

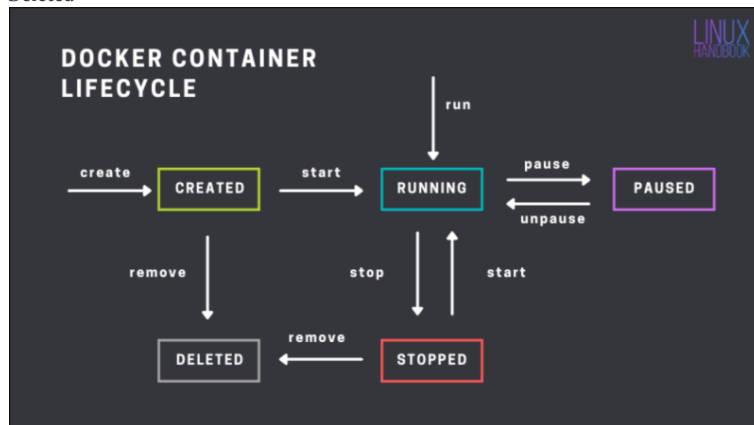


APRIL 9, 2023

DevOps Classroomnotes 09/Apr/2023

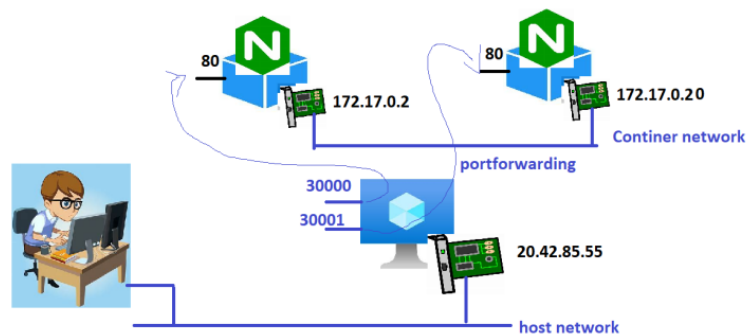
Docker container lifecycle

- Docker lifecycle states
 - Created
 - Running
 - Paused
 - Stopped
 - Deleted

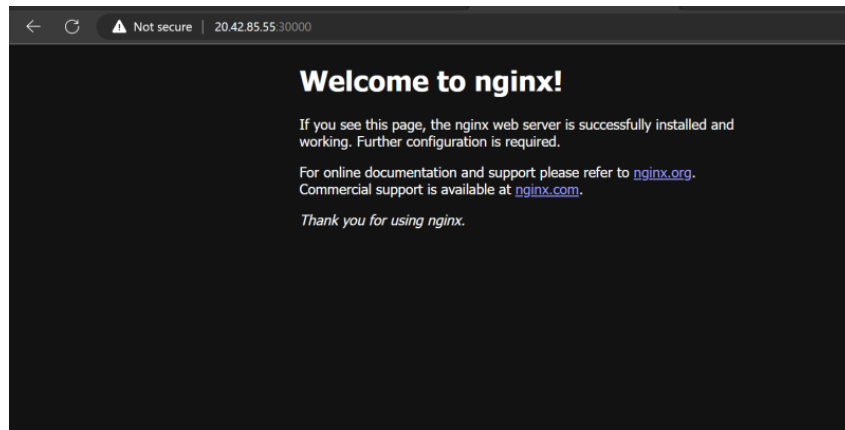


Accessing the applications inside docker containers

- From now the machine where we have installed docker will be referred as host and the docker container will be referred as container
- We have access to host network & as of now containers are created in private container network, so to access applications inside containers we use port-forwarding



- command: `docker container run -d -p <host-port>:<container-port> <image>`
- Create a nginx container and expose on port 30000 `docker container run -d -p 30000:80 --name nginx1 nginx`



- Create a jenkins container & expose 8080 port on 30001 port of host docker container run -d -p 30001:8080 --name jenkins1 jenkins/jenkins



Please wait while Jenkins is getting ready to work ...

