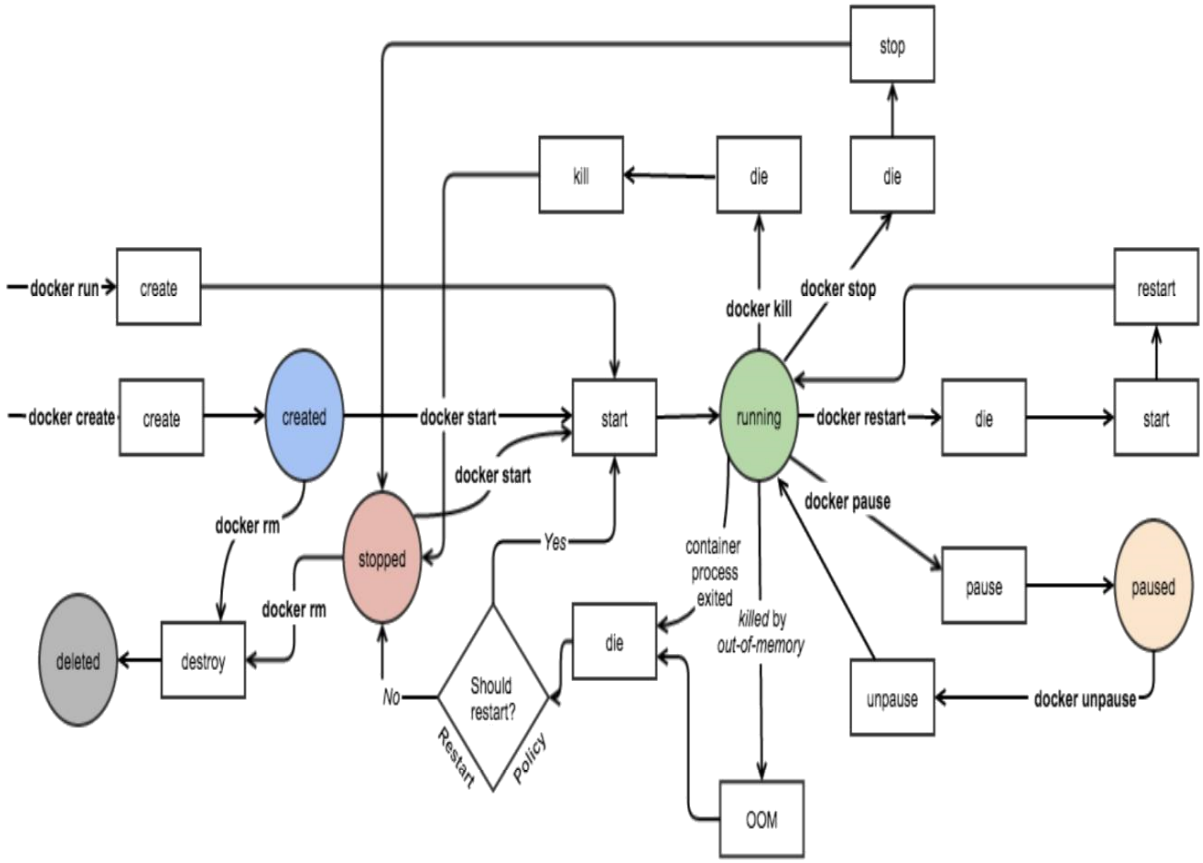
Lab: Managing Docker Containers

Introduction:

Docker provides the ability to **package** and **run an application** in an isolated environment called a **container**. Container management is a process for automating the creation, deployment and scaling of containers.

Container management facilitates the addition, replacement and organization of containers on a large scale.



In this Lab you will learn the below Commands.

Commands	Description
attach	Attach local standard input, output, and error streams to a running container
commit	Create a new image from a container's changes
cp	Copy files/folders between a container and the local filesystem
Create	Create a new container
diff	Inspect changes to files or directories on a container's filesystem
exec	Execute a command in a running container
export	Export a container's filesystem as a tar archive
inspect	Display detailed information on one or more containers
kill	Kill one or more running containers
logs	Fetch the logs of a container
ls	List containers
pause	Pause all processes within one or more containers
port	List port mappings or a specific mapping for the container
prune	Remove all stopped containers
rename	Rename a container
restart	Restart one or more containers
rm	Remove one or more containers
run	Create and run a new container from an image
start	Start one or more stopped containers
stats	Display a live stream of container(s) resource usage statistics
stop	Stop one or more running containers
top	Display the running processes of a container
unpause	Unpause all processes within one or more containers
upadte	Update configuration of one or more containers
wait	Block until one or more containers stop, then print their exit codes

1 Ensure that you have logged-in as **root** user with password as **linux** on the training lab environment.

1.1 Let's **list** the containers if any by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID    IMAGE     COMMAND    CREATED    STATUS     PORTS      NAMES
```

1.2 Let's **create** a nginx container by executing the below command.

