

# Docker Compose

## Compose

Compose

Docker

Compose

YML

YML

YAML

Compose

- Dockerfile
- docker-compose.yml
- docker-compose up

docker-compose.yml

```
# yaml
version: '3'
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/code
      - logvolume01:/var/log
    links:
      - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

## Compose

## Linux

Linux

[Github](#) [compose-releases](#)

[Docker](#) [Compose](#)

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.24.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

[Compose](#) 1.24.1

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

```
$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
```

```
$ docker-compose --version
docker-compose version 1.24.1, build 4667896b
```

[alpine](#) | [py-pip](#) | [python-dev](#) | [libffi-dev](#) | [openssl-dev](#) | [gcc](#) | [libc-dev](#) | [make](#)

## macOS

Mac [Docker](#) | [Docker Toolbox](#) | [Compose](#) | [Docker](#) | [Compose](#) | [Docker](#) | [MacOS Docker](#)

## windows PC

Windows [Docker](#) | [Docker Toolbox](#) | [Compose](#) | [Docker](#) | Wir [Compose](#) | [Docker](#) | [Windows Docker](#)

# 1

```
$ mkdir composetest
$ cd composetest
```

[app.py](#)

[composetest/app.py](#)

```
import time

import redis
from flask import Flask
```

```

app = Flask(__name__)
cache = redis.Redis(host='redis', port=6379)

def get_hit_count():
    retries = 5
    while True:
        try:
            return cache.incr('hits')
        except redis.exceptions.ConnectionError as exc:
            if retries == 0:
                raise exc
            retries -= 1
            time.sleep(0.5)

@app.route('/')
def hello():
    count = get_hit_count()
    return 'Hello World! I have been seen {} times.\n'.format(count)

```

redis                  redis                  6379

composetest                  requirements.txt

```

flask
redis

```

## 2 | Dockerfile

composetest                  Dockerfile

```

FROM python:3.7-alpine
WORKDIR /code
ENV FLASK_APP app.py
ENV FLASK_RUN_HOST 0.0.0.0
RUN apk add --no-cache gcc musl-dev linux-headers
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt
COPY . .

```

```
CMD ["flask", "run"]
```

#### Dockerfile

- FROM python:3.7-alpine: Python 3.7
- WORKDIR /code: /code
- ```
ENV FLASK_APP app.py
ENV FLASK_RUN_HOST 0.0.0.0

flask
```
- RUN apk add --no-cache gcc musl-dev linux-headers: gcc MarkupSafe SQLAlchemy P
- ```
COPY requirements.txt requirements.txt
RUN pip install -r requirements.txt

requirements.txt Python
```
- COPY .: .
- CMD ["flask", "run"]: flask run

### 3 docker-compose.yml

docker-compose.yml

#### docker-compose.yml

```
# yaml
version: '3'
services:
  web:
    build: .
    ports:
      - "5000:5000"
  redis:
    image: "redis:alpine"
```

Compose web redis

- web web Dockerfile 5000 Flask Web 5000
- redis redis Docker Hub Redis

### 4 Compose

```
docker-compose up
```

```
-d
```

```
docker-compose up -d
```

```
yml
```

## version

```
yml compose
```

## build

```
webapp ./dir/Dockerfile
```

```
version: "3.7"
services:
  webapp:
    build: ./dir
```

Dockerfile args

```
version: "3.7"
services:
  webapp:
    build:
      context: ./dir
      dockerfile: Dockerfile-alternate
      args:
        buildno: 1
      labels:
        - "com.example.description=Accounting webapp"
        - "com.example.department=Finance"
        - "com.example.label-with-empty-value"
      target: prod
```

- context
- dockerfile Dockerfile
- args
- labels

- target

## cap\_add cap\_drop

```
cap_add:
  - ALL #

cap_drop:
  - SYS_PTRACE # ptrace
```

## cgroup\_parent

cgroup

```
cgroup_parent: m-executor-abcd
```

## command

```
command: ["bundle", "exec", "thin", "-p", "3000"]
```

## container\_name

```
container_name: my-web-container
```

## depends\_on

- docker-compose up db redis web
- docker-compose up SERVICE SERVICE docker-compose up web db redi
- docker-compose stop web db redis

```
version: "3.7"
services:
  web:
    build: .
    depends_on:
      - db
      - redis
  redis:
```

```
image: redis
db:
  image: postgres
```

web          redis db

# deploy

swarm

