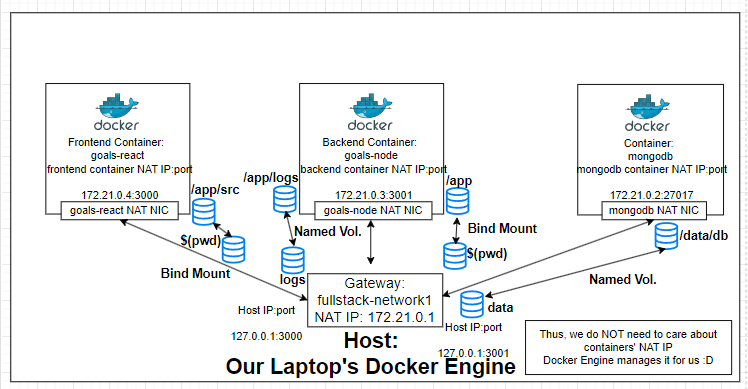
=== 6. Docker Compose

92. Module Intro

Previously, our demo project



Previously, we have to implement several steps separately across Docker Containers

e.g.

Creating a network

docker network create

Running different containers separately

Docker-Node-MongoDB-React > 6_dockerCompose > docker-cornrnands-txt 
4 
10 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
23 
24 
25 
26 
27 
29 
30 
31 
32 
33 
34 
35 
36 
Create Network 
docker network create goats—net 
Run XongoDB Container 
docker run —name mongodb 
—v data:/data/db 
—network goats—net 
mongo 
Build Node API Image 
docker build —t goats—node . 
Run Node API Container 
docker run —name goats-backend 
-e NONGODB_USEF'UXE-max 
-e NONGODB PASSWORD-secret 
togs:/app/logs \ 
—v 'Users/naximiuanschvarzmuuer/devetopment/teaching/udemy/dt 
app \ 
—v 'app/node_modules 

**Using Docker-compose**

Automating mutli-container setups

Intro to Docker compose

Using Docker compose

93. Docker-Compose: What & Why?

To skip

docker build ...

docker run ...

One Configuration File to Orchestrate commands

(build, start, stop, ...)

Using 1 command to run .yaml

Docker Compose does NOT replace Dockerfiles for custom images

Docker Compose works along with Dockerfiles

Docker Compose does NOT replace Images/Containers

Docker Compose is NOT suited for managing multiple-containers on **different hosts** (VMs)

**Writing Docker Compose files**

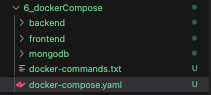
* Services (Containers)
* Publised Ports
* Environment Variables
* Volumes
* Networks

94. Creating a Compose file

Working directory

v DOCKER-NODE-MONCODB-RE. 
> 2_images 
> 3_managingData 
> 4_Networking 
> 5 _ Multi-containers 
v 6_dockerCompose 
> backend 
> frontend 
> mongodb 
> 
> 
> 
S- docker-commands.txt 
kali-docker 
nodejs-app-starting-setup 
prerequisites 

Create a docker-compose.yaml file in projectFolder



docker-compose.yaml is Case-sensitive &

indentation sensitive

6_dockerCompose > dock«-compose.yaml 
2 
# Version of Docker—compose 
Version: "3.8" 
Serv ices: 
# direct child service strict 
mongodb: 
backend: 
frontend: 

**\*\*** Service name defined under services: with 1 indentation

can be used for inter-connection inside Docker Engine

services:

**mongodb**:

backend:

frontend:

# Start by specifying docker-compose file version

version: "3.8"

# Start specifying services

services:

mongodb:

image: 'mongodb'

volumes:

* data:/data/db

6_dockerCommse > dock«-compose.yaml 
# Version Of Docker—compose 
version: "3.8" 
services: 
# direct child service strict indentation 
mongodb: 
docker container instances & need to specify detach mode 
7 
10 
11 
12 
•mongo 
vo unes 
docker run —name 
—v data: /d.ta/db —rm —d —ne!ypr.k 
"NGO INITDE -e 
mo o: latest • 
backend: 
goats—backend —v —v logs :/app. 
docker run —name 
node nodules —r' 
—d —p 3001: 3001 —network fullstack—network 
frontend: 

# Start by specifying docker-compose file version

version: "3.8"

services:

mongodb:

image: 'mongodb'

volumes:

* data:/data/db

environment:

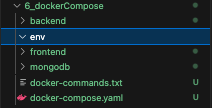
MONGO\_INITDB\_ROOT\_USERNAME: root

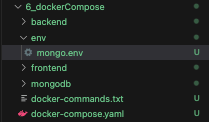
MONGO\_INITDB\_ROOT\_PASSWORD: rootPassword

6_dwkerCompose > 
6 
7 
8 
9 
10 
11 
12 
1B 
15 
Version of Docker-compose 
version: • 
services: 
g direct child service strict indentation 
mong odb: 
docker container ins tances need to specify detach mode 
image: 'mongo 
vo lumes: 
May have multiple volumes 
hostVolume: 'containerVolumepath 
— data: 'data/db 
Qwironmet": 
2 kinds 01 
syntax 
YONGO INITD8 
ROOT USERN"E: 
MONGO INITDB ROOT PASSWORD: 
root 
rootPassword 

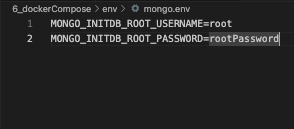
**Adding a rootDir/env folder**

for specifying env variables





rootDir/env/mongo.env



**Specifying .env file for a service**

version: "3.8"

services:

mongodb:

image: 'mongodb'

volumes:

* data:/data/db

**env\_file**:

- ./env/mongo.env

6 
11 
12 
13 
14 
15 
# Version of Docker—compose 
version: 03 80 
services: 
# direct child service strict indentatiol 
mongodb: 
docker container instances need to sp. 
Image: •mongo 
vo tunes : 
4 May have multiple volumes 
hostVOIume:/containerV01umePath 
— data:/data/db 
env t: 
# 2 kinds of syntax 

We may but do NOT need to specify a network

network:

* networkName

6_dockerCommse > dock«-compose.yaml 
# Version Of Docker—compose 
version: "3.8" 
services: 
# direct child service strict indentation 
mongodb: 
docker container instances & need to specify detach 
image: •mongo 
vo lunes 
May have multiple volumes 
# hostVolume:/containerVoLumePath 
— data:/data/db 
environment: 
10 
12 
13 
14 
15 
16 
17 
19 
4 
2 kinds of syntax 
vs 
# MCNGO INITDB ROOT USEFW.YE: 
root 
: rootPassword 

docker-compose.yaml

# Version of Docker-compose

version: "3.8"

services:

