**Docker Command Cheat Sheet**

**Best container management tool: https://www.portainer.io/**

* Create and run container: alpine image (Mini basit Linux Isletim sistemi)
  + *docker container run alpine*
  + *docker container run python:3*
* Parameter can use after docker container or docker container run
  + *docker container --help*
  + *docker container run --help*
* Create python image container name is proje1
  + *docker container run --name proje1 python:2*
* List of container
* *docker container ls (0nly working container)*
* *docker container ls -a*
* *docker container ls -a -q (Only list the id of container)*
* docker ps -a

Show the list of all containers available on Docker machine

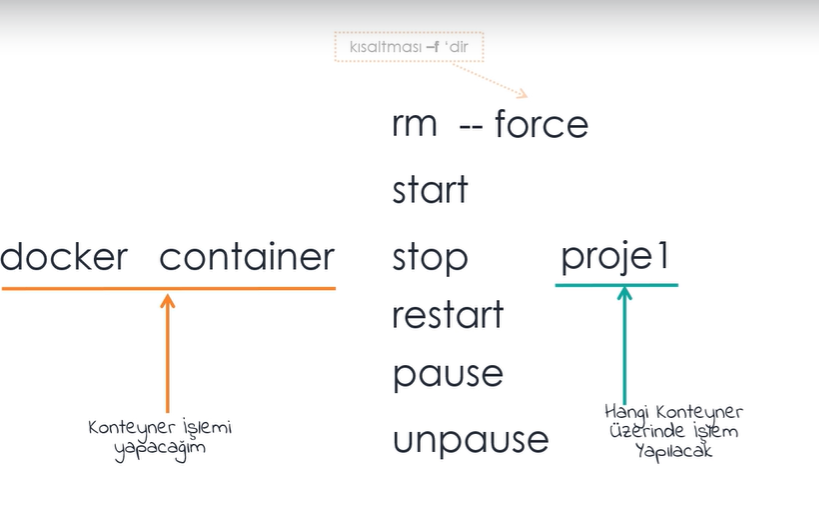
* docker ps

Show the list of running containers on Docker machine

* docker attach container1

Connect to the interactive shell of running `container1` container

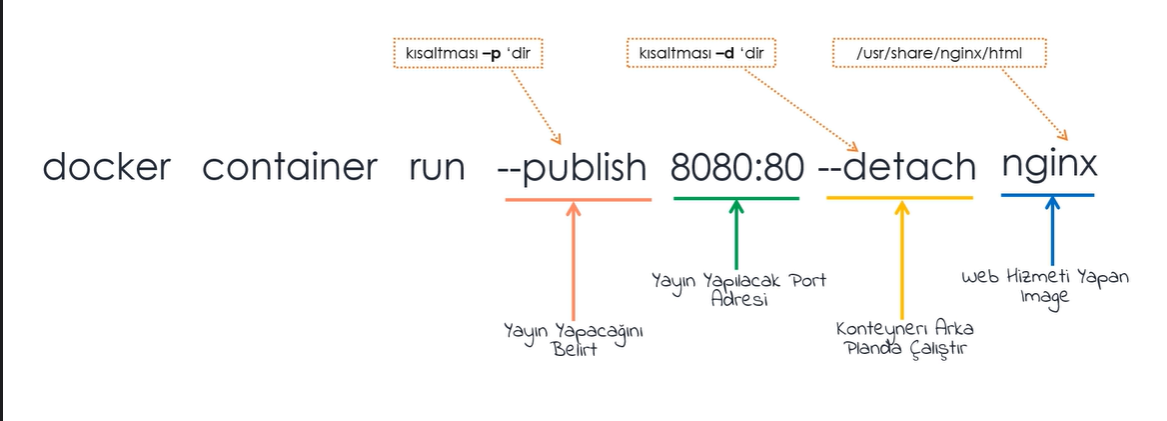
* Get information about container
* *docker info*
* Start-Stop-Pause-Delete Container