```
version: "3.7"
services:
  redis:
    image: redis:alpine
    deploy:
      mode replicated
      replicas: 6
      endpoint_mode: dnsrr
      labels:
        description: "This redis service label"
    resources:
      limits:
        cpus: '0.50'
        memory: 50M
      reservations:
        cpus: '0.25'
        memory: 20M
    restart_policy:
      condition: on-failure
      delay: 5s
      max_attempts: 3
      window: 120s
```

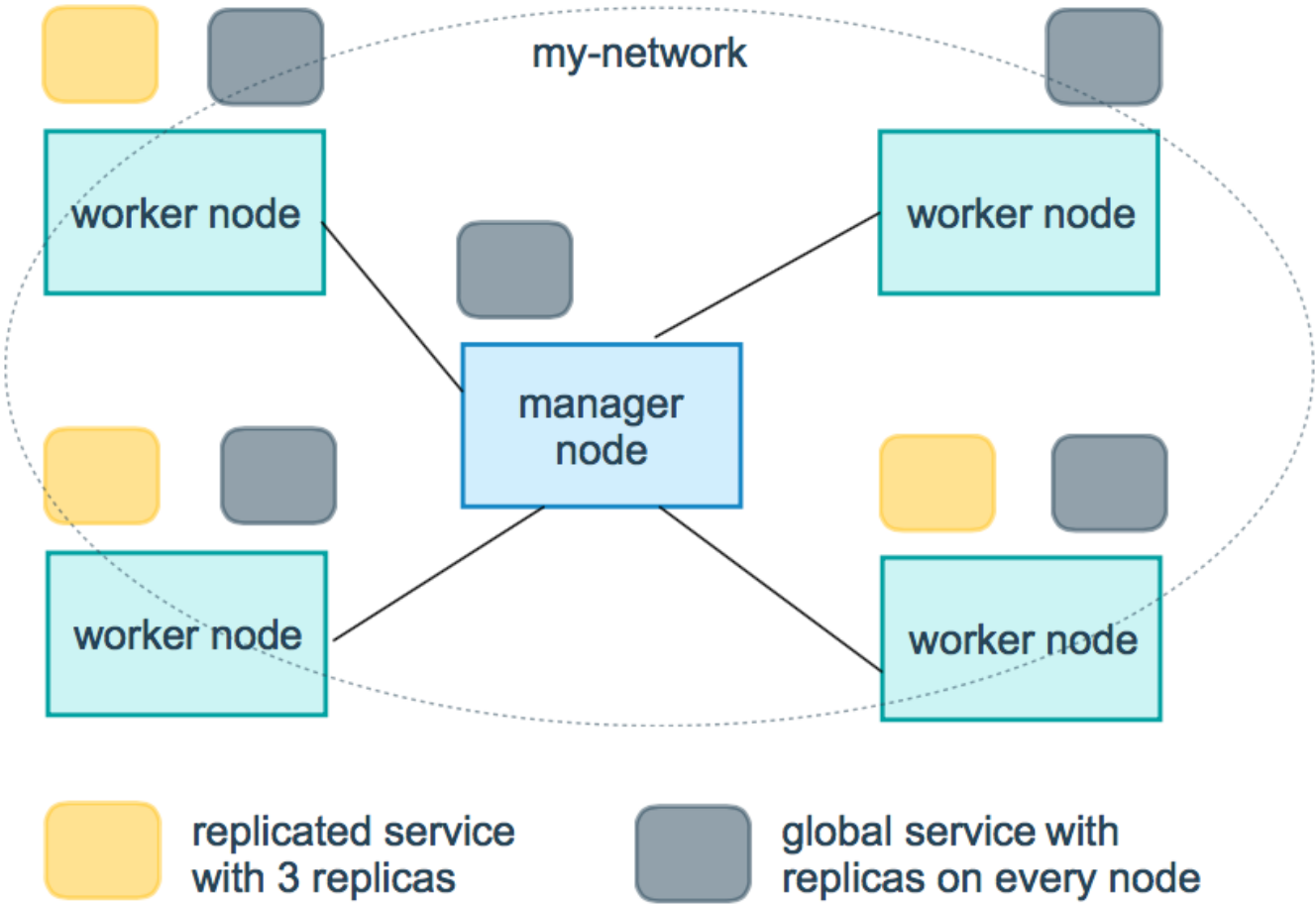
## endpoint\_mode

```
endpoint_mode: vip
# Docker          ip          ip
endpoint_mode: dnsrr
# DNS      DNSRR          ip      ip
```

labels labels deploy deploy labels

mode

- replicated
- global
- replicated global



replicas mode replicated

resources redis cpu

restart\_policy

- condition none on-failure any any
- delay 0
- max\_attempts
- window 0
- rollback\_config
- parallelism 0
- delay 0s



- |                     |            |                  |             |            |
|---------------------|------------|------------------|-------------|------------|
| • failure_action    |            | continue         | pause       | pause      |
| • monitor           |            | (ns us ms s m h) | 0s          |            |
| • max_failure_ratio |            | 0                |             |            |
| • order             | stop-first |                  | start-first | stop-first |
| • update_config     |            |                  |             |            |
| • parallelism       |            |                  |             |            |
| • delay             |            |                  |             |            |
| • failure_action    |            | continue         | rollback    | pause      |
| • monitor           |            | (ns us ms s m h) | 0s          |            |
| • max_failure_ratio |            |                  |             |            |
| • order             | stop-first |                  | start-first | stop-first |

V3.4

# devices