# direct child service => strict indentation

mongodb:

image: 'mongo'

volumes:

# May have multiple volumes

# hostVolume:/containerVolumePath

- data:/data/db

# environment:

# 2 kinds of syntax

#- MONGO\_INITDB\_ROOT\_USERNAME=root

# vs

# MONGO\_INITDB\_ROOT\_USERNAME: root

# MONGO\_INITDB\_ROOT\_PASSWORD: rootPassword

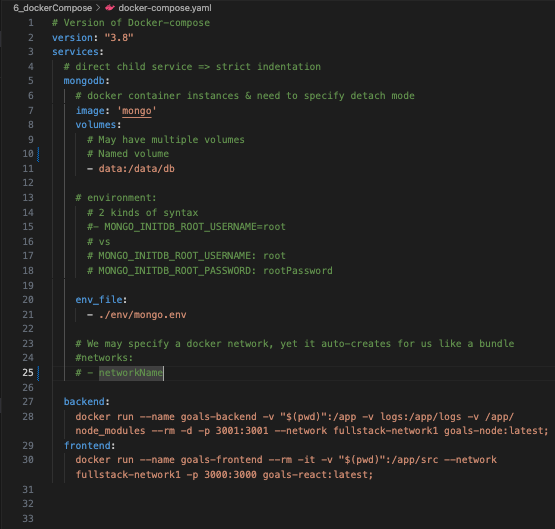
env\_file:

- ./env/mongo.env

# We may specify a docker network, yet it auto-creates for us like a bundle

#networks:

# - networkName



Named volumes can be specified in parallel to indentation level of services:

services:

mongodb:

backend:

frontend:

# Named volumes, no need to specify

volumes:

**# \*\* Leave it as it is, all services share this named volume as data:**

**# Do not specify Anonymous Volumes & Bind Mounts here**

**# Anonymous Volumes e.g. node\_modules**

**# Bind Mounts $(pwd):/containerPath**

data:

2 
4 
6 
10 ! 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
23 
24 
26 
21 
28 
* Version of Docker—compose 
version: "3.8" 
services: 
direct child service strict indentation 
nongodb : 
docker container Instances & need to specify detach mode 
image: •mongo ' 
vo lumes: 
May have multiple volumes 
Named vo lume for MongoDB 
— data:/data/db 
envi ronnent: 
2 kinds of syntax 
MCNGO INITOB ROOT USERNAME-root 
vs 
INITDB ROOT IJSE*O.YE: 
INITDB ROOT PASSWORD: 
env_f ile: 
— ./env/%ngo.env 
root 
root Password 
# We may specify a docker network, 
•networks: 
- networkName 
yet it auto—crea tes 
lor 
backend: 
docker run 
--name goals—backend 

95. Installing Docker Compose on Linux

On Linux machines, you need to install it separately.

These steps should get you there:

1. sudo curl -L "[https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)](https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname%20-s)-$(uname%20-m))" -o /usr/local/bin/docker-compose

2. sudo chmod +x /usr/local/bin/docker-compose

3. sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

4. to verify: docker-compose --version

96. Docker Compose Up & Down

Coming up with our own docker-compose.yaml

# Version of Docker—compose 
version: 03 80 
services: 
# direct child service strict indentation 
mongodb: 
docker container instances 
Detach node to be specified docker—compose up -d 
image: •mongo: latest' 
vo tumes : 
A microservice may have multiple volumes 
12 
13 
14 
15 
16 
17 
20 
21 
22 
23 
24 
25 
26 } 
27} 
28 
30 
32 
33 } 
34 
36 
38 t 
39 i 
Named vo lume for MongoDB 
— data:/data/db 
env_f i le: 
— ./env/mngo. env 
We may specify a docker network, 
*networks : 
— networkName 
backend: 
image: •node: latest ' 
vo tunes : 
— . :/app 
— logs:/app/logs 
— /app/node_modules 
working_dir: /app 
co—and: npm start 
ports: 
- "3001 :3ee1" 
depends_on : 
— nongodb 
environmen t: 
frontend: 
image: •node: latest ' 
vo tunes : 
— .:/app/src 
yet it auto-creates 

docker-compose.yaml

# Version of Docker-compose

version: "3.8"

services:

# direct child service => strict indentation

mongodb:

# docker container instances

# Detach mode to be specified docker-compose up -d

image: 'mongo:latest'

volumes:

# A microservice may have multiple volumes

# Named volume for MongoDB

- data:/data/db

env\_file:

- ./env/mongo.env

# We may specify a docker network, yet it auto-creates for us like a bundle

#networks:

# - networkName

backend:

image: 'node:latest'

volumes:

- .:/app

- logs:/app/logs

- /app/node\_modules

working\_dir: /app

command: npm start

ports:

- "3001:3001"

depends\_on:

- mongodb

environment:

- NODE\_ENV=development

frontend:

image: 'node:latest'

volumes:

- .:/app/src

- /app/node\_modules

working\_dir: /app/src

command: npm start

ports:

- "3000:3000"

depends\_on:

- backend

# Named volumes parallel to services indentation level

# Anonymous Volumes & Bind Mounts are NOT specified here

volumes:

data:

logs:

To start Docker-compose services

First, navigate into our projectFolder, where

projectFolder/docker-compose.yaml sits

,'llsers,' 11.q1 lv/DesktoWvs code/Docker 
ο 11.qilv$ I 

Before docker-compose up

**Let's remove all abandoned docker images**

docker image prune -a;

• j docker image prune —a; 
This alt images without at least one containe 
Are you sure you want to continue? (YIN) y 
Total reclaimed space: •8 

**Let's remove all abandoned docker containers**

docker container prune;

• jiaqitv$ docker 
This remove att stopped containers. 
Are you sure you to continue? y 
Deleted Containers: 
e3b1b2 t e t ba leb8577c1 

Ensure there're no running containers

• liaqilvS docker ps; 
ID IRAGE 
CREATED STATUS 

docker-compose up

• j docker—compose 
19.Ί9 
backend Pulled 
, fronteM pulled 
6d11t181ebb3 
2b238499ec52 
d71ate•e8bbd 
, 922N9a2a9b7 
9859652.3720 
6e59tM5818e 
9c2163a8c555 
6188Ι5ΙΙ6Μ5 
a4dc.la887e40 
, e7dca2d7d2bl 
3dt8.6b714b3 
Vollne 
pull 
pull 
ΡΙΑ Ι 
ΡΙΑ Ι 
pull 
ΡΙΑ Ι 
ΡΙΑ Ι 
ΡυΙΙ 
pull 
pull 
ΡΙΑ Ι 
ΡΙΑ Ι 
pull 
ΡΙΑ Ι 
ΡΙΑ Ι 
ΡυΙΙ 
leTe 
C omp lete 
lete 
leTe 
C omp lete 
lete 
leTe 
lete 
leTe 
C omp lete 
lete 
leTe 
C omp lete 
lete 
leTe 
lete 
treaTed 

