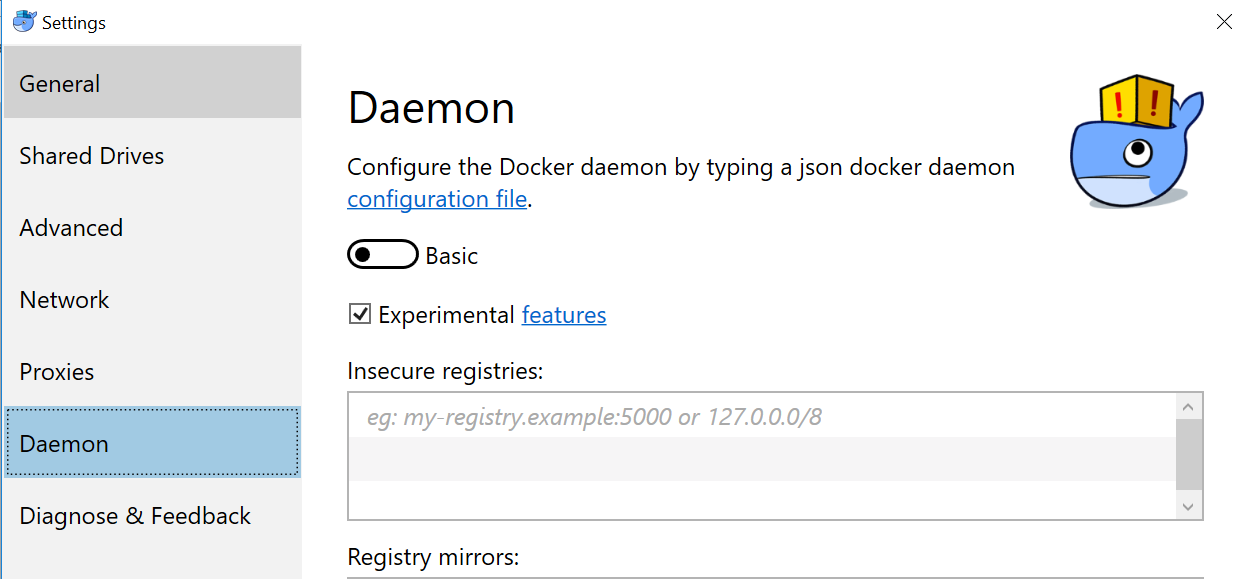
Guidelines for using Docker Swarm

## Settings in Docker daemon:

* *Experimental features* must be turned ON

Right click on docker in sys tray -> settings -> Daemon -> check the checkbox “Experimental features”



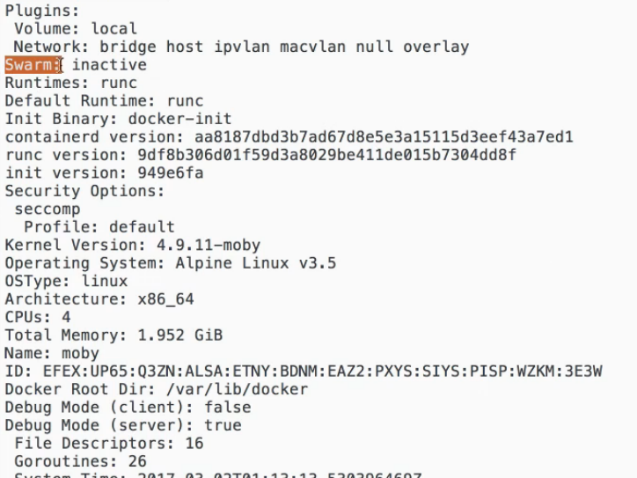
## Activating the Swarm mode in Daemon:

In order to make swarming work, Docker needs to have one node work as swarm manager and rest of nodes can still remain “worker nodes”. For current example, since I have only one node, it will work as swarm manager. To make it swarm manager, it requires to enable the “Swarm” mode.