```
devices:
  - "/dev/ttyUSB0: /dev/ttyUSB0"
```

dns

## DNS

```
dns: 8.8.8.8
```

```
dns:
  - 8.8.8.8
  - 9.9.9.9
```

## dns\_search

## DNS

dns\_search: example.com

```
dns_search:
- dc1.example.com
- dc2.example.com
```

## entrypoint

## entrypoint

```
entrypoint: /code/entrypoint.sh
```

```
entrypoint:
  - php
  - -d
  - zend_extension=/usr/local/lib/php/extensions/no-debug-non-zts-20100525/xdebug.so
  - -d
  - memory_limit=-1
  - vendor/bin/phpunit
```

## env\_file

```
env_file: .env
```

```
env_file:
  - ./common.env
  - ./apps/web.env
  - /opt/secrets.env
```

## environment

YML

True False

```
environment:
  RACK_ENV: development
  SHOW: 'true'
```

## expose

```
expose:
  - "3000"
  - "8000"
```

## extra\_hosts

docker client --add-host

```
extra_hosts:
- "somehost: 162. 242. 195. 82"
- "otherhost: 50. 31. 209. 229"
```

/etc/hosts      ip

```
162. 242. 195. 82   somehost
50. 31. 209. 229    otherhost
```

## healthcheck

docker

```
healthcheck:
  test: ["CMD", "curl", "-f", "http://localhost"] #
  interval: 1m30s #
  timeout: 10s #
  retries: 3 #
  start_period: 40s #
```

## image

```
image: redis
image: ubuntu:14.04
image: tutum/influxdb
image: example-registry.com:4000/postgresql
image: a4bc65fd #    id
```

## logging

driver                      json-file

```
driver: "json-file"
driver: "syslog"
driver: "none"
```

json-file

```
logging:
  driver: json-file
  options:
    max-size: "200k" # 200k
    max-file: "10" # 10
```

syslog                syslog-address

```
logging:
  driver: syslog
  options:
    syslog-address: "tcp://192.168.0.42:123"
```

## network\_mode

```
network_mode: "bridge"
network_mode: "host"
network_mode: "none"
network_mode: "service:[service name]"
network_mode: "container:[container name/id]"
```

networks

networks

```
services:
  some-service:
    networks:
      some-network:
        aliases:
          - alias1
      other-network:
        aliases:
          - alias2
networks:
  some-network:
    # Use a custom driver
    driver: custom-driver-1
  other-network:
    # Use a custom driver which takes special options
```

```
driver: custom-driver-2
```

## aliases

# restart

- no
- always
- on-failure 0 Docker
- unless-stopped

```
restart: "no"
restart: always
restart: on-failure
restart: unless-stopped
```

swarm restart\_policy

# secrets

```
version: "3.1"
services:

mysql:
  image: mysql
  environment:
    MYSQL_ROOT_PASSWORD_FILE: /run/secrets/my_secret
  secrets:
    - my_secret

secrets:
  my_secret:
    file: ./my_secret.txt
```

# security\_opt

schema

```
security-opt
- label: user: USER #
```

- label: role: ROLE #
- label: type: TYPE #
- label: level: LEVEL #

## stop\_grace\_period

SIGTERM ( stop\_signal ) SIGKILL

```
stop_grace_period: 1s # 1
stop_grace_period: 1m30s # 1 30
```

10

## stop\_signal

SIGTERM

SIGUSR1 SIGTERM

```
stop_signal: SIGUSR1
```

## sysctls

```
sysctls:
  net.core.somaxconn: 1024
  net.ipv4.tcp_syncookies: 0

sysctls:
- net.core.somaxconn=1024
- net.ipv4.tcp_syncookies=0
```

## tmpfs

```
tmpfs: /run

tmpfs:
- /run
- /tmp
```

# ulimits

ulimit

```
ulimits:
  nproc: 65535
  nofile:
    soft: 20000
    hard: 40000
```

# volumes

```
version: "3.7"
services:
  db:
    image: postgres:latest
    volumes:
      - "/localhost/postgres.sock:/var/run/postgres/postgres.sock"
      - "/localhost/data:/var/lib/postgresql/data"
```

---

Revision #2

Created 20 March 2020 13:55:55 by

Updated 20 March 2020 14:56:35 by