Bridge networks facilitate communication between Docker containers on the same host. In this lesson, we will explore bridge networks in a little more detail. We will discuss the commands needed to create and use bridge networks. We will also examine Docker's embedded DNS, and how to use it to communicate with containers using container names and network aliases. We will also talk about some commands that can be used to manage existing networks on a Docker host.

Relevant Documentation

- https://docs.docker.com/network/bridge/

Lesson Reference

1. Create a bridge network and demonstrate that two containers can communicate using the network.

```
docker network create my-net
docker run -d --name my-net-busybox --network my-net radial/busyboxplus:curl sleep 3600
docker run -d --name my-net-nginx nginx
docker network connect my-net my-net-nginx
docker exec my-net-busybox curl my-net-nginx:80
```

2. Create a container with a network alias and communicate with it from another container using both the name and the alias.

```
docker run -d --name my-net-nginx2 --network my-net --network-alias my-nginx-alias nginx
docker exec my-net-busybox curl my-net-nginx2:80
docker exec my-net-busybox curl my-nginx-alias:80
```

3. Create a container and provide a network alias with the `docker network connect` command.

```
docker run -d --name my-net-nginx3 nginx
docker network connect --alias another-alias my-net my-net-nginx3
docker exec my-net-busybox curl another-alias:80
```

4. Manage existing networks on a Docker host.

```
docker network ls
docker network inspect my-net
docker network disconnect my-net my-net-nginx
docker network rm my-net
```