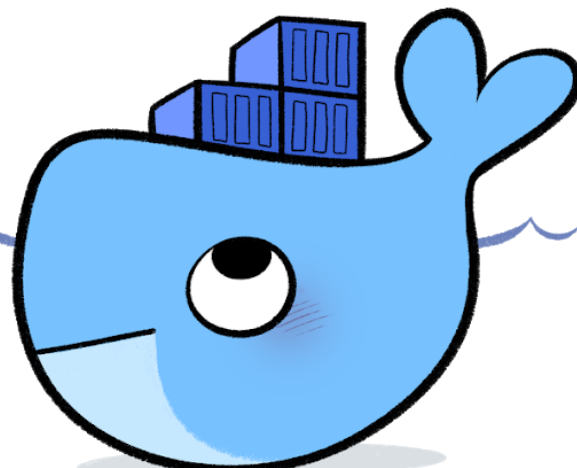
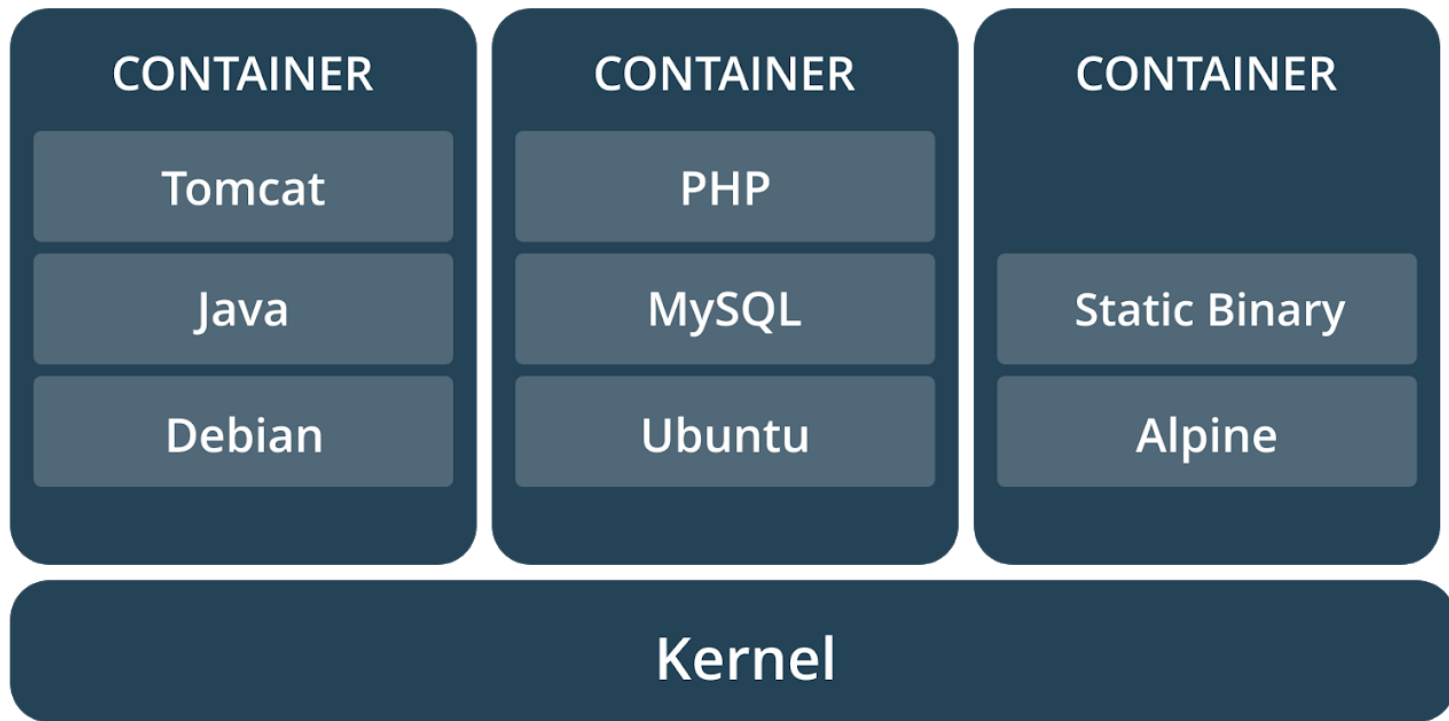


