**INTERVIEW QUESTIONS**

**GIT:**

1. **What is git?**

Git is nothing but Global Information Tracker.

Git is also called as version control system and source code management.

It is used to track the files.

It will maintain the multiple versions of the same file.

Git is free and open source.

1. **What are git stages?**

There are 3 stages in git which is

1. Working directory
2. Staging areas
3. Repository

Working directory: in this stage we can create a files for our project.

Staging area: it is nothing but a rough draft space

We can modified file in its current version to go into your next commit snapshot.

We can track the files in staging area.

Repository: it is nothing but out project folder, when we commit the file it will store on repository.

the data is safely stored in your local database.

We have 3 types of repositories

1. Local
2. Remote
3. Central
4. **How to track a file in git?**

To track a file, we use a command “git add filename”

Once we add, it will gets started tracking

So that we can know the every modification on the file

1. **How to commit the file?**

To commit a file we use a command called “git commit -m “message” file\_name

Once we commit the file it will safely stored in local database.

1. **What is git status?**

Git status is used to know the tracking and untacking files in our server.

1. **What is git logs?**

Git log is a history, we can see the list of every commit in our project along some other information like who commit the file and when we commit the file.

1. **How to configure in git**

Git config user.name “name”

Git confit user.email “name@gmail.com”

1. **What is git merge?**

It allows us to get the code from one branch to another. This is useful when developers work on the same code and want to integrate their changes before pushing them up in a branch.

Command: git merge branch\_name

1. What is merge conflicts?

Merge conflicts happen when you merge branches that have competing commits, and Git needs your help to decide which changes to incorporate in the final merge.

1. **What is git diff?**

It is used to show all the differences between any two commits or files within your Git repository.

1. **What is git push?**

Git push is used to share the local files into central repositories like GitHub and Bitbucket.

1. **What is git stash?**

It is used to save your changes but not record them in the Git repository.

1. **What is git clone?**

Git clone is used to get the repository from central to local.

1. **What is Git checkout?**

Git checkout is used to switch the branches

Command: git checkout branch\_name

1. W**hat is git rm?**

Git rm is used to untrack the files

1. **Difference b/w git and GitHub?**

| Git is a software | GitHub is a service |
| --- | --- |

| [Git can be installed](https://www.simplilearn.com/tutorials/git-tutorial/git-installation-on-windows) locally on the system | GitHub is hosted on the web |
| --- | --- |
| Git has 3 types of repositories | GitHub has 2 types of repositories |
| Git has a default branch called master | GitHub has a default branch called main |

1. **What is git pull?**

Git pull is the combination of git fetch and git merger

Git pull is used to get the commits from central repo to local repo

Command: git pull origin branch

1. **Git fetch vs pull**

| Git Fetch | Git Pull |
| --- | --- |
| The Git fetch command only downloads new data from a remote repository. | Git pull updates the current HEAD branch with the latest changes from the remote server. |
| It does not integrate any of these new data into your working files. | Downloads new data and integrate it with the current working files. |
| Command - git fetch origin  git fetch --all | Tries to merge remote changes with your local ones.  Command - git pull origin master |

1. How to resolve the git conflicts?

* Open the file in vi editor and remove the conflict messages.
* Save the file
* Add the files using the git add command.
* The last step is to commit the changes in the file.

1. **What is git tag?**

Tagging is generally used to capture a point in history that is used for a marked version release.

1. **How to change the commit message**

git commit —amend commit\_id

1. **What is git fork?**

Git fork is used to get the repositories from another account

1. **What is cherry-pick?**

Cherry-pick is used to get a particular file from another branch

command: git cherry-pick commit\_id

1. **What is git branch?**

A branch is a way to isolate development work on a particular aspect of a project.

1. **What is git blame?**

It is used to see the history of the GitHub file.

==========================================================================================

Git reset commit\_id : deletes the commits

git log --graph --oneline --all : see all the history in graph

git merge --abort : cancel the conflicts in commits

git remote show origin : used to see the branches in GitHub

git log origin/main : history of the GitHub

git push --delete origin mustafa : delete the branch on GitHub

Git branch -m old new : to change the branch name

Extra commands:

git show <commit> --stat : you’ll see the commit summary along with the files that changed and details on how they changed.

git commit --amend -m “New commit message” : to edit the commit message

git commit --amend --no-edit : used to add some files in previous commit. (--no-edit means that the commit message does not change.)

git update-ref -d HEAD : used to delete 1st commit in the git

git reset commit: used to delete all the commits (upto the commit id)

Let’s say that you forgot to configure the email and already did your first commit. Amend can be used to change the author of your previous commit as well. The author of the commit can be changed using the following command:

git commit --amend --author "Author Name <Author Email>"

**git merge vs git rebase**

When there are changes on the main branch that you want to incorporate into your branch, you can either *merge* the changes in or *rebase* your branch from a different point.

merge takes the changes from one branch and merges them into another branch in one *merge commit*.

rebase adjusts the point at which a branch actually branched off (i.e. moves the branch to a new starting point from the base branch).

