

# DOCKER

---

## 1. What command builds a Docker image from a Dockerfile?

- A. docker run
- B. docker build
- C. docker create
- D. docker compile

**Answer: B**

Explanation: 'docker build' reads the Dockerfile directives and creates an image layer by layer based on the instructions.

## 2. Which command lists running containers?

- A. docker ps
- B. docker ls
- C. docker list
- D. docker status

**Answer: A**

Explanation: 'docker ps' shows currently running containers; 'docker ps -a' additionally lists stopped containers.

## 3. What does 'docker run -d image' do?

- A. Run container in detached mode
- B. Delete container
- C. Download image
- D. Display logs

**Answer: A**

Explanation: The '-d' flag detaches the container from the terminal, letting it run in the background.

## 4. Which command stops a running container?

- A. docker stop <container>
- B. docker kill <container>
- C. docker pause <container>
- D. docker quit <container>

**Answer: A**

Explanation: 'docker stop' sends SIGTERM followed by SIGKILL after a grace period, allowing graceful shutdown.

## 5. What is a Docker image?

- A. Immutable template used to create containers

- B. Running process
- C. Network bridge
- D. Volume

**Answer: A**

Explanation: Images encapsulate filesystem and metadata required to instantiate containers, serving as blueprints.

## 6. Which file defines instructions to build Docker image?

- A. Dockerfile
- B. docker-compose.yml
- C. image.json
- D. container.cfg

**Answer: A**

Explanation: The Dockerfile holds build instructions (FROM, RUN, COPY, etc.) that 'docker build' executes to produce an image.

## 7. What does 'docker pull' do?

- A. Download image from registry
- B. Remove image
- C. Push image
- D. Create volume

**Answer: A**

Explanation: 'docker pull' fetches images from registries like Docker Hub or private registries, making them available locally.

## 8. How do you remove a container?

- A. docker rm <container>
- B. docker delete <container>
- C. docker remove <container>
- D. docker rmc <container>

**Answer: A**

Explanation: 'docker rm' deletes stopped containers; 'docker rm -f' forces removal of running containers by stopping them first.

## 9. Which command shows container logs?

- A. docker logs <container>
- B. docker output <container>
- C. docker tail <container>
- D. docker history <container>

**Answer: A**

Explanation: 'docker logs' displays the stdout/stderr stream produced by the container, optionally following output with '-f'.

**10. What does 'docker exec -it container bash' do?**

- A. Start interactive bash shell inside running container
- B. Execute command and exit
- C. Inspect container
- D. Restart container

**Answer: A**

Explanation: 'docker exec' runs commands in existing containers; '-it' attaches an interactive terminal, launching Bash if available.

**11. Which command lists Docker images locally?**

- A. docker images
- B. docker image ls
- C. Both A and B
- D. docker list images

**Answer: C**

Explanation: Both commands are equivalent, showing local images with repository, tag, image ID, and size.

**12. What is the default network driver for Docker?**

- A. bridge
- B. host
- C. overlay
- D. macvlan

**Answer: A**

Explanation: Standalone containers attach to the default bridge network unless otherwise specified, providing NAT connectivity.

**13. Which instruction sets working directory in Dockerfile?**

- A. WORKDIR /app
- B. RUN cd /app
- C. ENV WORKDIR=/app
- D. SET WORKDIR /app

**Answer: A**

Explanation: 'WORKDIR' changes the working directory for subsequent Dockerfile instructions and for the container runtime.

**14. What does 'docker-compose up' do?**

- A. Create and start services defined in docker-compose.yml
- B. Stop services
- C. Delete services
- D. Build images only

**Answer: A**

Explanation: 'docker-compose up' creates containers, networks, and volumes as per the compose file, starting the defined services.

**15. How do you build images with docker-compose?**

- A. docker-compose build
- B. docker-compose make
- C. docker-compose compile
- D. docker-compose image

**Answer: A**

Explanation: 'docker-compose build' builds service images defined in the compose file using the specified build contexts.

**16. Which command removes unused images, containers, networks?**

- A. docker system prune
- B. docker clean
- C. docker prune all
- D. docker gc

**Answer: A**

Explanation: 'docker system prune' cleans up dangling resources; adding '-a' also removes unused images not referenced by containers.