```
# docker container create nginx
```

Output:

```
[root@docker-engine ~]#docker container create nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
01b5b2efb836: Pull complete
db354f722736: Pull complete
abb02e674be5: Pull complete
214be53c3027: Pull complete
a69afcef752d: Pull complete
625184acb94e: Pull complete
Digest: sha256:038c50b33894118f98670e5905857d20404d3847b64e780e4a63cba30c5e90a0
Status: Downloaded newer image for nginx:latest
62fa561c0589652da9c24aabfad5d8aa788c1d2bdb3469e6cd5e81e9edc3da03
```

1.3 Executing the below command to list container created from the above step.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS    PORTS     NAMES
62fa561c0589   nginx     "/docker-entrypoint.…"   3 minutes ago   Created             cranky_zhukovsky
```

Note: Docker by default, assigns a random name comprised of two dictionary words separated by a '_'.

1.4 Let's **create** a container and **name** it as web01 by executing the below command.

```
# docker container create --name web01 nginx
```

Output:

```
[root@docker-engine ~]#docker container create --name web01 nginx
4e4322ecc08c0ca04a91a601ff35c84aba3e12449657c5042373373656f55c15
```

Note: As the image was already pulled in the previous steps, it did not pull an image.

1.5 Let's **list** the container created by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS    PORTS     NAMES
4e4322ecc08c   nginx     "/docker-entrypoint.…"   4 minutes ago    Created             web01
62fa561c0589   nginx     "/docker-entrypoint.…"   9 minutes ago    Created             cranky_zhukovsky
```

1.6 Let's **start** the container which is created by executing the below command.

```
# docker container start web01
```

Output:

```
[root@docker-engine ~]#docker container start web01
web01
```

1.7 Let's **list** the container created by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS              PORTS     NAMES
4e4322ecc08c   nginx     "/docker-entrypoint.…"   7 minutes ago    Up About a minute   80/tcp    web01
```

1.8 Let's **create** a new container using **run** option by executing the below command.

```
# docker container run --name server01 centos
```

Output:

```
[root@docker-engine ~]# docker container run --name server01 centos
Unable to find image 'centos:latest' locally
latest: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:latest
```

Note: docker container run command is the combination of docker create + docker start.

1.9 Let's **list** the containers (both Running & Exited) by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
dbcb765c162b   centos    "/bin/bash"              3 minutes ago    Exited (0) 3 minutes ago              server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   14 minutes ago   Up 8 minutes                80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   19 minutes ago   Created                               cranky_zhukovsky
```

Note: The container is in **exited** status as containers by default run to completion.

1.10 Let's create a new container by supplying (**-i and -t**) option to get interactive terminal by executing the below command.

Note: "-i" stands for interactive & "-t" stands for terminal. So, we are requesting an interactive terminal for a container.

```
# docker container run --name server02 -i -t centos
```

Output:

```
[root@docker-engine ~]#docker container run --name server02 -i -t centos
[root@071cf1f110b5 /]#
```

```
# ps
```

Output:

```
[root@071cf1f110b5 /]# ps
    PID TTY          TIME CMD
      1 pts/0    00:00:00 bash
     15 pts/0    00:00:00 ps
```

```
# exit
```

Output:

```
[root@071cf1f110b5 /]# exit
exit
```

Note: The container is created and interactive terminal is displayed. You can run commands inside the container and exit out.

1.11 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
071cf1f110b5   centos    "/bin/bash"              4 minutes ago    Exited (0) 2 minutes ago              server02
dbcb765c162b   centos    "/bin/bash"              20 minutes ago   Exited (0) 20 minutes ago             server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   31 minutes ago   Up 26 minutes               80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   36 minutes ago   Created                               cranky_zhukovsky
```

1.12 Let's create a new container with **detached interactive terminal** (**-d -i -t**) option, by executing the below command.

Note: d stand for detached mode.

```
# docker container run --name server03 -dit centos
```

Output:

```
[root@docker-engine ~]#docker container run --name server03 -dit centos
81290f888bbf16ff8a81ee3a3ee8c2f0c7b1f7668b0524865bf9c5929eccd4cd
```

Note: The container was created in background as we had mentioned -d (detach) option.

