RUN AND OPERATE CONTAINERS**Introduction & Learning Objectives**

# Chapter Overview

### RUN AND OPERATE CONTAINERSKnowledge Check

Congratulations on completing Chapter 3 - Run and Operate Containers. Take this quiz to check your understanding of the concepts you've learned about so far.

Chapter 3 - Run and Operate Containers RUN AND OPERATE CONTAINERS Knowledge Check Question 3.1 When you launch a container and use -p 8000, it will map 8000 port on the Docker host to the port defined in the image with EXPOSE xx as a metadata. True or False? A. True B. False

ChatGPT

A. True

The statement is True.

When you launch a container and use the **-p** flag followed by the port number (e.g., **-p 8000**), it maps the specified port on the Docker host to the port defined in the container image using the **EXPOSE** instruction in the Dockerfile.

The **EXPOSE** instruction in the Dockerfile is used to indicate which ports the container is listening on for incoming connections. It does not actually publish or expose the ports by itself. Instead, it serves as a form of metadata to inform users of the container's expected network behavior.

By using the **-p** flag during container launch, you publish (or expose) the container's specified ports to the Docker host. This allows you to access the container's services from the host or from external networks using the mapped port on the host.

For example, if the Docker image has **EXPOSE 80** in its Dockerfile and you launch the container with **-p 8000:80**, it will map port 8000 on the Docker host to port 80 in the container, enabling you to access the web service running inside the container on the Docker host's port 8000.



Question 3.2 Which of the following options is used to continuously update logs of a running container? A. -c B. -f C. -follow D. -continue

ChatGPT

B. -f

The **-f** option is used to continuously update logs of a running container in Docker.

When you run a container with the **-f** or **--follow** flag, it allows you to view the real-time log output of the container. This means that as the container produces new log messages, they will be displayed on the terminal or console where you executed the **docker logs** command.

For example, to continuously follow the logs of a running container named "my\_container," you would use the following command:

Copy code

docker logs -f my\_container

This is useful for monitoring the output of a containerized application in real-time, troubleshooting issues, or understanding the behavior of the application while it is running in the container.



Question 3.3 Kalam has launched a container with an image specification "schoolofdevops/testapp". Can you identify the tag that Kalam used to launch the container with? A. testapp B. schoolofdevops C. latest D. v1.0.1

ChatGPT

Without more information about the available tags in the "schoolofdevops/testapp" Docker image, it is not possible to identify the specific tag that Kalam used to launch the container. The tag names for Docker images can vary depending on the image's versioning and repository structure.

The tags for Docker images are used to specify different versions or variations of the same image. Common tag names include "latest" for the most recent version and version numbers like "v1.0.1" for specific releases.

To find out the available tags for the "schoolofdevops/testapp" image, you can use the following command:

bashCopy code

docker image ls schoolofdevops/testapp

This will list the available tags for the specified image, and from there, you can identify which tag Kalam used to launch the container.



Question 3.4 The docker stop command wipes out all the changes you have made since launching the container. True or False? A. True B. False

ChatGPT

B. False

The statement is False.

The **docker stop** command is used to stop a running container gracefully, allowing it to perform any necessary cleanup tasks before it shuts down. When you stop a container using **docker stop**, it does not wipe out any changes you have made inside the container.

Stopping a container using **docker stop** is the preferred method for shutting down a container because it gives the container a chance to handle any shutdown procedures properly. This allows the container to save any temporary data or perform cleanup operations before stopping.

If you need to make changes to a container and want to preserve those changes even after stopping the container, you should make sure to create a Docker image with those changes by using the **docker commit** command. This will create a new image with the changes you made, which you can later use to create new containers that include those modifications.

On the other hand, if you want to discard changes made within a container and revert it to its original state, you can use the **docker rm** command to remove the existing container and then create a new container from the original image, effectively starting with a fresh state.

Remember that Docker containers are designed to be stateless and ephemeral, which means that any changes made inside a container should ideally be captured in a new image or persisted externally, rather than being directly applied to the original container.



