn a pool of Docker hosts into a single, virtual Docker host. Docker Swarm will serve the standard Docker API, any tool which already communicates with a Docker daemon will be use Swarm to transparently scale to multiple hosts.

43. What is Docker Engine?

Answer: Docker daemon or Docker engine will represent the server. The docker daemon and the clients will run on the same or remote host, which will communicate through command-line client binary and full RESTful API.

44. Explain Registries

Answer: Two types of registry are

1. Public Registry

2. Private Registry

Docker's public registry is called Docker hub, which will allow to store images privately. In Docker hub, we will store millions of images.

45. What command should you run to see all running container in Docker?

Answer: $docker ps

46. Write the command to stop the Docker Container.

Answer: $ sudo docker stop container name

47. What is the command to run the image as a container?

Answer: $ sudo docker run -i -t alpine /bin/bash

48. Explain Docker object labels.

Answer: Docker object labels is a method to apply metadata to docker objects which will include images, containers, volumes, network, swarm nodes, and services.

49. Write a Docker file to create and copy a directory and built it using python modules?

Answer: FROM pyhton:2.7-slim

WORKDIR /app

COPY . /app

docker build –tag

50. Where the docker volumes are stored?

Answer: We required to navigate

/var/lib/docker/volumes

51. How do you run multiple copies of Compose file on the same host?

Answer: Compose will use the project name which will allow us for creating unique identifiers for all of a project's containers and other resources. To run multiple copies of a project, set a custom project name using the -a command-line option or using COMPOSE_PROJECT_NAME environment variable.

52. Did Docker come up with the 'container' technology?

Answer: No, Docker did not come up with the container technology. Multiple other development tools offer containers similar to Docker.

53. How is Docker better than other tools that use containers, then?

Answer: Docker will utilise the cloud to run its container-related operations – which is not used by many other development tools. Docker will become much more flexible and adaptable to different scenarios using the cloud, which might come up during the development or shipment processes. This is the main reason to use the Docker when compared to other container-based developer tools.

54. What is a Dockerfile?

Answer: A Dockerfile is a set of instructions. Developers provided Docker with such instructions therefore the program could do the job correctly, with those specific parameters in mind.

55. What are the three main types of Docker components?

Answer: The Client, the Host, and the Registry.

The client is the component which will issue "run" and "build" commands to the host. The host is where all of the containers and images will be created. They will be then sent to the registry, for execution.

56. Will you lose all of your work if you accidentally exit a container?

Answer: No, We won't lose any information, data and other parameters if we accidentally exit the Docker container. The only way to lose progress would be to issue a specific command to delete the container – exiting it won't do the files within any harm.

57. Can you use any other files for composing instead of the default YAML?

Answer: Yes, The more popular version to use than YAML is the good-old JSON.

58. What is 'NameSpaces' used for?

Answer: NameSpaces will isolate the Docker containers from other activities or tampering with them.

59. What is the single most important requirement for building a Docker container?

Answer: The most important requirement for building a container with Docker is the default image. This default image will vary depending on the code that we are using. To find out and access the default image, we should go to the Docker Hub and search for the specific domain that we required. After we find the image, all that's left to do is deal with the documentation and that's it – we can create a Docker container.

60. How does Docker manage 'Dockerized nodes'?

Answer: A Dockerized node can be any machine which has Docker installed and running. Docker will manage both in-house and cloud-based nodes. Therefore, whether the node will exist in the area of the main computer running Docker or it is present on the cloud – it will not matter. Docker can manage it without a problem.

61. What are the main factors that dictate the number of containers you can run?

Answer: There is no defined limit of containers that we can run with Docker. The limitations may start due to hardware.

Two factors might limit the number of containers that we can run – the size of app and CPU strength. If application isn't ginormous and we have a never-ending supply of CPU power, we could probably run a huge amount of Docker container simultaneously.

62. How is Docker different from Hypervisor?

Answer: Hypervisor will require to have extensive hardware to function properly, while Docker will run on the actual operating system. This will allow Docker to be exceptionally fast and perform tasks in a fluid manner – something which Hypervisor tends to lack.

63.Can I use JSON instead of YAML for my compose file in Docker?