**

* Downlading Docker webpage to my local
* *docker container run -d -p 4000:4000 docs/docker.github.io*
* Rename the docker container; juliane to dockerweb
* *docker container rename juliane dockerweb*
* Delete Stopped Container:
* *docker container rm (name or id container)*
* *docker container 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*

**Nginx//Windows IIS**

**host container micrsoft/iis**

****

Run the website by Nginx

* *docker container run -p 8080:80 -d nginx*

*8081:80 possible but 8080:81 not possible*

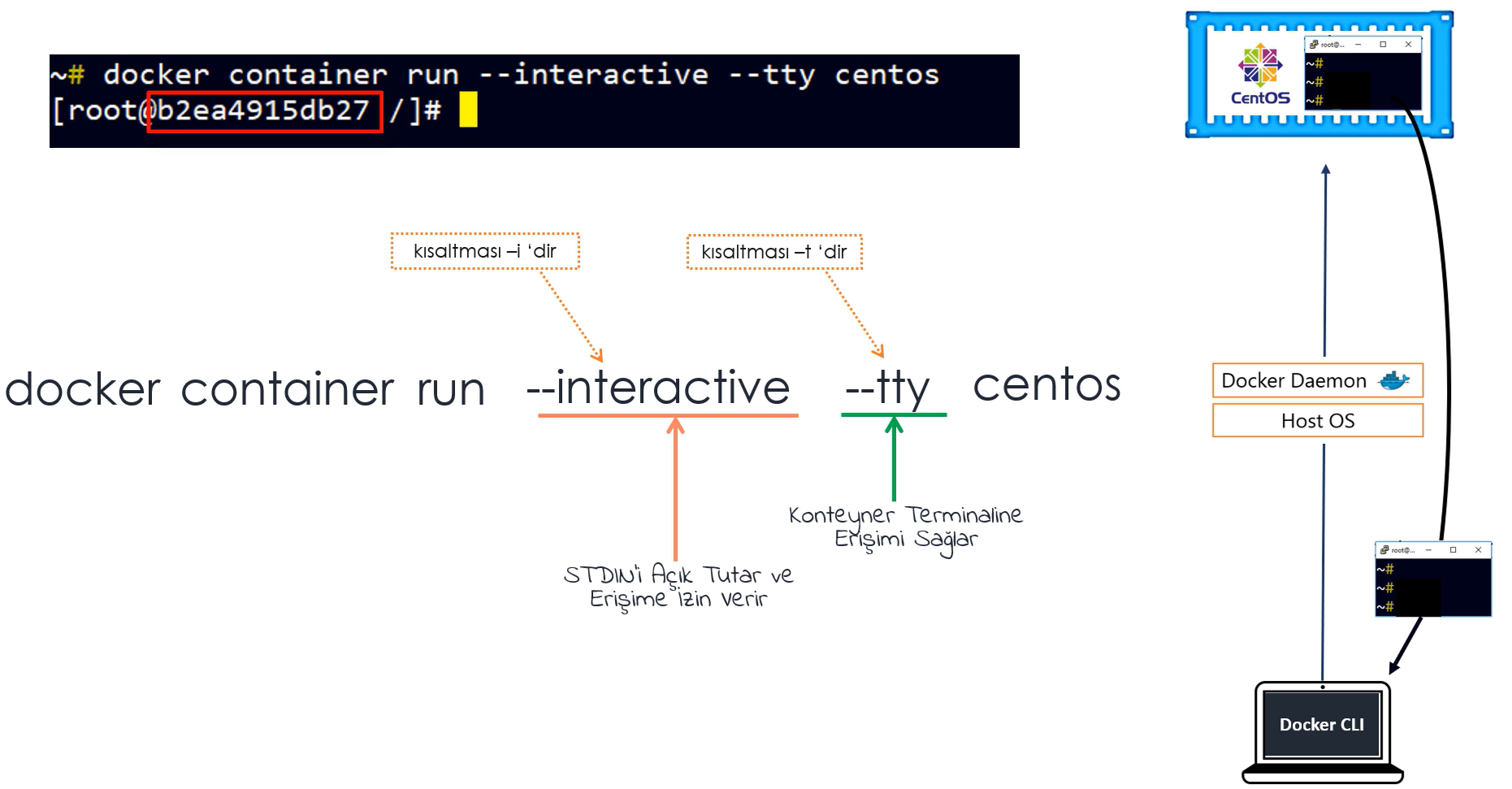
p: We determine the port

P: Automatically random port determine

* Create nginx container with the name of myweb and port determine ramdomly, it will run backround the console