Question 3.4 The docker stop command wipes out all the changes you have made since launching the container. True or False? A. True B. False

ChatGPT

B. False

The statement is False.

The **docker stop** command gracefully stops a running container, allowing it to perform any necessary cleanup tasks before shutting down. When you stop a container using **docker stop**, it does not wipe out any changes you have made inside the container.

The purpose of **docker stop** is to initiate a controlled shutdown of the container, giving it a chance to save any temporary data or perform cleanup operations before stopping. This ensures that the container shuts down in an orderly manner.

If you have made changes to a container and want to preserve those changes, you need to create a new Docker image from the modified container using the **docker commit** command. This will create a new image that includes the changes you made inside the container.

However, it is important to note that directly modifying a running container and then committing it to an image is generally not considered a recommended practice in production scenarios. Instead, it is better to use Dockerfiles and version-controlled configurations to define the desired state of a container and create images from them. This way, the changes are properly documented and can be easily reproduced.



Question 3.5 Take a look at the sequence below: Pull > Create > Attach > \_\_\_\_\_\_\_\_\_ Which of the following comes next when launching a container? Fill in the blank. A. Detach B. Network connect C. Execute D. Die

ChatGPT

C. Execute

The sequence is:

Pull > Create > Attach > Execute

When launching a container, after pulling the necessary image (if not already available locally), creating the container, and attaching it to the terminal, you can execute commands or applications inside the container using the **docker exec** command.

For example, to execute a shell (like **/bin/bash**) inside the container, you can use the following command:

bashCopy code

docker exec -it <container\_id\_or\_name> /bin/bash

This will start an interactive shell session inside the running container, allowing you to interact with the container's filesystem and run commands as if you were working directly within the container.



Question 3.6 Which of the following commands can be used to connect to a running container to execute commands (similar to SSH connection)? A. docker exec -it <container\_name> B. docker exec -it <container\_name> bash C. docker run -it <conatainer\_name> D. None of the above

ChatGPT

B. docker exec -it <container\_name> bash

Option B is the correct command to connect to a running container and execute commands similar to an SSH connection.

The **docker exec** command is used to execute a command inside a running container. The **-it** flag allows for an interactive session, and **<container\_name>** is the name or ID of the running container.

For example, to connect to a running container named "my\_container" and start an interactive shell (like **/bin/bash**) inside the container, you would use the following command:

bashCopy code

docker exec -it my\_container bash

This command opens an interactive session within the container, allowing you to run commands, navigate the container's file system, and perform various tasks as if you were logged in through SSH.



Question 3.7 Which of the following statements is true? A. Docker images and containers are the same concept B. Docker containers can run without images C. Docker images are the pre-baked templates for running Docker containers D. Docker containers are considered prerequisites to run the images

ChatGPT

C. Docker images are the pre-baked templates for running Docker containers.

Option C is true. Docker images are pre-built templates or blueprints that contain all the necessary files, dependencies, and configurations needed to run a container. Images are like snapshots of a container's file system and runtime environment. They serve as the basis for creating and running containers.

Option A is not true. Docker images and containers are related concepts, but they are not the same. As mentioned earlier, images are templates, while containers are instances of running containers created from those images.

Option B is not true. Docker containers cannot run without images. Containers are created based on existing images. When you run a container, Docker creates an instance of that image and runs the application or service defined in the image.

Option D is not true. Docker containers are not prerequisites to run images. It is the other way around. Docker images are the prerequisites for running containers. Containers are the runnable instances of images, and they are created based on the specifications defined in the images.



Question 3.8 Which of the following options should be used to attach and detach from the container's process as well as to interact with it? A. -i B. -q C. -d D. -p

ChatGPT

A. -i

The **-i** option should be used to attach and interact with a container's process.

When you run a container with the **-i** flag (interactive mode), it allows you to interact with the container's standard input (stdin). This means that you can provide input to the container's process and also view the output on your terminal.

