docker create container-name

docker start -a container-name

docker start container-name

docker run container-name

e.g. docker run busybox echo hi there

docker ps

docker ps --all

already created and exited container ran again

docker run -a container-id

same process would run which was ran earlier

docker system prune

This will remove:

- all stopped containers

- all networks not used by at least one container

- all dangling images

- all build cache

docker logs <container-id>

docker stop container-id

sigterm sent to running process

docker kill container-id

send kill signal to primary running process

docker exec -it <container id> <command>

-it allows directly type command into container

-i attach my terminal to stdin of process , -t displays formatted data and does something more

exec executes the command

docker exec -it <container-id> sh

want to run commands in context of container sh open sh shell to do it

docker run -it <container-id> <command>

if u do this you cant run any other process on this container

if u have only one primary process u can do this

Docker file : base image + commands to install dependency + command to run container startup

Creating own image : Docker file > docker client > docker server > usable image

$ mkdir redis-image

$ cd redis-image

$ touch Dockerfile

$ vi Dockerfile

$ cat Dockerfile

# Use an existing docker image as a base

FROM alpine

# Download and install dependency

RUN apk add --update redis

# Tell the image what to do when it starts as a container

CMD ["redis-server"]

$ docker build .

Sending build context to Docker daemon 2.048kB

Step 1/3 : FROM alpine

---> 3fd9065eaf02

Step 2/3 : RUN apk add --update redis

---> Running in be47b820312f

fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/main/x86\_64/APKINDEX.tar.gz

fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/community/x86\_64/APKINDEX.tar.gz

(1/1) Installing redis (4.0.14-r0)

Executing redis-4.0.14-r0.pre-install

Executing busybox-1.27.2-r7.trigger

OK: 7 MiB in 12 packages

Removing intermediate container be47b820312f

---> 5dac1d1e1de7

Step 3/3 : CMD ["redis-server"]

---> Running in 19fe9b92ed51

Removing intermediate container 19fe9b92ed51

---> 606e3404166b

Successfully built 606e3404166b

$ docker run 606e3404166b

1:C 29 Apr 03:06:13.972 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

1:C 29 Apr 03:06:13.972 # Redis version=4.0.14, bits=64, commit=5865ea3a, modified=0, pid=1, just started

1:C 29 Apr 03:06:13.972 # Warning: no config file specified, using the default config. In order tospecify a config file use redis-server /path/to/redis.conf

1:M 29 Apr 03:06:13.974 \* Running mode=standalone, port=6379.

1:M 29 Apr 03:06:13.974 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

1:M 29 Apr 03:06:13.974 # Server initialized

1:M 29 Apr 03:06:13.974 # WARNING overcommit\_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit\_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit\_memory=1' for this to take effect.

1:M 29 Apr 03:06:13.974 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

1:M 29 Apr 03:06:13.974 \* Ready to accept connections

base image provides initial infrastructure to further customize our image

we use alpine because it has package manager apk that we used to install redis

docker build <build context>

Temporary containers are created and their snapshot stored as image for intermediate steps

Run apk.. looked for the image at previous steps and created intermediate container to install redis and snapshot sent to temporary image

CMD ['redis-server"] tells container if it has to ever run redis-server will be its primary command, it doesn't run the command on intermediate container

$ vi Dockerfile

$ cat Dockerfile

# Use an existing docker image as a base

FROM alpine

# Download and install dependency

RUN apk add --update redis

RUN apk add --update gcc

# Tell the image what to do when it starts as a container

CMD ["redis-server"]

$ docker build .

Sending build context to Docker daemon 2.048kB

Step 1/4 : FROM alpine

---> 3fd9065eaf02

Step 2/4 : RUN apk add --update redis

---> Using cache

---> 5dac1d1e1de7

Step 3/4 : RUN apk add --update gcc

---> Running in 575254576ee3

fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/main/x86\_64/APKINDEX.tar.gz

fetch http://dl-cdn.alpinelinux.org/alpine/v3.7/community/x86\_64/APKINDEX.tar.gz

(1/12) Installing binutils-libs (2.30-r2)

(2/12) Installing binutils (2.30-r2)

(3/12) Installing gmp (6.1.2-r1)

(4/12) Installing isl (0.18-r0)

(5/12) Installing libgomp (6.4.0-r5)

(6/12) Installing libatomic (6.4.0-r5)

(7/12) Installing pkgconf (1.3.10-r0)

(8/12) Installing libgcc (6.4.0-r5)

(9/12) Installing mpfr3 (3.1.5-r1)

(10/12) Installing mpc1 (1.0.3-r1)

(11/12) Installing libstdc++ (6.4.0-r5)

(12/12) Installing gcc (6.4.0-r5)

Executing busybox-1.27.2-r7.trigger

OK: 91 MiB in 24 packages

Removing intermediate container 575254576ee3

---> 4231c7b1aa64

Step 4/4 : CMD ["redis-server"]

---> Running in 9a36eaaf2447

Removing intermediate container 9a36eaaf2447

---> b066b9a60c52

Successfully built b066b9a60c52

docker build uses cache till the step to which no changes are made (even if order is changed it won't use cache)

$ docker build .

Sending build context to Docker daemon 2.048kB

Step 1/4 : FROM alpine

---> 3fd9065eaf02

Step 2/4 : RUN apk add --update redis

---> Using cache

---> 5dac1d1e1de7

Step 3/4 : RUN apk add --update gcc

---> Using cache

---> 4231c7b1aa64

Step 4/4 : CMD ["redis-server"]

---> Using cache

---> b066b9a60c52

Successfully built b066b9a60c52

to tag image

docker build -t <dockerID>/<repo/project name>:<version> <build context>

if latest version you can mention version as latest

$ docker build -t katacode/redis:latest .

Sending build context to Docker daemon 2.048kB

Step 1/4 : FROM alpine

---> 3fd9065eaf02

Step 2/4 : RUN apk add --update redis

---> Using cache

---> 5dac1d1e1de7

Step 3/4 : RUN apk add --update gcc

---> Using cache

---> 4231c7b1aa64

Step 4/4 : CMD ["redis-server"]

---> Using cache

---> b066b9a60c52

Successfully built b066b9a60c52

Successfully tagged katacode/redis:latest

$ docker run katacode/redis

1:C 29 Apr 03:49:16.870 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

1:C 29 Apr 03:49:16.870 # Redis version=4.0.14, bits=64, commit=5865ea3a, modified=0, pid=1, just started

1:C 29 Apr 03:49:16.870 # Warning: no config file specified, using the default config. In order tospecify a config file use redis-server /path/to/redis.conf

1:M 29 Apr 03:49:16.872 \* Running mode=standalone, port=6379.

1:M 29 Apr 03:49:16.872 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

1:M 29 Apr 03:49:16.872 # Server initialized

1:M 29 Apr 03:49:16.872 # WARNING overcommit\_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit\_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit\_memory=1' for this to take effect.