*docker container run -P -d --name myweb nginx*

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



* ***NOTE: Tüm konteynır komutları image parametresin den sonra yazılır.***

***Image parametresinden sonra yazılan tüm komutlar konteynır içerisinde çalışır.***

* We can reach the terminal of nginx or python on container.
* *docker container run -i -t nginx bash*
* *docker container run -i -t python:3 bash*
* Create docker container of alpine with the name of Op.Sys1 and terminal of alpine(mini linux) is sh
* *docker container run -it –name OpSys1 alpine sh*
* *docker container run -it python:3 bash*

*/# python -V (give us version )*

*/# python (open the terminal f or python code)*

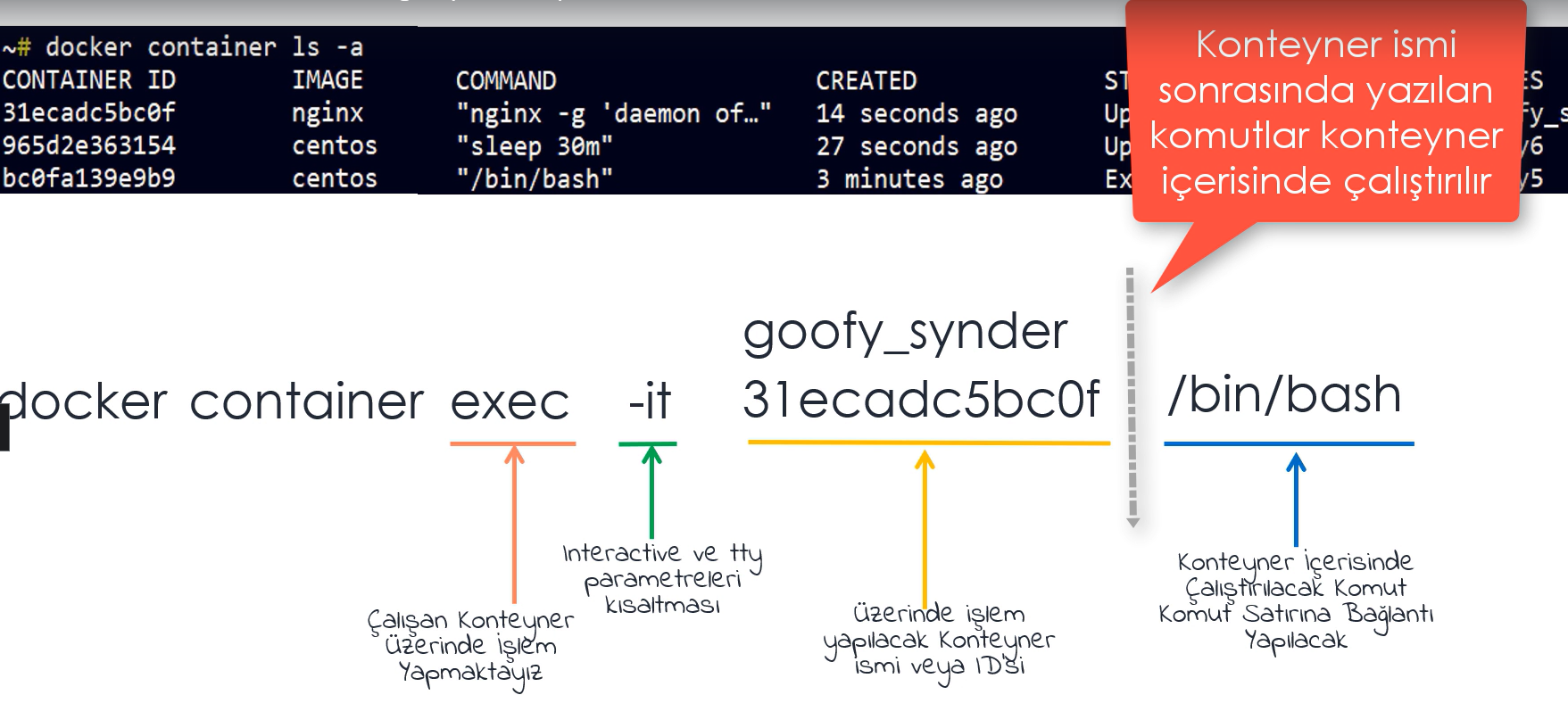
*/# docker container run -it python:3 (Saved as python terminal and open the terminal of python)*

