Docker and Docker Compose

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. The service has both free and premium tiers.

Resources:

- https://www.youtube.com/watch?v=3c-iBn73dDE&t=1485s
- https://www.youtube.com/watch?v=WmcdMiyqfZs
- https://www.youtube.com/watch?v=EoY1i8lds1w
- https://www.youtube.com/watch?v=MVIcrmeV_6c
- https://www.youtube.com/watch?v=Qw9zlE3t8Ko
- https://www.youtube.com/playlist?list=PL4cUxeGkcC9hxjeEtdHFNYMtCpjNBm3h7

Links:

- https://docs.docker.com/
- https://docs.docker.com/compose/
- https://medium.com/@kmdkhadeer/docker-get-started-9aa7ee662cea
- https://towardsdatascience.com/docker-101-all-you-wanted-to-know-about-docker-2dd0cb476f03

Courses:

- https://kodekloud.com/courses/docker-for-the-absolute-beginner/?utm_source=google&utm_medium=&utm_id=16890563714&utm_content=&utm_term=&creativeId=

Getting Started:

- Docker overview
- Docker Installation
- Docker compose installation
- Docker Volumes
- Docker Networking

Certification:

- https://training.linuxfoundation.org/training/containers-fundamentals/

Docker and Docker Composer:

- Install docker and docker compose in your local machine linux
- Use docker cli to run a nginx server locally and expose it on port 8080
- Docker push an image to docker hub
- Run docker commands without sudo
- Learn all basic docker cli commands
- Write your first dockerfile to create a custom nginx server to output “hello World”
- Create a docker compose file which has 3 nginx services which outputs hello-world 1, 2, 3 respectively in a network.
- Attach docker volume and read the file dynamically in the container from outside
- Shell into a running container and execute basic commands
- Create 2 docker files of nginx with CMD and ENTRYPOINT respectively.
- Create a multi-stage build dockerfile for nginx

● Install docker and docker compose in your local machine linux:

Running Following Commands:

1. Update package list:

```
sudo apt update
```

2. Install dependencies:

```
sudo apt update
```

3. Add Docker's GPG key:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
add -
```

4. Add Docker repository:

```
sudo add-apt-repository "deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
```

5. Update package list again:

```
sudo apt update
```

6. Install Docker Engine:

```
sudo apt install -y docker-ce
```

7. Verify Docker installation:

```
sudo systemctl status docker
```

8. Add user to Docker group:

```
sudo usermod -aG docker $USER
```

9. Download Docker Compose:

```
sudo curl -L
"https://github.com/docker/compose/releases/latest/download/docker-compo
se-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

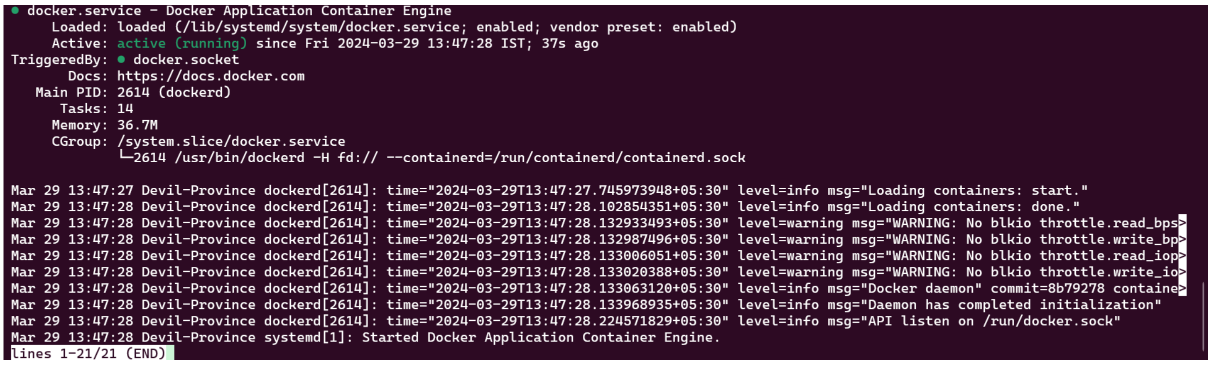
10. Set executable permissions for Docker Compose:

```
sudo chmod +x /usr/local/bin/docker-compose
```

11. Verify Docker Compose installation:

```
docker-compose --version
```

Docker Installed Successfully:

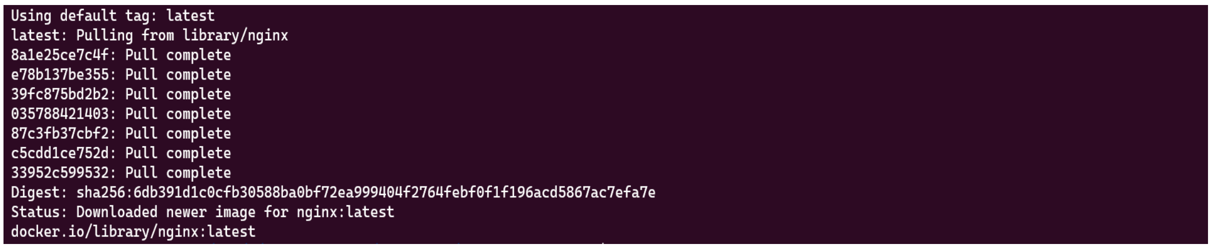


```
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2024-03-29 13:47:28 IST; 37s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2614 (dockerd)
      Tasks: 14
     Memory: 36.7M
     CGroup: /system.slice/docker.service
             └─2614 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Mar 29 13:47:27 Devil-Province dockerd[2614]: time="2024-03-29T13:47:27.745973948+05:30" level=info msg="Loading containers: start."
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.102854351+05:30" level=info msg="Loading containers: done."
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.132933493+05:30" level=warning msg="WARNING: No blkio throttle.read_bps>
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.132987496+05:30" level=warning msg="WARNING: No blkio throttle.write_bp>
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.133006051+05:30" level=warning msg="WARNING: No blkio throttle.read_iop>
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.133020388+05:30" level=warning msg="WARNING: No blkio throttle.write_io>
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.133063120+05:30" level=info msg="Docker daemon" commit=8b79278 containe>
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.133968935+05:30" level=info msg="Daemon has completed initialization"
Mar 29 13:47:28 Devil-Province dockerd[2614]: time="2024-03-29T13:47:28.224571829+05:30" level=info msg="API listen on /run/docker.sock"
Mar 29 13:47:28 Devil-Province systemd[1]: Started Docker Application Container Engine.
lines 1-21/21 (END)
```

● Use docker cli to run a nginx server locally and expose it on port 8080

1. Pull Nginx Image
   `Sudo docker pull nginx`
2. Run Nginx Container
   `docker run -d -p 8080:80 --name my-nginx nginx`
3. Verify Nginx Container Status:
   `docker ps`



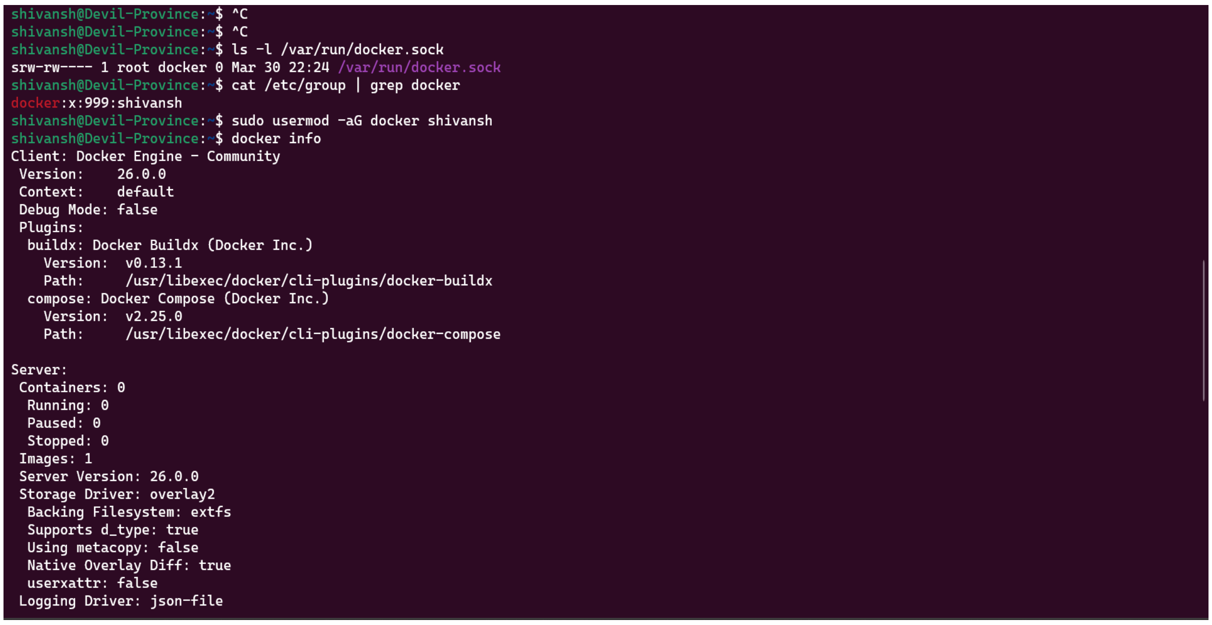
```
Using default tag: latest
latest: Pulling from library/nginx
8a1e25ce7c4f: Pull complete
e78b137be355: Pull complete
39fc875bd2b2: Pull complete
035788421403: Pull complete
87c3fb37cbf2: Pull complete
c5cdd1ce752d: Pull complete
33952c599532: Pull complete
Digest: sha256:6db391d1c0cfb30588ba0bf72ea999404f2764febf0f1f196acd5867ac7efa7e
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
```