Answer: YES, We can very comfortably use JSON instead of the default YAML for Docker compose file. In order to use JSON file with compose, we required to specify the filename to use as the following:

docker-compose -f docker-compose.json up

64. Tell us how you have used Docker in your past position?

Answer: We could also explain the ease that this technology has brought in the automation of the development to production lifecycle management. We can also discuss about any other integrations that we might have worked along with Docker like Puppet, Chef or even the most popular of all technologies – Jenkins.

65.How to create Docker container?

Answer: We will create a Docker container out of any specific Docker image of choice and the same can be achieved using the command given below:

docker run -t -i command name

The command above can create the container . In order to check whether the Docker container will be created and whether it is running or not, we could make use of the following command. This command can list out all the Docker containers along with its statuses on the host that the Docker container runs.

docker ps –a

66. How to stop and restart the Docker container?

Answer: The following command will be used to stop a certain Docker container with the container id as

CONTAINER_ID:

docker stop CONTAINER_ID

The following command will be used to restart a certain Docker container with the container id as

CONTAINER_ID:

docker restart CONTAINER_ID

67. How far do Docker containers scale?

Answer: The Web deployments such as Google, Twitter and best examples in the Platform Providers such as Heroku, dotCloud run on Docker that can scale from the ranges of hundreds of thousands to

millions of containers running in parallel, provided the condition which the OS and the memory will not run out from the hosts which runs all these innumerable containers hosting your applications.

68. What platforms does Docker run on?

Answer: Docker will currently available on the following platforms and also on the following Vendors or Linux:

1. Ubuntu 12.04, 13.04

2. Fedora 19/20+

3. RHEL 6.5+

4 CentOS 6+

5. Gentoo

6. ArchLinux

7. openSUSE 12.3+

8. CRUX 3.0+

Docker will currently available and run on the following Cloud environment setups as following:

1. Amazon EC2

2. Google Compute Engine

3. Microsoft Azure

4. Rackspace

Docker is extending it will support to Windows and Mac OSX environments and support on Windows has been on the growth in a very drastic manner.

69. What is the advantage of Docker over hypervisors?

Answer: Docker is lightweight and more efficient in resource uses because it will use the underlying host kernel rather than creating its hypervisor.

70. Is Container technology new?

Answer: No, container technology is not new. Different variations of containers technology ate out there in *NIX world for a long time. Solaris container (aka Solaris Zones)-FreeBSD Jails-AIX Workload Partitions (aka WPARs)-Linux OpenVZ are examples.

71. How is Docker different from other container technologies?

Answer: Docker is a quite fresh project. It was created in the Era of Cloud, Therefore a lot of things are better than in other container technologies. Following are the features which are good in Docker:

1. Docker will run on any infrastructure, we can run docker on laptop, or can run it in the cloud.

2. Docker has a Container HUB which is a repository of containers that can download and use. We can even share containers with any applications.

3. 3. Docker is quite well documented.

72. What are the networks that are available by default?

Answer:

bridge It is the default network all containers connect to if we will not specify the network.

none connects to a container-specific network stack which will lack a network interface

host connects to the host's network stack – there will be no isolation between the host machine and the container, as far as network is concerned

73. Difference between Docker Image and container?

Answer: The runtime instance of docker image is the Docker container. Docker Image will not have a state, and its state will never change as it will just set of files whereas docker container will have its execution state.

74. What is the use case for Docker?

Answer: There are use cases where we can use Docker in production.

75. How exactly are containers (Docker in our case) different from hypervisor virtualization (vSphere)? What are the benefits?

Answer: To run an application in a virtualized environment (example for vSphere), we first required for creating a VM, install an OS inside and only then deploy the application. To run the same application in docker, all we required is to deploy that application in Docker. There is no need for additional OS layer. We just deploy the application with its dependent libraries, docker engine (kernel, etc.) which will provide the rest.

Another benefit of Docker is speed of deployment.

ACME inc. will require to virtualize application GOOD APP for testing purposes.