1.13 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS                      PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              6 minutes ago    Up 6 minutes                          server03
071cf1f110b5   centos    "/bin/bash"              15 minutes ago   Exited (0) 12 minutes ago             server02
dbcb765c162b   centos    "/bin/bash"              31 minutes ago   Exited (0) 31 minutes ago             server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   42 minutes ago   Up 36 minutes               80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   47 minutes ago   Created                               cranky_zhukovsky
```

1.14 Let's **attach** the container server03 by executing the below command.

```
# docker container attach server03
```

Output:

```
[root@docker-engine ~]#docker container attach server03
[root@81290f888bbf /]#
```

```
# ps
```

Output:

```
[root@81290f888bbf /]# ps
    PID TTY          TIME CMD
      1 pts/0    00:00:00 bash
     15 pts/0    00:00:00 ps
[root@81290f888bbf /]#
```

Note: Press ctrl+pq (To come out without exiting the container)

Output:

```
[root@81290f888bbf /]#
[root@81290f888bbf /]# read escape sequence
```

1.15 Let's list the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              24 minutes ago   Up 24 minutes             server03
4e4322ecc08c   nginx     "/docker-entrypoint.…"   59 minutes ago   Up 53 minutes   80/tcp    web01
```

Note: The container continues to run as we had used ctrl+pq to exit from the container.

1.16 Let's run a command inside the container without attaching to the container by using **exec** option.

```
# docker container exec server03 cat /etc/resolv.conf
```

Output:

```
[root@docker-engine ~]#docker container exec server03 cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 192.168.100.2
```

1.17 Let's **rename** the existing container by executing the below command.

```
# docker container rename server03 server003
```

1.18 Let's **list** the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              29 minutes ago      Up 29 minutes             server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up 58 minutes   80/tcp    web01
```

1.19 Let's **pause** the existing container by executing the below command.

```
# docker container pause server003
```

Output:

```
[root@docker-engine ~]#docker container pause server003
server003
```

1.20 Let's **list** the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS                      PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              32 minutes ago      Up 32 minutes (Paused)                server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour            80/tcp    web01
```

1.21 Let's **unpause** the existing container by executing the below command.

```
# docker container unpause server003
```

Output:

```
[root@docker-engine ~]#docker container unpause server003
server003
```

1.22 Let's **list** the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS              PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              35 minutes ago      Up 35 minutes               server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour    80/tcp    web01
```

1.23 Let's **stop** the existing container by executing the below command.

```
# docker container stop server003
```

Output:

```
[root@docker-engine ~]#docker container stop server003
server003
```

1.24 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS                          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              38 minutes ago      Exited (0) About a minute ago             server003
071cf1f110b5   centos    "/bin/bash"              47 minutes ago      Exited (0) 44 minutes ago                 server02
dbcb765c162b   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago              server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour                80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   About an hour ago   Created                                   cranky_zhukovsky
```

1.25 Let's **start** the existing container by executing the below command.

```
# docker container start server003
```

Output:

```
[root@docker-engine ~]#docker container start server003
server003
```

1.26 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS                          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              42 minutes ago      Up About a minute                         server003
071cf1f110b5   centos    "/bin/bash"              50 minutes ago      Exited (0) 48 minutes ago                 server02
dbcb765c162b   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago              server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour                80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   About an hour ago   Created                                   cranky_zhukovsky
```

1.27 Let's **kill** the container by executing the below command.

```
# docker container kill server003
```

Output:

```
[root@docker-engine ~]#docker container kill server003
server003
```

1.28 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS                          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              45 minutes ago      Exited (137) 2 minutes ago                server003
071cf1f110b5   centos    "/bin/bash"              53 minutes ago      Exited (0) 51 minutes ago                 server02
dbcb765c162b   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago              server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour                80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   About an hour ago   Created                                   cranky_zhukovsky
```

1.29 Let's **restart** the existing container by executing the below command.

```
# docker container restart server003
```

Output:

```
[root@docker-engine ~]#docker container restart server003
server003
```

1.30 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS                          PORTS     NAMES
81290f888bbf   centos    "/bin/bash"              49 minutes ago      Up 2 minutes                              server003
071cf1f110b5   centos    "/bin/bash"              58 minutes ago      Exited (0) 55 minutes ago                 server02
dbcb765c162b   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago              server01
4e4322ecc08c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour                80/tcp    web01
62fa561c0589   nginx     "/docker-entrypoint.…"   About an hour ago   Created                                   cranky_zhukovsky
```

Note: We can also use "start" instead of restart to start the container.

1.31 Let's expose the **custom port** to the container by executing the below command.