*Generally,* you’ll use rebase when there are changes are made in main/master branch that you want to include in your branch. You’ll use merge when there are changes in a branch that you want to put back into main.

==========================================

**JENKINS:**

1) **What is Jenkins and why should we use it?**

Jenkins is a leading open-source, free automation tool that is used to develop and test software projects.

Reasons why Jenkins is widely used:

-Used to detect faults in software development and systematize the testing of buildss

-Used to constantly monitor the code simultaneously and add changes to the build.

-Jenkins consists of an installer package for most operating systems.

-Used to keep the team updated and synchronized about the changes incorporated.

-Used to build CI/CD pipelines since it has plugin capabilities and is easy to use.

**2) What are the features of Jenkins?**

Jenkins features include:

-It is an open-source automation tool and it is free.

-Gives pipeline support

-installation is easy on systems with multiple operations.

-large number of plugins

-Jenkins upgrades effortlessly.

-speedy release cycle

-Configuration setup is easy.

**3) What are the prerequisites to use Jenkins?**

The requirements to use Jenkins are:

-First requirement is an accessible source code repository, for instance, a Git repository.

-A build script in working condition, example- Maven script checked into the repository.

**4) How do we manually restart Jenkins?**

The following commands are used to restart Jenkins manually:

-(jenkins\_url)/safe restart — Wait until all the builds are completed before restarting.

-(jenkins\_url)/restart-To force restart without waiting for build completion.

**5) What does “continuous integration” mean?**

Continuous Integration is the continuous process of checking the code made by developers into a version control system numerous times. The build is automated in the process to inspect and detect bugs in the developed code. Continuous integration comprises of:

**6) what is Jenkins Pipeline?**

Jenkins Pipeline is collection of features of Jenkins. They are installed as plugins that allows delivery of pipeline implementation continuously.

**7) What are the advantages of using Jenkins?**

The advantages of using Jenkins are:

-User-friendly, free, and it is an open source

-Trouble-free Installation

-Code deployment is convenient and takes very less time. It simultaneously generates reports.

-Helps in collaboration between the operation and development teams.

-Free of cost

-Detection of code errors at an early stage

-Reduced integration issues due to automation of integration work

-Rich plugin ecosystem

-Platform independence

**8) What are the components that you can integrate Jenkins with?**

Jenkins is mainly integrated with the following components:

-First is the version control system, for example: GIT, SVN

-Another one is build tools. An example is Apache Maven.

**9) How does Hudson relate to Jenkins?**

Jenkins was previously known as Hudson. It is a continuous integration tool and is open source, written in Java.

10) Explain how to set up Build jobs in Jenkins.

The following steps will help you to build jobs in Jenkins:

Step 1 -First, open the Jenkins dashboard and click on the New Item.

Step 2 -Enter the item name and choose the “Freestyle project option”.

Step 3 -Specify the details of the job.

Step 4 -Next, specify the location of files that need to be built.

Step 5 -In case the repository is hosted on GitHub, enter the URL of that repository here.

Step 6 -Build section and click on the Add build step.

Step 7 -In the command window, enter the following commands and then click on the Save button.

(Javac HelloWorld.java

Java HelloWorld)

Step 8 -Now click on the “Build” option to see whether the job is defined well or not.

Step 9 -Once the build is scheduled, it will run.

Step 10 -To see details of the build, click on the Console Output link.

===================================================

**MAVEN:**

**Q1) what is Maven?**

Apache maven is a software project management and comprehension tool. based on the concept of a project object model (POM)

maven can manage a project's build. Reporting and documentation from a central piece of information

**Q2) what are the main features of Maven?**

•simple and fast

•easy to learn

•dependency management

•multiple projects

•large library

•extensible

•instant

**Q3) what are the functions of a build tool?**

•generate

•Documentation

•compile

•package

•install

**Q4) why should one use Maven?**

it helps to setup project very quickly and it avoids complicated build files like build.xml

**Q5) what is POM?**

POM stands for project object model. it is fundamental unit of work in maven

**Q6) what information does POM contain?**

•project dependencies

•plugins

•goals

•build profiles

•project version

•developers

•mailing list

**Q7) what is maven build lifecycle?**

A Build Lifecyle is a well-defined sequence of phases.it defines the order in which the goals are to be executed. Each build phase consists of a sequence of goals

**Q8) what would the command mvn clean do?**

this command removes the target directory with all the build data before starting the build process

**Q9) what phases does a clean lifecycle consist?**

•pre-clean

clean

•post-clean

**Q10) what are the types of maven repositories?**

Maven repository are of three types:

•local

•central

•remote

**Q11) what is Maven Repository?**

A repository is palace i.e. directory where all the project jars. libraryjar, plugins

**Q12) what is the default location for your local repository?**

~/m2./repository

**=======================================================**

**ANSIBLE:**

**Ansible interview questions?**

**1) What Is Ansible?**

Ansible is a configuration management system. It is used to set up and manage infrastructure and applications. It allows users to deploy and update applications using SSH, without needing to install an agent on a remote system.