76. Docker is the new craze in virtualization and cloud computing. Why are people so excited about it?

Answer: Docker is fast, easy for using and a developer-centric DevOps-ish tool. it is easy to package and ship code. Developers will want tools which abstract away a lot of the details of that process. They will just want for seeing their code working. That will lead to all sorts of conflicts with Sys Admins when code will be shipped around and turns out not to work somewhere other than the developer's environment. Docker will turn to work around that by making code as portable as possible and making that portability user-friendly and simple.

77. Do you think open source development has heavily influenced cloud technology development?

Answer: I think open source software will be closely tied to cloud computing. Both in terms of the software running in the cloud and the development models which have enabled the cloud. Open source software will be cheap, it's low friction both from an efficiency and a licensing perspective.

78. Can you give us a quick rundown of what we should expect from your Docker presentation at OSCON this year?

Answer: It is aimed at Developers and SysAdmins who will want to get started with Docker in very hands on way. We'll teach the basics of how to use Docker and how to integrate it into daily workflow.

79. How is Docker different from other container technologies?

Answer: Docker containers are easy to deploy in the cloud. It is capable of getting more applications running on the same hardware compared to other technologies such as Kubernete, Amazon Elastic Contain, etc. Therefore making learners who take Kubernetes Training Hyderabad and developers create, ready-to-run containerized applications and make them manage, deploy and share easily.

80. Mention some commonly used Docker Commands?

Answer: Some among the most commonly used Docker Commands will be as follows:

Command	Description
Dockerd	Launch the Docker Daemon

Info	Displays information System-Wide
Version	Displays the Docker Version information
Build	Builds images for Docker files
Inspect	Returns low-level information on an image or container
History	Shows Image History
Commit	Creates new images from Container changes
Attach	Attaches to a running container
Load	Load an image from STDIN or tar archive
Create	Create a new container
Diff	Inspect changes on a container's file system
Kill	Kill a running container

81. What is a Docker Hub?

Answer: Docker Hub can be considered as a cloud registry which lets us link the code repositories, create the images, and test them. We will also store our pushed images, or we will link to the Docker Cloud, therefore that the images will be deployed to the host. We have a centralized container image discovery resource which will be used for the collaboration of our teams, automating the workflow and distribution, and changing management by creating the development pipeline.

Docker Vs VM (Virtual Machine)

Virtual Machines Vs Docker Containers

Virtual Machines Docker Containers

Need more resources Less resources are used

Process isolation is done at hardware level Process Isolation is done at Operating System level

Separate Operating System for each VM Operating System resources can be shared within Docker

VMs can be customized. Custom container setup is easy

Takes time to create Virtual Machine Creation of docker is very quick

Booting takes minutes CBooting is done within seconds

82. Do I lose my data when the Docker container exits?

Answer: There is no loss of data when any of Docker containers will exit as any of the data that application can write to the disk in order to preserve it. This can be done until the container is explicitly deleted. The file system for the Docker container will persist even after the Docker container is halted.

What, in your opinion, is the most exciting potential use for Docker?

Answer: The most exciting potential use of Docker is its build pipeline. Most of the Docker professionals will be seen using hyper-scaling with containers, and indeed get a lot of containers on the host which it can run on. These will be known to be blatantly fast. Most of the development – test build pipeline will be completely automated using the Docker framework.

84. Why is Docker the new craze in virtualization and cloud computing?

Answer: Docker will be the newest and the latest craze in the world of Virtualization and also Cloud computing because it will be an ultra-lightweight containerization app which is brimming with potential to prove its mettle.

85. Why do my services take 10 seconds to recreate or stop?

Answer: Docker compose stop attempts to stop a specific Docker container by sending a SIGTERM message. Once this message will be delivered, it CaaS wait for the default timeout period of 10 seconds and once the timeout period will be crossed, it will then send out a SIGKILL message for the container – in order to kill it forcefully. If we are actually waiting for the timeout period, then it will mean that the containers will not shutting down on receiving SIGTERM signals / messages.

In an attempt to solve this issue, the following is what we will do:

1. We can ensure that we are using the JSON form of the CMD and also the ENTRYPOINT in dockerfile.

2. Use [“program”, “argument1”, “argument2”] instead of sending it as a plain string as like this – “program argument1 argument2”.

3. Using the string form, which will make Docker run the process using bash which will not handle signals properly. Compose always uses the JSON form.