```
# docker container run --name web02 -dit -p 8080:80 nginx
```

Output:

```
[root@docker-engine ~]#docker container run --name web02 -dit -p 8080:80 nginx
8812ce7b497e76e0bb16ea75b5e57ffc8c4107e7bd44fe0e5c25ba39caa763aa
```

1.32 Let's **list** the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                                   NAMES
26afed3fcddb   nginx     "/docker-entrypoint.…"   10 seconds ago   Up 9 seconds    0.0.0.0:8080->80/tcp, :::8080->80/tcp   web02
81290f888bbf   centos    "/bin/bash"              57 minutes ago   Up 10 minutes                                           server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   2 hours ago      Up About an hour   80/tcp                               web01
```

Note: The container port is 80 and same is exposed to 8080 ports on the host.

1.33 Let's expose the **random host port** to the container by executing the below command.

```
# docker container run --name web03 -dit -P nginx
```

Output:

```
[root@docker-engine ~]#docker container run --name web03 -dit -P nginx
9429831d734f401beb9d8fbce936e57e6fc5199149ab93a2a17a52213bc0efd6
```

Note: The host port default dynamic port range is 32768-60999

Reference – cat /proc/sys/net/ipv4/ip_local_port_range

1.34 Let's **list** the container by executing the below command.

```
# docker container ls
```

Output:

```
[root@docker-engine ~]#docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS              PORTS                                   NAMES
9429831d734f   nginx     "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:32768->80/tcp, :::32768->80/tcp   web03
26afed3fcddb   nginx     "/docker-entrypoint.…"   3 minutes ago        Up 3 minutes        0.0.0.0:8080->80/tcp, :::8080->80/tcp     web02
81290f888bbf   centos    "/bin/bash"              About an hour ago    Up 13 minutes                                                 server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   2 hours ago          Up 2 hours          80/tcp                                  web01
```

1.35 Let's **inspect** the container by executing the below command.

```
# docker container inspect web02 | grep -e "HostPort" -e
"IPAddress" | uniq
```

Output:

```
[root@docker-engine ~]#docker container inspect web02 | grep -e "HostPort" -e "IPAddress" | uniq
                    "HostPort": "8080"
            "SecondaryIPAddresses": null,
            "IPAddress": "172.17.0.2",
                    "IPAddress": "172.17.0.2",
```

1.36 Let's **access** the webserver by using the container ip by executing the below command.

```
# curl 172.17.0.2
```

Output:

```
[root@docker-engine ~]#curl 172.17.0.2
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

1.37 Let's **access** the webserver by using the docker host ip and port exposed by executing the below command.

```
# curl 192.168.100.10:8080
```

Note: Change the IP address and port number based on your configuration ip a s eth0

Output:

```
[root@docker-engine ~]#curl 192.168.100.10:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

1.38 Let's **access** the webserver by using docker host ip and port exposed by executing the below command.

Note: Change the **IP address and port number** based on your configuration.

```
# curl 192.168.100.10:32768
```

Output:

```
[root@docker-engine ~]#curl 192.168.100.10:32768
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

1.39 Let's **create** an **index.html** file by executing the below command.

```
# cat > index.html << EOF
Welcome to Docker Training!!
EOF
```

1.40 Let's **copy** files between local filesystem and a container by executing the below command. ex: we are copying the above created file inside a web02 container.

```
# docker container cp index.html \
 web02:/usr/share/nginx/html/index.html
```

Output:

```
[root@docker-engine ~]#docker container cp index.html web02:/usr/share/nginx/html/index.html
Preparing to copy...
Copying to container - 0B
Copying to container - 0B
Copying to container - 512B
Copying to container - 541B
Copying to container - 1.024kB
Copying to container - 1.536kB
Copying to container - 2.048kB
Successfully copied 2.048kB to web02:/usr/share/nginx/html/index.html
```

1.41 Let's **access** the webserver by using the container IP by executing the below command.

```
# curl 172.17.0.2
```

Output:

```
[root@docker-engine ~]#curl 172.17.0.2
Welcome to Docker Training!!
```

1.42 Let's **update** memory configuration of the container first let us **inspect** the default memory limit set by executing the below command.