**17. What does 'EXPOSE 80' do in Dockerfile?**

- A. Document port container listens on
- B. Bind port automatically
- C. Expose to host
- D. Secure port

**Answer: A**

Explanation: 'EXPOSE' is informational, advertising intended ports; actual mapping occurs with 'docker run -p'.

**18. How do you map host port to container port?**

- A. docker run -p 8080:80 image
- B. docker run --port 8080 80
- C. docker port map 8080 80
- D. docker expose 8080 80

**Answer: A**

Explanation: '-p host:container' exposes container port 80 on host port 8080, enabling access from outside the container.

**19. What is a Docker volume?**

- A. Persistent storage managed by Docker outside container filesystem
- B. Network driver
- C. CPU limit

D. Memory allocation

**Answer: A**

Explanation: Volumes store data independently of container lifecycles, easing data persistence and sharing between containers.

**20. Which command creates a named volume?**

- A. docker volume create myvol
- B. docker create volume myvol
- C. docker volume new myvol
- D. docker volume add myvol

**Answer: A**

Explanation: 'docker volume create' provisions named volumes accessible by containers via '-v' or '--mount'.

**21. What is the base command to inspect container details?**

- A. docker inspect <container>
- B. docker info <container>
- C. docker details <container>
- D. docker show <container>

**Answer: A**

Explanation: 'docker inspect' outputs JSON describing container configuration, networking, volumes, and runtime state.

**22. How do you specify base image in Dockerfile?**

- A. FROM ubuntu:22.04
- B. BASE ubuntu
- C. IMAGE ubuntu
- D. START ubuntu

**Answer: A**

Explanation: The 'FROM' instruction declares the parent image for subsequent Dockerfile commands, such as official base images or custom ones.

**23. Which command tags an image?**

- A. docker tag source:tag target:tag
- B. docker label image:tag
- C. docker rename image tag
- D. docker set-tag image tag

**Answer: A**

Explanation: 'docker tag' assigns a new repository:tag reference to an existing image ID, facilitating pushes to registries.

**24. What is 'ENTRYPOINT' used for?**

- A. Define default executable for container

- B. Define environment variables
- C. Create volumes
- D. Copy files

**Answer: A**

Explanation: 'ENTRYPOINT' sets the main command that always executes when the container starts, optionally combined with 'CMD' arguments.

**25. What does 'docker-compose down' do by default?**

- A. Stop and remove containers, networks created by compose
- B. Delete images
- C. Remove volumes always
- D. Stop Docker daemon

**Answer: A**

Explanation: 'docker-compose down' stops services and removes associated containers and networks; '--volumes' removes volumes too.

**26. How do you limit CPU usage for container?**

- A. docker run --cpus="1.5" image
- B. docker run --cpu 1.5 image
- C. docker run --limit-cpu 1.5 image
- D. docker run --cpuquota 1.5 image

**Answer: A**

Explanation: '--cpus' sets a quota on CPU resources; e.g., '1.5' allows 1 and a half cores worth of CPU time.

**27. Which command pulls latest busybox image?**

- A. docker pull busybox:latest
- B. docker get busybox
- C. docker fetch busybox latest
- D. docker update busybox

**Answer: A**

Explanation: Proper syntax is 'docker pull repository:tag'; 'latest' is the default tag, so 'docker pull busybox' also works.

**28. What does 'docker login' accomplish?**

- A. Authenticate to Docker registry
- B. Login to container
- C. Start Docker daemon
- D. Access Docker Hub website

**Answer: A**

Explanation: Logging in stores credentials so pushes and pulls to private registries succeed; credentials are cached in Docker config.

**29. Which storage driver is default on many Linux distributions?**

- A. overlay2
- B. aufs
- C. devicemapper
- D. zfs

**Answer: A**

Explanation: Overlay2 is widely used due to performance and stability, layering filesystems efficiently for containers.