**2) What’s the use of Ansible?**

Ansible is used for managing IT infrastructure and deploy software apps to remote nodes.

For example, Ansible allows you to deploy as an application to many nodes with one single command. However, for that, there is a need for some programming knowledge to understand the ansible scripts.

**3) What is Ansible Galaxy?**

Ansible can communicate with configured clients from the command line by using ansible command. It also allows you to automate configuration by using ansible-playbook command. To create the base directory structure, you can use a tool bundled with Ansible which is known as ansible-galaxy.

**Command:**

$ ansible-galaxy init azavea. packer

azavea.packer was created successfully

**4) What is Continuous Delivery?**

Continuous delivery is a practice of delivering the software as soon as it developed. In this method, we need to use versioning control system. The software is constantly updated in live production systems.

**5) What is the way to access shell environment variables in Ansible?**

In Ansible, if you want to access existing variables, the user needs to use the ‘env’ lookup plugin. Example, to access the value of the Office environment on the management machine:

You need to write following code:

---

# ...

vars:

local\_home: "{{ lookup('env','Office') }}"

I

{{ ansible\_env.SOME\_VARIABLE }}

**6) What is the code you need to write for accessing a variable name?**

Variable names can be built by adding using the following method:

{{ hostvars[inventory\_hostname]['ansible\_' + which\_interface]['ipv4']['address'] }}

The method of using hostvars is important because it’s a dictionary of the entire namespace of variables. ‘inventory\_hostname’ variable specifies the current host you are looking over in the host loop.

**7) Explain how you can disable cowsay?**

If cowsay is installed then executing playbooks inside the Ansible you can disable coway by using following options:

1. Uninstall cowsay
2. Setting up value for the environment variable

export ANSIBLE\_NOCOWS=1

**8) Explain how you can copy file recursively onto a target host?**

The “copy” module has a recursive parameter. However, if you want this to perform more efficient for a large number of files, then “synchronize” module is the best option for you.

**9) How Can you submit a change to the Documentation in Ansible?**

Documentation for Ansible is kept in the project git repository. It contains complete instructions for contributing can be found in the docs.

**10) What Is the Best Method to Make Content Reusable/redistributable?**

You can read everything about “Roles” in the playbooks documentation section. This helps to make playbook content self-contained and shareable with other ansible users.

**11) What is Ansible Tower?**

Ansible tower is a tool which makes Ansible very easy to use. It acts as a hub for the task automation. The tower is free for usage till 10 nodes.

**12) What’s the method to check the inventory vars defined for the host?**

**For that use this command:**

ansible -m debug -a "var=hostvars['hostname']" localhost

| **Variable Name** | **Environment Variables** |
| --- | --- |
| It can be built by adding strings. | To access the environment variable, you need to access existing variables. |
| {{ hostvars[inventory\_hostname][‘ansible\_’ + which\_interface][‘ipv4’][‘address’] }} | # … vars: local\_home: “{{ lookup(‘env’,’HOME’) }}” |
| Allows to add strings | To set environment variables, we need to see the advanced playbooks section. |
| Ipv4 address type use for Variable names we use the ipv4 address. | For Remote environment variables, use {{ ansible\_env.SOME\_VARIABLE }} |

**13) State the difference between Variable name and Environment Variables.**

**14) What are ad-hoc commands?**

You can think of ad-hoc commands as a way for us to take actions on our hosts without writing a playbook. For example, if we want to reboot all hosts in a particular group(webservers). Then you can write a playbook or simply run a one-off ad-hoc command.

**15) Explain Ansible facts?**

You can think of ansible facts as a way for ansible to get information about a host and store them in variables for easy access. This information stored in predefined variables are available to use in the playbook. To generate facts, ansible runs the setup module.

**16) How do you see all variables for a host?**

You can see them using the hostvars variable. This stores host variables with the hostname as key. For example, to look at the variables defined for localhost, you can run;

ansible -m debug -a "var=hostvars[inventory\_hostname]"

**17) Explain modules in ansible**

Modules in Ansible are idempotent. From a RESTful service standpoint, for the operation to be idempotent, clients can perform the same result by using modules in Ansible. Multiple identical requests become a single request.

There are two different types of modules in Ansible:

* **Core modules**
* **Extras modules**

**Core Modules**

The Ansible team maintains these types of modules, and they will always ship with Ansible software. They will also give higher priority for all requests than those in the “extras” repos.

**Extras Modules:**

These modules currently is bundled with Ansible but might available separately in the future. They are also mostly maintained by the Ansible community. These modules are still usable, but it can receive a lower rate of response to issues and pull requests.

**18) When should you test playbooks and roles?**

In ansible, Tests can be added either in new Playbooks or to existing Playbooks. Therefore, most of the testing job offers a clean hosting each time. By using this testing methodology, you need to make very little to no code changes.