docker

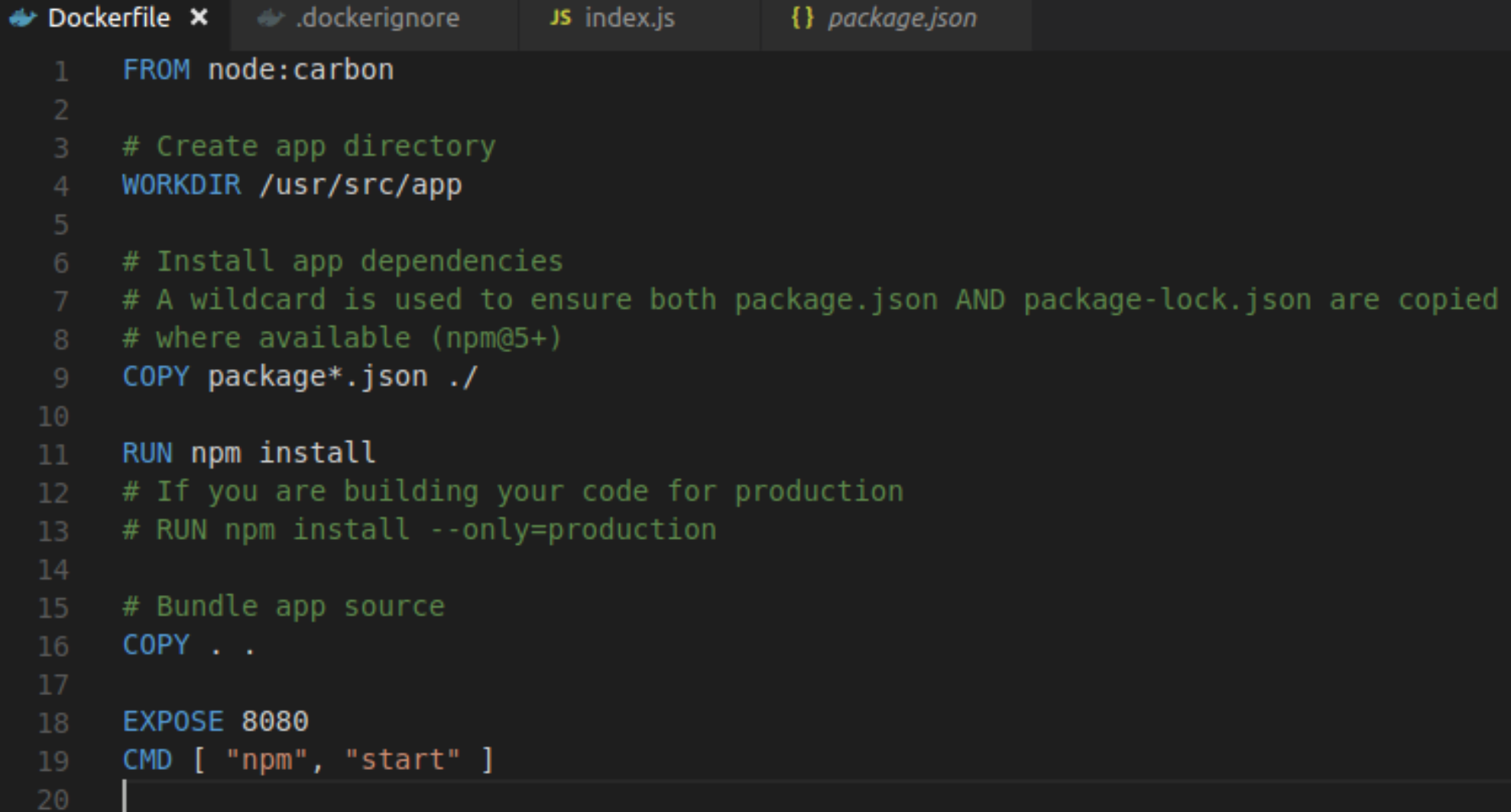
**What is docker ?**

## What is docker ?



**How images are build**

## How images are build



The image shows a code editor with a dark theme. At the top, there are four tabs: 'Dockerfile' (active), '.dockerignore', 'index.js', and 'package.json'. The 'Dockerfile' tab contains the following code:

```
1 FROM node:carbon
2
3 # Create app directory
4 WORKDIR /usr/src/app
5
6 # Install app dependencies
7 # A wildcard is used to ensure both package.json AND package-lock.json are copied
8 # where available (npm@5+)
9 COPY package*.json ./
10
11 RUN npm install
12 # If you are building your code for production
13 # RUN npm install --only=production
14
15 # Bundle app source
16 COPY . .
17
18 EXPOSE 8080
19 CMD [ "npm", "start" ]
20
```

## How images are build

```
pdarii@P5MD-PORT142:~/public_html/docker-examples/node-js-server$ docker build .
Sending build context to Docker daemon 18.43kB
Step 1/7 : FROM node:carbon
carbon: Pulling from library/node
3d77ce4481b1: Pull complete
534514c83d69: Pull complete
d562b1c3ac3f: Pull complete
4b85e68dc01d: Pull complete
f6a66c5de9db: Pull complete
7a4e7d9a081d: Pull complete