● Docker push an image to docker hub

```
docker tag my-image:latest myusername/my-image:latest
docker login
docker push myusername/my-image:latest
```

● Run docker commands without sudo

```
cat /etc/group | grep docker
sudo usermod -aG docker <username>
logout
groups
docker info
```



```
shivansh@Devil-Province:~$ ^C
shivansh@Devil-Province:~$ ^C
shivansh@Devil-Province:~$ ls -l /var/run/docker.sock
srw-rw---- 1 root docker 0 Mar 30 22:24 /var/run/docker.sock
shivansh@Devil-Province:~$ cat /etc/group | grep docker
docker:x:999:shivansh
shivansh@Devil-Province:~$ sudo usermod -aG docker shivansh
shivansh@Devil-Province:~$ docker info
Client: Docker Engine - Community
 Version:    26.0.0
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.13.1
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.25.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 1
 Server Version: 26.0.0
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
```

- Learn all basic docker cli commands

```
# Display Docker version information
docker version

# Show detailed Docker system information
docker info

# List all Docker images
docker images

# List running containers
docker ps


# Pull a Docker image from a registry (e.g., Docker Hub)
docker pull <image-name>

# Create and start a new container based on an image
docker run <image-name>

# Create and start a container in detached mode (background)
docker run -d <image-name>

# Stop a running container
docker stop <container-id>

# Start a stopped container
docker start <container-id>
```

```
# Restart a container
docker restart <container-id>

# Remove a stopped container
docker rm <container-id>

# Remove a Docker image
docker rmi <image-id>

# Execute a command inside a running container in interactive mode
docker exec -it <container-id> <command>

# Display logs of a running container
docker logs <container-id>

# Build a Docker image from a Dockerfile and tag it
docker build -t <image-name> <path-to-dockerfile>

# Start services defined in a Docker Compose file
docker-compose up

# Stop and remove services defined in a Docker Compose file
docker-compose down
```

● Write your first dockerfile to create a custom nginx server to output “hello World”
Create Project Directory and Navigate:

```
mkdir custom-nginx
cd custom-nginx
```

Create Dockerfile:

```
cat > Dockerfile <<EOF
FROM nginx:latest
COPY nginx.conf /etc/nginx/nginx.conf
RUN mkdir -p /usr/share/nginx/html
COPY index.html /usr/share/nginx/html/index.html
EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
EOF
```

Create nginx.conf:

```
cat > nginx.conf <<EOF
worker_processes 1;
events {
    worker_connections 1024;
}
http {
    server {
        listen 80;
        server_name localhost;
        location / {
            root /usr/share/nginx/html;
            index index.html;
        }
    }
}
EOF
```

Create index.html:

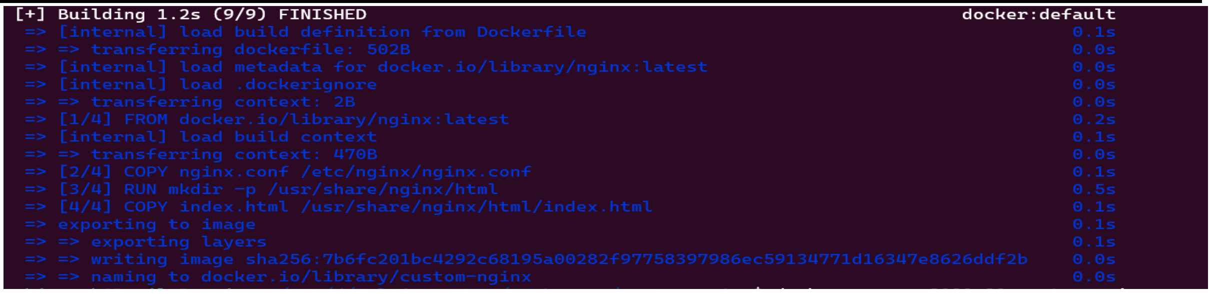
```
cat > index.html <<EOF
<!DOCTYPE html>
<html>
<head>
    <title>Hello, World!</title>
</head>
<body>
    <h1>Hello, World!</h1>
</body>
</html>
EOF
```