Docker Network that takes projectFolder name 6\_docker\_compose\_default

is auto-created

• jiaqiLv$ up; 
19/19 
backend pulled 
fronteod Putted 
6.d11c181ebb3 
Putt 
Pult 
2b238499ec52 
Putt 
41b754dØ79e8 
d71aceee8bbd 
Pult 
d629be75&933 
Putt 
922b"a2a9b7 
98596520728 Putt 
252cN91c82c Pun 
mongodb Put led 
€e59cbe5818e Putt 
9c2163a8c555 
Putt 
97ee%6ae366 
618815 lc6de5 
Pult 
a4d c 4.887 e4b 
Putt 
e7dcd2d7d2b1 
3df8a6b714b3 
c71e6ea971f7 
Pult 
tete 
tete 
Cor. Le te 
tete 
Cor. Le te 
tete 
tete 
tete 
Cor. Le te 
tete 
com tete 

Starting docker-compose in detached mode

• j iaqilvS docker—cmose —d; 
fusers/j 
Obso Lete 
R„nnLmg 3/3 
Container sta 

Will be able to netcat MongoDB port



To remove all docker-composed microservices with volumes

docker-compose down -v

• j LaqiLvS docker compose down —v: 
WARN [eeeøl 
obso tete 
Running 2n 
Volume Removed 
Volume 

97. bash - docker prune

We'll be using this bash to start fresh everytime

we do docker-compose

projectFolder/prune-before-docker-compose-up.sh

2 
4 
6 
9 
10 
11 
12 
13 
14 
15 
16 
17 
18 
19 
20 
21 
22 
# ! 'bin/ bash 
# Stopping all runn Ing docker containers 
docker stop "docker ps -q) 
echo "Currently running 
containers: 
docker ps -a; 
# Force stop al L docker 
containers 
docker kill "docker ps —q) 
echo "Current ty running 
containers: o; 
docker ps -a; 
# pruning all existing docker image to avoic 
docker image prune -a; 
echo "Cur rent docker images: 
docker image Is; 
# Pruning alt abandoned docker containers tc 
docker container prune; 
echo "Currently running containers: 
docker ps -a; 

#!/bin/bash

# Stopping all running docker containers

docker stop $(docker ps -q)

echo "Currently running containers: ";

docker ps -a;

# Force stop all docker containers

docker kill $(docker ps -q)

echo "Currently running containers: ";

docker ps -a;

# Pruning all existing docker image to avoid conflicts

docker image prune -a;

echo "Current docker images: ";

docker image ls;

# Pruning all abandoned docker containers to avoid conflicts

docker container prune;

echo "Currently running containers: ";

docker ps -a;

# Remove all docker volumes spawn by docker-compose

docker-compose down -v;

# Force remove all docker volumes

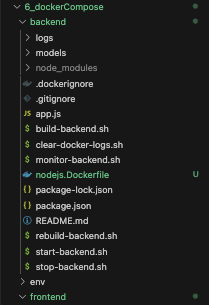
docker volume rm $(docker volume ls -q);

echo "Current docker volumes: ";

docker volume ls;

98. Docker composing MongoDB & Node - Working with multiple containers

backend directory



**Backend Node service**

We can specify how a service creates itself:

1. image: Pulls a Docker image from DockerHub

27 
29 
32 
34 
35 
36 
37 
39 
backend: 
'node • 
container_name: node—goa Is 
*bui [d: 
. 'backend 
* context: 
# dockerf ile: node) s. Dockerlile 
# args: 
BACKEND PORT: yeøl 
/ app 
con—and: npm start 
ports: 
- '3øø1:3øø1' 
vo lumes: 
— togs: /app/togs g Named Volume 
— ./backend:/app g Sharing everything in 
• backend • 
tolder to 

1. build: Build a container using ./backend/nodejs.Dockerfile

26 
27 
28 
29 
30 
31 
32 
33 
34 
35 
36 
37 
39 | 
backend: 
"image: •node 
container_name: node—goats 
991 q: 
context: ./backend 
dockerfile: nodejs.Dockerfite 
arg" 
BACKENO_PORT: 3eø1 
working_dir: /app 
co—and: npm Start 
ports: 
- '3001 :3em• 
vo tunes : 
— logs:/app/logs Naned Volume 
— ./backend:/app # Sharing everything in 
• backend • 
folder to 

frontend directory

v 6—dockerConvose 
> backend 
> env 
v frontend 
> cornpments 
js App* 
js ir&x.jS 
bug fix-frontend.sh 
build-frontend.sh 
S monaor-frontend.sh 
packa*jsm 

mongodb directory

く 6 ー g0k260 コ も 0 い の 
・ 、 ヨ on 、 b 
) 竄 0k2 コ 0 
》 呂 t 雪 d 
3 舞 名 ー d の b .sh 
3 呉 早 d 言 b , ・ h 
3 、 一 一 と ー d ・ 富 ba い 卩 当 
3 ョ 0- 第 01d3 新 ba . 当 
の cl ・ ま ー 各 第 と bd ・ - , ・ プ 
3 ぎ = d ー 0 第 等 の ー . ・ h 

backend volumes

Named Volume

Bind Mount (sharing local files to containers)

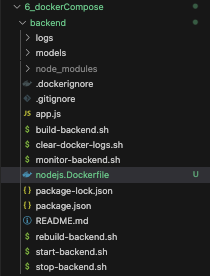
Anonymous Volume

docker-compose.yaml

26 
27 
28 
29 
31 
32 
33 
34 
35 
36 
37 
38 
39 
40 
backend: 
image 
node 
container_nane: node—goats 
build: 
context: . 'backend 
dockerfile: nodejs.DockerfiIe 
args: 
BACKENO_PORT: 3001 
•working_dir: lapp 
co—and: npm Start 
ports: 
- '3ee1:3øø1' 
•tunes: 
— logs: /app/logs g Named Vo lume 
— . 'backend: 'app g Sharing everything in 
• backend • 
folder to 