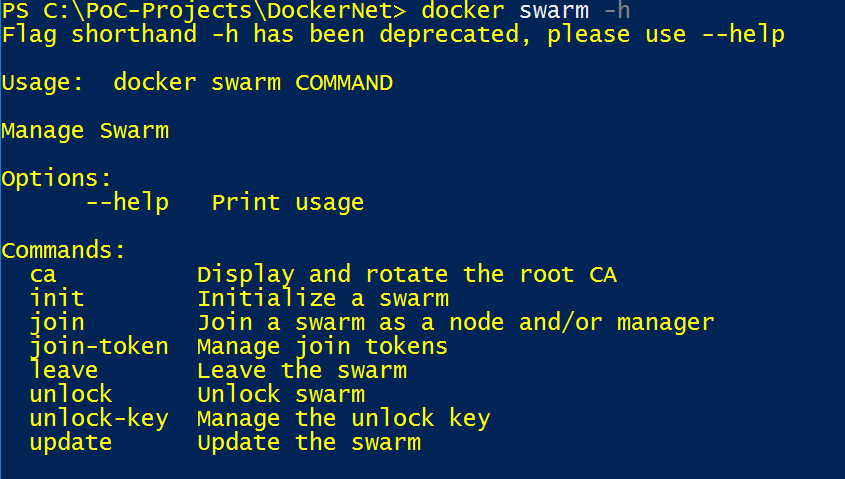
Use command *docker info*

**

That would tell whether swarm is enabled or not. Scroll through the output of *info* command to see that:

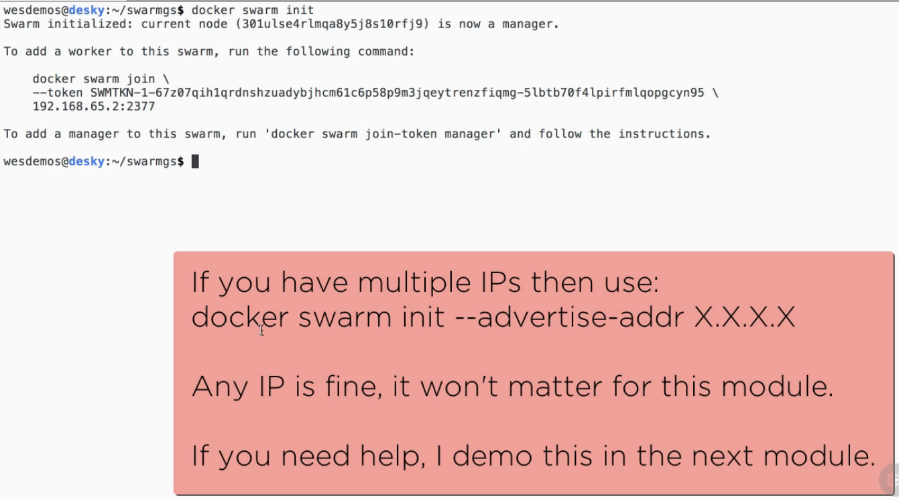


To activate Swarm: take a look at host of docker management commands. *init* is one that activates the swarm



Here is how you initiate swarm mode:

***docker swarm init***

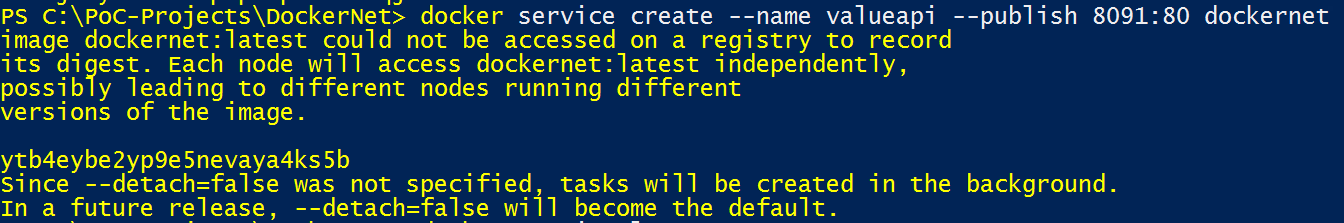


## Same Node Swarm:

### Creating Docker Service and task:

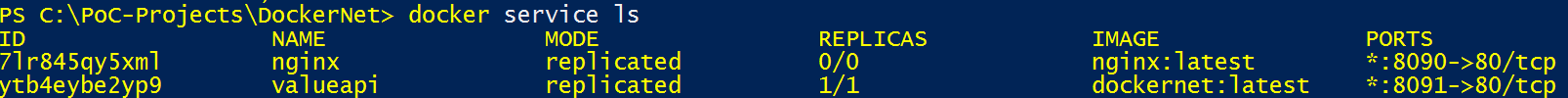
Let’s use one core .net api image *valueapi* for this example.