**30. How do you copy file from host into container using Dockerfile?**

- A. COPY src dest
- B. ADD src dest
- C. Both (ADD has extra features)
- D. put src dest

**Answer: C**

Explanation: 'COPY' copies files locally; 'ADD' can also extract tar archives or download URLs, though 'COPY' is preferred unless advanced features required.

**31. What is the default restart policy?**

- A. no
- B. on-failure
- C. always
- D. unless-stopped

**Answer: A**

Explanation: Containers don't restart by default ('no'); specifying policies ensures they restart automatically under desired conditions.

**32. How do you set environment variable in Dockerfile?**

- A. ENV VAR=value
- B. SET VAR=value
- C. EXPORT VAR=value
- D. Environment VAR=value

**Answer: A**

Explanation: 'ENV' defines environment variables available during build and runtime, accessible by the running container.

**33. Which command removes an image?**

- A. docker rmi image:tag
- B. docker rm image:tag
- C. docker delete image:tag
- D. docker image remove image:tag

**Answer: A**

Explanation: 'docker rmi' removes images by name or ID; tags referencing the same image must be removed before the image can be deleted.

**34. What does 'docker-compose logs -f' do?**

- A. Follow logs for compose services
- B. Delete logs
- C. List log files
- D. Format logs

**Answer: A**

Explanation: Following logs streams aggregated output from service containers, aiding live troubleshooting.

**35. What is the purpose of multi-stage build?**

- A. Use multiple FROM stages to create smaller final images
- B. Run containers in stages
- C. Combine several compose files
- D. Build multiple images simultaneously

**Answer: A**

Explanation: Multi-stage builds compile dependencies in intermediate stages and copy only necessary artifacts into a lightweight final image.

**36. How do you connect container to custom network?**

- A. docker network create mynet; docker run --network=mynet image
- B. docker net create mynet; docker connect mynet
- C. docker network add container mynet
- D. docker network attach mynet image

**Answer: A**

Explanation: Creating a network and specifying '--network' attaches the container to that network, enabling isolation or custom connectivity.

**37. What does 'docker save' do?**

- A. Export image to tar archive
- B. Save container logs
- C. Snapshot container
- D. Backup volumes

**Answer: A**

Explanation: 'docker save' writes image layers to a tarball, suitable for transport or offline loading with 'docker load'.

**38. Which command exports container filesystem as tar?**

- A. docker export container > file.tar
- B. docker save container > file.tar
- C. docker backup container



D. docker snapshot container

**Answer: A**

Explanation: 'docker export' captures a container's filesystem (without metadata) as a tar file for archival or migration.

**39. What does 'docker history image' show?**

- A. Layer history of image
- B. Container history
- C. Log history
- D. Registry history

**Answer: A**

Explanation: 'docker history' lists layers, sizes, and commands used to build an image, assisting in analyzing image composition.

**40. How do you specify build arguments?**

- A. docker build --build-arg KEY=value .
- B. docker build ARG KEY=value
- C. docker build KEY=value
- D. docker run --build-arg

**Answer: A**

Explanation: Build arguments provide values to Dockerfile 'ARG' instructions, enabling parameterized builds.

**41. What is 'docker context' used for?**

- A. Manage multiple Docker endpoints (local/remote)
- B. Manage swarm nodes
- C. Manage Compose files
- D. Manage volumes

**Answer: A**

Explanation: Contexts let you switch between Docker daemons (local, remote servers, cloud services) by changing configuration profiles.

**42. How do you list Docker contexts?**

- A. docker context ls
- B. docker context list
- C. docker context show
- D. docker contexts

**Answer: A**

Explanation: 'docker context ls' enumerates available contexts, showing endpoints and default settings.

**43. In Docker Swarm, what is a service?**

- A. Definition of tasks (containers) running on cluster

- B. Single container
- C. Volume
- D. Network

**Answer: A**

Explanation: A Swarm service defines desired state (image, replicas, constraints) that Swarm orchestrates across nodes.

**44. What is the difference between 'CMD' and 'ENTRYPOINT'?**

- A. ENTRYPOINT sets default executable; CMD provides default arguments or command
- B. CMD overrides ENTRYPOINT always
- C. ENTRYPOINT runs second
- D. No difference

**Answer: A**

Explanation: 'ENTRYPOINT' defines the fixed command; 'CMD' supplies default parameters or command that can be overridden on the CLI.