80f398d0747c: Pull complete
0d76f8ac4029: Pull complete
Digest: sha256:a9d2c607e10e842349dd37da8fe7c2e8ec573d38d12086660fa891eed2b60780
Status: Downloaded newer image for node:carbon
---> 78f8aef50581
Step 2/7 : WORKDIR /usr/src/app
Removing intermediate container 89b78367b7d1
---> 268af091e125
Step 3/7 : COPY package*.json ./
---> b3edff5b5cf6
Step 4/7 : RUN npm install
---> Running in 0670d7969369
npm WARN js-server@1.0.0 No description
npm WARN js-server@1.0.0 No repository field.

added 50 packages in 1.169s
Removing intermediate container 0670d7969369
---> 16fb3f0f4190
Step 5/7 : COPY . .
---> 67b06521ab96
Step 6/7 : EXPOSE 8080
---> Running in d9f8a3aa4fc7
Removing intermediate container d9f8a3aa4fc7
---> fa5f49d966cb
Step 7/7 : CMD [ "npm", "start" ]
---> Running in 0bebca56afc0
Removing intermediate container 0bebca56afc0
---> ea3dd6562815
Successfully built ea3dd6562815
pdarii@P5MD-PORT142:~/public_html/docker-examples/node-js-server$
```

How images are build

## docker images

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	bfe80a1b20ab	27 minutes ago	676MB
node	carbon	78f8aef50581	10 days ago	673MB
node	test	785454545454	6 minutes ago	118MB