* ***NOTE: Image parametresinden sonra ne yazarsak terminale gitmeden konteynir icerisinde bu komut calisacaktir.***

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

* Delete the container after working on the terminal
* *docker container run --rm -it python:3*
* *We can reach the Linux desktop by container (actually it is just image created by someone)*
* *docker container run -p 6080:80 -d dorowu/ubuntu-desktop-lxde-vnc*

**Connect & work on any working container or work before creating container:**



* To run the command inside of any working container
* *docker container exec linuxcontainer(name/id) yum install -y vim*

*Daha once create ettigimiz linuxcontainer icerisine girmeden vim yukle*

* Let’s create container name centoscontainer and 30min to stand up time
* *docker container run -d --name centoscontainer centos sleep 30m*
* *docker container exec -it centoscontainer bash*

*# mkdir app1 (worked inside of container)*

* *docker container exec centoscontainer mkdir app2 (worked without entering the container)*
* Installing nginx on the centoscontainer without entering inside container
* *docker container exec centoscontainer yum install -y nginx*
* Create script file inside container to be able to install or run packet from outside the container.
* *~# docker container exec -it centoscontainer bash (enter inside container)*

*]# cd tmp (enter the tmp file)*

*]# echo “ \*

*> yum -y update && \*

*> yum install -y vim && \*

*> mkdir app3 && \*

*> touch app3/list.txt “ >>script doc(create list.txt file inside app3 folder*

*]#chmod 755 script ]#exit*

*~# docker container exec -it centoscontainer /bin/bash ./tmp/script*

*-------------------------------------------------------------------------------------------------------------*

* Let’s create nginx container name webapp then change the default page of nginx

*docker container run -dp 5080:80 --name webapp nginx*

*docker container exec -it webapp bash*

*cd /usr/share/nginx/html/ (webpage yayin yaptigi path e gidiyorum)*

*usr/share/nginx/html# ls (list the html files)*

*usr/share/nginx/html# rm -r \* (delete the html files)*

*echo “Hello World, Have a good lesson”>>index.html*

**Multi-Container Management:**

* Create mariadb and connect to database container on terminal:

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

*docker container exec -it mariadb1 mariadb -uroot -p*

Enter Password:

> show databases;

>create database demo

To check databases on bash terminal;

~# *docker container exec -it mariadb1 bash*

*~# cd /var/lib/mysql*

* Create mssql container and connect to database container on terminal:

*docker container run --name mssqldb1 -d -p 1433:1433 -e ‘ACCEPT\_EULA=Y -e ‘SA\_PASSWORD=123456’ mcr.microsoft.com/mssql/server*

*docker container exec -it mssqldb1 /opt/mssql-tools/bin/sqlcmd -S localhost -U YT -P “123456”*

> select name from sys.Databases

>Go

>CREATE DATABASE testDB

To check databases on terminal;

~# *docker container exec -it mssqldb1 bash*

*/$ cd /var/opt/mssql/data*

**Container Details**

* To check details of container name is web

*docker container inspect web*

*docker container inspect web | grep IpAddress/Gateway/Hostname*

*docker container inspect -f “{{.NetworkSettings.IPAddress}}” web*  (to get data only)