Build Docker Image:

```
docker build -t custom-nginx .
```

Run Docker Container:

```
docker run -p 8080:80 custom-nginx
```



```
[+] Building 1.2s (9/9) FINISHED                                         docker:default
 => [internal] load build definition from Dockerfile                              0.1s
 => => transferring dockerfile: 502B                                              0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                   0.0s
 => [internal] load .dockerignore                                                 0.0s
 => => transferring context: 2B                                                   0.0s
 => [1/4] FROM docker.io/library/nginx:latest                                     0.2s
 => [internal] load build context                                                 0.1s
 => => transferring context: 470B                                                 0.0s
 => [2/4] COPY nginx.conf /etc/nginx/nginx.conf                                   0.1s
 => [3/4] RUN mkdir -p /usr/share/nginx/html                                      0.5s
 => [4/4] COPY index.html /usr/share/nginx/html/index.html                        0.1s
 => exporting to image                                                            0.1s
 => => exporting layers                                                           0.1s
 => => writing image sha256:7b6fc201bc4292c68195a00282f97758397986ec59134771d16347e8626ddf2b    0.0s
 => => naming to docker.io/library/custom-nginx                                   0.0s
```

● Create a docker compose file which has 3 nginx services which outputs hello-world 1, 2, 3

```
respectively in a network.
cat > docker-compose.yml <<EOF
version: '3.8'
```

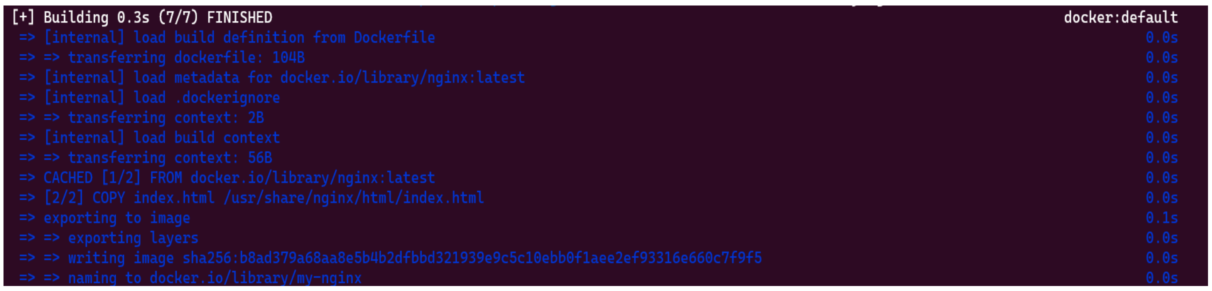
```
services:
  nginx1:
    image: nginx:latest
    ports:
      - "8081:80"
    environment:
      - MESSAGE=hello-world 1
    networks:
      - my-network

  nginx2:
    image: nginx:latest
    ports:
      - "8082:80"
    environment:
      - MESSAGE=hello-world 2
    networks:
      - my-network

  nginx3:
    image: nginx:latest
    ports:
      - "8083:80"
    environment:
      - MESSAGE=hello-world 3
    networks:
      - my-network

networks:
  my-network:
EOF
```

- Attach docker volume and read the file dynamically in the container from outside



```
[+] Building 0.3s (7/7) FINISHED                                                     docker:default
 => [internal] load build definition from Dockerfile                                           0.0s
 => => transferring dockerfile: 104B                                                           0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                0.0s
 => [internal] load .dockerignore                                                              0.0s
 => => transferring context: 2B                                                                0.0s
 => [internal] load build context                                                              0.0s
 => => transferring context: 56B                                                               0.0s
 => CACHED [1/2] FROM docker.io/library/nginx:latest                                           0.0s
 => [2/2] COPY index.html /usr/share/nginx/html/index.html                                     0.0s
 => exporting to image                                                                         0.1s
 => => exporting layers                                                                        0.0s
 => => writing image sha256:b8ad379a68aa8e5b4b2dfbbd321939e9c5c10ebb0f1aee2ef93316e660c7f9f5   0.0s
 => => naming to docker.io/library/my-nginx                                                    0.0s
```

- Shell into a running container and execute basic commands

```
# List running containers and find the ID or name of the container you
want to shell into
docker ps
```

```
# Shell into the container
docker exec -it <container-id-or-name> /bin/bash

# Now you are inside the container's shell
# You can execute any basic commands you need
ls
pwd
whoami
uname

# Exit the container's shell
exit
```

● Create 2 docker files of nginx with CMD and ENTRYPOINT respectively.
Dockerfile using CMD:

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html/index.html
CMD ["nginx", "-g", "daemon off;"]
```

Dockerfile using ENTRYPOINT:

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html/index.html
ENTRYPOINT ["nginx", "-g", "daemon off;"]
```

● Create a multi-stage build dockerfile for nginx

```
# Stage 1: Build stage
FROM node:14 as build-stage

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .
RUN npm run build

# Stage 2: Production stage
FROM nginx:latest as production-stage

COPY --from=build-stage /app/build /usr/share/nginx/html

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```