Creating a Network in Docker and Connecting a Container to That Network

Networks are created so that the devices which are inside that network can connect to each other and transfer of files can take place. In docker also we can create a network and can create a container and connect to the respective network and two containers that are connected to the same network can **communicate with each other**. These containers can also communicate with the host in which the docker is deployed. **The communication will take place by using the IP address of each other.** A container can connect to one or more networks in that docker host.

Step 1: In the first step we run a command to see the list of networks in your docker host.

```
-bash-4.2$ docker network ls
NETWORK ID     NAME          DRIVER    SCOPE
d1c2923c9b2c   bridge        bridge    local
557df3b72879   host          host      local
a3f562f38a4b   network_new   bridge    local
362dbe0bb189   none          null      local
```

A bridge is the **default network in docker.**

Step 2: If you want to see the information regarding the particular network you can use the inspect command.

docker network inspect network name or network ID

```
-bash-4.2$ docker network inspect d1c2923c9b2c
[
    {
        "Name": "bridge",
        "Id": "d1c2923c9b2c2b7e3caa80acf233d131d6c0b2f50698651d4d6e67c
90f3272ac",
        "Created": "2022-03-30T14:17:14.083369888Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
```

```
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        },
        "Labels": {}
    }
]
```

Notice that there is no container currently connected to the bridge driver as the brackets are empty.

Step 3: Let create a new network

docker network create --driver bridge network_new

```
-bash-4.2$ docker network create --driver bridge network_new
a3f562f38a4be02e194dd8e10b7918e97a6c5eddcb323b56aad2e9483811ca7a
```

Description of the commands:

- **driver_name:** name of the driver in this case we will use bridge driver.
- **network_name:** name of the network you want to give to your network.
-

Step 4:

Confirm the new network has been created

```
-bash-4.2$ docker network ls
NETWORK ID     NAME          DRIVER    SCOPE
d1c2923c9b2c   bridge        bridge    local
557df3b72879   host          host      local
a3f562f38a4b   network_new   bridge    local
362dbe0bb189   none          null      local
```

Step 5: In this step, we will run and connect a container to the network which we have created in the previous step. Run this in detach mode

sudo docker run -itd --network= network_new ubuntu:latest /bin/bash

```
-bash-4.2$ docker run -itd --network=network_new ubuntu:latest /bin/ba
sh
```

Step 6: Now inspect the network we created. The container we attach to the respected network is mentioned there

```
        }
    ]
    -bash-4.2$ docker network inspect network_new
```

This is the complete process to create the network and connect the container to it.