Referencing backend/nodejs.Dockerfile for

ENV variables



./backend/nodejs.Dockerfile

59 
60 
61 
62 
EXPOSE $BACKBO_PORT 
pav • 
ENV 

Create a backend.env file

./env/backend.env

6_d0CkerConvOSe 
) backend 

./env/backend.env

docker-corrpose.yznl M 
> env > O backend—v 
"KiODB_USEFOU-mE-root 

backend env\_file:

docker-compose.yaml

26 
27 
28 
29 
31 
32 
33 
34 
35 
36 
37 
39 
40 
backend: 
image 
node 
container_nane: node—goats 
build: 
context: . 'backend 
dockerfile: nodejs.Dockerfite 
args: 
BACKENO_PORT: 3001 
•working_dir: lapp 
co—and: npm Start 
ports: 
- '3ee1:3øø1' 
vo lumes: 
— logs: /app/logs Named Vo lume 
— . [backend: 'app g Sharing everything in 
• backend • 
folder to 

Now, we're done with services for mongodb & backend

docker-compose.yaml

4 
11 i 
12 
13 
14 
15 
16 
171 
18 
19 
20 
21 
22 
23 
24 
25 
26 
29 
30 
> docW-compose-yaml 
version: •'3.8'• 
services: 
direct child service strict indentation 
docker container instances 
Detach mode to be specif led docker-co•pose up -d 
mongodb: 
container_name: —ongodb—goals 
use either image: (pull image) OR bui [d: (using Docke 
image: • 'ongo• 
build: 
context: ./mongodb 
dockerfile: mongodb. Dockerfile 
vo lunes 
* A microservice may have multiple volumes 
# Named vo Lume for YongoDB 
— ./data:/data/db 
ports: 
- "27017 :27e17" 
env file: 
— ./env/%ngo.env 
We may specify a docker network, yet it auto—creates 
'networks : 
— networkName 
backend: 
gimage: •node • 
container_name: node—goats 
build: 
context: ./backend 
dockerf ile: nodejs.Dockerfite 
args: 
BACKEND PORT: seei 

MongoDB & Node services .yaml

version: "3.8"

services:

# direct child service => strict indentation

# docker container instances

# Detach mode to be specified docker-compose up -d

mongodb:

container\_name: mongodb-goals

# Use either image: (pull image) OR build: (using Dockerfile)

# image: 'mongo'

build:

context: ./mongodb

dockerfile: mongodb.Dockerfile

volumes:

# A microservice may have multiple volumes

# Named volume for MongoDB

- ./data:/data/db

ports:

- "27017:27017"

env\_file:

- ./env/mongo.env

# We may specify a docker network, yet it auto-creates for us like a bundle

#networks:

# - networkName

backend:

#image: 'node'

container\_name: node-goals

build:

context: ./backend

dockerfile: nodejs.Dockerfile

args:

BACKEND\_PORT: 3001

#working\_dir: /app

command: npm start

ports:

- '3001:3001'

volumes:

- logs:/app/logs # Named Volume

- ./backend:/app # Sharing everything in 'backend' folder to docker backend container

- /app/node\_modules # Anonymous Volumes

env\_file:

- ./env/backend.env

depends\_on:

- mongodb

**Try docker-compose up again**

docker-compose up -d;

• j iaqitv$ docker—cor.ose up —d; 
obsolete 
Running 18/18 
frontend pu I ted 
6d11c181ebb3 
20238499ec52 
41b7S4de79e8 
d71aceee8bbd 
d629be7%933 
922b&9a2a9b7 
9859652.372• 
252cb•91c82c 
pu led 
6e59cbe5818e 
9c2163a8cS55 
97eeed6ae366 
6188151c6dØ5 
a4dc4a887e4b 
e7dcd2d7d2b1 
3dt8a6b714b3 
c71e6ea971f7 
Building •.35 
Already exists 
A L ready exists 
Already exists 
Already exists 
A L ready exists 
Already exists 
Already exists 
AL ready exists 
Putt cMQtete 
Lete 
Pult cc—tete 
Putt comlete 
Pu Lete 
Pult cc—tete 
Putt comlete 
Pu Lete 
(21/21) FINIS&ED 
(backer"' Internet I detlnltlon trom nodejs.oockertlte 
transferring docket tile: 
(backend internall metadata for docker.io/llbrary/node: latest 
(backend internall . dockerrgnore 
transfer ring context: 688 
(backend ducker. 10/1 Ibrary/node: Latest 
(backend internall context 
transferring 
context: 
CACHED 
CACHO 
CA CHED 
CACHO 
CA CHED 
CACHO 
CA CHED 
CACHEO 
(backena 
(backend 
[ kend 
(backena 
(backend 
[ kend 
(backena 
(backend 
[ kend 
Wackena 
[backend 
3/ 161 
copy package. Ison . 
RLt• 
RLt• 
RLt• 
echo "Hostname Of this Contarn,er: node—goals" 
echo node—goals 
ngm Instau 
/ app/host 
apt —ge t 
apt —get 
apt—get 
apt —ge t 
apt —get 
apt—get 
update apt —y upgröde 
Install 
-y Inetutlls-tools 
Install 
-y reutlls—grng 
InstaLL 
—y net—toots 
Install 
—Y telnet 
Install - 
netcat-tr.dltlonat 

Verify MongoDB by netcat

nc -v 127.0.0.1 27017;

Verify Backend Node by netcat

nc -v 127.0.0.1 3001;

11.qilv$ —v 
connecTlon το 127.0.e.l ροττ 27117 [TcV*] succe. 
—C 
j iaqilvS 0c —v ΙΣ 

mongo

• 6_dockeÆompose j docker ps —a 
ct%TAINER ID IRAGE 
8e66ea99rea3 node 
react—goa Is 
193a1ö265t66 
e: 3ee1.3ee1/tcp node.-goaLs 
"docker—entrypoint. 
•idocker—ent rypoint. 
CREATED 
2 minutes ago 
2 minutes ago 

Now, we can try get into Backend Node container &

ping mongodb-goals container

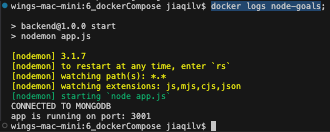
docker exec -it node-goals bash;

ping mongodb-goals;

"ec —it 
rootea1e26c919aa:/app• ping mongodb—goaLs 
(172.23.e.2) 56184) bytes of data. 
64 bytes trom mongodb—goats.6 (172.23.e.2): icr._se 
64 bytes from (172.23. e. 2): ic._se 
— mngodb—goats ping statistics 