* To check port list of container name is web

*docker container port web (port of website broadcasting)*

* To check logs of container name is web

*docker container logs web*

* To check container operations name is web

*docker container top web*

* To check container resorce usage name is web

*docker container stats web (cpu ram usage)*

**Running a Node.js application inside container:**

*docker container run -it --name nodejs -d -p 1111:8080 --workdir /nodejs --volume C:\Users\pakya\My\_Devops\class-notes\Docker\nodejsdocker:/nodejs node:latest node server.js*

* To check project folders

*docker container exec -it 7ff16.. /bin/bash*

**……**

**Running a Python application inside container:**

……

**DOCKER IMAGES:**

* Download image from dockerhub to local

*docker image pull python:3*

* To send image from local to the my(yasint23) dockerhub repository

*docker login*

*docker image push yasint23/image name*

*docker logout*

* To make container as image

*docker container commit container-name image-name*

*docker container commit container-name yasint23/image-name*

* To put tag on docker images without tag

*docker image tag Image-Id target-image-name*

* To change tag on docker images

*docker image tag changing-image-name target-image-name*

* To see the stages of image

*docker image history image-name*

* To search docker images on dockerhub

*docker search image-name/user*

* To check details of image

*docker image inspect image-name*

*Ex:*

* **docker image pull docker.io/murataksunet/merhaba** (image name “merhaba” dowloaded from murataksunet repo to my local)

docker container run -d -p 8080:80 murataksunet/merhaba

**Ex:**

* *docker container run -d -p 7080:80 --name hello-world nginx* (hello-world container created from nginx image)
* *docker container exec -it hello-world bash*

root@77ee1a86350d:/# *cd /usr/share/nginx/html*

root@77ee1a86350d:/usr/share/nginx/html# ls

50x.html index.html

root@77ee1a86350d:/usr/share/nginx/html# *rm index.html*

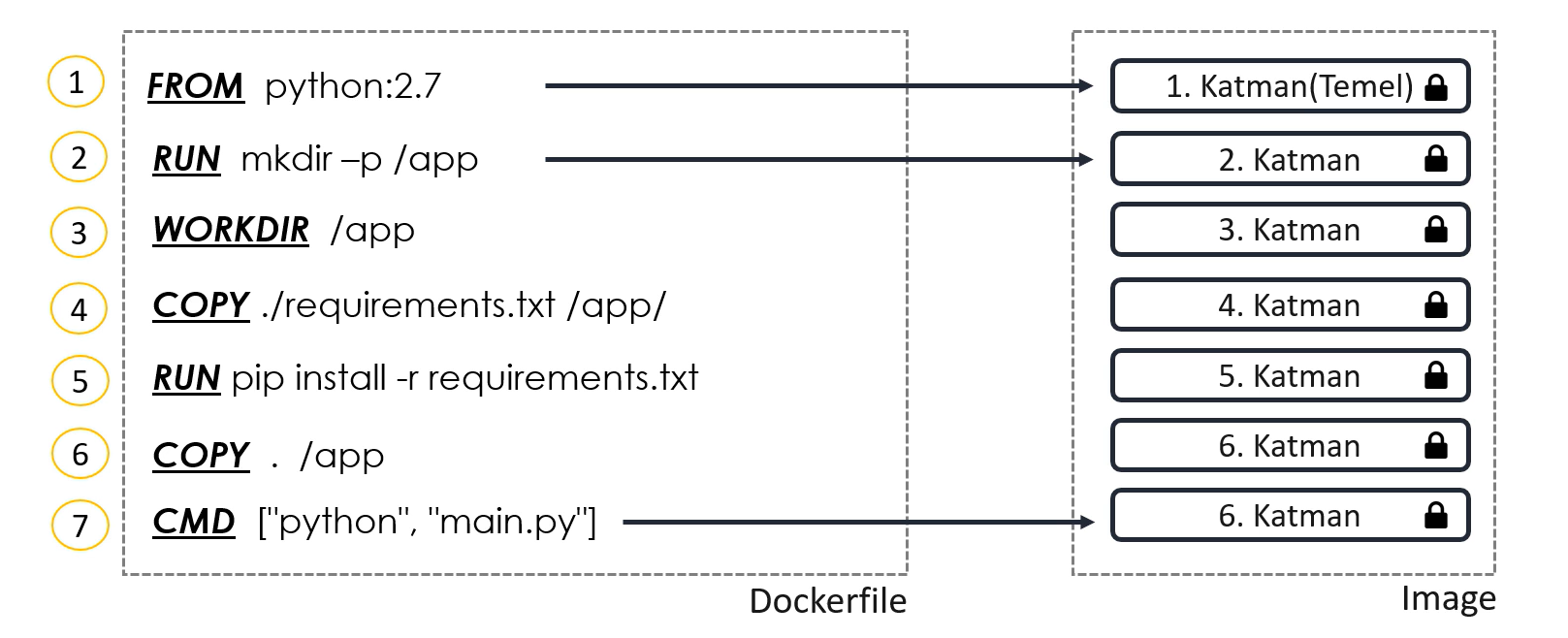
root@77ee1a86350d:/usr/share/nginx/html# *echo "Hello World" >> index.html*