Command line for creating the service:

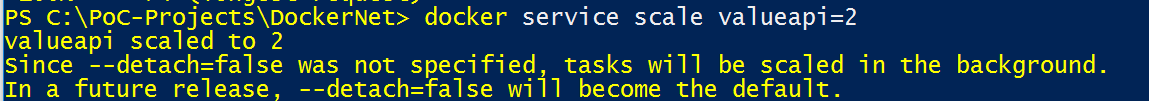


This creates a service with container *valueapi* publishing to port 8091 of host machine.

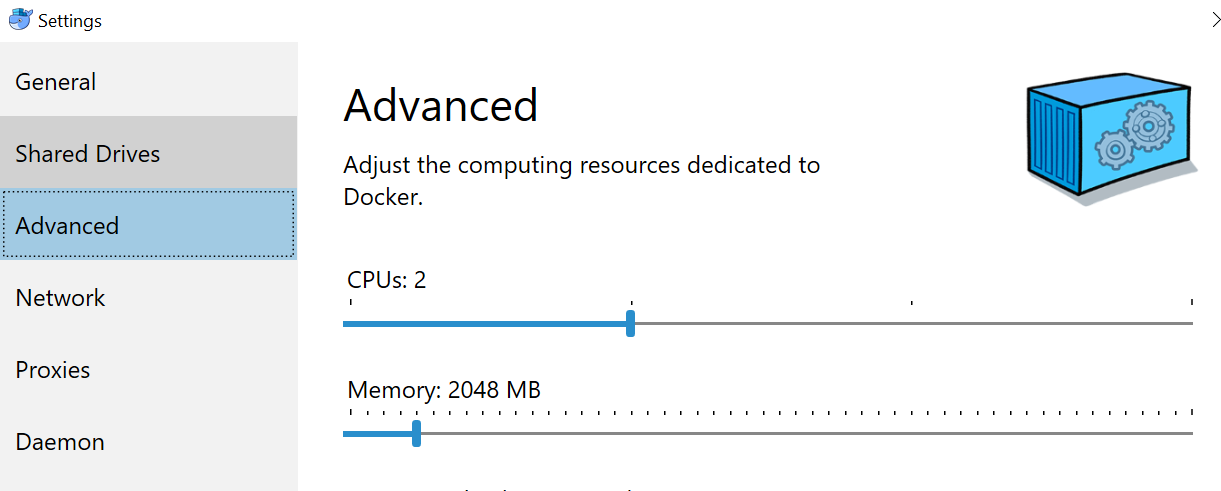
Service should be listed in **docker service ls** output



Service can be scaled up or down as per the optimization requirements and hardware support on host node. For example, in order to have 2 replicas of same service running, use following command:



Note: the scaling up of services should be in sync with number of CPUs configured to Docker Daemon:



In most probability, 2 replicas may give the best optimization…still that contingent upon other variables too.

Try following apache benchmarking command line:

ab -n 1000 -c 4 <http://127.0.0.1:8091/api/value>

### **Deploying Services using Compose file**

#### Create a docker compose with Service segment**:**

Create a docker-compose file (in my example, docker-composeNew.yml) that has service segment therein

*version: '3.1'*

*services:*

*apiserv:*

*image: dockernet*

*deploy:*

*replicas:2*

*ports:*

*- "8092:80"*

In the *docker-compose* file above, a service named as “*apiserv*” is created using image “*dockernet*”. Notice the *deploy* section - that’s the one that facilitates number of *replicas* (in same cluster) to be setup. The cluster will run on port 8092 of host computer.

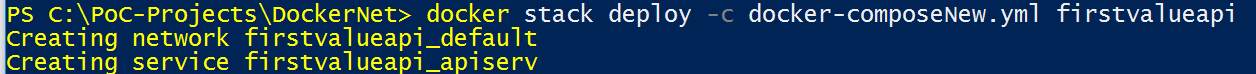
#### Deploy the container Cluster:

Use *docker stack deploy* command line as following to deploy the cluster defined in *docker-compose* file:



In the example above, -c switch accepts compose file and “firstvalueapi” is the name given to containers in run mode.

The command above shall output as following:



#### Verify the services and tasks running:



It shows 2 replicas are running.

Check out the individual tasks:



There are 2 replicas with suffix of .1 and .2 are running (see the C*urrent State* column)