**19) Discuss method to Create an Empty File with Ansible**

**To create and empty file you need to follow given steps.**

Step 1. Save An Empty File into The Files Directory

Step 2. Copy It to The Remote Host.

You can think of ad-hoc commands as a way for us to take actions on our hosts without writing a playbook. For example, if we want to reboot all hosts in a particular group(webservers). Then you can write a playbook or simply run a one-off ad-hoc command.

**==========================================================**

**DOCKER:**

**WHY DOCKER**: docker ensures that your application will be same on your machine and any other machine in the world.

**DOCKER DAEMON**: it receives the commands from the clients and manages the objects like images, containers, networks and volumes.

**CONTAINER:**

It is a way of packing an application with all the necessary dependencies and configuration files.

It a directory which has all the libraries/binaries to run the service.

directory has its own IP address and port numbers to access the services.

Container is a virtual machine which doesn’t have any OS because it shares the host machine OS system.

Docker containers are standard, security and lightweight.

Docker engine combines the namespaces(name of the container), Cgroups(CPU, Memory & networks) and UnionFS(File systems) to form a container.

**IMAGES:**

Docker images are used to pack the application.

It consists of multiple layers. Each layer will be some set of data. If you add some files it will add a layer to the image, when you change any configuration files that will create a layer.

When we create image from the container. That image will be read only mode.

Images becomes containers when we run it.

When you add some data in a container and commit it will automatically create a layer.

**Docker exec** - used to run any command in the container from the outside of the container

**Docker it** - interactive terminal, if we are running a container without ‘it’ container will be exited state because it runs on bin/bash command, due to the absence of ‘it’ the shell was dead and container also dead.

ctrl+p+q ——> used to exit from the container without stopping the container.

===================================================================

**RUN**: it is used to execute the commands while we build the images and add a new layer into the image.

FROM centos:centos7

RUN yum install git -y

or

RUN [“yum”, “install”, “git” “-y”]

**CMD:** it is used to execute the commands when we run the container.

It is used to set the default command.

if we have multiple CMD’s only last one will gets executed.

FROM centos:centos7

CMD yum install maven -y

or

CMD [“yum”, “install”, “maven”, “-y”]

If you want to overwrite the parameters:

docker run image\_name httpd (FAILED)

docker run image\_name yum install httpd -y (only httpd will gets installed)

**ENTRYPOINT**: it overwrites the CMD when you pass additional parameters while running the container.

FROM centos:centos7

ENTRYPOINT [“yum”, “install”, “maven”, “-y”]

If you want to overwrite the parameters:

docker run image\_name httpd (both maven and httpd will gets installed)

docker run image\_name yum install httpd -y (both maven and httpd will gets installed)

FROM centos:centos7

ENTRYPOINT [“yum”, “install”, “-y”]

CMD [“httpd”]

Bydefault it will executes httpd command, if you specify the command while running the container it will gets executed.

docker run image\_name (httpd will install)

docker run image\_name git (only git will install)

docker run image\_name git tree(both git & tree will install)

**COPY**: It copies the files from local server to container.

**ADD:** It will also copies the files but the difference is it can handle remote URLs & it can auto-extract tar files.

**ARG**: it is used to define a variable that is used to build a docker image, it will not available once we build it. In containers also it’s not possible to access it.

we can change these variables in command line arguments (docker build -t test --build-arg var\_name=value)

**ENV:** These variables are used for inside the container

we can’t change the values in command line arguments. f we need to change the ENV variable using CMD line then we have to use ARG and place ARG variable in ENV variable

ARG abc=devops

ENV xyz=aws

RUN echo $abc

RUN echo $xyz

===========================================================================

**DOCKER NETWORK**: it allows you to attach your container into many networks, it is used to isolate the containers

To create a network: docker network create —attachable network\_name

**To see the list**: docker network ls

**To delete a network**: docker network rm network\_name

**To inspect**: docker network inspect network\_name

To connect a container to the network: docker network connect network\_name container\_id/name

To disconnect from the container: docker network disconnect network\_name container\_name

To prune: docker network prune

============================================================================

============================================================================

DOCKER RENAME:

To rename docker container: Docker rename old\_container new\_container

To rename docker port:

stop the container

go to path (var/lib/docker/container/container\_id)

open hostconfig.json

edit port number

restart docker and start container

**DOCKER SWARM**: It is a group of servers that runs the docker application.

This can be implemented by the cluster.

The activities of the cluster are controlled by a swarm manager, and machines that have joined the cluster is called swarm worker.

