

# docker-compose.yml 语法说明

## YAML 模板文件语法

默认的模板文件是 `docker-compose.yml`，其中定义的每个服务都必须通过 `image` 指令指定镜像或 `build` 指令（需要 `Dockerfile`）来自动构建。

其它大部分指令都跟 `docker run` 中的类似。

如果使用 `build` 指令，在 `Dockerfile` 中设置的选项(例如： `CMD`, `EXPOSE`, `VOLUME`, `ENV` 等) 将会自动被获取，无需在 `docker-compose.yml` 中再次设置。

`image`

指定为镜像名称或镜像 ID。如果镜像在本地不存在，Compose 将会尝试拉去这个镜像。

例如：

```
image: ubuntu
image: orchardup/postgresql
image: a4bc65fd
```

## build

指定 `Dockerfile` 所在文件夹的路径。Compose 将会利用它自动构建这个镜像，然后使用这个镜像。

`build: /path/to/build/dir`

## command

覆盖容器启动后默认执行的命令。

`command: bundle exec thin -p 3000`

## links

链接到其它服务中的容器。使用服务名称（同时作为别名）或服务名称：服务别名（`SERVICE:ALIAS`）格式都可以。

`links:`

- db
- db:database
- redis

使用的别名将会自动在服务容器中的 `/etc/hosts` 里创建。例如：

`172.17.2.186 db`

相应环境变量也将被创建。

## external\_links

链接到 `docker-compose.yml` 外部的容器，甚至 并非 Compose 管理的容器。参数格式跟 `links` 类似。

`external_links:`

- redis\_1
- project\_db\_1:mysql
- project\_db\_1:postgresql

## ports

暴露端口信息。

使用宿主：容器（`HOST:CONTAINER`）格式或者仅仅指定容器的端口（宿主将会随机选择端口）都可以。

`ports:`

- "3000"
- "8000:8000"
- "127.0.0.1:8001:8001"

注：当使用 `HOST:CONTAINER` 格式来映射端口时，如果你使用的容器端口小于 60 你可能会得到错误得结果，因为 `YAML` 将会解析 `xx:yy` 这种数字格式为 60 进制。所以建议采用字符串格式。

## expose

## 公告

昵称: freefei  
园龄: 4年2个月  
粉丝: 2  
关注: 3  
[+加关注](#)

< 2016年11月 >						
日	一	二	三	四	五	六
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	1	2	3
4	5	6	7	8	9	10

## 搜索

找找看

谷歌搜索

## 常用链接

[我的随笔](#)  
[我的评论](#)  
[我的参与](#)  
[最新评论](#)  
[我的标签](#)

## 我的标签

[docker\(5\)](#)  
[git\(5\)](#)  
[shell\(3\)](#)  
[私有盘\(2\)](#)  
[pdo\(2\)](#)  
[php\(2\)](#)  
[rancher\(2\)](#)  
[gitlab\(2\)](#)  
[linux\(2\)](#)  
[lomox\(2\)](#)  
[更多](#)

## 随笔分类

[c/c++](#)  
[docker\(15\)](#)  
[go\(1\)](#)  
[javascript\(5\)](#)  
[linux\(27\)](#)  
[node.js\(1\)](#)  
[php\(5\)](#)  
[shell](#)  
[架构文章\(2\)](#)  
[嵌入式](#)

## 随笔档案

[2016年10月 \(2\)](#)  
[2016年9月 \(4\)](#)  
[2016年8月 \(1\)](#)  
[2016年6月 \(3\)](#)  
[2016年5月 \(2\)](#)  
[2016年4月 \(9\)](#)  
[2016年3月 \(12\)](#)  
[2016年2月 \(1\)](#)  
[2016年1月 \(5\)](#)  
[2015年12月 \(1\)](#)  
[2015年11月 \(1\)](#)  
[2015年8月 \(2\)](#)  
[2015年7月 \(1\)](#)