**45. Which command updates running container image to latest?**

- A. No direct update; stop, pull new image, recreate container
- B. docker update image
- C. docker upgrade container
- D. docker refresh container

**Answer: A**

Explanation: Containers are immutable; to update you pull the new image, remove or stop the old container, and run a new container.

**46. How do you run container with read-only filesystem?**

- A. docker run --read-only image
- B. docker run --readonly image
- C. docker run --fs readonly image
- D. docker run --disk readonly image

**Answer: A**

Explanation: '--read-only' mounts the container's root filesystem as read-only, improving security by preventing writes.

**47. Which command sets container restart policy to always?**

- A. docker run --restart=always image
- B. docker run restart always image
- C. docker restart always image
- D. docker run --always image

**Answer: A**

Explanation: The '--restart' flag configures restart policies; 'always' restarts the container regardless of exit status.

**48. How do you scale service in docker-compose?**

- A. docker-compose up --scale web=3
- B. docker-compose scale web=3
- C. Both (scale command deprecated but still works)
- D. docker-compose replicate web 3

**Answer: C**

Explanation: '--scale' scales a service to the specified number of containers; legacy 'docker-compose scale' remains available.

**49. What does 'docker inspect' return by default?**

- A. JSON data with detailed info
- B. YAML data
- C. Table output
- D. XML

**Answer: A**

Explanation: 'docker inspect' outputs detailed JSON metadata for containers, images, networks, or volumes.

**50. How can you pass environment variables to container at runtime?**

- A. docker run -e VAR=value image
- B. docker run --env-file file image
- C. Both A and B
- D. docker run env VAR

**Answer: C**

Explanation: Using '-e' sets individual variables; '--env-file' loads many from a file, making configuration flexible.

**51. What is overlay network used for?**

- A. Multi-host networking for Swarm services
- B. Local host networks
- C. Port mapping
- D. Volume sharing

**Answer: A**

Explanation: Overlay networks span multiple Docker hosts, enabling container-to-container communication across the Swarm cluster.

**52. Which log driver is default?**

- A. json-file
- B. syslog
- C. journald
- D. none

**Answer: A**

Explanation: Docker stores container logs in JSON-file format by default, accessible via 'docker logs'.

**53. How do you limit memory for container?**

- A. docker run -m 512m image
- B. docker run --memory=512 image
- C. docker run --limit-mem 512 image
- D. docker run --max-mem=512 image

**Answer: A**

Explanation: The '-m' or '--memory' flag caps container memory usage, preventing excessive resource consumption.

**54. What does 'HEALTHCHECK' instruct in Dockerfile?**

- A. Command to determine container health status
- B. Security check
- C. Performance check
- D. Build check

**Answer: A**

Explanation: 'HEALTHCHECK' specifies a command that Docker uses to determine container health, reflecting status in 'docker ps'.

**55. How do you list volumes?**

- A. docker volume ls
- B. docker volumes
- C. docker volume list
- D. docker list volumes

**Answer: A**

Explanation: 'docker volume ls' displays existing volumes, showing driver and name.

**56. Which command creates Docker secrets in Swarm?**

- A. docker secret create
- B. docker secret new
- C. docker swarm secret create
- D. docker secret add

**Answer: A**

Explanation: Secrets are created using 'docker secret create', then referenced in service definitions to mount securely.

**57. How do you deploy stack in Swarm using compose file?**

- A. docker stack deploy -c docker-compose.yml mystack
- B. docker compose deploy mystack
- C. docker swarm deploy mystack
- D. docker stack up mystack

**Answer: A**

Explanation: 'docker stack deploy' leverages compose files to define Swarm services, networks, and volumes, deploying them as a stack.

**58. What does 'docker cp file container:/path' do?**

- A. Copy file from host to container path
- B. Copy file from container to host
- C. Copy between containers
- D. Copy file to registry

**Answer: A**

Explanation: 'docker cp' transfers files between host and container; reversing arguments copies from container to host.

