Virtual Pair Programmers

Docker for Java Developers - Quick Command Reference 1

Manage images

docker image pull <image name>
(Chapter 4): download an image from DockerHub

docker image ls Chapter 5
(Chapter 5): list all local images

docker image build -t <image name> .
(Chapter 6): build an image with a tag (note the dot!)

docker image push <image name>
(Chapter 9): publish an image to dockerhub

docker image tag <image id> <tag name>
(Chapter 9): tag an image - either alias an exisiting image or apply a :tag to one

Manage Containers

docker container run -p <public port>:<container port> <image name>
(Chapter 4): run a container from an image, publishing the specified ports

docker container ls -a
(Chapter 4): list all containers, even the stopped ones

docker container stop <container id>
(Chapter 4) stop a running container

docker container start <container id>
(Chapter 4) restart a stopped container

docker container rm <container id>
(Chapter 4) remove a stopped container

Virtual Pair Programmers

Docker for Java Developers - Quick Command Reference 2

Manage Containers (ctd)

docker container prune
(Chapter 4) remove all stopped containers

docker container run -it <image name>
(Chapter 5): run a container with interactive terminal

docker container run -d <image name>
(Chapter 5): run a container detached (or in a daemon like way)

docker container exec -it <container id> <command>
(Chapter 5): run a command in a container

docker container exec -it <container id> bash
(Chapter 5): special form of the above, runs a bash shell, connected to your local terminal (your distro needs to have bash, alpine will require /bin/sh)

docker container logs -f <container id>
(Chapter 5) Follow the log (STDIN/System.out) of the container

docker container commit -a "author" <container id> <image name>
(Chapter 6) Take a snapshot image of a container

Manage your (local) Virtual Machine

docker-machine ip
(Chapter 4): Find the IP address of your VirtualMachine, required for Docker Toolbox users only

Virtual Pair Programmers

Docker for Java Developers - Quick Command Reference 3

Manage Networks

docker network ls
(Chapter 10): list all networks

docker network create <network name>
(Chapter 10): create a network using the bridge driver

Manage Volumes

docker volume ls
(Chapter 11): list all volumes

docker volume prune
(Chapter 11): delete all volumes that are not currently mounted to a container

docker volume inspect <volume name>
(Chapter 11): inspect a volume (can find out the mount point, the location of the volume on the host system)

docker volume rm <volume name>
(Chapter 11): remove a volume

Docker Compose

docker-compose up
(Chapter 13): process the default docker-compose.yaml file, starting any containers as required. If containers are already running they are ignored, meaning this command also serves as a "redeploy".

docker-compose up -d
(Chapter 13): run containers in the detached state. Note the order of the command line arguments!

docker-compose logs -f <service name>
(Chapter 13): follow the log for the specified service. Omit the -f to tail the log.

docker-compose down
(Chapter 13): stop all the containers (services) listed in the default compose file.

Virtual Pair Programmers

Docker for Java Developers - Quick Command Reference 4

Manage a Swarm

docker swarm init (--advertise-addr <ip address>)
(Chapter 14): Switch the machine into Swarm mode. We didn't cover how to stop swarm mode: ***docker swarm leave --force***

docker service create <args>
(Chapter 14): Start a service in the swarm. The args are largely the same as those you will have used in docker container run.

docker network create --driver overlay <name>
(Chapter 14): Create a network suitable for using in a swarm.

docker service ls
(Chapter 14): List all services

docker node ls
(Chapter 14): List all nodes in the swarm

docker service logs -f <service name>
(Chapter 14): Follow the log for the service. This feature is a new feature in Docker and may not be available on your version (especially if using Linux Repository Packages).

docker service ps <service name>
(Chapter 15): List full details of the service - in particular the node on which it is running and any previous failed containers from the service.

docker swarm join-token <worker|manager>
(Chapter 16): Get a join token to enable a new node to connect to the swarm, either as a worker or manager.

Manage Stacks

docker stack ls
(Chapter 15): list all stacks on this swarm.

docker stack deploy -c <compose file> <stack name>
(Chapter 15): deploy (or re-deploy) a stack based on a standard compose file.

docker stack rm <stack name>
(Chapter 15): delete a stack and its corresponding services/networks/etc.