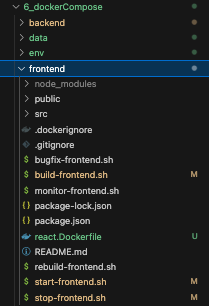
Can also inspect docker logs to inspect DB connection inside Docker Engine

docker logs node-goals;



99. Docker composing Frontend (React) - Adding another container

frontend directory



using ./frontend/react.Dockerfile

6_dockerCommse dockg-compose.yaml 
version: "3.8" 
Services: 
23 
24 > 
44 
46 
47 
49 
so 
51 
52 
53 
56 
57 
58 
# serv ice name 'mongodb• can be used in cod 
nongodb: 
backend: 
frontend: 
image: •node' 
container _ name: react—goals 
build: 
context: ./frontend 
@ckerfiYé: react.Dockerfite 
vo tumes : 
— ./frontend'src:/app/src 
— /app/src/node_nodules 
working_dir: /app/src 
co—and: npm Start 
ports: 
- "3eøø:3eee•• 
depends_on : 

Bind mount frontend/src folder to containerPath

6_dockerCotnpose > e docket-cornpose-yaml 
1 version: 3 8 
se rv ices: 
23 
24 > 
44 
45 
46 
47 
51 
52 
54 
55 
56 
57 
service name •mongodb' can be used in coc 
mongodb: 
backend: 
f rontend: 
image 
node 
container_name: react-goats 
build: 
context: . / f rontend 
dockerfile: react.DockerfiIe 
vo lumes: 
— ./frontend/irq: # Bind Mount 
— /app/src/node_modules 
working_dir: 'app/'" 
co—and: npm start 
ports: 
- "3øøø: 
depends _ on : 

**Running frontend service in interactive mode**

services:

frontend:

...

**# To allow React run interactively**

stdin\_open: true

tty: true

6_dockerCommse 
23 
24 > 
44 
46 
47 
49 
50 
51 
52 
53 
56 
57 
58 
59 
61 
version: "3.8" 
services: 
# serv ice name 'mongodb• can be used in cod 
nongodb: 
backend: 
frontend: 
gimage : 
node 
container_name: react—goals 
build: 
context: ./frontend 
dockerfile: react.Dockerfile 
volumes: 
B 2nd Mount 
— ./frontend/src:/app/src 
co—and: npm Start 
ports: 
- oeøø:seee• 
docker run —it interactive mode 
Stdin_open: true # frontend needs an open 
tty: true # 
depends on: 

docker-compose.yaml

version: "3.8"

services:

# service name 'mongodb' can be used in code for connections inside Docker Engine

mongodb:

container\_name: mongodb-goals

# Use either image: (pull image) OR build: (using Dockerfile)

# image: 'mongo'

build:

context: ./mongodb

dockerfile: mongodb.Dockerfile

volumes:

# A microservice may have multiple volumes

# Named volume for MongoDB

- ./data:/data/db

ports:

- "27017:27017"

env\_file:

- ./env/mongo.env

# We may specify a docker network, yet it auto-creates for us like a bundle

#networks:

# - networkName

backend:

#image: 'node'

container\_name: node-goals

build:

context: ./backend

dockerfile: nodejs.Dockerfile

args:

BACKEND\_PORT: 3001

working\_dir: /app

command: npm start

ports:

- '3001:3001'

volumes:

- logs:/app/logs # Named Volume

- ./backend:/app # Bind Mount - Sharing everything in 'backend' folder to docker backend container

- /app/node\_modules # Anonymous Volumes

env\_file:

- ./env/backend.env

depends\_on:

- mongodb

frontend:

#image: 'node'

container\_name: react-goals

build:

context: ./frontend

dockerfile: react.Dockerfile

volumes:

# Bind Mount

- ./frontend/src:/app/src

command: npm start

ports:

- '3000:3000'

# docker run -it interactive mode

stdin\_open: true # frontend needs an open input connection

tty: true #

depends\_on:

- backend

# Named volumes parallel to services indentation level

# Anonymous Volumes & Bind Mounts are NOT specified here

volumes:

data:

logs:

**Test Docker-compose again**

docker-compose up -d;

'users/ j 
obsolete 
Building 5.35 (44/44) FINISÆD 
(mcngcdb Internall Load bulld definition from mongodb.DockerfLLe 
transferring dackertlle: 
(mongoab internall metadata tor docker.lo/llbrary/mongo: latest 
(mongodb Internall Load . dockerzgnore 
transferring context: 688 
CACHED 1/11 cocker. 
(mongcdbl exportzng to Image 
exearting layers 
writing Image sna256:e5b8954t4e3a7raa9t13a317c7ede15dc1291b84a96e8e2eee3d3ca148å 
naming to 
(backend internall de tinltlon trom node]s.Dockertile 
transferring dockertlle: 1.6ekB 
(t rantend Internal) Load rnetadata tor docker. io/ILbrary/node: Latest 
(backend internall load . dackerlgnore 
transferring context: 68B 
(t rantend FROE docker.Lo,'L1brary/node: 
resolve Cacker. 
(backend internall context 
transferring context: EUB 
CACHED 
CACHEO 
CA CHED 
CACHED 
CACHEO 
CA CHED 
CACHED 
CACHEO 
CA CHED 
CACHED 
CACHEO 
CA CHED 
CACHED 
CACHEO 
CA CHED 
(trontend 2/1El / app 
( backend 
[ kend 
(backena 
( backend 
[ kend 
(backena 
( backend 
[ kend 
Wackena 
[backend 
[ kend 
Wackena 
[backend 
[ backend 
3/ 161 
copy package. Ison . 
RLt• 
RLt• 
RLt• 
RLt• 
echo "Hostname Of this Contarn,er: node—goals" 
echo node—goals 
ngm instau 
/app/hos• 
apt —get 
apt —get 
apt—get 
apt —get 
apt —get 
apt—get 
apt —get 
apt —get 
apt—get 
update apt —y upgröde 
Install 
Install 
InstaLL 
Install 
Install 
InstaLL 
Install 
clean 's 
-y Inetutlls-tools 
-y reutlls—grng 
—y net—toots 
—Y telnet 
-y netcat-tr.dltlonat 
—y coreutLLs 
—y busynx 
rm —rt 'var/llb/apt/tlsts/. 
(backend] exporting to 
exeortxng layers 
cop,' . . 
Image 
wrltrng Image sna25€:5e9e752dcnee38nb3187b2fc217b3dc69aeea6d9e€d9d9267ed433276, 
naming to Cocker. La/ 
(trontena internal) load build detrnitLon trom react.Dockert1te 
transferring docket tile: 
(trontena Internal I load .dockerignore 
transferring context: 68B 
(t rantend Internal) Load build context 
transferring context: 
CACHEO 
CA CHED 
CACHED 
CACHEO 
CACHED 
CACHED 
(trontend 
[t run tend 
(trontend 
(trontend 
[t run tend 
(trontend 
3/161 
4/181 
5/161 
6/161 
7/181 
8/ LEI 
COPY Rckage.)son . 
echo ' 'Hostname Ot this Container: react—goats" 
echo react—goals 
nom Install 
act—get update apt u0grade 
apt—get install —y Lnetutils-tools 