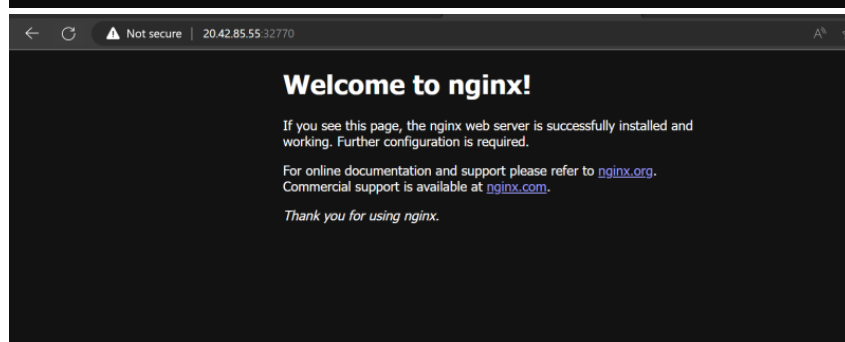
Your browser will reload automatically when Jenkins is ready.

- To assign any random free port on host to container port docker container run -d -P image
- Lets create 3 nginx containers

```
Dell@qtdocker:~$ docker container run -d --name nginx1 -P nginx
9f8749dd68d54e340165cb2089e9e300e9960840f58d9308d63c74b53299718f
Dell@qtdocker:~$ docker container run -d --name nginx2 -P nginx
2bd1fd25927d864a07fea140aa1d581d04e9fd15565098887ac02abb9ebde8d1
Dell@qtdocker:~$ docker container run -d --name nginx3 -P nginx
18e869233fa1393cc4e471b0cfeaf7ad350bbeac519287acf3e071fc558b1b1b
Dell@qtdocker:~$ docker container ls
```

CONTAINER ID	IMAGE	COMMAND	NAMES	CREATED	STATUS	PORTS
18e869233fa1	nginx	"/docker-entrypoint..."	nginx3	10 seconds ago	Up 6 seconds	0.0.0.0:32770->80/tcp
0:32770->80/tcp			nginx3			
2bd1fd25927d	nginx	"/docker-entrypoint..."	nginx2	18 seconds ago	Up 15 seconds	0.0.0.0:32769->80/tcp
0:32769->80/tcp			nginx2			
9f8749dd68d5	nginx	"/docker-entrypoint..."	nginx1	27 seconds ago	Up 23 seconds	0.0.0.0:32768->80/tcp
0:32768->80/tcp			nginx1			

```
Dell@qtdocker:~$
```



Exercise

1. Install docker on a linux vm
2. Run 1 httpd containers (apache container) which runs on 80 port
3. try accessing any application
4. stop the containers
5. try accessing
6. start the containers and access this should work
7. pause the containers, access the application
8. unpause the containers, access the application
9. delete the container

Containerizing spring petclinic

- I have spring petclinic version 2.4.2 which requires java 11 and runs on port 8080
- to start application java -jar spring-petclinic-2.4.2.jar
- What is required:
 - jdk 11
 - jar file
- How to access application
 - http over port 8080
- Lets start the amazoncorretto based container with port 8080 exposed [Refer Here](#)

```
docker container run -it -p 30000:8080 amazoncorretto:11 /bin/bash
```

