如果你有多个容器，要启动这些容器需要每个容器都输入一遍命令，那不是很麻烦，docker-compose.yml就是为了解决此问题

**Compose 安装**

1. 下载Compose，（发版本地址：https://github.com/docker/compose/releases）

# 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

将版本替换为你需要的版本

1. 添加可执行权限

# sudo chmod +x /usr/local/bin/docker-compose

1. 创建软链接

# sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

1. 测试是否安装成功

# docker-compose --version

**容器文件详解**

详细文档参考：<https://www.runoob.com/docker/docker-compose.html>

# 指定本 yml 依从的 compose 哪个版本制定的

version: "3.7"

services:

webapp:

# 指定为从上下文路径 ./dir/Dockerfile 所构建的镜像

build: ./dir

# 覆盖容器启动后默认执行的命令, 支持 shell 格式和 [] 格式

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

# 指定自定义容器名称

container\_name: my-web-container

# 设置依赖关系

depends\_on:

- db

- redis

# 自定义 DNS 服务器，可以是单个值或列表的多个值

dns: 8.8.8.8

# 覆盖容器默认的 entrypoint

entrypoint: /code/entrypoint.sh

# 添加环境变量，您可以使用数组或字典、任何布尔值，布尔值需要用引号引起来

environment:

RACK\_ENV: development

SHOW: 'true'

# 端口映射

ports:

- '3000:3000'

# 添加主机名映射

extra\_hosts:

- "somehost:162.242.195.82"

- "otherhost:50.31.209.229"

# 指定容器运行的镜像

image: ubuntu:14.04

# 设置网络模式，取值 "bridge" | "host" | "none" | "service:[service name]" | "container:[container name/id]"

network\_mode: "bridge"

networks:

# 加入some-network网络

default:

# 网络上的名称

aliases:

- alias1

other-network:

aliases:

- alias2

# 重启策略，取值 "no" | always | on-failure | unless-stopped

restart: "no"

# 将主机的数据卷或着文件挂载到容器里

volumes:

- "/localhost/postgres.sock:/var/run/postgres/postgres.sock"

- "/localhost/data:/var/lib/postgresql/data"

redis:

image: redis

db:

image: postgres

# docker-compose有提供默认网络（名为default），如果你想自定义网络，可以使用如下

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

**常用命令**

销毁容器

*# docker-compose down*

构建并运行容器（-d后台运行）

*# docker-compose up -d*

运行容器（如果容器没有构建会报错）

*# docker-compose start*

# 停止容器

*# docker-compose stop*

# 移除已停止的容器

*# docker-compose rm*