1:M 29 Apr 03:49:16.872 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

1:M 29 Apr 03:49:16.872 \* Ready to accept connections

you can run each step of dockerfile manually to create image manually but thats not recommended

E.g. Create a node js web app and run it on docker container

Steps:

1) Create node.js web app

2) Create dockerfile

3) Build image from docker file

4) Run image as container

5) connect web app from browser

$ mkdir simpleweb

$ cd simpleweb

$ vi package.json

$ vi index.js

$ ls

index.js package.json

$ cat index.js

const express = require('express')

const app = express();

app.get('/',(req,res) => {

res.send('Hi there');

});

app.listen(8080,() => {

console.log('Listening on port 8080');

});

$ cat package.json

{

"dependencies":{

"express":"\*"

},

"scripts":{

"start":"node index.js"

}

}

$ vi dockerfile

$ vi Dockerfile

$ ls

Dockerfile index.js package.json

$ cat Dockerfile

# Specify base image

FROM alpine

# Install dependencies

RUN npm install

# Default command

CMD ["npm","start"]

$ docker build .

Sending build context to Docker daemon 4.096kB

Step 1/3 : FROM alpine

---> 11cd0b38bc3c

Step 2/3 : RUN npm install

---> Running in 56502af42294

/bin/sh: npm: not found

The command '/bin/sh -c npm install' returned a non-zero code: 127

$ vi Dockerfile

$ cat Dockerfile

# Specify base image

FROM node:alpine

# Install dependencies

RUN npm install

# Default command

CMD ["npm","start"]

$ docker build .

Sending build context to Docker daemon 4.096kB

Step 1/3 : FROM node:alpine

alpine: Pulling from library/node

cbdbe7a5bc2a: Pull complete

0deddb4d0d4b: Pull complete

b49b5c4238a4: Pull complete

2021d4207c54: Pull complete

Digest: sha256:a8553f21bea4e402dac9618fadb47b179bd956c07f97064263f9522a6e53693a

Status: Downloaded newer image for node:alpine

---> 0854fcfc1637

Step 2/3 : RUN npm install

---> Running in 1c7a613a6d2b

npm WARN saveError ENOENT: no such file or directory, open '/package.json'

npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN enoent ENOENT: no such file or directory, open '/package.json'

npm WARN !invalid#2 No description

npm WARN !invalid#2 No repository field.

npm WARN !invalid#2 No README data

npm WARN !invalid#2 No license field.

up to date in 0.458s

found 0 vulnerabilities

Removing intermediate container 1c7a613a6d2b

---> be3e7346036b

Step 3/3 : CMD ["npm","start"]

---> Running in f0843e3ad2ba

Removing intermediate container f0843e3ad2ba

---> 206649d81922

Successfully built 206649d81922

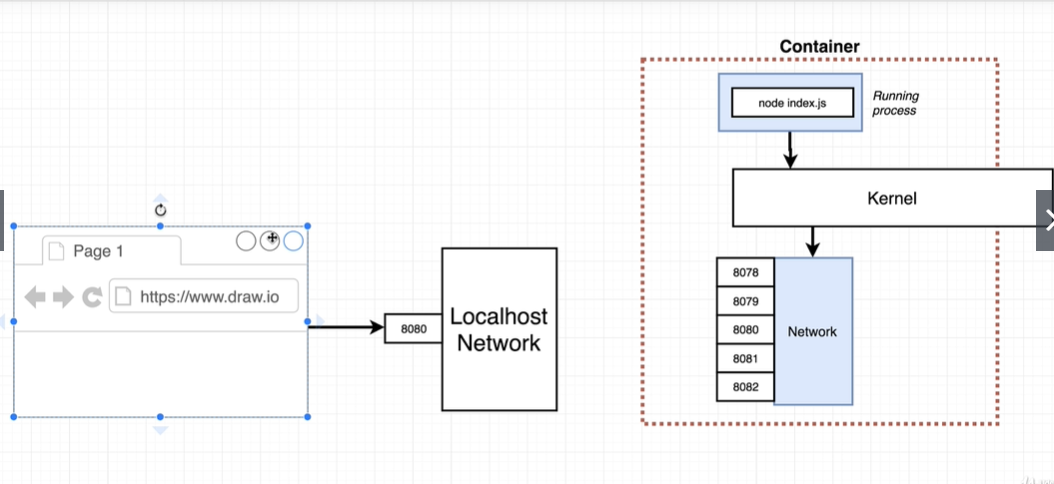
If can’t find base image search on hub.docker.com

Alpine version is the version which doesn’t contain any extra packages. The small lightweight version is called alpine version.

By default, image pulled won’t have files we have created on our system so we need to use COPY instruction to make those files available in container.

Source path in copy command is relative to build context

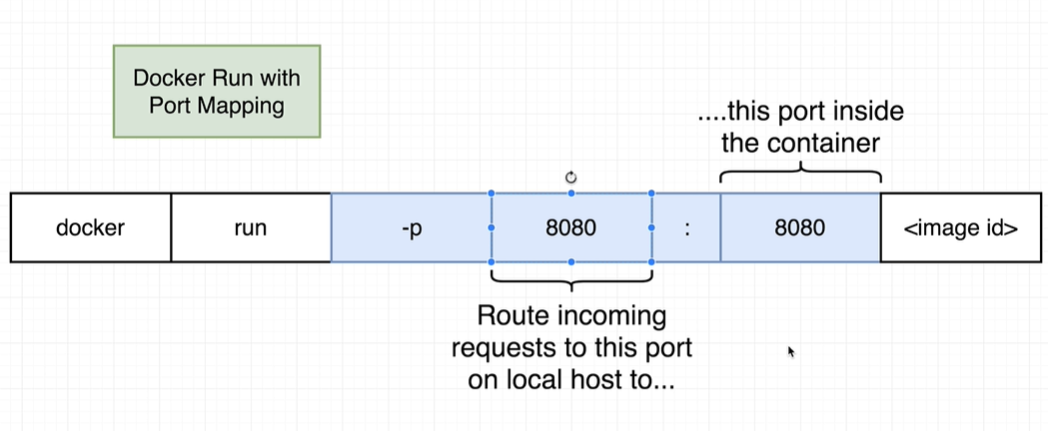
Simpleweb directory in current example.



Port mapping is when someone make request to my machine at particular port map it to some port in container, so that application can accept request and respond

No limitation on reaching out to web for container

It’s for incoming request to get in to container



$ vi Dockerfile

$ cat Dockerfile

# Specify base image

FROM node:alpine

# Install dependencies

COPY ./ ./

RUN npm install

# Default command

CMD ["npm","start"]

$ docker build .

