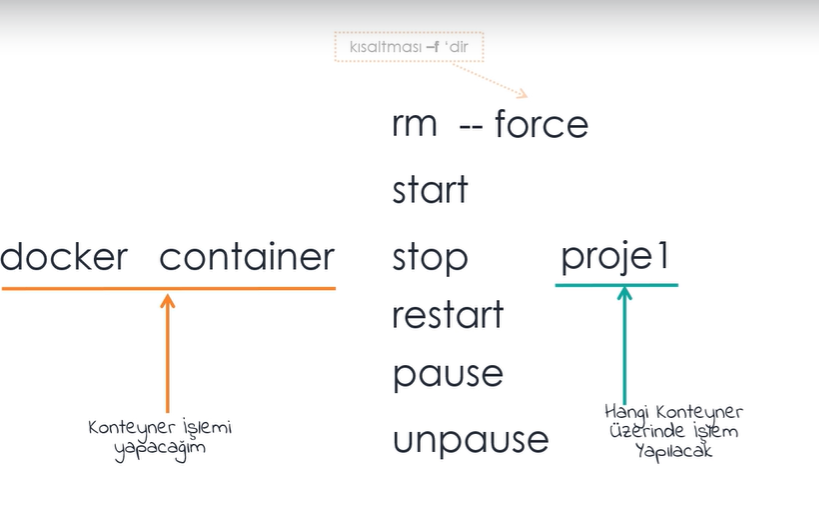
**Docker Command Cheat Sheet**

* Login to docker registry from browser by email and password and we can login or logout from terminal by;

docker login or docker logout

* Create and run container: alpine image (Mini basit Linux Isletim sistemi)
  + *docker run alpine*
  + *docker run python:3*
* Parameter can use after docker container or docker container run
  + *docker run --help*
* Create python image container name is proje1
  + *docker run --name proje1 python:2*
* List of container
* *docker container ps (0nly list of running container)*
* *docker container ps -a*
* *docker container ls -a -q (Only list of the id of container)*
* Get information about container
* *docker info*
* Start-Stop-Pause-Delete Container

**

* Rename the docker container; juliane to dockerweb
* *docker rename juliane dockerweb*
* Delete Stopped Container:
* *docker rm (name or id container)*
* *docker rm --f (name) -delete running container also*
* Delete all the container in one line:
* *docker container rm $(docker container ls -a -q)*
* Delete Stopped Container -best practise-
* *docker container prune*