```
# docker container inspect web02 | grep Memory | head -1
```

Output:

```
[root@docker-engine ~]#docker container inspect web02 | grep Memory | head -1
            "Memory": 0,
```

1.43 Let's **update** the **memory** and memory-swap by executing the below command.

```
# docker container update --memory 200M --memory-swap 200M
web02
```

Output:

```
[root@docker-engine ~]#docker container update --memory 200M --memory-swap 200M web02
web02
```

`--memory-swap` is a modifier flag that only has meaning if --memory is also set. Using swap allows the container to write excess memory requirements to the disk, when the container has exhausted all the RAM that is available to it.

1.44 Let's **inspect** the container and validate if the values are updated correctly by executing the below command.

```
# docker container inspect web02 | grep Memory | head -1
```

Output:

```
[root@docker-engine ~]#docker container inspect web02 | grep Memory | head -1
            "Memory": 209715200,
```

1.45 Let's check the **size** of the container by executing the below command.

```
# docker container ls -sa
```

Output:

```
[root@docker-engine ~]#docker container ls -sa
CONTAINER ID   IMAGE     COMMAND                  CREATED             STATUS
22af37518dbf   nginx     "/docker-entrypoint.…"   6 minutes ago       Up 6 minutes
04304f7081d6   nginx     "/docker-entrypoint.…"   7 minutes ago       Up 7 minutes
6daf8f7fe012   centos    "/bin/bash"              About an hour ago   Up 34 minutes
a35b1102c495   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago
e5de40974369   centos    "/bin/bash"              About an hour ago   Exited (0) About an hour ago
8937920bcc2c   nginx     "/docker-entrypoint.…"   About an hour ago   Up About an hour
f91aee8f7e0a   nginx     "/docker-entrypoint.…"   2 hours ago         Created

 PORTS                                       NAMES              SIZE
 0.0.0.0:32768->80/tcp, :::32768->80/tcp     web03              1.09kB (virtual 142MB)
 0.0.0.0:8080->80/tcp, :::8080->80/tcp       web02              1.12kB (virtual 142MB)
                                             server003          3B (virtual 231MB)
                                             server02           8B (virtual 231MB)
                                             server01           0B (virtual 231MB)
 80/tcp                                      web01              1.09kB (virtual 142MB)
                                             great_goldwasser   0B (virtual 142MB)
```

1.46 Let's verify the **top running process** inside the container by executing the below command.

```
# docker container top server003
```

Output:

```
[root@docker-engine ~]#docker container top server003
UID                 PID                 PPID                C                   STIME               TTY
root                94485               94464               0                   18:30               pts/0
```

1.47 Let's verify the **stats** of the running containers by executing the below command.

```
# docker container stats
```

Output:

```
CONTAINER ID   NAME        CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
9429831d734f   web03       0.00%     4.441MiB / 3.55GiB    0.12%     3.33kB / 1.27kB   8.19kB / 36.9kB   5
26afed3fcddb   web02       0.00%     4.527MiB / 200MiB     2.26%     4.55kB / 3.24kB   8.19kB / 49.2kB   5
81290f888bbf   server003   0.00%     1.34MiB / 3.55GiB     0.04%     2.78kB / 0B       0B / 0B           1
4e4322ecc08c   web01       0.00%     4.496MiB / 3.55GiB    0.12%     6.07kB / 0B       8.19kB / 36.9kB   5
```

Note: Press **ctrl + c** to exit from the output screen or run the below command to disable streaming stats.

```
# docker container stats --no-stream
```

Output:

```
[root@docker-engine ~]#docker container stats --no-stream
CONTAINER ID   NAME        CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
9429831d734f   web03       0.00%     4.441MiB / 3.55GiB    0.12%     3.33kB / 1.27kB   8.19kB / 36.9kB   5
26afed3fcddb   web02       0.00%     4.527MiB / 200MiB     2.26%     4.55kB / 3.24kB   8.19kB / 49.2kB   5
81290f888bbf   server003   0.00%     1.34MiB / 3.55GiB     0.04%     2.78kB / 0B       0B / 0B           1
4e4322ecc08c   web01       0.00%     4.496MiB / 3.55GiB    0.12%     6.07kB / 0B       8.19kB / 36.9kB   5
```

1.48 Let's verify the **logs** of the containers by executing the below command.