Sending build context to Docker daemon 4.096kB

Step 1/4 : FROM node:alpine

---> 0854fcfc1637

Step 2/4 : COPY ./ ./

---> 2b847da08e0d

Step 3/4 : RUN npm install

---> Running in 3577a0429a04

npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN !invalid#2 No description

npm WARN !invalid#2 No repository field.

npm WARN !invalid#2 No license field.

added 50 packages from 37 contributors and audited 126 packages in 2.695s

found 0 vulnerabilities

Removing intermediate container 3577a0429a04

---> 114af01ee372

Step 4/4 : CMD ["npm","start"]

---> Running in 283a6692a54d

Removing intermediate container 283a6692a54d

---> 213f1908b550

Successfully built 213f1908b550

$ docker build -t katacode/simpleweb .

Sending build context to Docker daemon 4.096kB

Step 1/4 : FROM node:alpine

---> 0854fcfc1637

Step 2/4 : COPY ./ ./

---> Using cache

---> 2b847da08e0d

Step 3/4 : RUN npm install

---> Using cache

---> 114af01ee372

Step 4/4 : CMD ["npm","start"]

---> Using cache

---> 213f1908b550

Successfully built 213f1908b550

Successfully tagged katacode/simpleweb:latest

$ docker run katacode/simpleweb

> @ start /

> node index.js

Listening on port 8080

exit

^C$ docker run -p 8080:8080 katacode/simpleweb

> @ start /

> node index.js

Listening on port 8080

^C$ docker run -p 5000:8080 katacode/simpleweb

> @ start /

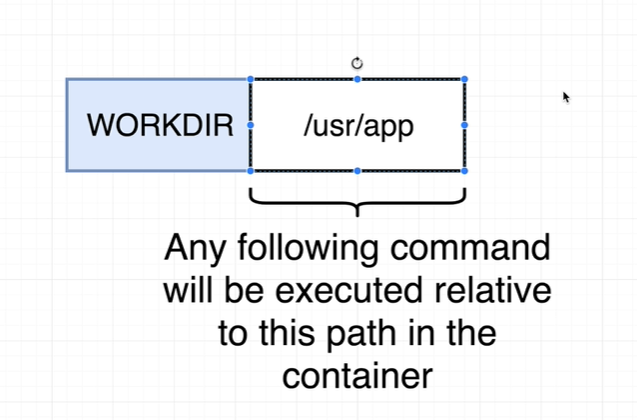
> node index.js

Listening on port 8080

Access the port 5000 from your machine and you will be able to see the output of port 8080 from container

The listening port in container doesn’t have to be 8080, set it whatever you want in files and then map them correctly with -p flag

To organize files in container during setup



$ docker run -it katacode/simpleweb sh

/ # ls

Dockerfile mnt sbin

bin node\_modules srv

dev opt sys

etc package-lock.json tmp

home package.json usr

index.js proc var

lib root

media run

/ # exit

$

$ vi Dockerfile

$ docker build -t katacode/simpleweb .

Sending build context to Docker daemon 4.096kB

Step 1/5 : FROM node:alpine

---> 0854fcfc1637

Step 2/5 : WORKDIR /usr/app

Removing intermediate container bfe861218754

---> f441cf09d3dc

Step 3/5 : COPY ./ ./

---> 74065d1a8cf7

Step 4/5 : RUN npm install

---> Running in a9b70aae245c

npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN app No description

npm WARN app No repository field.

npm WARN app No license field.

added 50 packages from 37 contributors and audited 126 packages in 3.101s

found 0 vulnerabilities

Removing intermediate container a9b70aae245c

---> 54feaecf5207

Step 5/5 : CMD ["npm","start"]

---> Running in 618a033bbe8d

Removing intermediate container 618a033bbe8d

---> 8317baab3fe1

Successfully built 8317baab3fe1

Successfully tagged katacode/simpleweb:latest

$ docker run -p 8080:8080 katacode/simpleweb

> @ start /usr/app

> node index.js

Listening on port 8080

^C$ cat Dockerfile

# Specify base image

FROM node:alpine

WORKDIR /usr/app

# Install dependencies

COPY ./ ./

RUN npm install

# Default command

CMD ["npm","start"]

$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

7c830b5832d1 katacode/simpleweb "docker-entrypoint.s…" 41 seconds ago Up 39 seconds 0.0.0.0:8080->8080/tcp flamboyant\_clarke

$ docker exec -it 7c830b5832d1 sh

/usr/app # ls

Dockerfile node\_modules package.json

index.js package-lock.json

/usr/app # cd /

/ # ls

bin etc lib mnt proc run srv tmp var

dev home media opt root sbin sys usr

/ # $

When changes made on system, it is not reflected on container

Since snapshot have already been taken during build.

Run docker build again to reflect the changes

$ cat index.js

const express = require('express')

const app = express();

app.get('/',(req,res) => {

res.send('Bye there');

});

app.listen(8080,() => {

console.log('Listening on port 8080');

});

$ docker build -t katacode/simpleweb .

Sending build context to Docker daemon 4.096kB

Step 1/5 : FROM node:alpine

alpine: Pulling from library/node

cbdbe7a5bc2a: Pull complete

0deddb4d0d4b: Pull complete

b49b5c4238a4: Pull complete

2021d4207c54: Pull complete

Digest: sha256:a8553f21bea4e402dac9618fadb47b179bd956c07f97064263f9522a6e53693a

Status: Downloaded newer image for node:alpine

---> 0854fcfc1637

Step 2/5 : WORKDIR /usr/app

Removing intermediate container 2bd72d80eecd

---> 0919c7581f8e

Step 3/5 : COPY ./ ./

---> d4645ccd0a36

Step 4/5 : RUN npm install

---> Running in 6f00cab7dd0c

**npm notice created a lockfile as package-lock.json. You should commit this file.**

**npm WARN app No description**

**npm WARN app No repository field.**

**npm WARN app No license field.**

**added 50 packages from 37 contributors and audited 126 packages in 3.904s**

**found 0 vulnerabilities**

Removing intermediate container 6f00cab7dd0c

---> bb436eb0aee8

Step 5/5 : CMD ["npm","start"]

---> Running in 331735ece32d

Removing intermediate container 331735ece32d

---> 5356a18c4ef1

Successfully built 5356a18c4ef1

Successfully tagged katacode/simpleweb:latest

We changed some source code and dependencies are installing again

$ cat Dockerfile

# Specify base image

FROM node:alpine

WORKDIR /usr/app

# Install dependencies

**COPY ./package.json ./**

**RUN npm install**

**COPY ./ ./**

# Default command

CMD ["npm","start"]

Now npm would run install dependencies if any changes made to steps above it.

$ docker build -t katacode/simpleweb .

