Docker Course

Docker is a platform for developing, shipping, and running applications in containers. Containers allow you to package an application and its dependencies together, ensuring consistency across different environments. Docker provides tools and a runtime for creating and managing containers, making it easier to deploy and scale applications.

To install Docker, you can follow these general steps:

1. Linux:
- For Ubuntu, you can use:

```
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

1.
- For CentOS, you can use:

sudo yum install docker-ce docker-ce-cli containerd.io

2. macOS:
   - Install Docker Desktop for Mac from the official Docker website.
3. Windows:
   - Install Docker Desktop for Windows from the official Docker website.

To create a Docker network, you can use the following command:

docker network create <network_name>

Replace <network_name> with the desired name for your Docker network. Here's an example:

docker network create mynetwork

This will create a new Docker network named "mynetwork."

To create a Docker container, you use the docker run command. Here's a basic example:

docker run <options> <image_name>

Replace <options> with any additional configuration you need and <image_name> with the name of the Docker image you want to use. For instance:

docker run -d --name mycontainer nginx

This command creates a detached (background) container named "mycontainer" using the official NGINX image. Adjust the options and image name based on your requirements.

Remember that Docker will pull the specified image from Docker Hub if it's not available locally.

Docker Commands

1. Build an image:

docker build -t <image_name>
<path_to_Dockerfile>

2. Run a container:

docker run <options> <image_name>

3. List running containers:

docker ps

4. List all containers (including stopped ones):

docker ps -a

5. Stop a running container:

docker stop <container_id or container_name>

6. Remove a container:

docker rm <container_id or container_name>

7. List Docker images:

docker images

8. Remove a Docker image:

docker rmi <image_name or image_id>

9. Inspect a container:

docker inspect <container_id or container_name>

10. View container logs:

docker logs <container_id or container_name>

11. Pull an image from Docker Hub:

docker pull <image_name>

12. Execute a command in a running container:

docker exec <options> <container_id or container_name> <command>

13. Attach to a running container:

docker attach <container_id or container_name>

14. Create a Docker network:

docker network create <network_name>

15. List Docker networks:

docker network ls

16. Inspect a Docker network:

docker network inspect <network_name>

17. Remove a Docker network:

docker network rm <network_name>

18. Build a Docker image with a specified Dockerfile:

docker build -f <Dockerfile_path> -t <image_name> <build_context>

19. Prune unused Docker objects (containers, networks, images):

docker system prune

20. Show Docker disk usage:

docker system df

21. Run a container with port mapping:

docker run -p <host_port>:<container_port> <image_name>

22. Run a container with volume mapping:

docker run -v <host_path>:<container_path> <image_name>

23. Inspect Docker volumes:

docker volume inspect <volume_name>

24. List Docker volumes:

docker volume ls

25. Prune unused Docker volumes:

```
docker volume prune
```

26. Create a named Docker volume:

```
docker volume create <volume_name>
```

27. Pause a running container:

```
docker pause <container_id or container_name>
```

28. Unpause a paused container:

```
docker unpause <container_id or container_name>
```

29. Inspect Docker events:

```
docker events
```

30. Tag a Docker image with a repository name:

docker tag <image_id or image_name> <repository_name>:<tag>

31. Push a Docker image to a registry:

docker push <repository_name>:<tag>

32. Login to a Docker registry:

docker login <registry_url>

33. Inspect Docker config:

docker config inspect <config_name>

34. List Docker configs:

```
docker config ls
```

35. Remove a Docker config:

```
docker config rm <config_name>
```

36. Create a Docker secret:

```
echo "<secret_data>" | docker secret create
<secret_name> -
```

37. List Docker secrets:

```
docker secret ls
```

38. Inspect Docker secret:

```
docker secret inspect <secret_name>
```

39. Remove a Docker secret:

docker secret rm <secret_name>

40. Run a container in the background and print container ID:

docker run -d <image_name>

41. Run a container with environment variables:

docker run -e KEY=VALUE <image_name>

42. Run a container with a specified user:

docker run --user <username or UID> <image_name>

43. Run a container with resource constraints (CPU and memory):

docker run --cpu-shares=<value> --memory=<value> <image_name>

44. Inspect Docker image layers:

docker history <image_name>

45. Show live resource usage statistics for a container:

docker stats <container_id or container_name>

46. Attach to a running container and open a shell:

docker exec -it <container_id or container_name> sh

47. Build and run a Docker container in one command:

```
docker build -t <image_name>
<path_to_Dockerfile> && docker run
<image_name>
```

48. Run a container with a specific name:

```
docker run --name <container_name>
<image_name>
```

49. Copy files or directories between a container and the local filesystem:

```
docker cp <container_id or
container_name>:<source_path>
<destination_path>
```

50. Set up a Docker Compose file and run services:

```
docker-compose up -d
```

51. Scale services defined in a Docker Compose file:

docker-compose scale <service_name>=<number_of_instances>

52. Pause all processes in all containers:

docker pause $(docker ps -q)

53. Unpause all processes in all containers:

docker unpause $(docker ps -q)

54. Inspect Docker image metadata:

docker image inspect <image_name>

55. Export a container’s file system as a tar archive:

docker export <container_id or container_name> > exported_container.tar

56. Import a container from a tarball archive:

cat exported_container.tar | docker import - <image_name>

57. Run a command in a new container and remove it after execution:

docker run --rm <image_name> <command>

58. Run a container with a specific network:

docker run --network <network_name> <image_name>

59. Run a container with a specific DNS server:

docker run --dns <dns_server> <image_name>

60. Run a container with a specific hostname:

docker run --hostname <custom_hostname> <image_name>

61. Run a container with a specific IP address:

docker run --ip <ip_address> <image_name>

62. Run a container with a custom MAC address:

docker run --mac-address <mac_address> <image_name>

63. Run a container with a specific user and group:

docker run --user <user>:<group> <image_name>

64. Run a container with read-only file system:

docker run --read-only <image_name>

65. Run a container in interactive mode:

docker run -it <image_name>

66. Mount a temporary filesystem inside a container:

docker run --tmpfs <mount_point> <image_name>

67. Create a Docker image from a running container:

docker commit <container_id or container_name> <new_image_name>

68. Export a Docker image as a tarball:

docker save <image_name> > image.tar

69. Import a Docker image from a tarball:

docker load < image.tar

70. Run a container with a custom entrypoint:

docker run --entrypoint <custom_entrypoint> <image_name>

71. Run a container with environment variables from a file:

docker run --env-file <env_file> <image_name>

72. Run a container with resource limits using Docker Compose:

```
docker-compose up --scale
<service_name>=<number_of_instances>
```

73. List containers sorted by creation time:

```
docker ps --format "table
{{.ID}}\t{{.Names}}\t{{.CreatedAt}}" --sort=created
```

74. Run a container with CPU affinity:

```
docker run --cpuset-cpus <cpu_set>
<image_name>
```

75. Run a container with custom ulimit settings:

```
docker run --ulimit <ulimit_option>
<image_name>
```

76. Run a container with custom security options:

docker run --security-opt <security_option> <image_name>

77. Run a container with a specific AppArmor profile:

docker run --security-opt apparmor=<profile_name> <image_name>

78. Run a container with a specific Seccomp profile:

docker run --security-opt seccomp=<profile_path> <image_name>

79. Run a container with custom capabilities:

docker run --cap-add <capability> <image_name>

80. Run a container with capabilities dropped:

docker run --cap-drop <capability> <image_name>

81. Run a container with a specific cgroup parent:

docker run --cgroup-parent <cgroup_parent> <image_name>

82. Run a container with a specific device:

docker run --device <device> <image_name>

83. Run a container with custom health check options:

docker run --health-cmd=<command> --health-interval=<interval> --health-retries=<retries> <image_name>

84. Run a container with a specific isolation technology:

docker run --isolation <isolation_type> <image_name>

85. Run a container with a specific log driver:

docker run --log-driver=<log_driver> <image_name>

86. Run a container with a specific log driver options:

docker run --log-opt <option>=<value> <image_name>

87. Run a container with custom user namespace options:

docker run --userns <user_namespace_option> <image_name>

88. Run a container with custom networking mode:

docker run --network <network_mode> <image_name>

89. Run a container with custom labels:

docker run --label <key>=<value> <image_name>

90. Run a container with a specific platform architecture:

docker run --platform <platform_architecture> <image_name>

91. Run a container with a specific storage driver:

docker run --storage-driver <storage_driver> <image_name>

92. Run a container with a specific storage option:

docker run --storage-opt <option>=<value> <image_name>

93. Run a container with a specific restart policy:

docker run --restart <restart_policy> <image_name>

94. Run a container with a specific health check interval:

docker run --health-interval <interval> <image_name>

95. Run a container with a specific health check timeout:

docker run --health-timeout <timeout> <image_name>

96. Run a container with a specific health check start period:

docker run --health-start-period <start_period> <image_name>

97. Run a container with a specific health check retries:

docker run --health-retries <retries> <image_name>

98. Run a container with a specific health check command:

docker run --health-cmd <command> <image_name>

99. Run a container with a specific log level:

docker run --log-level <log_level> <image_name>

100. Run a container with a specific environment variable file:

bash docker run --env-file <env_file> <image_name>

101. Run a container with read-only root filesystem:

bash docker run --read-only <image_name>

102. Run a container with a specific CPU set and memory limit:

bash docker run --cpus=<value> --memory=<value> <image_name>

103. Run a container with custom capabilities and privileged mode:

bash docker run --cap-add <capability> --privileged <image_name>

104. Run a container with a specific timezone:

bash docker run -e TZ=<timezone> <image_name>

105. Run a container with a specific kernel version:

bash docker run --kernel-memory=<value> <image_name>

106. Run a container with a specific network alias:

bash docker run --network-alias <alias> <image_name>

107. Run a container with a specific health check mode:

bash docker run --health-mode <mode> <image_name>

108. Run a container with a specific isolation technology and custom options:

bash docker run --isolation <isolation_type> --isolation-opt <option>=<value> <image_name>

109. Run a container with a specific user and group mapping:

bash docker run --userns <userns_option> <image_name>

110. Run a container with a specific device and device options:

bash docker run --device <device> --device-option <option>=<value> <image_name>

111. Run a container with a specific volume and volume options:

bash docker run -v <host_path>:<container_path>:<option> <image_name>

112. Run a container with a specific shm-size (shared memory size):

bash docker run --shm-size=<size> <image_name>

113. Run a container with a specific security profile:
bash docker run --security-opt <profile_name>=<profile_option> <image_name>

114. Run a container with a specific secret:

bash docker run --secret <secret_name> <image_name>

115. Run a container with a specific configuration:

bash docker run --config <config_name> <image_name>

116. Run a container with a specific namespace:

bash docker run --namespace <namespace> <image_name>

117. Run a container with a specific platform architecture and variant:

bash docker run --platform <platform_architecture>/<platform_variant> <image_name>

118. Run a container with a specific target build stage:

bash docker run --target <build_stage> <image_name>

119. Run a container with a specific networking driver:

bash docker run --network-driver <network_driver> <image_name>

120. Run a container with a specific DNS search domain:

bash docker run --dns-search <search_domain> <image_name>

121. Run a container with a specific DNS option:

bash docker run --dns-opt <option>=<value> <image_name>

122. Run a container with a specific host networking mode:

bash docker run --net host <image_name>

123. Run a container with a specific log format:

bash docker run --log-opt <log_format>=<format> <image_name>

124. Run a container with a specific log max-size:

bash docker run --log-opt max-size=<size> <image_name>

125. Run a container with a specific log max-file:

bash docker run --log-opt max-file=<number> <image_name>

126. Run a container with a specific log labels:

bash docker run --log-opt labels=<label1,label2> <image_name>

127. Run a container with a specific log env:

bash docker run --log-opt env=<env_var1,env_var2> <image_name>

128. Run a container with a specific log syslog facility:

bash docker run --log-opt syslog-facility=<facility> <image_name>

129. Run a container with a specific log syslog address:

bash docker run --log-opt syslog-address=<address> <image_name>

130. Run a container with a specific log syslog tag:

bash docker run --log-opt syslog-tag=<tag> <image_name>

131. Run a container with a specific log syslog priority:

bash docker run --log-opt syslog-priority=<priority> <image_name>

132. Run a container with a specific log syslog hostname:

bash docker run --log-opt syslog-hostname=<hostname> <image_name>