Now, all Frontend, Backend, MongoDB services are running

• jiaqiLv$ ps —a 
CCNTAINER ID INAGE 
d1441a6e3613 
rea L s 
t cp 
14ca3dcA5dcc 
"docker—entrypoint. 
•idocker—ent rypoint. 
CREATED 
6 seconds 
6 seconds 
ago 
a-go 
STATU 
up 6 

Testing React frontend

<http://localhost:3000>

C 
@ localhost:3000 
New Goal 
Add Goal 
No goals found. Start adding so 

Reloading the page after adding something does work

@ localhost:3000 
New Goal 
Add Goal 

100. Building Images & Understanding Container Names

docker-compose --help;

• j —he 
usage: docker (OPTITeSl 
Define and rtm multi—container applications with Docker 
—ansi string 
—compat 1b i City 
—d ry—run 
—env—fLLe stringArray 
—tite strÜ"Array 
—parallel int 
—profite stringArray 
—progress string 
—project—directory string 
—P, —project—nne string 
Control to print "61 control character! 
(default "auto") 
Run coroose in backward compatibility mode 
Execute co«mand in dry run Nde 
Specify an alternate enviroment 
Compose configuration tiles 
Control max parallelism, —1 for unlimited (d. 
Specify a profile to enable 
Set type ot progress output (auto. tty, ptai' 
Specify an alternate working directory 
(default: the path of the. first specified. 
project nuæ 
Comands 
attach 
build 
con t 
cp 
c reate 
events 
images 
togs 
pause 
ps 
put L 
push 
restart 
Attach locat standard Input. output, and error strens to a servil 
Build or rebuild services 
Parse, resolve and render c.ose file in canonicat format 
Copy files/folders between a service container and the locat tile! 
Creates containers tor a service 
Stop and remove containers, networks 
Receive reat time events from containers 
Execute a cMmand in a running container 
List images used by the created containers 
Force stop service containers 
output from containers 
List projects 
pause services 
Print the public port for a prt binding 
List containers 
putl service images 
Push service images 
Restart service containers 
Removes st 
service containers 

docker-compose build = Build or rebuild services

**In case, there's any code changes either from**

**Frontend or Backend**

**Docker build all images again**

docker-compose up --build;

or

if we just wanna rebuild images without docker-compose up

docker-compose build;

• j LaqiLvS docker—co.ose build: 
WAR* [eeeøl 
obso tete 
building 4. Is (44/44) FINISHED 
('song oft Internall load aetlnltion from nongodb.Dockertile 
transferring dockertlle: 26eB 
[nongodb Internall load metadata tor docker. W Library/rongo: Latest 
(r.ongoab Internall load 
transferring context: 68B 
CACHED 1/11 FROE docker.10,'L1brary/mongo: 
(r.ongcabl eWarting to Image 
exportLng layers 
writing image 
naming to docker. 
(oackena Internall toad build detlnrtlon trom nodeJs.Dockertite 
transferring dockerfLLe: I.EdkB 
(trontemd internall metadata for docker.io/llbrary/node: latest 
(oackena Internall toad . dockerignare 
transferring Context: 68B 
(trontemd ducker. 
resolve cocker. 
[backend Internall Context 
transferring context: 
CACHED 
CACHED 
CACHED 
CACHED 
CACHED 
(t rontena 
(backend 
(backend 
(bac 
(backend 
(backend 
(backend 
3/161 
5/161 
6/161 
8/161 
WORX.01R 
COPY package. j Son . 
Rut' echo "hastmame ot this Container: node—goats 
RI." echo Is 
Rt_t• ng«n InstaLI 
apt-get u>date SS apt -y upgrade 
Rlut• apt—get Install -y Inetutlts-toots 
" 'app/hosl 

Services naming in docker-compose.yaml

services:

frontend:

**container\_name**:

23 
24 > 
45 
46 
47 
48 
52 
53 
54 
58 
60 
version: "3 8" 
services: 
# service nane •mongodb' can be used in cod 
mongodb: 
backend: 
frontend: 
*image: •node ' 
react-goals 
build: 
context: ./frontend 
dockerf i le: react. 
vo lunes : 
Bind Mount 
— ./frontend/src:/app/src 
Co..and: npm start 
ports: 
- '3eeø:3eee• 
docker run —it interactive mode 
stdin_open: true # frontend needs an open 
tty: true 
depends_on : 

• jiaqitv$ 
d1441a6e3613 
e/ t cp react—pa Is 
14 ca3dc9Sdcc 
I/tcp node—goa ts 
ac8c6e3e143a 
e 17/ t cp nongodb—goa t s 
- jiaqitv$ 
docker ps —a; 
"docker—entrypoint. 
" docker—entrypoint. 
"docker—entrypoint. 
11 minutes 
minutes ago 
11 minutes a-go 
up 
up 
up 