Sending build context to Docker daemon 4.096kB

Step 1/6 : FROM node:alpine

---> 0854fcfc1637

Step 2/6 : WORKDIR /usr/app

---> Using cache

---> 0919c7581f8e

Step 3/6 : COPY ./package.json ./

---> Using cache

---> 6f4554844049

Step 4/6 : RUN npm install

---> Using cache

---> 3bb2cd282460

Step 5/6 : COPY ./ ./

---> 43ff6477498c

Step 6/6 : CMD ["npm","start"]

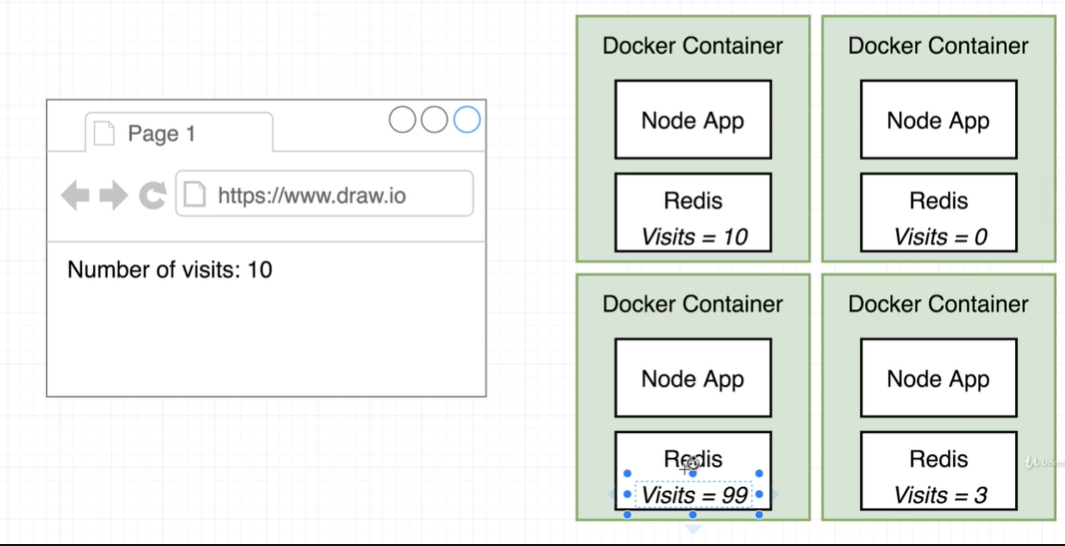
---> Running in 4beb333f08ca

Removing intermediate container 4beb333f08ca

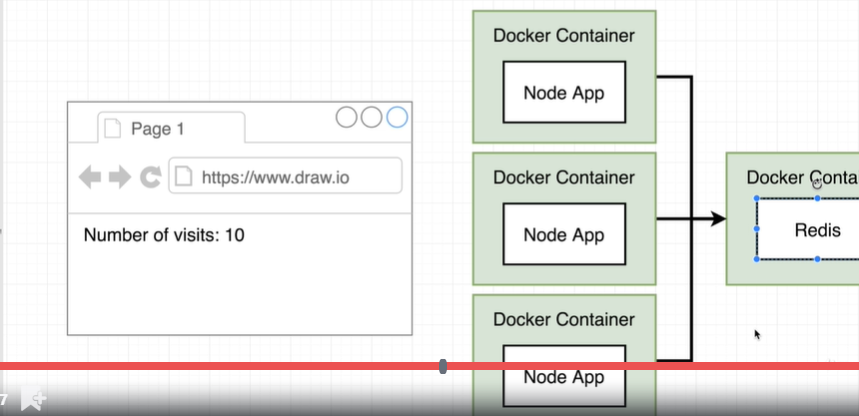
---> f6e2e9c24db7

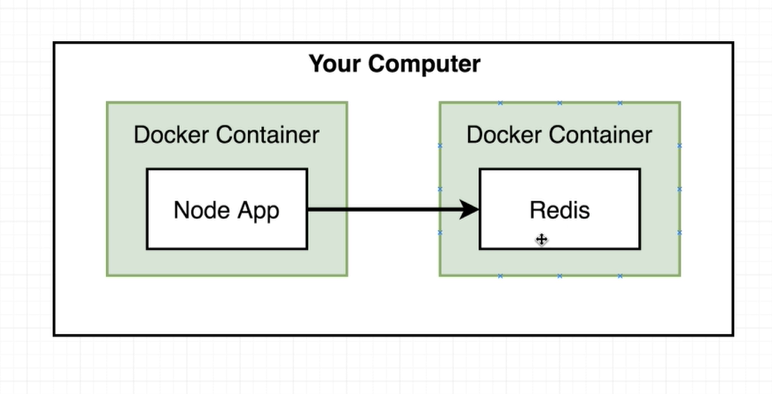
Successfully built f6e2e9c24db7

Successfully tagged katacode/simpleweb:latest



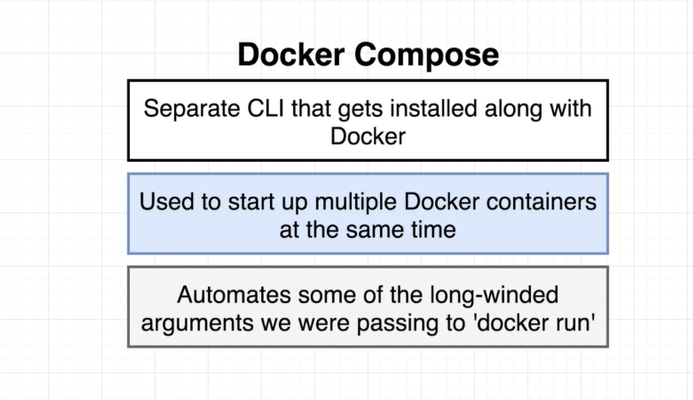
If traffic increases we might deploy multiple instances of container but then redis server wont be in sync.

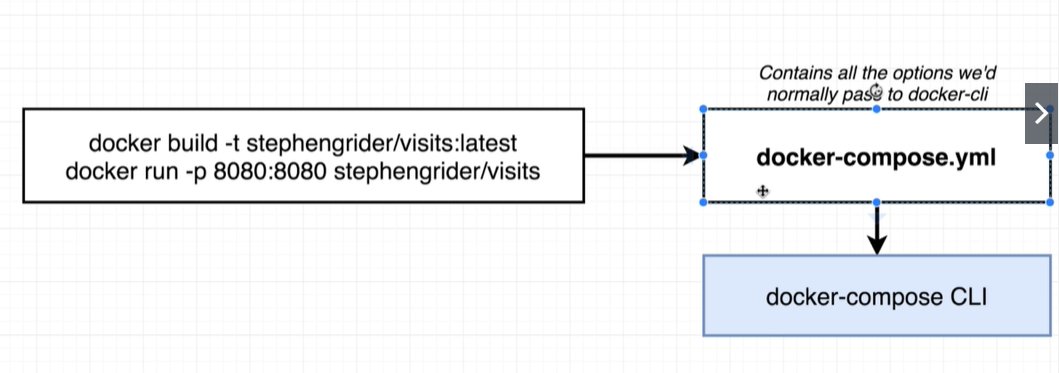
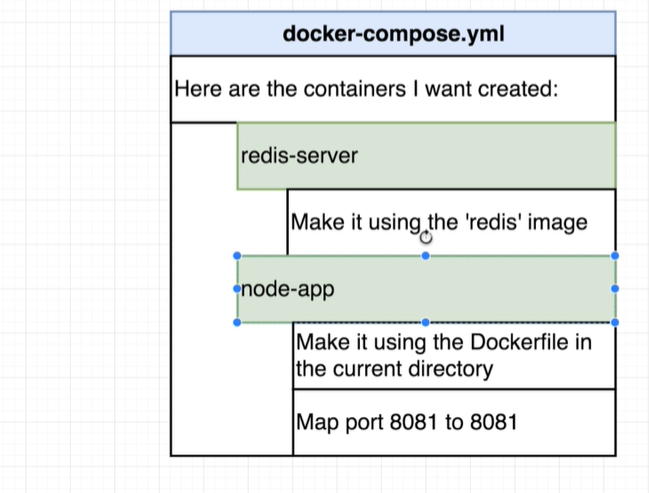