How images are build

## docker history

```
$ docker history bfe80a1b20ab
IMAGE                CREATED              CREATED BY                                      SIZE
bfe80a1b20ab         28 minutes ago      /bin/sh -c #(nop)  CMD ["npm" "start"]         0B
5bad2c7c2cd3         28 minutes ago      /bin/sh -c #(nop)  EXPOSE 8080                  0B
1dafcdc908e7         28 minutes ago      /bin/sh -c #(nop)  COPY dir:c190e8c7d76314943...    1.65MB
1769a43875f9         28 minutes ago      /bin/sh -c npm install                          2.24MB
4d7c9a04112b         29 minutes ago      /bin/sh -c #(nop)  COPY multi:47393a063c0494b...  13.7kB
ac7fc6f4a26e         29 minutes ago      /bin/sh -c #(nop)  WORKDIR /usr/src/app          0B
78f8aef50581         10 days ago         /bin/sh -c #(nop)  CMD ["node"]                 0B
<missing>            10 days ago         /bin/sh -c set -ex  && for key in      6A010...    4.44MB
<missing>            10 days ago         /bin/sh -c #(nop)  ENV YARN_VERSION=1.5.1        0B
<missing>            10 days ago         /bin/sh -c ARCH= && dpkgArch="$(dpkg --print...  56.6MB
<missing>            10 days ago         /bin/sh -c #(nop)  ENV NODE_VERSION=8.11.1       0B
<missing>            10 days ago         /bin/sh -c set -ex  && for key in      94AE3...    129kB
<missing>            10 days ago         /bin/sh -c groupadd --gid 1000 node  && use...    335kB
<missing>            11 days ago         /bin/sh -c set -ex; apt-get update; apt-ge...    320MB
<missing>            11 days ago         /bin/sh -c apt-get update && apt-get install...  123MB
<missing>            11 days ago         /bin/sh -c set -ex; if ! command -v gpg > /...    0B
<missing>            11 days ago         /bin/sh -c apt-get update && apt-get install...  41.2MB
<missing>            2 weeks ago         /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>            2 weeks ago         /bin/sh -c #(nop)  ADD file:3e6141c0c9cb74b14...  127MB
$
```



# Docker containers

## Docker containers

### Run container

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
<none>	<none>	0b9475145111	6 minutes ago	676MB
node-js-server	latest	1acb0a86f9ca	32 minutes ago	676MB

```
$ docker run node-js-server

> js-server@1.0.0 start /usr/src/app
> node index.js

Running on http://0.0.0.0:8080
█
```

## Docker containers

# List containers

```
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
7eaaa066045a   node-js-server "npm start"             36 seconds ago Up 34 seconds  8080/tcp       priceless_easley
$
```