For example, to attach to an existing container in interactive mode, you would use the following command:

arduinoCopy code

docker attach -i <container\_id\_or\_name>

To detach from the container's process without stopping it, you can use the escape sequence **Ctrl + P + Q**.

So, using the **-i** option allows you to both attach and detach from a container's process, making it the correct option for interacting with the container. The other options have different purposes:

* **-q** is used to print the container ID, and it is commonly used to obtain the container ID for scripting purposes.
* **-d** is used to run a container in detached mode (background) and is used when you don't need an interactive session.
* **-p** is used to publish ports from the container to the host, allowing network access to the container's services.



Question 3.9 What does the docker container attach command do? A. It prints the log of the container B. It attaches to the shell inside the container C. It attaches to the process running inside the container D. It gives resource utilization data

ChatGPT

C. It attaches to the process running inside the container.

Option C is correct. The **docker container attach** command is used to attach to the main process running inside a container. When you attach to a container, you connect to its main process's standard input (stdin), standard output (stdout), and standard error (stderr). This allows you to interact with the container's process, see its output, and provide input to it.

For example, if you have a container running a web server, using **docker container attach <container\_id\_or\_name>**, you can attach to the server's process and see the log output directly in your terminal as the web server processes requests and responds.

Keep in mind that when you attach to a container, you are directly connecting to the process's input and output streams. If you exit the attached terminal session, the container will stop because the main process is also stopped.

If you want to interact with a shell inside the container, you would typically use the **docker exec** command with the **-it** options, as mentioned in a previous question. This provides you with an interactive shell session within the running container.

# Question 3.1

When you launch a container and use **-p 8000**, it will map 8000 port on the Docker host to the port defined in the image with **EXPOSE xx** as a metadata. True or False?

Incorrect Answer

* **A.** True

Your Answer:

Incorrect

* **B.** False

Correct Answer

# Question 3.2

Which of the following options is used to continuously update logs of a running container?

Correct Answer

* **A.** -c
* **B.** -f

Your Answer:

Correct

* **C.** -follow
* **D.** -continue

# Question 3.3

Kalam has launched a container with an image specification "schoolofdevops/testapp". Can you identify the tag that Kalam used to launch the container with?

Incorrect Answer

* **A.** testapp

Your Answer:

Incorrect

* **B.** schoolofdevops
* **C.** latest

Correct Answer

* **D.** v1.0.1

# Question 3.4

The **docker stop** command wipes out all the changes you have made since launching the container. True or False?

Correct Answer

* **A.** True
* **B.** False

Your Answer:

Correct

# Question 3.5

Take a look at the sequence below:

Pull > Create > Attach > \_\_\_\_\_\_\_\_\_

Which of the following comes next when launching a container? Fill in the blank.

Correct Answer

* **A.** Detach
* **B.** Network connect
* **C.** Execute

Your Answer:

Correct

* **D.** Die

# Question 3.6

Which of the following commands can be used to connect to a running container to execute commands (similar to SSH connection)?

Correct Answer

* **A.** docker exec -it <container\_name>
* **B.** docker exec -it <container\_name> bash

Your Answer:

Correct

* **C.** docker run -it <conatainer\_name>
* **D.** None of the above

# Question 3.7

Which of the following statements is true?

Correct Answer

* **A.** Docker images and containers are the same concept
* **B.** Docker containers can run without images
* **C.** Docker images are the pre-baked templates for running Docker containers

Your Answer:

Correct

* **D.** Docker containers are considered prerequisites to run the images

# Question 3.8

Which of the following options should be used to attach and detach from the container's process as well as to interact with it?

Correct Answer

* **A.** -i

Your Answer:

Correct

* **B.** -q
* **C.** -d
* **D.** -p

# Question 3.9

What does the **docker container attach** command do?

Incorrect Answer

* **A.** It prints the log of the container
* **B.** It attaches to the shell inside the container

Your Answer:

Incorrect

* **C.** It attaches to the process running inside the container

Correct Answer

* **D.** It gives resource utilization data