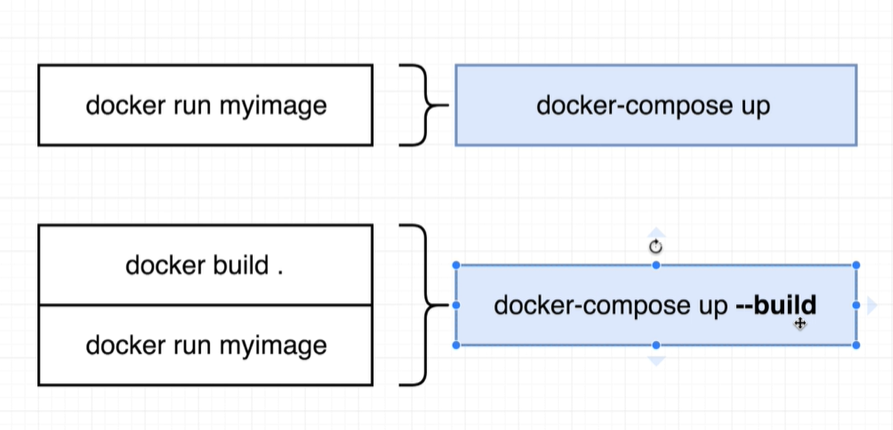
Options for networking 2 docker containers

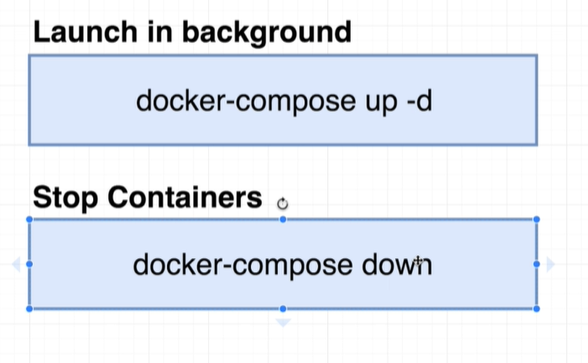
* Use docker cli options
* Use docker compose



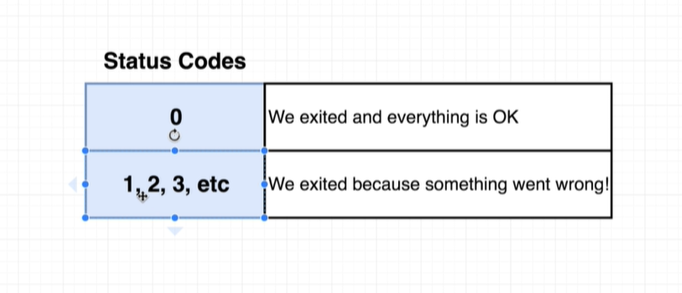
 

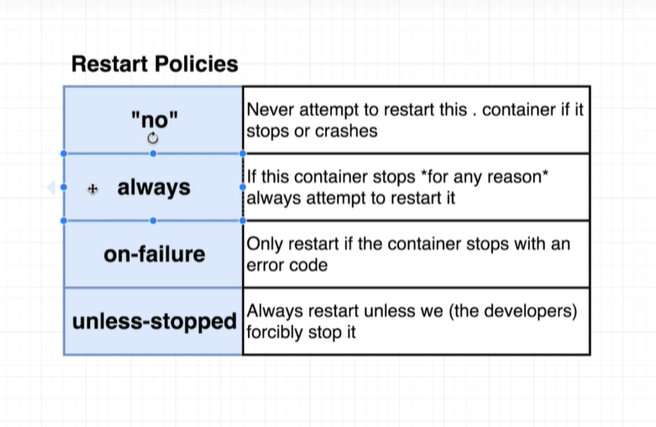
When services mentioned in docker.yml file these containers doesn’t need any extra networking configured they can communicate freely with each other.





Process exit status codes





$ mkdir visits

$ cd visits

$ vi docker-compose.yml

$ vi Dockerfle

$ vi Dockerfile

$ vi index.js

$ vi package.json

$ docker-compose docker-compose.yml

No such command: docker\_compose.yml

Commands:

build Build or rebuild services

bundle Generate a Docker bundle from the Compose file

config Validate and view the Compose file

create Create services

down Stop and remove containers, networks, images, and volumes

events Receive real time events from containers

exec Execute a command in a running container

help Get help on a command

images List images

kill Kill containers

logs View output from containers pause Pause services port Print the public port for a port binding ps List containers

pull Pull service images

push Push service images

restart Restart services

rm Remove stopped containers

run Run a one-off command

scale Set number of containers for a service

start Start services

stop Stop services

top Display the running processes

unpause Unpause services

up Create and start containers

version Show the Docker-Compose version information

$ docker-compose up

Creating network "visits\_default" with the default driver

Building node-app

Step 1/6 : FROM node:alpine

alpine: Pulling from library/node

cbdbe7a5bc2a: Pull complete

0deddb4d0d4b: Pull complete

b49b5c4238a4: Pull complete

2021d4207c54: Pull complete

Digest: sha256:a8553f21bea4e402dac9618fadb47b179bd956c07f97064263f9522a6e53693a

Status: Downloaded newer image for node:alpine

---> 0854fcfc1637

Step 2/6 : WORKDIR '/app'

Removing intermediate container eeec322d373c

---> 657fc6d00ee1

Step 3/6 : COPY package.json .

---> 47d8dbc6481e

Step 4/6 : RUN npm install

---> Running in 369cee78ac70

npm notice created a lockfile as package-lock.json. You should commit this file.

npm WARN app No description

npm WARN app No repository field.

npm WARN app No license field.

added 54 packages from 41 contributors and audited 130 packages in 3.266s

found 0 vulnerabilities

Removing intermediate container 369cee78ac70

---> d201ab656602

Step 5/6 : COPY . .