## Connect to container

`docker exec -it <mycontainer> bash`

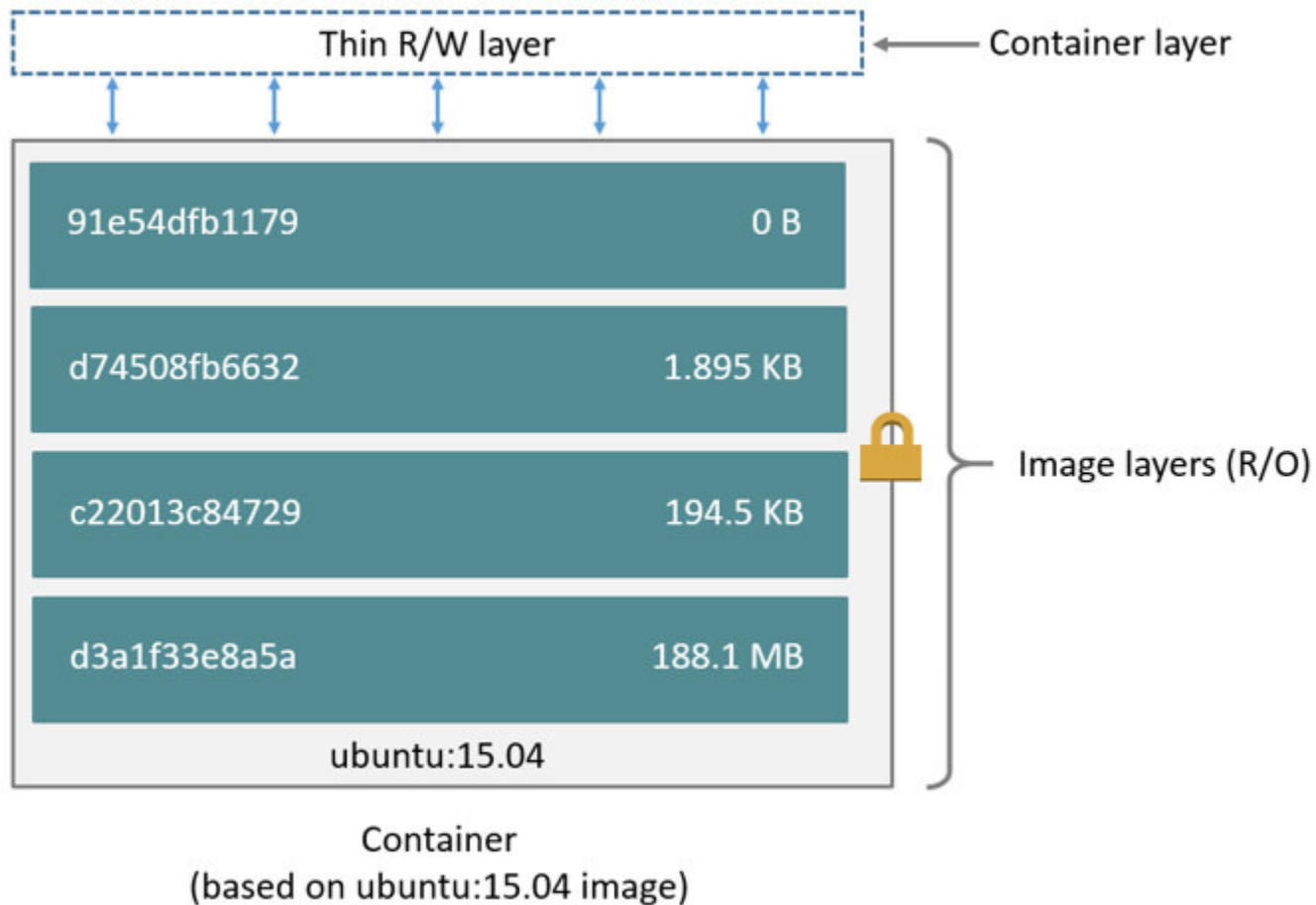
`--tty , -t` Allocate a pseudo-TTY

`--interactive , -i` Keep STDIN open even if not attached

```
pdarii@P5MD-PORT142:~$ docker exec -it 7fbe6aac6e00 bash
root@7fbe6aac6e00:/usr/src/app# node -v
v8.11.1
root@7fbe6aac6e00:/usr/src/app#
```