=== 7. Working with "Utility Containers" & Executing commands in containers

103. Module introduction & what are "Utility containers"?

We've been working with

**Application containers**

Environment

<Your App>

docker run myapp

Runs CMD & starts app

**Utility Containers**

Environment

docker run mynpm init

Executes / Appends custom command

104. Utility containers - Why use?

Tuesday, 1 October 2024

11:36 PM

.gitignore

# Logs

logs

\*.log

npm-debug.log\*

yarn-debug.log\*

yarn-error.log\*

# Runtime data

pids

\*.pid

\*.seed

\*.pid.lock

# Directory for instrumented libs generated by jscoverage/JSCover

lib-cov

# Coverage directory used by tools like istanbul

coverage

\*.lcov

# nyc test coverage

.nyc\_output

# Grunt intermediate storage (https:www.npmjs.com/package/grunt)

.grunt

# Bower dependency directory (https: bower.io)

bower\_components

# node-waf configuration

.lock-wscript

# Compiled binary addons (https:nodejs.org/api/addons.html)

build/Release

# Dependency directories

node\_modules/

jspm\_packages/

# Typescript v1 declaration files

typings/

# Optional npm cache directory

.npm

# Optional eslint cache

.eslintcache

# Optional REPL history

.node\_repl\_history

# Output of 'npm pack'

\*.tgz

# Yarn Integrity file

.yarn-integrity

# dotenv environment variables file

.env

.env.test

# parcel-bundler cache (https: parceljs.org)

.cache

.parcel-cache

# Next.js build output

.next

out

# Nuxt.js build / generate output

.nuxt

dist

# Gatsby files

.cache/

# Comment in the public line in if your project uses Gatsby and not Next.js

# public

# vuepress build output

.vuepress/dist

# Serverless directories

.serverless/

# FuseBox cache

.fusebox/

# DynamoDB Local files

.dynamodb/

# TernJS port file

.tern-port

# IDE / Text Editor settings

.vscode/

.idea/

\*.swp

\*.swo

\*.sublime-work

105. Different ways of running commands in containers

# Keeping node container up & running but waiting for inputs

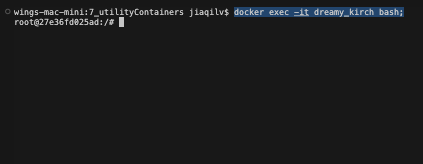
docker run -it -d node;

• wings-mac—mini: 7_utitityContainers j L.qitvi "cker rm —it 
unabte to find image •node: Latest • Locally 
latest: putting from library/node 
• Already 
6d11c181ebb3. 
2b238499ecs2: 
A ready 
• A I ready 
41b754de79e8. 
• Already 
d71aceee8bbd. 
d629be7se933: 
A ready 
• A I ready 
922b&9a2a9b7. 
• Already 
985%52a372ø. 
252cb&91c82c. 
• AL ready 
exists 
exists 
exists 
exists 
exists 
exists 
exists 

# List running docker containers

• wings-mac-mini: 7_utiLityContainers jiaqitv$ docker ps —a; 
cr«TAINER ID 
IRAGE 
27 e36tde25ad 
node 
About a minute 
wings—mac-mini: 7_utiLityContainers jiaqiLv$ 
ago 
STATUS 
About a 

# Hopping into docker container inside a shell environment



106. Building a Utility container

# Create Dockerfile for Utility container



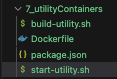
Dockerfile:

7_utilityContahers > Dockerfile ) 
4 
7 
# Extra slim Node image 
FRæ node: 
WORKDIR 'app 
# Give user full. control of 
*CMD npn init 

• utilityContainers jiaqitv$ ecker build —t 
Building 8.4; 16/6) 
(internall OuLlC de t 1 nLt10n trom Dockertlle 
transferring dockertlle: 148B 
[ Internall Load metadata for docker.lo/ IA—alpine 
(internall load 
transferring context: 2B 
[1/21 FROE 
resolve 
36.99" 
2.d3P8 
1.d3kB 
1.1EkB 
3.2EME 
extracting 
extracting 
extracting 
extracting 
(2/21 
export Lng to 
lavers 

# Making a docker container share files to our host machine

docker run -it -v $(pwd):/app node-util npm init;



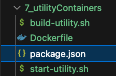
start-utility.sh

7_utilityContahers > S start-utity-sh 
3 
# ! /bin'bash 
docker run -It -v node—w 

bash ./start-utility.sh

• jiaqilvS bash ./start• 
This utility witt walk you through creating a package.js. 
It only covers the c—on items, and tries to guess 
See help Knit' for definitive doctnentation thesi 
and exactly What they do. 
use instatt epkgs• afterwards to install a package 
save it as a dependency In the package. j 
Press at any time to quit. 
package (app) utility 
version: (I.e.e) l.ø.e 
description: utility container testing 
entry point: (index. j s) 
test 
git repository: 
keywo rdS : 
autho r: 
license: 
About to write to 'app/package.json: 
"utility"', 
"version": 
"description": "utility container testing% 
"naLn": "index. j 

Files inside our container just mounts to our host machine



Running a Utility container

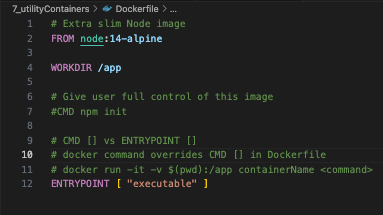
bash ./run-utility.sh

7_utilÄyContüers > S run-utilitysh 
S 
'!/bzn/bash 
container_nane• node—uti • ; 
—p ••Enter npm Comand [e.g. npm init): 
docker run —It —v :/app $container_nan 

107. Utilizing ENTRYPOINT

docker commands in terminal overrides

CMD [] in Dockerfile



ENTRYPOINT [ "npm" ]

We can append any npm commands

7_utilityContahers > Dockerfle > 
2 
4 
6 
9 
11 
12 
# Extra s Lim Node image 
FRI* 
WORKDIR lapp 
# Give user full control of this image 
*CMD 
CMD IJ 
# CMD IJ 
docker 
docker 
init 
vs ENTRYPOINT 
command overrides CBD [J in Doc 
run -it —v containe 

Let's rebuild our Utility container

• wings-mac—mini: 7_utitityContainers jiaqitv$ ecker build —t •f . 
Building 1.9S 16/6) 
(internall OuLlC de t 1 nLt10n trom Dockertlle 
transferring dockertlle: 362B 
[ Internall Load metadata for docker.lo/ IA—alpine 
(internall load 
transferring context: 2B 
[1/21 FROE 
CACHED (2/21 'app 
exporting to Image 
export Ing layers 
writing sha256:457e336agee712e2e1b4d4et434t98tt789S3e€e€btec922c313e89€e75! 