4. If it is possible then modify the application which we are intended to run by adding an explicit signal handler for the SIGTERM signal

5. Also set the stop_signal to a proper signal which the application will understand and also know how to handle it.

86. How do I run multiple copies of a Compose file on the same host?

Answer: Docker's compose will makes use of the Project name for creating unique identifiers for all of the project's containers and resources. In order to run multiple copies of the same project, we will required to set a custom project name using the –p command line option or we could use the COMPOSE_PROJECT_NAME environment variable for this purpose.

87. What's the difference between up, run, and start?

Answer: On any given scenario, we will always want docker-compose up. Using the command UP, we will start or restart all the services which are defined in a docker-compose.yml file. In the "attached" mode, that will also the default mode – we will be able to see all the log files from all the containers. In the "detached" mode, it can exit after starting all the containers that will continue to run in the background showing nothing over in the foreground.

Using docker-compose run command; we can be able to run the one-off or the ad-hoc tasks which will be required to be run as per the Business needs and requirements. This will require the service name to be provided which we would want to run and based on that, it can only start those containers for the services which the running service will depend on. Using the run command, we can run tests or perform any of the administrative tasks as like removing / adding data to the data volume container. It will also very similar to the docker run –ti command that opens up an interactive terminal to the containers an exit status which will matche with the exit status of the process in the container.

Using the docker-compose start command; we will only restart the containers which were previously created and were stopped. This command will never create any new Docker containers on its own.

88. What's the benefit of "Dockerizing?"

Answer: Dockerizing enterprise environments will help teams to leverage over the Docker containers to form a service platform such as a CaaS (Container as a Service). It will provide teams that necessary agility, portability and also lets them control staying within their own network / environment.

Most of the developers opt for using Docker and Docker alone because of the flexibility and also the ability that it will provide to quickly build and ship applications to the rest of the world. Docker containers will be portable and these will run on any environment without making any additional changes when the application developers will have to move between Developer, Staging and Production environments. This whole process will be seamlessly implemented without the need of performing any recoding activities for any of the environments. These not only will help to reduce the time between these lifecycle states, but also will ensure that the whole process can be performed with utmost efficiency. There is every possibility for the Developers for debugging any certain issue, fix it and will also update the application with it and propagate this fix to the higher environments with utmost ease.

The operations teams will handle the security of the environments while also allow the developers build and ship the applications in an independent manner. The CaaS platform which is provided by Docker framework will deploy on-premise and can also loaded with full of enterprise level security features like role-based access control, integration with LDAP or any Active Directory, image signing and etc.

Operations teams will have heavily rely on the scalability provided by Docker and will also leverage over the Dockerized applications across any environments.

Docker containers will be so portable that it will allow teams to migrate workloads which will run on an Amazon's AWS environment to Microsoft Azure without change its code and also with no downtime at all. Docker will allow teams to migrate these workloads from their cloud environments to their physical datacenters and vice versa. This also will enable the Organizations to focus on the infrastructure from the gained advantages both monetarily and also the self-reliability over Docker. The lightweight nature of Docker containers are compared to traditional tools such as virtualization, combined with the ability for Docker containers to run within VMs, allowing teams for optimizing their infrastructure by 20X, and save money in the process.

89. How many containers can run per host?

Answer: Depending on the environment where Docker hosts the containers, there will be as many containers as the environment will support. The application size, available resources such as CPU, memory will decide on the number of containers which will run on an environment. Though containers will create newer CPU on their own but they will definitely provide efficient ways of utilizing the resources. The containers themselves are super lightweight and will only last as long as the process they are running.

90. Is there a possibility to include specific code with COPY/ADD or a volume?

Answer: This will be easily achieved by adding either the COPY or the ADD directives in dockerfile. This can count to be useful if we want to move code along with any of Docker images, example, sending code an environment up the ladder – Development environment to the Staging environment or from the Staging environment to the Production environment. Having said that, we might come across situations where we will required to use both the approaches. We will have the image include the code using a COPY, and use a volume in Compose file to include the code from the host during development. The volume will override the directory contents of the image.