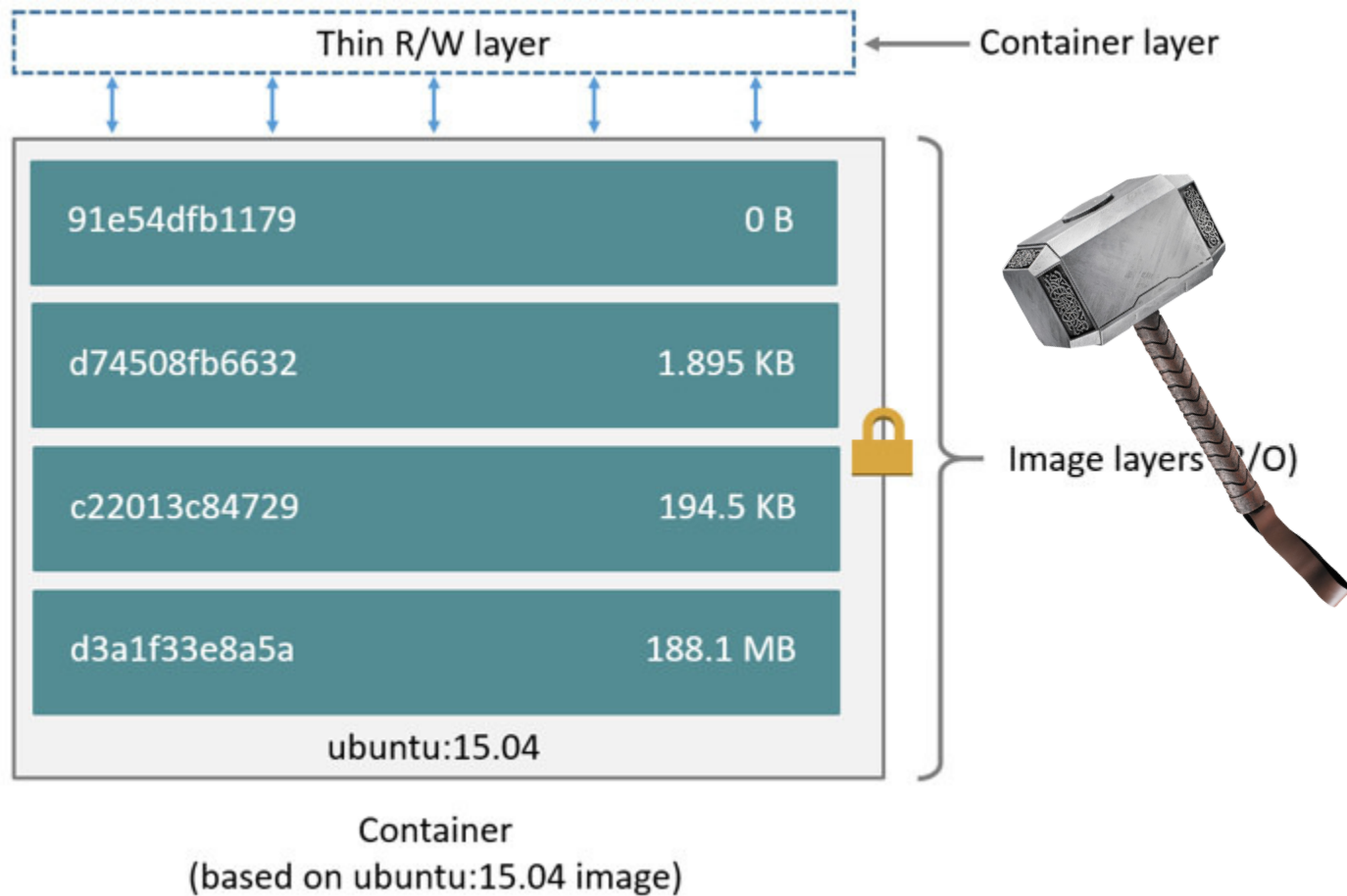
# Question !

<https://docs.docker.com/v17.09/engine/userguide/storagedriver/imagesandcontainers/#images-and-layers>



# Question !

We can break this lock?



# Docker compose

**Docker compose is a tool for defining and running multi-container Docker applications**



# Docker compose

## Config example

```
version: '3.3'

services:
  db:
    image: mysql
    container_name: database.dev
    volumes:
      - ./data/mysql/:/var/lib/mysql
    ports:
      - "3306:3306"
    networks:
      - custom
    environment:
      MYSQL_ROOT_PASSWORD: pass
      MYSQL_USER: root
      MYSQL_PASSWORD: pass
      MYSQL_ALLOW_EMPTY_PASSWORD: "yes"
```

## Docker compose

Run config file

```
pdarii@P5MD-PORT142:~/public_html/compose$ docker-compose up -d
Starting redis          ... done
Starting rabbitmq       ... done
Starting database.dev   ... done
Starting mongo.dev      ... done
```

# Docker network

# Docker network

- bridge - The default network driver.
- host - Remove network isolation between the container and the Docker host
- none - Disable all networking

# Docker network

## List networks

```
pdarii@P5MD-PORT142:~/public_html/compose$ docker network ls
```

NETWORK ID	NAME	DRIVER	SCOPE
ff9a9392a0da	appdesignerback backend	bridge	local
823c7d6cb68d	bridge	bridge	local
2d34b65e4a4f	br-2d34b65e4a4f	bridge	local