**59. How do you view resource usage statistics of containers?**

- A. docker stats
- B. docker top
- C. docker info
- D. docker usage

**Answer: A**

Explanation: 'docker stats' streams CPU, memory, network, and I/O metrics for running containers in real time.

**60. Which command lists layers of image?**

- A. docker history image
- B. docker layers image
- C. docker image layers image
- D. docker inspect layers

**Answer: A**

Explanation: 'docker history' reveals each layer's size and command, aiding optimization and troubleshooting.

**61. What is Docker Hub?**

- A. Public registry for Docker images
- B. Orchestration tool
- C. Build system
- D. CLI plugin

**Answer: A**

Explanation: Docker Hub hosts public and private images, enabling sharing and distribution of container images.

**62. How do you create named container?**

- A. docker run --name mycontainer image
- B. docker run name mycontainer image

- C. docker name create image
- D. docker run -n mycontainer image

**Answer: A**

Explanation: '--name' assigns a custom name to the container, making management easier than relying on random names.

**63. Which command updates Docker Engine on Linux?**

- A. Use OS package manager (e.g., apt-get install docker-ce)
- B. docker update engine
- C. docker upgrade
- D. docker engine update

**Answer: A**

Explanation: Docker Engine is installed and upgraded using the distribution's package manager or official repositories, not via Docker commands.

**64. How do you view Docker daemon configuration?**

- A. /etc/docker/daemon.json
- B. docker config show
- C. docker info config
- D. docker daemon config

**Answer: A**

Explanation: Daemon settings reside in 'daemon.json', where you define default network, logging, registry mirrors, etc.

**65. What does 'docker login registry.example.com' require?**

- A. Credentials for specified registry
- B. Access token only
- C. SSH key
- D. Nothing

**Answer: A**

Explanation: Logging into a registry requires username/password or token, stored for authenticating pushes and pulls.

**66. How do you remove all stopped containers?**

- A. docker container prune
- B. docker rm \$(docker ps -aq -f status=exited)
- C. Both (A simpler)
- D. docker clean containers

**Answer: C**

Explanation: 'docker container prune' removes all stopped containers; using 'docker rm' with command substitution accomplishes the same with more typing.

**67. What is 'docker context use' command for?**

- A. Switch active Docker context
- B. Use context file
- C. Run in context
- D. Create context

**Answer: A**

Explanation: 'docker context use' changes the currently active context, switching to different Docker endpoints or environments.

**68. Which command inspects network?**

- A. docker network inspect mynet
- B. docker inspect network mynet
- C. docker net inspect mynet
- D. docker network show mynet

**Answer: A**

Explanation: 'docker network inspect' shows detailed configuration, connected containers, and options for networks.

**69. What does 'docker rm -f container' do?**

- A. Force remove running container (stop + remove)
- B. Remove image
- C. Remove file
- D. Remove network

**Answer: A**

Explanation: '-f' stops the container and removes it in one command, useful when containers don't stop gracefully.

**70. How do you configure logging driver per container?**

- A. docker run --log-driver=syslog image
- B. docker run --log syslog image
- C. docker run --logger syslog image
- D. docker log-driver syslog image

**Answer: A**

Explanation: '--log-driver' sets the logging backend for an individual container, overriding daemon defaults as needed.

**71. What is default location of Docker named volumes?**

- A. /var/lib/docker/volumes
- B. /var/docker/volumes
- C. /var/lib/volumes
- D. /docker/volumes

**Answer: A**

Explanation: Docker stores volume data under '/var/lib/docker/volumes/<name>/\_data'

unless an alternative driver is used.

**72. How do you attach to running container console?**

- A. docker attach container
- B. docker exec -it container sh
- C. Both (depending on use)
- D. docker connect container

**Answer: C**

Explanation: 'docker attach' connects to the main process's stdout/stderr; 'docker exec -it' runs a new shell for interactive access.

**73. What does 'docker info' show?**

- A. System-wide information about Docker environment
- B. Container logs
- C. Compose config
- D. Image history

**Answer: A**

Explanation: 'docker info' displays daemon settings, number of containers, images, storage driver, and platform specifics.