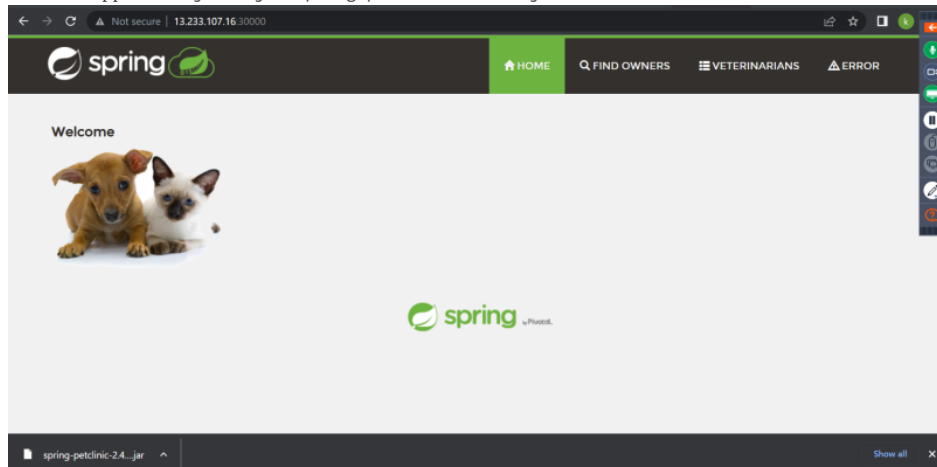
```
ubuntu@ip-172-31-35-3:~$ docker container run -it -p 30000:8080 amazoncorretto:11 /bin/bash
n/bash
bash-4.2# java -version
openjdk version "11.0.18" 2023-01-17 LTS
OpenJDK Runtime Environment Corretto-11.0.18.10.1 (build 11.0.18+10-LTS)
OpenJDK 64-Bit Server VM Corretto-11.0.18.10.1 (build 11.0.18+10-LTS, mixed mode)
bash-4.2#
```

* now lets download the spring petclinic [Refer Here](#)

```
curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar
```

```
bash-4.2# curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2.4.2.jar -o spring-petclinic-2.4.2.jar
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 47.4M 100 47.4M 0 0 3384k 0 0:00:14 0:00:14 --:--:-- 4006k
bash-4.2# ls
bin dev home lib64 media opt root sbin srv tmp var
boot etc lib local mnt proc run spring-petclinic-2.4.2.jar sys usr
bash-4.2#
```

* Run the application java -jar spring-petclinic-2.4.2.jar



* Now to create a image from a running container, lets login into linux vm, so lets use docker container commit

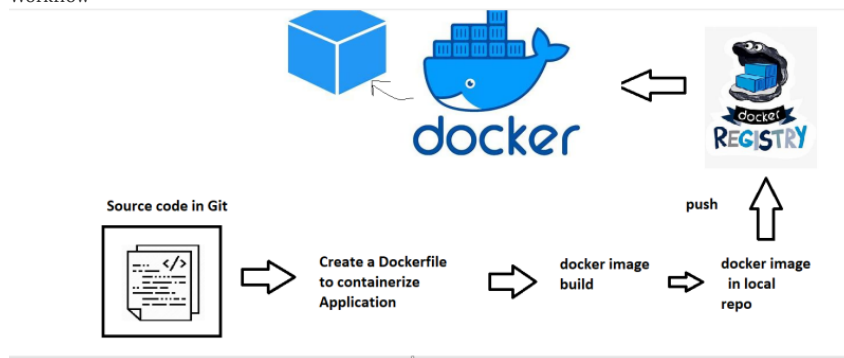
```
ubuntu@ip-172-31-35-3:~$ docker container commit crazy_gagarin myspc:latest
sha256:5d93a130447d6c1b88497b036240b10afe89b15db844ab072d0a30816e3b56f4
ubuntu@ip-172-31-35-3:~$ docker image ls
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
myspc                latest          5d93a130447d   6 seconds ago  499MB
httpd                latest          dc1a95e13784   2 days ago     145MB
jenkins/jenkins      latest          e701a1b6fb83   4 days ago     471MB
amazoncorretto       11             ebc51ffa390b   11 days ago    449MB
ubuntu@ip-172-31-35-3:~$
```

* remove all the containers and run the myspc image based container

```
docker container run -d -p 30001:8080 --name spc1 myspc:latest java -jar spring-petclinic
```

- This is not a useful approach as we are creating images manually
- DOcker has a better way i.e. Dockerfile

- Workflow



- Dockerfile is a text file with instructions [Refer Here](#)
- The basic syntax `INSTRUCTION arguments`
- In Docker we have concept of base image i.e. to run your application using some existing image
- We can use a base image called as scratch which has nothing in it
- In majority of the cases we take what is required to run our application as base image.