```
# docker container logs server003 --timestamps
```

Output:

```
[root@docker-engine ~]#docker container logs server003 --timestamps
[root@81290f888bbf /]#
2023-02-04T12:26:04.299947309Z [root@81290f888bbf /]#
2023-02-04T12:26:27.271871033Z [root@81290f888bbf /]# ps
2023-02-04T12:26:27.278251920Z     PID TTY          TIME CMD
2023-02-04T12:26:27.278283899Z       1 pts/0    00:00:00 bash
2023-02-04T12:26:27.278500909Z      15 pts/0    00:00:00 ps
2023-02-04T12:26:28.575624721Z [root@81290f888bbf /]#
2023-02-04T12:50:21.456425314Z [root@81290f888bbf /]# exit
2023-02-04T12:56:55.728938020Z [root@81290f888bbf /]# [root@docker-engine ~]#
```

1.49 Let's **prune** all the stopped containers by executing the below command.

```
# docker container prune -f
```

Output:

```
[root@docker-engine ~]#docker container prune -f
Deleted Containers:
071cf1f110b50f994daed8a6b9ac06373acc06df70df75ec7e9f1c02bb276eca
dbcb765c162b6099c7f292fc9fcb2a292426e09912a71528f063d365cc9aad6b
62fa561c0589652da9c24aabfad5d8aa788c1d2bdb3469e6cd5e81e9edc3da03

Total reclaimed space: 8B
```

1.50 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS             PORTS                                      NAMES
9429831d734f   nginx     "/docker-entrypoint.…"   55 minutes ago   Up 55 minutes      0.0.0.0:32768->80/tcp, :::32768->80/tcp    web03
26afed3fcddb   nginx     "/docker-entrypoint.…"   57 minutes ago   Up 57 minutes      0.0.0.0:8080->80/tcp, :::8080->80/tcp      web02
81290f888bbf   centos    "/bin/bash"              2 hours ago      Up About an hour                                              server003
4e4322ecc08c   nginx     "/docker-entrypoint.…"   3 hours ago      Up 2 hours         80/tcp                                     web01
```

1.51 Let's **remove** the server003 container gracefully by executing the below command.

```
# docker container stop server003
```

Output:

```
[root@docker-engine ~]#docker container stop server003
server003
```

```
# docker container rm server003
```

Output:

```
[root@docker-engine ~]#docker container rm server003
server003
```

1.52 Let's **remove** the running containers by force by executing the below command.

```
# docker container rm web01 web02 web03 -f
```

Output:

```
[root@docker-engine ~]#docker container rm web01 web02 web03 -f
web01
web02
web03
```

1.53 Let's **list** the container by executing the below command.

```
# docker container ls -a
```

Output:

```
[root@docker-engine ~]#docker container ls -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

1.54 Let's remove the downloaded images by executing the below command.

```
# docker image rm nginx centos
```

Output:

```
[root@docker-engine ~]#docker image rm nginx centos
Untagged: nginx:latest
Untagged: nginx@sha256:038c50b33894118f98670e5905857d20404d3847b64e780e4a63cba30c5e90a0
Deleted: sha256:9eee96112defa9d7b1880f18e76e174537491bcec6e31ad9a474cdea40f5abab
Deleted: sha256:9cadfb09eeeb8021f9bcda73607dc61ccff20c8fdc44af61267c788f174e9bdb
Deleted: sha256:f35b4ecd12ae382160d9a00f3b6956fec245c644e79e71d2bd0998403ea93a00
Deleted: sha256:0023ddac86d5e471a726b8b20d0422defc75eadbf85b3f98bfd02351f263bc57
Deleted: sha256:4bb459f6844bfd227980b8037560c81a5bf565fc81fd288052b84acbcf660b42
Deleted: sha256:0e16e930455b149d8ce0b75aaaa31b168c82d658527bfe184db954073457cb60
Deleted: sha256:bd2fe8b74db65d82ea10db97368d35b92998d4ea0e7e7dc819481fe4a68f64cf
Untagged: centos:latest
Untagged: centos@sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Deleted: sha256:5d0da3dc976460b72c77d94c8a1ad043720b0416bfc16c52c45d4847e53fadb6
Deleted: sha256:74ddd0ec08fa43d09f32636ba91a0a3053b02cb4627c35051aff89f853606b59
```

1.55 Let's **cleanup** by executing the below command.

```
# docker container rm `docker container ls -a -q` -f
```

```
# docker image rm `docker image ls -q` -f
```

Note: The above commands might error out indicating syntax error if there are no **containers** and **images** to **remove** which can be ignored.