It is used to manage the containers on multiple servers

To create a service: docker service create —name devops —replicas 2 image\_name

Note: image should be present on all the servers

To update the image service: docker service update —image image\_name service\_name

Note: we can change image,

To rollback the service: docker service rollback service\_name

To scale: docker service scale service\_name=3

To check the history: docker service logs

To check the containers: docker service ps service\_name

To inspect: docker service inspect service\_name

To remove: docker service rm service\_name

**. 1. Can you tell something about docker container?**

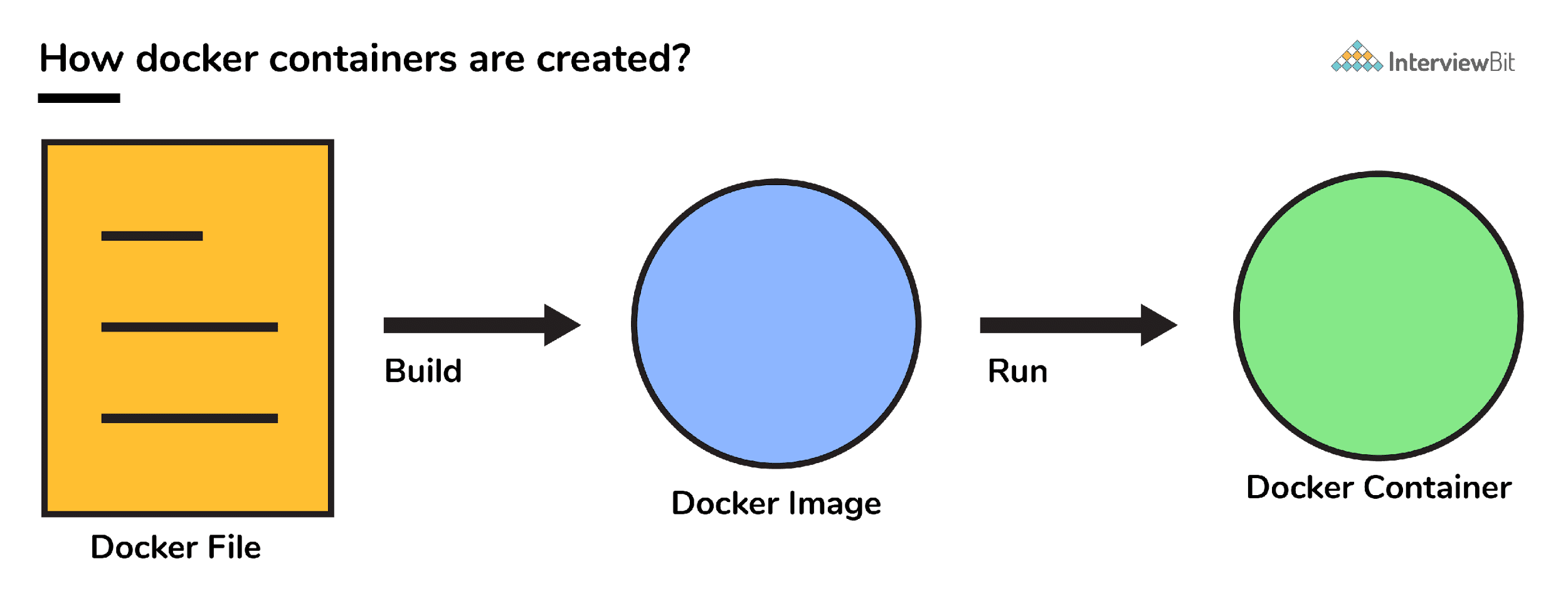
* In simplest terms, docker containers consist of applications and all their dependencies.
* They share the kernel and system resources with other containers and run as isolated systems in the host operating system.
* The main aim of docker containers is to get rid of the infrastructure dependency while deploying and running applications. This means that any containerized application can run on any platform irrespective of the infrastructure being used beneath.
* Technically, they are just the runtime instances of docker images.

**2. What are docker images?**

They are executable packages(bundled with application code & dependencies, software packages, etc.) for the purpose of creating containers. Docker images can be deployed to any docker environment and the containers can be spun up there to run the application.