---> d5e3d96cfac2

Step 6/6 : CMD ["npm","start"]

---> Running in a820e9a5ea62

Removing intermediate container a820e9a5ea62

---> 106822ae7529

Successfully built 106822ae7529

Successfully tagged visits\_node-app:latest

WARNING: Image for service node-app was built because it did not already exist. To rebuild this image you must use `docker-compose build`or `docker-compose up --build`.

Creating visits\_node-app\_1 ... done

Creating visits\_redis-server\_1 ... done

Attaching to visits\_redis-server\_1, visits\_node-app\_1

redis-server\_1 | 1:C 01 May 13:50:29.716 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

redis-server\_1 | 1:C 01 May 13:50:29.720 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=1, just started

redis-server\_1 | 1:C 01 May 13:50:29.720 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

redis-server\_1 | 1:M 01 May 13:50:29.725 \* Running mode=standalone,port=6379.

redis-server\_1 | 1:M 01 May 13:50:29.725 # WARNING: The TCP backlogsetting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

redis-server\_1 | 1:M 01 May 13:50:29.725 # Server initialized

redis-server\_1 | 1:M 01 May 13:50:29.725 # WARNING overcommit\_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit\_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit\_memory=1' forthis to take effect.

redis-server\_1 | 1:M 01 May 13:50:29.725 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled'as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

redis-server\_1 | 1:M 01 May 13:50:29.725 \* Ready to accept connections

node-app\_1 |

node-app\_1 | > @ start /app

node-app\_1 | > node index.js

node-app\_1 |

node-app\_1 | listening on port 8081

^CGracefully stopping... (press Ctrl+C again to force)

Stopping visits\_redis-server\_1 ... done

Stopping visits\_node-app\_1 ... done

$ docker-compose up -d

Starting visits\_node-app\_1 ... done

Starting visits\_redis-server\_1 ... done

$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

ce1246e4ca13 redis "docker-entrypoint.s…" 5 minutes ago Up 7 seconds 6379/tcp visits\_redis-server\_1

18f505bf65ab visits\_node-app "docker-entrypoint.s…" 5 minutes ago Up 7 seconds 0.0.0.0:4001->8081/tcp visits\_node-app\_1

$ docker-compose down

Stopping visits\_redis-server\_1 ... done

Stopping visits\_node-app\_1 ... done

Removing visits\_redis-server\_1 ... done

Removing visits\_node-app\_1 ... done

Removing network visits\_default

$ docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

$ cat \*

FROM node:alpine

WORKDIR '/app'

COPY package.json .

RUN npm install

COPY . .

CMD ["npm","start"]

version: "3"

services:

redis-server:

image: "redis"

node-app:

build: .

ports:

- "4001:8081"

const express = require("express");

const redis = require("redis");

const app = express();

const client = redis.createClient({

host: "redis-server",

port: 6379

});

client.set("visits", 0);

app.get("/", (req, res) => {

client.get("visits", (err, visits) => {

res.send("Number of visits " + visits);

client.set("visits", parseInt(visits) + 1);

});

});

app.listen(8081, () => {

console.log("listening on port 8081");

});

{

"dependencies": {

"express": "\*",

"redis": "2.8.0"

},

"scripts": {

"start": "node index.js"

}

}

$ vi index.js

$ docker-compose up --build

Creating network "visits\_default" with the default driver

Building node-app

Step 1/6 : FROM node:alpine

---> 0854fcfc1637

Step 2/6 : WORKDIR '/app'

---> Using cache

---> 657fc6d00ee1

Step 3/6 : COPY package.json .

---> Using cache

---> 47d8dbc6481e

Step 4/6 : RUN npm install

---> Using cache

---> d201ab656602

Step 5/6 : COPY . .

---> 31e9dbbb9b00

Step 6/6 : CMD ["npm","start"]

---> Running in 9c1c54621345

Removing intermediate container 9c1c54621345

---> 8e1379220f8e

Successfully built 8e1379220f8e

Successfully tagged visits\_node-app:latest

Creating visits\_redis-server\_1 ... done

Creating visits\_node-app\_1 ... done

Attaching to visits\_redis-server\_1, visits\_node-app\_1

redis-server\_1 | 1:C 01 May 14:01:00.346 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

redis-server\_1 | 1:C 01 May 14:01:00.348 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=1, just started

redis-server\_1 | 1:C 01 May 14:01:00.348 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

redis-server\_1 | 1:M 01 May 14:01:00.350 \* Running mode=standalone,port=6379.

redis-server\_1 | 1:M 01 May 14:01:00.350 # WARNING: The TCP backlogsetting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

redis-server\_1 | 1:M 01 May 14:01:00.350 # Server initialized

redis-server\_1 | 1:M 01 May 14:01:00.350 # WARNING overcommit\_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit\_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit\_memory=1' forthis to take effect.

redis-server\_1 | 1:M 01 May 14:01:00.350 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled'as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

redis-server\_1 | 1:M 01 May 14:01:00.350 \* Ready to accept connections

node-app\_1 |

node-app\_1 | > @ start /app

node-app\_1 | > node index.js

node-app\_1 |

node-app\_1 | listening on port 8081

visits\_node-app\_1 exited with code 0

^CGracefully stopping... (press Ctrl+C again to force)

Stopping visits\_redis-server\_1 ... done

$ docker-compose up

Starting visits\_redis-server\_1 ... done

Recreating visits\_node-app\_1 ... done

Attaching to visits\_redis-server\_1, visits\_node-app\_1

redis-server\_1 | 1:C 01 May 14:06:00.796 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

redis-server\_1 | 1:C 01 May 14:06:00.800 # Redis version=4.0.11, bits=64, commit=00000000, modified=0, pid=1, just started

redis-server\_1 | 1:C 01 May 14:06:00.800 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

redis-server\_1 | 1:M 01 May 14:06:00.801 \* Running mode=standalone,port=6379.

redis-server\_1 | 1:M 01 May 14:06:00.801 # WARNING: The TCP backlogsetting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

redis-server\_1 | 1:M 01 May 14:06:00.802 # Server initialized

redis-server\_1 | 1:M 01 May 14:06:00.802 # WARNING overcommit\_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit\_memory = 1' to /etc/sysctl.conf and then reboot or run the command 'sysctl vm.overcommit\_memory=1' forthis to take effect.

redis-server\_1 | 1:M 01 May 14:06:00.802 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled'as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

redis-server\_1 | 1:M 01 May 14:06:00.802 \* DB loaded from disk: 0.000 seconds

redis-server\_1 | 1:M 01 May 14:06:00.802 \* Ready to accept connections

node-app\_1 |

node-app\_1 | > @ start /app

node-app\_1 | > node index.js

node-app\_1 |

node-app\_1 | listening on port 8081

visits\_node-app\_1 exited with code 0

$ docker-compose ps

Name Command State Ports

---------------------------------------------------------------------

visits\_node-app\_1 docker-entrypoint.sh npm Exit 0

start

visits\_redis-server\_1 docker-entrypoint.sh Up 6379/tcp

redis ...