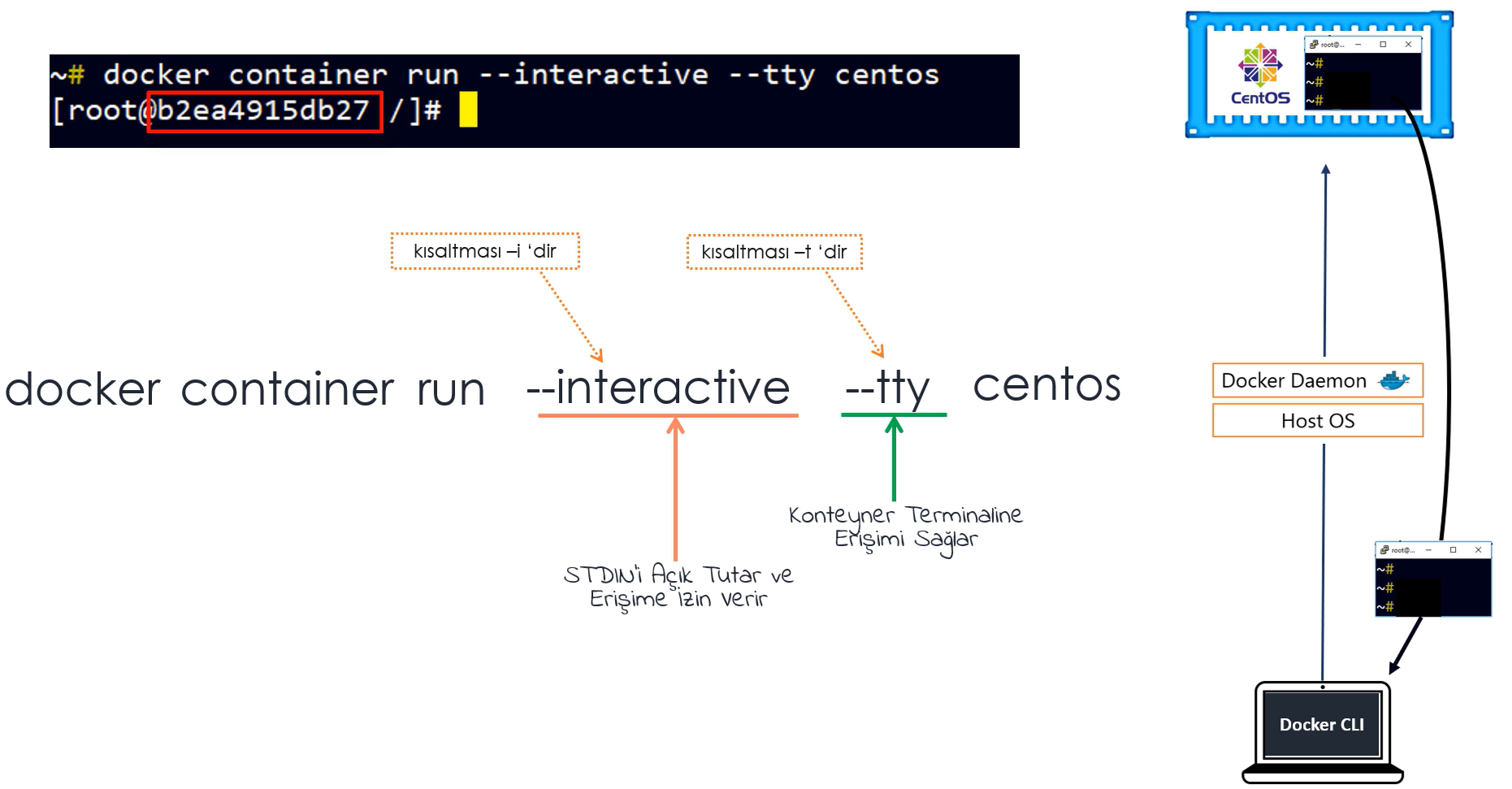
**Run the website by Nginx:**

* *docker run -p 8080:80 -d --name myweb nginx*

p: port, host:8080 container port: 80

d: de-attached mode, it will run background

**Connect & Work on Container without SSH Connection**



* We can reach the terminal of nginx or python on container.
* *docker run -it nginx python:3 bash*
* ***NOTE: Image parametresinden sonra ne yazarsak terminale gitmeden konteynir icerisinde bu komut calisacaktir.***

***bash, cmd, Powershell yazarsak isletim sisteminin icerisindeki terminale gideriz.***

* Delete the container after working on the terminal
* *docker run --rm -it python:3*
* Let’s create container name centoscontainer and 30min to stand up time
* *docker run -d --name centoscontainer centos sleep 30m*
* Installing nginx on the centoscontainer without entering inside container
* *docker exec centoscontainer yum install -y nginx*

**Multi-Container Management:**

* Create mariadb and connect to database container on terminal:

*docker run --name mariadb1 -d -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=’123456’ mariadb (-e : env)*

* Run a mysql container named mysql-db using the mysql image. Set database password to db\_pass123

docker run -d --name mysql-db -e MYSQL\_ROOT\_PASSWORD=db\_pass123 mysql

**DOCKER IMAGES:**

* Download image from dockerhub to local

*docker image pull python:3* **or**

*docker pull python:3*

* To send image from local to the my(yasint23) dockerhub repository
* *docker commit hello-world yasint23/hello-world* (Convert the container to the image)
* *docker login*
* *docker image push yasint23/hello-world*
* *docker image pull docker.io/yasint23/hello-world*
* *docker logout*

**DOCKER NETWORK:**

* Network list, information

*docker network ls*

*docker inspect mynetwork*

* Run a container named alpine-2 using the alpine image and attach it to the none network.

docker run --name alpine-2 --network=none alpine

* Create a new network named wp-mysql-network using the bridge driver. Allocate subnet 182.18.0.1/24. Configure Gateway 182.18.0.1

docker network create --driver bridge --subnet 182.18.0.1/24 --gateway 182.18.0.1 wp-mysql-network

* Deploy a mysql database using the mysql:5.6 image and name it mysql-db. Attach it to the newly created network wp-mysql-network

Set the database password to use db\_pass123. The environment variable to set is MYSQL\_ROOT\_PASSWORD.

docker run -d -e MYSQL\_ROOT\_PASSWORD=db\_pass123 --name mysql-db --network wp-mysql-network mysql:5.6

* Deploy a web application named webapp using the kodekloud/simple-webapp-mysql image. Expose the port to 38080 on the host.  
  The application makes use of two environment variable:  
  1: DB\_Host with the value mysql-db.  
  2: DB\_Password with the value db\_pass123.  
  Make sure to attach it to the newly created network called wp-mysql-network.

Also make sure to link the MySQL and the webapp container.

docker run --network=wp-mysql-network -e DB\_Host=mysql-db -e DB\_Password=db\_pass123 -p 38080:8080 --name webapp --link mysql-db:mysql-db -d kodekloud/simple-webapp-mysql

* Olusturdugumuz docker container’i network uzerinden cikarmak daha sonra network’u silme

*docker network disconnect none alpine*

*docker network rm none*

**DOCKER VOLUME:**

**Data Volume**

* Create data volume name depo (Konteyner disinda bir alanda kalici disk alani, konteyner silinse bile bu alan kalir- path yolu; /var/lib/docker/volumes)

*docker volume create depo*

*docker volume rm depo*

* Konteyner olustururken volume olusturma

*docker run –name voltest –detach –volume depo:depo nginx*

**Bind Volume**

* Host uzerindeki bir klasoru konteyner icersine direk baglama

*mkdir commonshare && touch commonshare/list.txt*

*docker run -d –name depo1 –volume $(pwd)/commonshare:depo nginx*

*docker exec -it depo1 bash*

*cd depo && ls ===> See the list.txt inside of container*

*Ex:* Run a container name "container1" with using alpine image, create this container with interactive mode, mount a volume name "volume1" to the "/test1" folder with readonly access and run the command "ls".

docker run --name container1 -it -v volume1:/test1:ro alpine ls