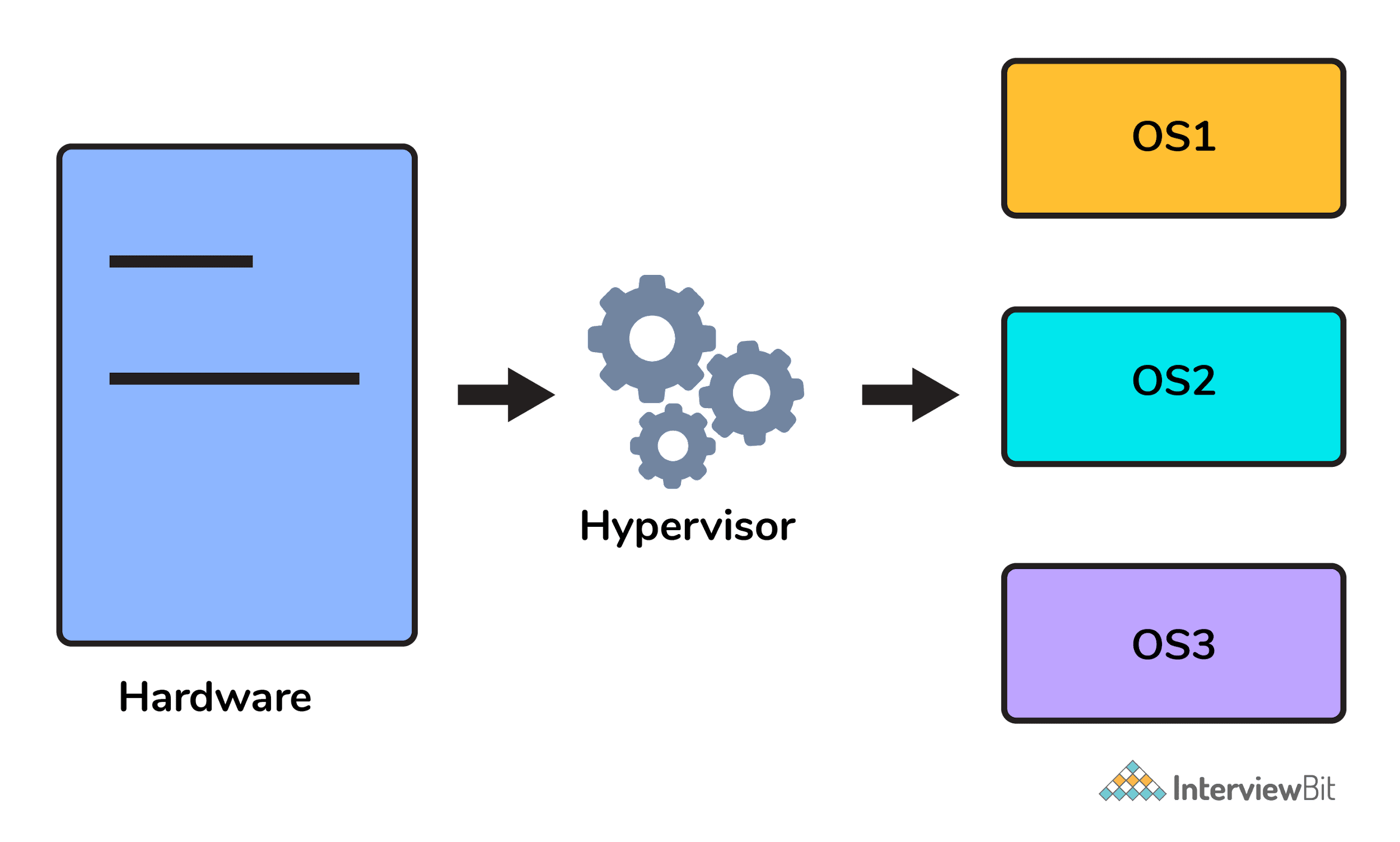
**3. What is a DockerFile?**

* It is a text file that has all commands which need to be run for building a given image.



**4. Can you tell what is the functionality of a hypervisor?**

A hypervisor is a software that makes virtualization happen because of which is sometimes referred to as the Virtual Machine Monitor. This divides the resources of the host system and allocates them to each guest environment installed.



* This means that multiple OS can be installed on a single host system. Hypervisors are of 2 types:  
    
  1. **Native Hypervisor:** This type is also called a Bare-metal Hypervisor and runs directly on the underlying host system which also ensures direct access to the host hardware which is why it does not require base OS.  
  2. **Hosted Hypervisor:** This type makes use of the underlying host operating system which has the existing OS installed.

**5. What can you tell about Docker Compose?**

It is a YAML file consisting of all the details regarding various services, networks, and volumes that are needed for setting up the Docker-based application. So, docker-compose is used for creating multiple containers, host them and establish communication between them. For the purpose of communication amongst the containers, ports are exposed by each and every container.

**6. Can you tell something about docker namespace?**

A namespace is basically a Linux feature that ensures OS resources partition in a mutually exclusive manner. This forms the core concept behind containerization as namespaces introduce a layer of isolation amongst the containers. In docker, the namespaces ensure that the containers are portable and they don't affect the underlying host. Examples for namespace types that are currently being supported by Docker – PID, Mount, User, Network, IPC.

**7. What is the docker command that lists the status of all docker containers?**

In order to get the status of all the containers, we run the below command: docker ps -a

**8. On what circumstances will you lose data stored in a container?**

The data of a container remains in it until and unless you delete the container.