- FROM: [Refer Here](#) for official docs. use tag all the time (donot use latest)
- RUN: The commands to be executed while building the image to install/configure your application [Refer Here](#)
- CMD: This command will be executed while starting the container. [Refer Here](#) for official docs
- EXPOSE: This adds ports to be exposed while starting the container [Refer Here](#) for official docs

- Lets do two ways
 - use any image with java11 already as base image amazoncorretto:11
 - use any image with slim os as base image alpine:3
- Dockerfile- based on amazoncorreto:11

```
FROM amazoncorretto:11
RUN curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2
EXPOSE 8080
CMD ["java", "-jar", "spring-petclinic-2.4.2.jar"]
```

- Lets build the image based on amazoncorreto

```
ubuntu@ip-172-31-35-3:~$ mkdir myspc_corretto
ubuntu@ip-172-31-35-3:~$ cd myspc_correto/
ubuntu@ip-172-31-35-3:~/myspc_correto$ nano Dockerfile
ubuntu@ip-172-31-35-3:~/myspc_correto$ ls
Dockerfile
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker image build -t myspc:corretto11 .
[+] Building 4.5s (4/5)
=> [internal] load build definition from Dockerfile                                0.1s
=> => transferring dockerfile: 253B                                              0.0s
=> [internal] load .dockerignore                                                  0.1s
=> => transferring context: 2B                                                    0.0s
=> [internal] load metadata for docker.io/library/amazoncorretto:11             0.0s
=> [1/2] FROM docker.io/library/amazoncorretto:11                             0.0s
=> [2/2] RUN curl https://referenceapplicationskhaja.s3.us-west-2.amazonaws.c   4.4s
=> => # 0 0 0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:--
=> => # 0 0 0 0 0 0 0 0 0 0 --:--:-- --:--:-- --:--:--
=> => # 0 47.4M 0 134k 0 83299 0 0:09:57 0:00:01 0:09:56 83
=> => # 7 47.4M 7 3839k 0 1400k 0 0:00:34 0:00:02 0:00:32 14
=> => # 13 47.4M 13 6598k 0 1803k 0 0:00:26 0:00:03 0:00:23 18
=> => # 02k
```

```
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
myspc                corretto11          5eb3d9bb3de7       About a minute ago 499MB
amazoncorretto      11                 ebc51ffa390b       11 days ago        449MB
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker image tag myspc:corretto11 myspc:latest
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
myspc                corretto11          5eb3d9bb3de7       2 minutes ago      499MB
myspc                latest             5eb3d9bb3de7       2 minutes ago      499MB
amazoncorretto      11                 ebc51ffa390b       11 days ago        449MB
ubuntu@ip-172-31-35-3:~/myspc_correto$
```

- Now lets create a container docker container run -d -P --name spc1 myspc:corretto11

```
Azure AWS Windows PowerShell ubuntu@ip-172-31-35-3 ~
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker container run -d -P --name spc1 myspc:corretto11
0e55bfda8aaf4e7f2d1206d5a46df9589a8f4fad635fb9e3b1fd597da8fbed91
ubuntu@ip-172-31-35-3:~/myspc_correto$ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS
0e55bfda8aaf   myspc:corretto11  "java -jar spring-pe..."  4 seconds ago  Up 3 seconds
0.0.0.0:32770->8080/tcp, :::32770->8080/tcp   spc1
ubuntu@ip-172-31-35-3:~/myspc_correto$
```