91. Will cloud automation overtake containerization any sooner?

Answer: Docker containers will gain the popularity each passing day and definitely can be a quintessential part of any professional Continuous Integration / Continuous Development pipelines. There is equal responsibility on all the key stakeholders at each Organization for taking up the challenge of weighing the risks and gains on adopting technologies which are budding up on a daily basis. Docker is extremely effective in Organizations that appreciate the consequences of Containerization.

92. Is there a way to identify the status of a Docker container?

Answer: We will identify the status of a Docker container by running the command 'docker ps –a', that will in turn list down all the available docker containers with its corresponding statuses on the host. From there we will easily identify the container of interest to check its status correspondingly.

93. What are the differences between the 'docker run' and the 'docker create'?

Answer: The most important difference is that by using the 'docker create' command we will create a Docker container in the Stopped state. We will also give it with an ID which can be stored for later usages as well.

This will be achieved by using the command 'docker run' with the option –cidfile FILE_NAME as like this:

'docker run –cidfile FILE_NAME'

94. Can you remove a paused container from Docker?

Answer: It is not possible for removing a container from Docker which is just paused. It is a must which a container will be in the stopped state, before it will be removed from the Docker container.

95. Is there a possibility that a container can restart all by itself in Docker?

Answer: No, it is not possible. The default –restart flag is set for never restart on its

96. What is the preferred way of removing containers – 'docker rm -f' or 'docker stop' then followed by a 'docker rm'?

Answer: The best and the preferred way for removing containers from Docker is to use the 'docker stop', as it will allow sending a SIG_HUP signal to its recipients providing them the time which is required for performing all the finalization and cleanup tasks. Once this activity can be completed, we will then comfortably remove the container using the 'docker rm' command from Docker and thereby updating the docker registry as well.

97. Difference between Docker Image and container?

Answer: Docker container is the runtime instance of docker image.

Docker Image can not have a state and its state never changes because it will be just set of files whereas docker container will have its execution state.

98. What are the main drawbacks of Docker?

Answer: Following are drawbacks of Docker :

1. Not provide a storage option

2. Provides a poor monitoring option.

3. No automatic rescheduling of inactive Nodes

4. Complicated automatic horizontal scaling set up

99. What is Docker Engine?

Answer: Docker daemon or Docker engine will represent the server. The docker daemon and the clients will run on the same or remote host, that will communicate through command-line client binary and full RESTful API.

100. Explain Registries.

Answer: There are following types of registry :

1. Public Registry

2. Private Registry

Docker's public registry is called Docker hub, which can allow to store images privately. In Docker hub, we can store millions of images.

101. What command should you run to see all running container in Docker?

Answer: $ docker ps

102. Write the command to stop the docker container.

Answer: $ sudo docker stop container name

103. What is the command to run the image as a container?

Answer: $ sudo docker run -i -t alpine /bin/bash

104. What are the common instruction in Dockerfile?

Answer: The common instruction in Dockerfile are FROM, LABEL, RUN, and CMD.

105. What is memory-swap flag?

Answer: Memory-swap is a modified flag which has meaning if- memory is set. Swap will allow the container to write express memory requirements on disk when the container has exhausted all the RAM which is available to it.

106. Explain Docker Swarm?

Answer: Docker Swarm is native gathering for docker which can help to a group of Docker hosts into a single and virtual docker host. It will offer the standard docker application program interface.

107. How can you monitor the docker in production environments?

Answer: Docker states and Docker Events can be used for monitoring docker in the production environment.

108. What the states of Docker container?

Answer: Following are Important states of Docker container:

1. Running

2. Paused

3. Restarting

4. Exited

109. What is Virtualization?

Answer:Virtualization is a method to logically divide mainframes to allow multiple applications for running simultaneously.

This scenario will changed when companies and open source communities can be able to offer a method of handling privileged instructions. It will allow multiple OS to run simultaneously on a single x86 based system

110. What is Hypervisor?Answer: The hypervisor will allow to create a virtual environment in that the guest virtual machines can operate. It will control the guest systems and checks if the resources are allocated to the guests as per requirement.

111. Explain Docker object labels.

Answer: Docker object labels is a method to apply metadata to docker objects including, images, containers, volumes, network, swam nodes, and services.