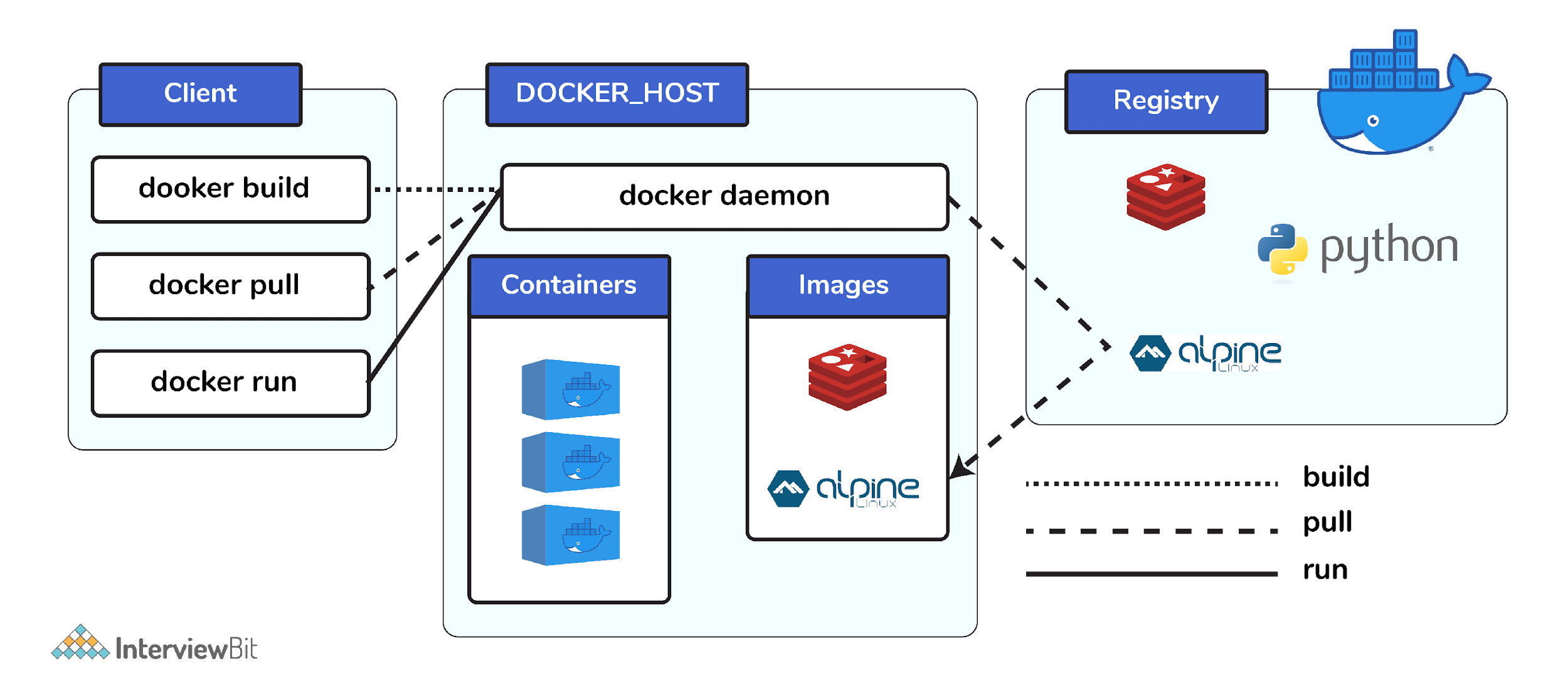
**9. What is docker image registry?**

* A Docker image registry, in simple terms, is an area where the docker images are stored. Instead of converting the applications to containers each and every time, a developer can directly use the images stored in the registry.
* This image registry can either be public or private and Docker hub is the most popular and famous public registry available.

**10. How many Docker components are there?**

There are three docker components, they are - Docker Client, Docker Host, and Docker Registry.

* **Docker Client:** This component performs “build” and “run” operations for the purpose of opening communication with the docker host.
* **Docker Host:** This component has the main docker daemon and hosts containers and their associated images. The daemon establishes a connection with the docker registry.
* **Docker Registry:** This component stores the docker images. There can be a public registry or a private one. The most famous public registries are Docker Hub and Docker Cloud.



**11. What is a Docker Hub?**

* It is a public cloud-based registry provided by Docker for storing public images of the containers along with the provision of finding and sharing them.
* The images can be pushed to Docker Hub through the docker push command.

**12. What command can you run to export a docker image as an archive?**

This can be done using the docker save command and the syntax is: docker save -o <exported\_name>.tar <container-name>

**13. What command can be run to import a pre-exported Docker image into another Docker host?**

This can be done using the docker load command and the syntax is docker load -i <export\_image\_name>.tar

**14. Can a paused container be removed from Docker?**

No, it is not possible! A container MUST be in the stopped state before we can remove it.

**15. What command is used to check for the version of docker client and server?**

* The command used to get all version information of the client and server is the docker version.
* To get only the server version details, we can run docker version --format '{{.Server.Version}}'

**=========================================================**

**LINUX:**

1) What is Linux?

Linux is a UNIX based operating system. Linus Torvalds first introduced it. It is an open source operating system that was designed to provide free and a low-cost operating system for the computer users.

2) What is the difference between UNIX and Linux?

UNIX was originally started as a propriety operating system for Bell Laboratories, which later release their commercial version while Linux is a free, open source and a non-propriety operating system for the mass uses.