Running a docker container with entrypoint npm commands

bash ./run-utility.sh

7_utilityContahers > S run-utility.sh 
4 
6 
# ! /bin'bash 
*container name: •node—utili ; 
read -p "Enter docker containerName [e.g. node-utill 
read —p •'Enter ent rypoint npn conmand [e. 
g. install] 
docker run -it -v $container_name 

Our entrypoint command "init" was sent to

ENTRYPOINT [ "npm" ] in Dockerfile

• jiaqiLvS bash ./ 
Enter docker containerNne [e.g. node—util]: node—util 
Enter entrypoint n:" co«mand (e.g. install]: init 
This utility walk you through creating a package. j s' 
It only covers the rost cmmon items, and tries to guess 
See 'npm help init' for definitive documentation on 
and exactly what they do. 
use ' npm install epkg>' afterwards to install a package 
save it as a dependency in the package. json tile. 
Press —C at any time to quit. 
package (utility) node—utility 
version: (l.e.e) l.ø.e 
git repository: 
keywo rds : 
author: 
license: (ISC) 
About to write to 
"one" : "node—utility" 
"version": "l.ø.e". 
"description": "utility container testing% 
"main": 

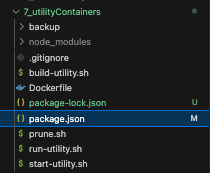
Using Utility ENTRYPOINT [ "npm" ] to install express

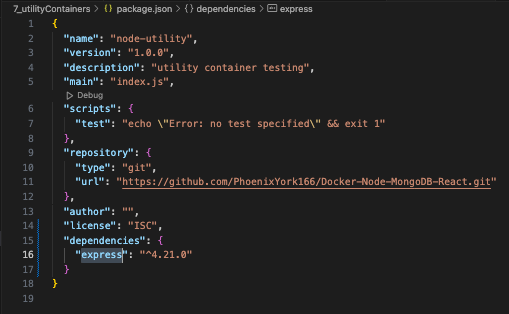
• 7_utiLityContainers jiaqitv$ bash ./nn—utili 
Enter docker containerNMæ (e.g. : node—util 
Enter entrypoint npm connand (e.g. instattl: install express 
t created a as package—lock. json. You 
express* 21. e 
added 65 packages from 41 contributors and 65 
13 packages are looking tor funding 
fund' for details 

Now our docker container runs commands =>

perform actions =>

bind files back to our host machine

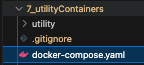




108. Docker compose for Utility container

Create docker-compose.yaml

inside 7\_utilityContainers



To allow standard input from users in terminal

inside

docker-compose.yaml

version: '3.8'

services:

utilty:

build:

context: ./utility

dockerfile: utility.Dockerfile

# Allow std\_in

stdin\_open: true

tty: true

> 0' docker-compose.yaml 
version: •3.8 • 
Services: 
utility: 
build: 
Node 
• IhtfEked 
3 
4 
6 
9 
10 
. ./utillty 
con text • 
dockerfile: utility.Dockerfite 
At low standard input from users in terminal 
stdin_open: true 
tty: true 
vo lumes : 

**Using docker-compose with utility containers**

docker-compose run <serviceName> <command>

In this case, we'd like to docker-compose run command

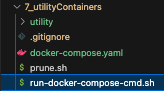
for serviceName 'utility'

docker-compose run utility <command>;

3 
4 
6 
9 
10 
> docker-compose.yaml 
version: •3.8 • 
services: 
utyl: 
build: 
dockerfile: utility.Dockerfite 
At low standard input from users in terminal 
stdin_open: true 
tty: true 
vo lumes : 

Let's write a simple bash to prompt users' input

for service\_name & npm\_command



./run-docker-compose-cmd.sh

> S run-docker-corn*'se-cndsh 
5 
read -p "Enter 
read -p "Enter 
docke 
servicd4ame le g 
. . utility]: Serv 
npm com-ands le g 
. . init install 
run —rm $service_name 

To start testing

Remove

projectFolder/utility/package.json

projectFolder/utility/package-lock.json

v utiäty 
> node_modules 
S build-utility_sh 
package-lock_ison 
S prune-sh 
$ run-utility Sh 
$ start-utilitysh 
WityDockerfile 

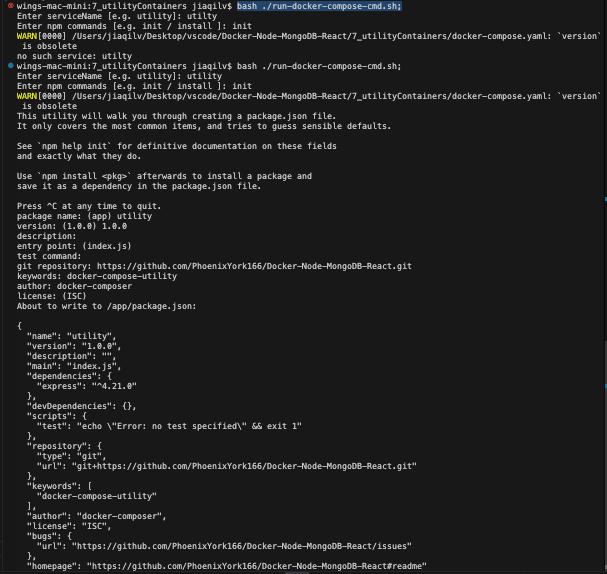
Now, we can docker-compose to project commands

into a specific service we'd like to create

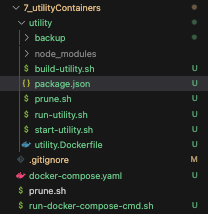
bash projectFolder/run-docker-compose-cmd.sh

Enter serviceName: utility

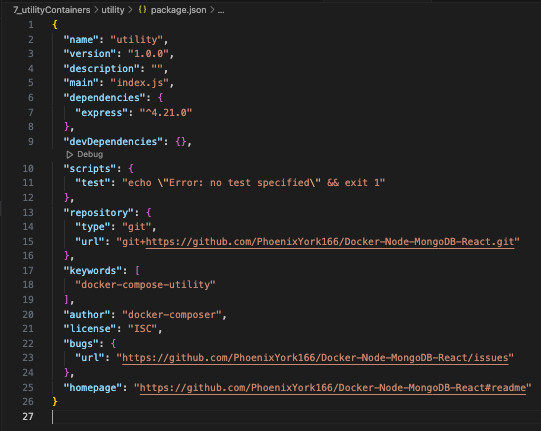
Enter npm\_command: init



A new package.json comes up in projectFolder/utility



projectFolder/utility/package.json



109. Utility containers, Permissions & Linux

In Linux, by default, Docker runs as "root"

Utility Containers' files got written to

"Bind Mount"

hostPath:/containerPath

with host machine "root" permission

On macOS & Windows, Docker is being used

within WSL2 (Linux sub-system),

the user mappings all happen automatically

due to Net File Share (NFS) mounts.

So, for example on Linux, if I do the following (as you described in the course): 
Dockerfile 
FROM node:14-slim 
WORKDIR /app 
$ docker build -t node-util:perm . 
$ docker run -it --rm -v $(pwd):/app node-util:perm npm init 
total 16 
drwxr-xr-x 3 scott scott 4096 Oct 3116:16 ./ 
drwxr-xr-x 12 scott scott 4096 Oct 31 16:14 ../ 
drwxr-xr-x 7 scott scott 4096 Oct 3116:14 .git/ 