## ifconfig

```
pdarii@P5MD-PORT142:~/public_html/compose$ ifconfig
```

br-2d34b65e4a4f Link encap:Ethernet HWaddr 02:42:dd:86:64:8a  
inet addr:172.19.0.1 Bcast:172.19.255.255 Mask:255.255.0.0  
UP BROADCAST MULTICAST MTU:1500 Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:0  
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

br-4f48456a7690 Link encap:Ethernet HWaddr 02:42:5d:3d:e7:0b  
inet addr:172.20.0.1 Bcast:172.20.255.255 Mask:255.255.0.0  
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
RX packets:5596 errors:0 dropped:0 overruns:0 frame:0  
TX packets:12870 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:0  
RX bytes:5838754 (5.8 MB) TX bytes:2783393 (2.7 MB)

br-b9f8ed767581 Link encap:Ethernet HWaddr 02:42:b1:38:22:87  
inet addr:172.21.0.1 Bcast:172.21.255.255 Mask:255.255.0.0  
UP BROADCAST MULTICAST MTU:1500 Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:0  
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

br-ff9a9392a0da Link encap:Ethernet HWaddr 02:42:1f:5b:66:93  
inet addr:172.22.0.1 Bcast:172.22.255.255 Mask:255.255.0.0  
UP BROADCAST MULTICAST MTU:1500 Metric:1  
RX packets:0 errors:0 dropped:0 overruns:0 frame:0  
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0  
collisions:0 txqueuelen:0  
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

# Tips and tricks



## "Services"

```
version: '3.3'

services:
  db:
    image: mysql
    container_name: database.dev
    volumes:
      - ./data/mysql:/var/lib/mysql
    ports:
      - "3306:3306"
    networks:
      - custom
    environment:
      MYSQL_ROOT_PASSWORD: pass
      MYSQL_USER: root
      MYSQL_PASSWORD: pass
      MYSQL_ALLOW_EMPTY_PASSWORD: "yes"
  mongo:
    image: mongo:3.2
    container_name: mongo.dev
    volumes:
      - ./data/mongo:/data/db
    ports:
      - "27017:27017"
  rabbitmq:
    container_name: rabbitmq
    image: rabbitmq
    ports:
      - "5672:5672"
    networks:
      - custom
  redis:
    container_name: redis
    image: redis
    ports:
      - "6379:6379"
    networks:
      - custom
networks:
  custom:
    external: true
```

You, a few seconds ago • Uncommitted changes

## "Containers"

```
version: '3.3'
services:
  application:
    build: .
    container_name: core
    volumes:
      - ../var/www/html
    ports:
      - 8090:80
    networks:
      - custom
    external_links:
      - database.dev:database
    environment:
      XDEBUG_CONFIG: remote_host=192.168.227.33
networks:
  custom:
    external: true
```

## "App config"

```
parameters:
  database_host: database
  database_name:
  database_port: 3600
  database_user:
  database_password: |
```



## "Test Dockerfile commands"

```
FROM php:7.1-apache

# Apache config
RUN a2enmod rewrite

# Install Git
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
        git \
        curl \
    && rm -rf /var/lib/apt/lists/*

ADD docker/apache/vhost.conf /etc/apache2/sites-available/000-default.conf

RUN yes | pecl install xdebug \
    && echo "zend_extension=$(find /usr/local/lib/php/extensions/ -name xdebug.so)" > /usr/local/etc/php/conf.d/xdebug.ini \
    && echo "xdebug.remote_enable=on" >> /usr/local/etc/php/conf.d/xdebug.ini \
    && echo "xdebug.remote_autostart=off" >> /usr/local/etc/php/conf.d/xdebug.ini
```

- Begin with clean image
- Connect to container "docker exec -it {container-name} bash"
- Run command inside container, check for errors
- Add RUN command to Dockerfile and rebuild image
- Add new commands at the end of the file.

## "Docker ADD vs VOLUME"

- ADD - copy files to the image at build time.
- VOLUME - mount a host folder inside your container

don't copy files with sensitive information on image!!

**Questions ?**