3) What is Linux Kernel?

Linux Kernel is low-level system software. It is used to manage the hardware resources for the users. It provides an interface for user-level interaction.

Skip Ad

4) Is it legal to edit Linux Kernel?

Yes. You can edit Linux Kernel because it is released under General Public License (GPL) and anyone can edit it. It comes under the category of free and open source software.

5) What is LILO?

LILO is a boot loader for Linux. It is used to load the Linux operating system into the main memory to begin its operations.

6) What is the advantage of open source?

Open source facilitates you to distribute your software, including source codes freely to anyone who is interested. So, you can add features and even debug and correct errors of the source code.

7) What are the basic components of Linux?

Just like other operating systems, Linux has all components like kernel, shells, GUIs, system utilities and application programs.

8) What is the advantage of Linux?

Every aspect comes with additional features, and it provides a free downloading facility for all codes.

9) Define shell

It is an interpreter in Linux.

10) Name some shells that are commonly used in Linux.

The most commonly used shells in Linux are bash, csh, ksh, bsh.

11) Name the Linux which is specially designed by the Sun Microsystems.

Solaris is the Linux of Sun Microsystems.

12) Name the Linux loader.

LILO is the Linux loader.

13) If you have saved a file in Linux. Later you wish to rename that file, what command is designed for it?

The **'mv'** command is used to rename a file.

14) Write about an internal command.

The commands which are built in the shells are called as the internal commands.

15) Define inode.

Each file is given a unique name by the operating system which is called as the inode.

**EXTRA LINUX COMMANDS:**

LINUX:

OS: it is a software which acts as a mediator between user and computer. It is used to interact with the computer. It provides environment to run the application.

Without OS no system will work.

EX: windows, Linux

SYSTEM COMMANDS:

df : used to show how much of free space is available in memory.

ping: is used to check the network connection

kill -9: command sends a kill signal to terminate any process immediately

dmesg: to display kernel related messages

du : used to show the disk usage of a particular file or folder.

netstat: is used to show the network connection and activity of a system

| ll | To see the files |
| --- | --- |
| ls -l | To display long list |
| ll -r | Reverse order |
| ll -t | Last modified |
| ll -tr | Reverse on last modified |
| ls -ltr | To display all latest modified files in reverse order |
| ls -al | To display hidden files |
| ll -il | It displays files along with ID’s |
| ll -lR | To see the files inside the folder in pwd |
| ll -lr | To see all the files inside all the directories |
| Cat -n filename | Content along with line numbers |
| Cat f1 f2 f3 | To display all the data in multiple files |
| Cat f1 > f2 | Copy data from f1 to f2 |
| Cat f1 | To read the data |
| tac f1 | To recursive the data |
| rev f1 | To reverse every line content |
| Head -n 8 f1 | To display 8 lines of the data |
| wc filename | To see lines, words and characters in a file. |
| Tail -n 4 filename | To get last 4 numbers of the file |
| Cmp file1 file2 | To compare the data of the files |
| Diff file1 file2 | To show the data in to two files |
| rpm -qa | To see the installed packages |
| rpm -qa | grep git | To check git is installed or not |
| grep ‘word’ filename | To search a word in a file |
| grep -I ‘word’ filename | To search for a word in a file |
| grep -c ‘word’ filename | To count the words |
| grep -n ‘word’ filename | To get the line number of a word in a file |
| grep -r ‘word’ | Displays matches lines with file names |

Why tail command: when we run our application, some log data will be appear at the end of the file. So if we want to see the log data of a file we will use tail command.

cat vs cp : cp will not copy the Data to multiple file and cat will copy the data to multiple files.

Cmp vs diff: cmp will returns only the difference of the file. Diff will returns the full difference of the file.

GREP: Global Regular Expression Print. It is used to search for a string in a group of files.

SYNTAX: grep ’word’ filename

VIM EDITOR:

k : one line up

l : one line down

h : left side one character ahead

l : right side one character ahead

0 : starting of the line

( : starting of the para

) : ending of the para

$ : at the end of the line

M : middle of the the screen

H : top of the screen

L : bottom of the screen

:%s/oldword/newword ——> to replace the word using vi editor

sed ’s/oldword/newword/‘ file1 ——> to replace the word in the first occurrence of the line

sed ’s/oldword/newword/g’ filename —> to replace all the words (g is global)

sed ’s/oldword/newword/2’ filename —> to replace all the words In the second line of the file

USERS:

We can change the username of the file by using the uid aswell.

FIND VS LOCATE: used to search the files in a linux

LOCATE: finds the files in linux database.

Locate filename

Locate -c filename

FIND: It will search the files and folders in the file system.

Find path/filename

Find /path -type d -empty

WGET VS CURL:

wget is used to download the files

Curl is also same but it will sends the request to urls

ping: to check network connection

SHELL: It acts as an interface b/w user and kernel

To find the files: find path -iname “file” -mtime -2 -print