112. Write a Docker file to create and copy a directory and built it using python modules?

Answer: FROM pyhton:2.7-slim

WORKDIR /app

COPY . /app

docker build –tag

113. Where the docker volumes are stored?

Answer: /var/lib/docker/volumes

114. List out some important advanced docker commands

Answer:

Command Description

docker info Information Command

docker pull Download an image

docker stats Container information

Docker images List of images downloaded

115. How does communication happen between Docker client and Docker Daemon?

Answer: We will communicate between Docker client and Docker Daemon with the combination of Rest API, socket.IO, and TCP.

116. Explain Implementation method of Continuous Integration(CI) and Continues Development (CD) in Docker?

Answer: We required to do the following things:

1. Runs Jenkins on docker

2. We will run integration tests in Jenkins using docker-compose

117. What are the command to control Docker with System?

Answer: systemctl start/stop docker

service docker start/stop.

118. How to use JSON instead of YAML compose file?

Answer: docker-compose -f docker-compose.json up

119. What is the command you need to give to push the new image to Docker registry?

Answer: docker push myorg/img

120. How to include code with copy/add or volumes?

Answer: In docker file, we required to use COPY or ADD directive. This is useful to relocate code. However, we should use a volume if we want to make changes.

121. Explain the process of scaling your Docker containers.

Answer: The Docker containers can be scaled to any level starting from a few hundred to even thousands or millions of containers. The only condition for this is that the containers required the memory and the OS at all times, and there should not be a constraint when the Docker is getting scaled.

122. What are the steps for the Docker container life cycle?

Answer: Following are the steps for Docker life cycle:

1. Build

2. Pull

3. Run

123. How can you run multiple containers using a single service?

Answer: By using docker-compose, we can run multiple containers using a single service. All docker-compose files can use yaml language.

124. What is CNM?

Answer: CNM will stand for Container Networking Model. It will form the basis of container networking in a Docker environment. This docker's approach will provide container networking with support for multiple network drivers.

125. Does Docker offer support for IPV6?

Answer: Yes, Docker will provide support IPv6. IPv6 networking is supported only on Docker daemons which will run on Linux hosts. However, if we want to enable IPv6 support in the Docker daemon, we required to modify /etc/docker/daemon.json and set the ipv6 key to true.

126. What are a different kind of volume mount types available in Docker?

Answer: Bind mounts- It will be stored anywhere on the host system

127. How to configure the default logging driver under Docker?

Answer: We required to set the value of log-driver to the name of the logging drive the daemon.jason.fie to configure the Docker daemon to default for a specific logging driver.

128. Explain Docker Trusted Registry?

Answer: Docker Trusted Registry is the enterprise-grade image storage toll for Docker. We should install it after firewall so that we can securely manage the Docker images we will use in the applications.

129. What are Docker Namespaces?

Answer: The Namespace in Docker is a technique that will offer isolated workspaces called the Container. Namespaces can also offer a layer of isolation for the Docker containers.

130. What are the three components of Docker Architecture?

Answer: Following are three components of Docker Architecture:

1. Client

2. Docker-Host

3. Registry

131. What is client?

Answer:Docker will provide Command Line Interface tools to the client to interact with Docker daemon.

132. What is the method for creating a Docker container?

Answer:We will use any of the specific Docker images to create a Docker container using the below command.

docker run -t -i command name

This command not will create the container but also start it.

133. What is the best place to find decent examples of 'compose files'?

Answer:Most of the high-key companies which require Docker experts use a specific tool to manage their internal workings. That tool is called GitHub.

Other than all of the main functions which it will perform, it is also a great place to find the before-mentioned compose files for Docker containers.

134. How do you think Docker will change virtualization and cloud environments?

Answer:There are a lot of workloads which Docker is ideal for. Both in the hyper-scale world of many containers and in the dev-test-build use case. I will fully expect a lot of companies and vendors to embrace Docker as an alternative form of virtualization on both bare metal and in the cloud.

135. What are the various states that a Docker container can be in at any given point in time?

Answer:There will be four states which a Docker container will be in, at any provided point in time. Those states will be as given as follows:

- Running
- Paused
- Restarting
- Exited