root@77ee1a86350d:/usr/share/nginx/html# exit

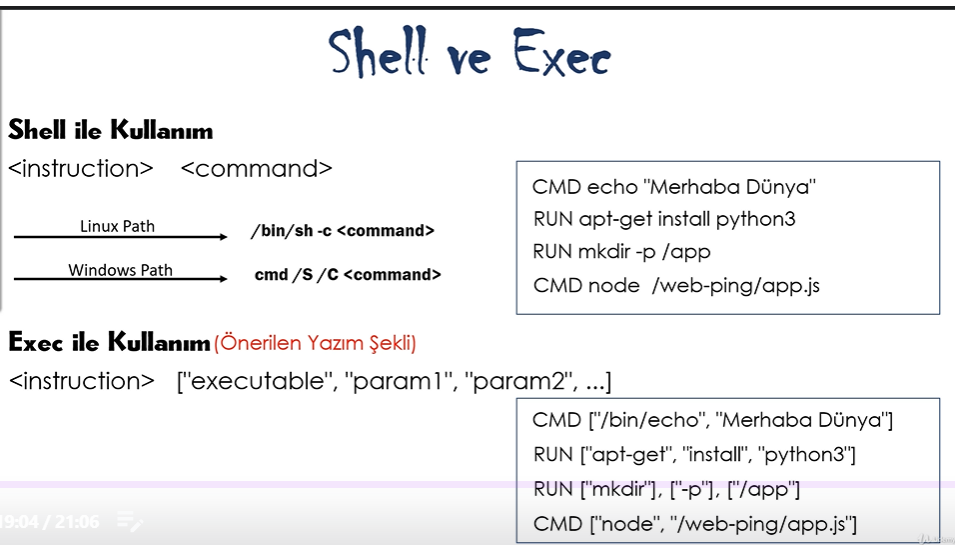
* *docker container commit hello-world yasint23/hello-world* (Convert the container to the image)
* *docker login*
* *docker image push yasint23/hello-world*

**DOCKER FILE:**

The detail txt files for each layer of docker images



<https://docs.docker.com/engine/reference/builder/> (Docker file commands details)



**DOCKER NETWORK:**

* Docker uzerinde network olusturma

*ipvlan / none*

*docker network create –driver overlay sw0-ntwrk*

*macvlan / bridge*

* Docker network listeleme (herbir network icin ayri bir ID olusur, default olarak Name: bridge, host, none vardir birde sw0-ntwrk biz olusturduk)

*docker network ls*

* Yeni olusturdugumuz docker container’i network uzerine dahil etme

*docker container run –network sw0-ntwrk –name cont-prod alpine*

* Olusturdugumuz docker container’i baska bir network uzerine dahil etme

*docker network connect sw1-ntwrk cont-prod*

* Yeni networkun detaylarina baktigimizda konteyner in yeni IP aldigini goruruz

*docker network inspect sw1-ntwrk*

* Olusturdugumuz konteyner in detaylarina baktigimizda networklardan iki ayri IP address bilgilerini gorecegiz

*docker container inspect cont-prod*

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

*docker network disconnect sw1-ntwrk cont-prod*

*docker network rm sw-ntwrk*

**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