$ cd ..

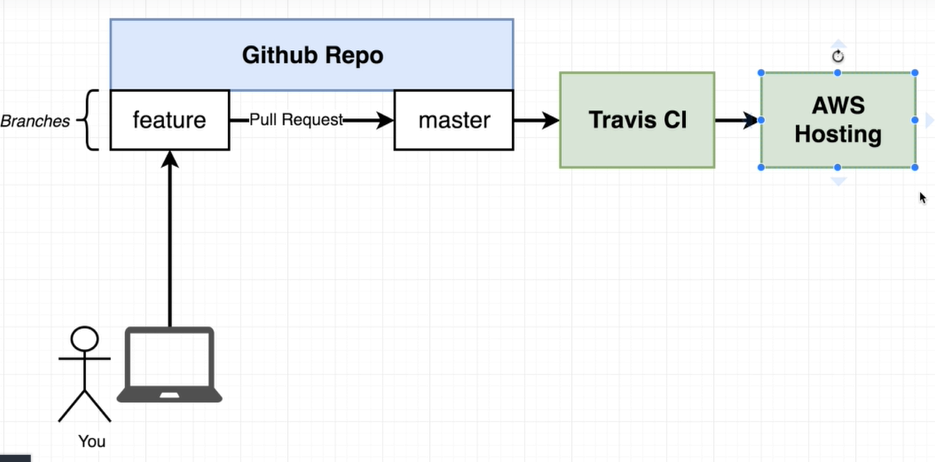
$ docker-compose ps

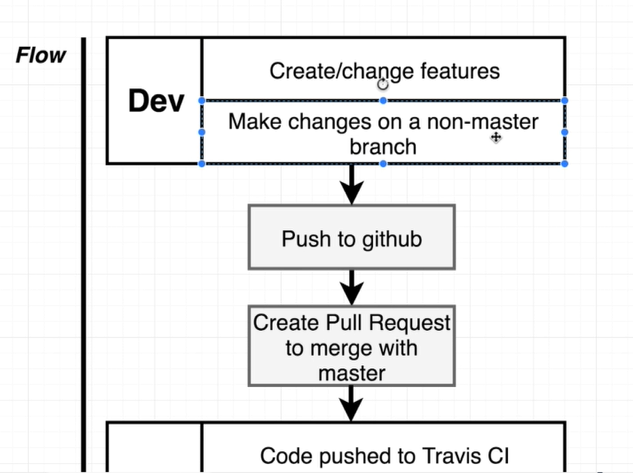
ERROR:

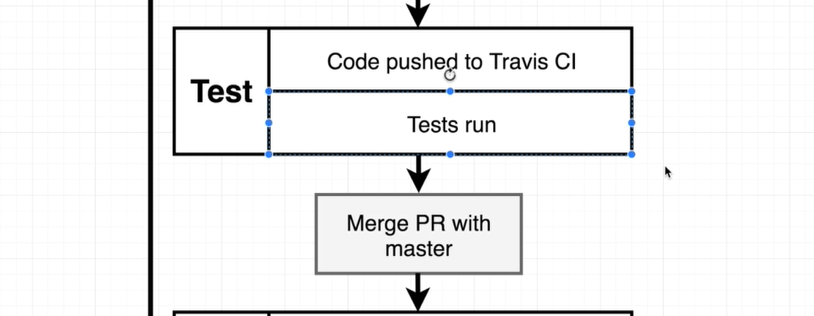
Can't find a suitable configuration file in this directory or any

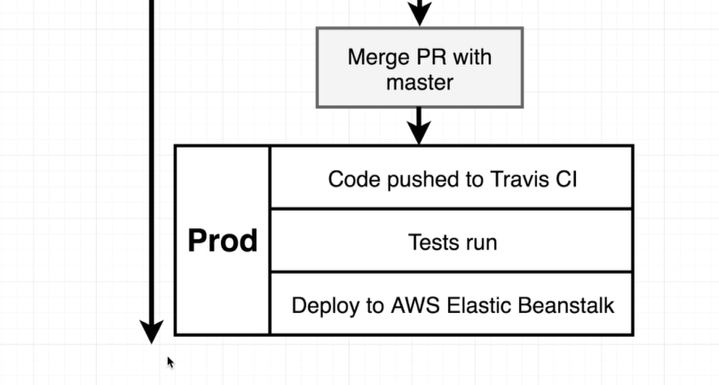
parent. Are you in the right directory?

Supported filenames: docker-compose.yml, docker-compose.yaml

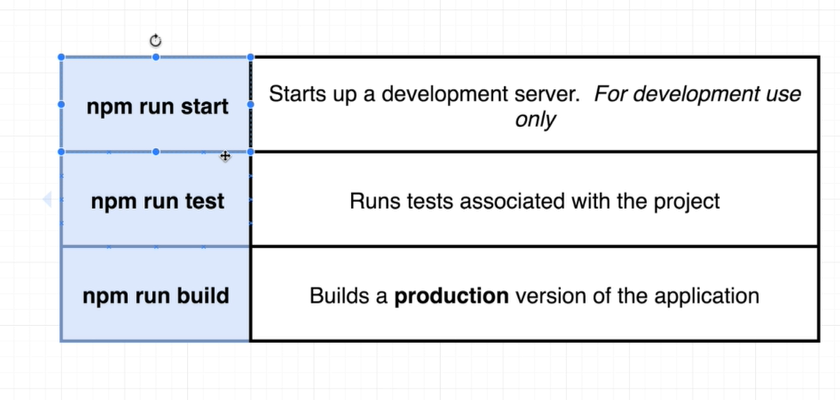


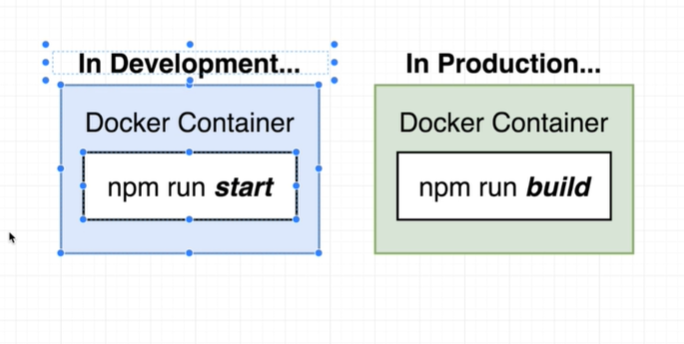


****



Install nodejs on your machine <https://medium.com/the-node-js-collection/time-to-hello-world-part2-gce-95e9df907b93>