**74. How do you update container environment variables without rebuild?**

- A. Typically recreate container with new vars
- B. docker update env
- C. docker env set
- D. docker container env

**Answer: A**

Explanation: Environment variables are set at container start; to change them you normally recreate the container with updated '-e' values.

**75. Which Dockerfile instruction adds metadata label?**

- A. LABEL key="value"
- B. META key=value
- C. INFO key=value
- D. TAG key=value

**Answer: A**

Explanation: 'LABEL' stores metadata in images, useful for versioning, ownership, or integration with image scanners.

**76. How do you configure Docker BuildKit?**

- A. Export DOCKER\_BUILDKIT=1 or set in daemon.json
- B. docker build --buildkit
- C. Both options
- D. Not configurable



**Answer: C**

Explanation: BuildKit is enabled by environment variable or daemon configuration, offering faster builds and advanced features like caching.

**77. What is 'docker-compose.override.yml' used for?**

- A. Override or extend default compose configuration
- B. Replace Dockerfile
- C. Define swarm services
- D. Manage volumes only

**Answer: A**

Explanation: Compose automatically merges 'docker-compose.yml' with 'docker-compose.override.yml', enabling environment-specific overrides (e.g., development).

**78. How do you list services in Swarm?**

- A. docker service ls
- B. docker services
- C. docker swarm service ls
- D. docker service list

**Answer: A**

Explanation: 'docker service ls' displays active Swarm services with replicas, image, and mode information.

**79. What does 'docker service scale svc=5' do?**

- A. Scale Swarm service named svc to 5 replicas
- B. Scale compose service
- C. Scale container
- D. Scale volume

**Answer: A**

Explanation: Scaling adjusts the desired number of replicas for a service, and Swarm schedules tasks accordingly.

**80. Which command inspects container resource usage stats once?**

- A. docker container stats --no-stream
- B. docker stats --once
- C. docker top --stats
- D. docker usage

**Answer: A**

Explanation: 'docker container stats --no-stream' prints a single snapshot of resource usage and exits.

**81. How do you configure docker-compose to use specific profiles?**

- A. docker-compose --profile dev up
- B. Set COMPOSE\_PROFILES=dev

- C. Both methods
- D. docker-compose profile dev

**Answer: C**

Explanation: Profiles allow selective service activation based on profile names, enabled via CLI or environment variables.

## 82. What does 'docker plugin install' do?

- A. Install Docker plugin (e.g., volume/log drivers)
- B. Install compose plugin
- C. Install CLI extension
- D. Install swarm

**Answer: A**

Explanation: Docker plugins extend functionality (storage, logging, networking) and are managed via 'docker plugin' commands.

## 83. How do you pass secrets to containers securely in Swarm?

- A. docker secret create + mount via service definition
- B. docker run -e SECRET
- C. docker volume secret
- D. docker env secret

**Answer: A**

Explanation: Swarm secrets are encrypted and mounted as files inside containers when referenced in service definitions, protecting sensitive data.

## 84. What is difference between 'docker compose' and 'docker-compose'?

- A. docker compose is plugin integrated into CLI (v2); docker-compose is legacy Python CLI
- B. No difference
- C. docker compose only on Windows
- D. docker-compose only on Linux

**Answer: A**

Explanation: Docker Compose V2 is integrated as a CLI plugin ('docker compose'), while 'docker-compose' refers to the standalone V1 Python binary.

## 85. How do you create build cache export?

- A. docker build --cache-to=type=local,dest=cache
- B. docker build --cache export
- C. docker cache save
- D. docker build cache

**Answer: A**

Explanation: BuildKit supports exporting caches to local directories or registries, enabling reuse in subsequent builds, especially in CI.

**86. Which command upgrades Compose v2 plugin?**

- A. Use system package manager or Docker Desktop update
- B. docker compose update
- C. docker-compose update
- D. docker plugin upgrade compose

**Answer: A**

Explanation: Compose plugin updates come from new Docker Desktop releases or package manager upgrades; there isn't a 'docker compose update' command.