- Approach 2: Start from some os

```
FROM alpine:3
RUN apk add openjdk11
RUN wget https://referenceapplicationskhaja.s3.us-west-2.amazonaws.com/spring-petclinic-2
EXPOSE 8080
CMD ["java", "-jar", "spring-petclinic-2.4.2.jar"]
```

- Build the image

```
Azure AWS Windows PowerShell ubuntu@ip-172-31-35-3 ~
ubuntu@ip-172-31-35-3:~/myspc_alpine$ nano Dockerfile
ubuntu@ip-172-31-35-3:~/myspc_alpine$ docker image build -t myspc:alpine .
[+] Building 2.7s (4/6)
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 236B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/alpine:3 0.7s
=> CACHED [1/3] FROM docker.io/library/alpine:3@sha256:124c7d2707904eea7431fffe91 0.0s
=> [2/3] RUN apk add openjdk11 1.9s
=> => # (3/32) Installing p11-kit (0.24.1-r1)
=> => # (4/32) Installing libtasn1 (4.19.0-r0)
=> => # (5/32) Installing p11-kit-trust (0.24.1-r1)
=> => # (6/32) Installing ca-certificates (20220614-r4)
=> => # (7/32) Installing java-cacerts (1.0-r1)
=> => # (8/32) Installing openjdk11-jre-headless (11.0.18_p10-r0)
```

```
[+] Building 36.7s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 236B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/alpine:3
=> CACHED [1/3] FROM docker.io/library/alpine:3@sha256:124c7d2707904eea7431
=> [2/3] RUN apk add openjdk11
=> [3/3] RUN wget https://referenceapplicationskhaja.s3.us-west-2.amazonaws
=> => exporting to image
=> => exporting layers
=> => writing image sha256:1040021b212ccca5304d3d6702b0f109dc73a5202ddcbbb8
=> => naming to docker.io/library/myspc:alpine
ubuntu@ip-172-31-35-3:~/myspc_alpine$ docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
myspc         alpine    1040021b212c   12 seconds ago  327MB
myspc         corretto11 5eb3d9bb3de7   13 minutes ago  499MB
myspc         latest    5eb3d9bb3de7   13 minutes ago  499MB
alpine        latest    9ed4aefc74f6   10 days ago    7.05MB
amazoncorretto 11        ebc51ffa390b   11 days ago    449MB
ubuntu@ip-172-31-35-3:~/myspc_alpine$
```

- Lets run the container docker container run -d -P --name myspc2 myspc:alpine

```
Azure AWS Windows PowerShell ubuntu@ip-172-31-35-3 ~
ubuntu@ip-172-31-35-3:~/myspc_alpine$ docker container run -d -P --name myspc2 myspc:alpine
d1007d298d5632fe305ca3b1c923383c7f6dda3e91b9ec4b97e88c8da776b230
ubuntu@ip-172-31-35-3:~/myspc_alpine$ docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS
d1007d298d56   myspc:alpine    "java -jar spring-pe..."  7 seconds ago  Up 5 seconds
0.0.0.0:32771->8080/tcp, :::32771->8080/tcp   myspc2
dcd49547f7da   alpine          "/bin/sh"                7 minutes ago  Up 7 minutes
0.0.0.0:30000->8080/tcp, :::30000->8080/tcp   sharp_gates
0e55bfda8aaf   myspc:corretto11  "java -jar spring-pe..."  10 minutes ago  Up 10 minutes
0.0.0.0:32770->8080/tcp, :::32770->8080/tcp   spc1
ubuntu@ip-172-31-35-3:~/myspc_alpine$
```

Immutable Infrastructure

- Any infra changes will not be done on infra directly rather we create some infra as code option and change the configuration.

Leave a Reply

Enter your comment here...

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)



About continuous learner

devops & cloud enthusiastic learner

[VIEW ALL POSTS](#)

◀ [PREVIOUS POST](#)

[Azure DevOps Book](#)

NEXT POST

**Grooming Classroomnotes
11/Apr/2023**

POWERED BY [WORDPRESS.COM](#).