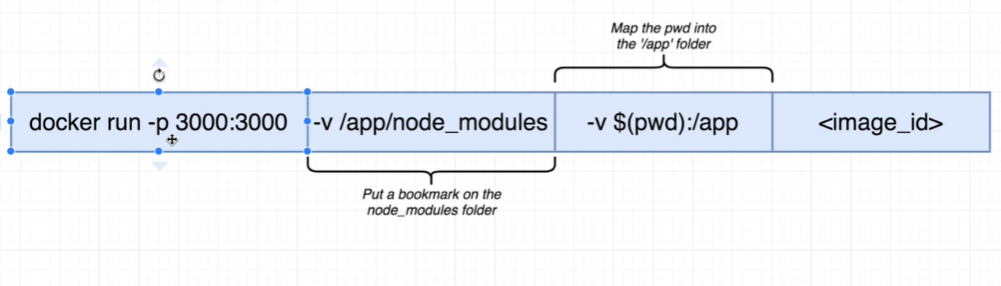
Npx create-app explicitly installed node-modules for web app

When we ran build, Sending build context has large amount

To avoid that deleted node-modules which react app installed since dockerfile.dev already has step to install node-modules

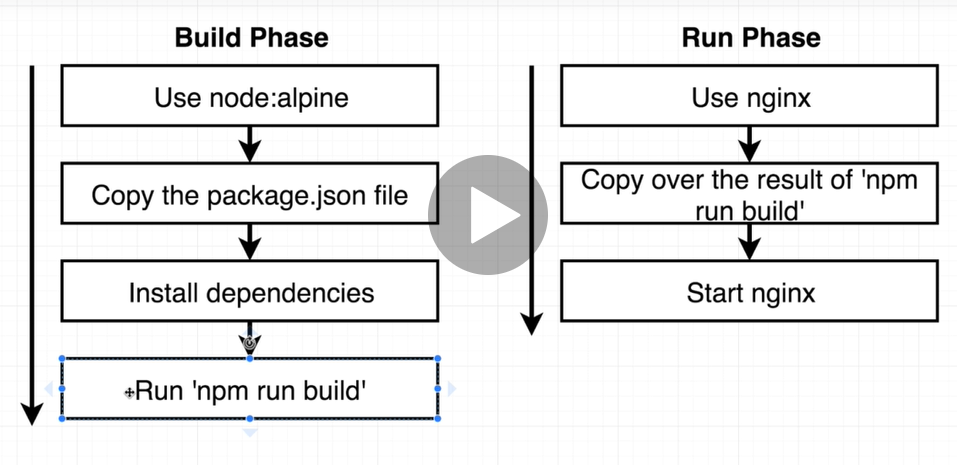
To not build each time you make change to source use volumes

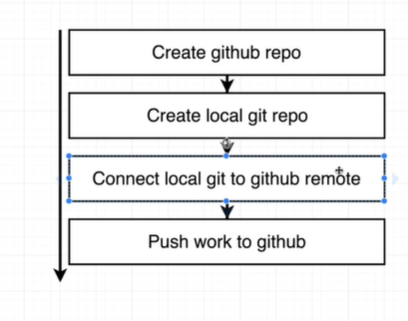
Volumes kind of create reference to directories from containers



docker run -it -p 3000:3000 -v /app/node\_modules -v ${PWD}:/app -e CHOKIDAR\_USEPOLLING=true edef25b5d308

Production container doesn’t need dev server since we are not making any changes to file at production container and dev server is really powerful we don’t need it. We need some production server like nginx which mainly deals with traffic pointing to some static files.





1 cd /c/Users/Shriya/Desktop/STUDY/kubernetes\_docker/

2 ls

3 cd frontend/

4 ls

5 find . -name node-modules

6 find . -name \*node\*

7 history

8 ls

9 npm start

10 ls

11 npm run test

12 cd ..

13 ls

14 rm frontend

15 rmdir frontend/

16 rm -rf frontend

17 ls

18 ls

19 npx create-react-app frontend

20 npm run test

21 cd frontend/

22 npm run test

23 npm run build

24 ls

25 ls build/

26 ls build/static/

27 ls build/static/js

28 npm run start

29 docker build .

30 ls

31 docker build -f Dockerfile.dev .

32 ls

33 mv node\_modules ..

34 ls ..

35 docker build -f Dockerfile.dev .

36 docker run -it -p 3000:3000 636215fd9b3b

37 docker build -f Dockerfile.dev .

38 docker run -it -p 3000:3000 636215fd9b3b

39 docker build -f Dockerfile.dev .

40 docker run -it -p 3000:3000 d05e484e27cf

41 docker build -f Dockerfile.dev .

42 docker run -it -p 3000:3000 d05e484e27cf

43 docker run -it -p 3000:3000 7333a67e04db

44 docker ps

45 docker ps -all

46 docker build -f Dockerfile.dev .

47 docker run -p 3000:3000 -v $(pwd):/app edef25b5d308

48 docker ps

49 docker run -it -p 3000:3000 -v /app/node\_modules -v ${PWD}:/app -e CHOKIDAR\_USEPOLLING=true edef25b5d308

50 docker-compose up

51 ls

52 docker-compose up

53 docker-compose up

54 docker-compose up

55 docker-compose up

56 docker-compose up

57 docker-compose up

58 docker-compose up

59 docker-compose up

60 docker-compose up

61 docker-compose up

62 docker-compose up

63 docker build -f Dockerfile.dev

64 docker build -f Dockerfile.dev

65 docker build -f Dockerfile.dev .

66 docker run dfacfe462456 npm run test

67 docker run 7dfe69c741cc npm run test

68 docker run -it 7dfe69c741cc npm run test

69 docker run -it 7dfe69c741cc npm run test

70 docker-compose up

71 docker-compose up --build

72 w

73 w

74 docker-compose -it up --build

75 docker-compose up

76 docker-compose up --build

77 ls

78 move node\_modules/ ..

79 move node\_modules ..

80 mv node\_modules ..

81 ls

82 node -v

83 npm build -f Dockerfile.dev

84 ls

85 npm build -f Dockerfile.dev

86 npm build -f Dockerfile.dev

87 npm build -f Dockerfile.dev

88 docker build -f Dockerfile.dev .

89 history

90 docker run -it -p 3000:3000 e9952b2a0670

91 docker run -it -p 3000:3000 e9952b2a0670

92 docker build .

93 docker build .

94 docker run -it -p 8080:80 5593e0c8edc7

95 history

97 docker run -it -p 3000:3000 1d3350a6c4cb

98 git init

99 git add .

100 git commit -m "initial commit"

101 git config --global user.email "shriyajain2503@gmail.com"

102 git commit -m "initial commit"

103 git remote add origin https://github.com/shriyaj/docker-react.git

104 git push orgin master

105 git push origin master

106 git push origin master