**87. How do you inspect layer sizes?**

- A. docker history --no-trunc image
- B. docker image inspect --format '{{.Size}}'
- C. docker system df
- D. All provide size info (A for per layer)

**Answer: D**

Explanation: 'docker history' shows per-layer sizes; 'docker image inspect' provides aggregate size; 'docker system df' summarizes disk usage for images/containers.

**88. What does 'docker login --password-stdin' enable?**

- A. Provide password via stdin for security
- B. Login via CLI interactively
- C. Login via GUI
- D. Store password in file

**Answer: A**

Explanation: Passing passwords through stdin avoids storing credentials in shell history, improving security for automation scripts.

**89. How do you enable experimental Docker features?**

- A. Set "experimental": true in daemon.json or export DOCKER\_CLI\_EXPERIMENTAL=enabled
- B. docker experimental on
- C. docker enable experimental
- D. Not possible

**Answer: A**

Explanation: Experimental features require enabling flags via daemon configuration or environment variables; there is no direct CLI toggle.

**90. Which command removes all unused volumes?**

- A. docker volume prune
- B. docker prune volumes
- C. docker volume clean
- D. docker volume rm all

**Answer: A**

Explanation: 'docker volume prune' removes dangling volumes not referenced by any containers, freeing disk space.

**91. How do you set log rotation with json-file driver?**

- A. Configure daemon.json with log-opts max-size/max-file
- B. docker run --log-opt max-size=10m
- C. Both methods
- D. Not supported

**Answer: C**

Explanation: Log rotation options can be set globally in 'daemon.json' or per-container with '--log-opt', preventing logs from consuming excessive disk.

**92. What does 'docker buildx' provide?**

- A. Extended build capabilities like multi-arch builds
- B. Compose features
- C. Swarm management
- D. Volume plugins

**Answer: A**

Explanation: Buildx leverages BuildKit for advanced features such as cross-platform builds, caching, and exporting.

**93. How do you run container in privileged mode?**

- A. docker run --privileged image
- B. docker run privileged image
- C. docker run --elevated image
- D. docker run --sudo image

**Answer: A**

Explanation: '--privileged' lifts many restrictions, giving the container full access to host devices and capabilities; use sparingly.

**94. What is the default Docker Swarm quorum requirement?**

- A. Majority of managers must be available
- B. All nodes must be available
- C. Any node available
- D. Only one manager needed

**Answer: A**

Explanation: Swarm requires a majority of manager nodes to maintain quorum and coordinate the cluster reliably.

**95. How do you label node in Swarm?**

- A. docker node update --label-add env=prod node1
- B. docker node label node1 env=prod
- C. docker swarm label node1 env=prod

D. `docker label node1 env=prod`

**Answer: A**

Explanation: 'docker node update --label-add' assigns labels to nodes, enabling scheduling constraints based on those labels.

**96. Which command promotes worker to manager?**

- A. `docker node promote node1`
- B. `docker swarm promote node1`
- C. `docker node manager node1`
- D. `docker promote node1`

**Answer: A**

Explanation: 'docker node promote' converts worker nodes into managers, expanding the manager quorum.

**97. How do you view compose config after merging files?**

- A. `docker compose config`
- B. `docker-compose config`
- C. Both
- D. `docker compose show`

**Answer: C**

Explanation: The 'config' command renders the effective configuration after merging overrides, validating and showing final services.

**98. What does 'docker context create' require?**

- A. Name and endpoint configuration
- B. Only name
- C. Only endpoint
- D. Nothing

**Answer: A**

Explanation: Creating a context involves specifying a name plus endpoint (e.g., Docker host, certificates), enabling remote management.

**99. How do you run container with specific user?**

- A. `docker run -u user image`
- B. `docker run --user user image`
- C. Both A and B
- D. `docker run user=user image`

**Answer: C**

Explanation: Setting '-u' or '--user' changes the user context within the container, improving security by avoiding root.

**100. What is 'docker scan'?**

- A. Security vulnerability scanning powered by Snyk

- B. Image building
- C. Orchestration
- D. Logging

**Answer: A**

Explanation: 'docker scan' analyzes images for known vulnerabilities using integrated Snyk scanning, informing remediation steps.