暴露端口，但不映射到宿主机，只被连接的服务访问。

仅可以指定内部端口为参数

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

volumes

卷挂载路径设置。可以设置宿主机路径（HOST:CONTAINER）或加上访问模式（HOST:CONTAINER:ro）。

```
volumes:
- /var/lib/mysql
- cache:/tmp/cache
- ~/configs:/etc/configs/:ro
```

volumes\_from

从另一个服务或容器挂载它的所有卷。

```
volumes_from:
- service_name
- container_name
```

environment

设置环境变量。你可以使用数组或字典两种格式。

只给定名称的变量会自动获取它在 Compose 主机上的值，可以用来防止泄露不必要的数据。

```
environment:
- RACK_ENV=development
- SESSION_SECRET
```

env\_file

从文件中获取环境变量，可以为单独的文件路径或列表。

如果通过 docker-compose -f FILE 指定了模板文件，则 env\_file 中路径会基于模板文件路径。

如果有变量名称与 environment 指令冲突，则以后者为准。

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

环境变量文件中每一行必须符合格式，支持 # 开头的注释行。

```
# common.env: Set Rails/Rack environment
RACK_ENV=development
```

extends

基于已有的服务进行扩展。例如我们已经有了一个 webapp 服务，模板文件为 common.yml。

```
# common.yml
webapp:
build: ./webapp
environment:
\ - DEBUG=false
\ - SEND_EMAILS=false
```

编写一个新的 development.yml 文件，使用 common.yml 中的 webapp 服务进行扩展。

development.yml

```
web:
extends:
file: common.yml
service: webapp
ports:
\ - "8000:8000"
links:
\ - db
```

- 2015年6月 (1)
- 2015年5月 (1)
- 2015年2月 (1)
- 2014年12月 (4)
- 2014年10月 (5)
- 2014年9月 (7)
- 2014年8月 (2)
- 2014年5月 (2)
- 2014年4月 (1)
- 2014年3月 (1)
- 2013年12月 (1)
- 2013年11月 (2)
- 2013年10月 (5)
- 2013年9月 (1)
- 2013年8月 (2)
- 2013年6月 (1)
- 2013年5月 (6)
- 2013年4月 (5)
- 2013年3月 (4)
- 2013年2月 (3)
- 2013年1月 (9)
- 2012年12月 (7)
- 2012年11月 (8)
- 2012年10月 (12)

文章分类

- bootstrap
- html5(1)
- javascript(3)
- jquery
- linux(16)
- lua
- mongodb
- node.js(9)
- php(6)
- shell
- thinkphp
- 个人随笔

相册

- 自建别墅(1)

技术博客

- edwardlost

阅读排行榜

- 1. docker-compose.yml 语法说明 (5952)
- 2. js 解析 bytearray 成 字符串(930)
- 3. ssss(529)
- 4. docker 下 安装rancher 笔记(324)
- 5. thinkphp 二维码封装函数(288)

推荐排行榜

- 1. docker-compose.yml 语法说明(2)

environment:

- DEBUG=true

db:

image: postgres

后者会自动继承 common.yml 中的 webapp 服务及相关环节变量。

### net

设置网络模式。使用 `docker client` 的 `--net` 参数一样的值。

net: "bridge"

net: "none"

net: "container:[name or id]"

net: "host"

### pid

跟主机系统共享进程命名空间。打开该选项的容器可以相互通过进程 ID 来访问和操作。

pid: "host"

### dns

配置 DNS 服务器。可以是一个值，也可以是一个列表。

dns: 8.8.8.8

dns:

- 8.8.8.8

- 9.9.9.9

### cap\_add, cap\_drop

添加或放弃容器的 Linux 能力（Capability）。

cap\_add:

- ALL

cap\_drop:

- NET\_ADMIN

- SYS\_ADMIN

### dns\_search

配置 DNS 搜索域。可以是一个值，也可以是一个列表。

dns\_search: example.com

dns\_search:

- domain1.example.com

\ - domain2.example.com

working\_dir, entrypoint, user, hostname, domainname, mem\_limit, privileged, restart, stdin\_open, tty,

cpu\_shares

这些都是和 `docker run` 支持的选项类似。

cpu\_shares: 73

working\_dir: /code

entrypoint: /code/entrypoint.sh

user: postgresql

hostname: foo

domainname: foo.com

mem\_limit: 1000000000

privileged: true

restart: always

stdin\_open: true

tty: true

分类: [docker](#)

标签: [docker-compose.yml](#)

« 上一篇: [docker 镜像和容器的批量清理](#)

» 下一篇: [docker-compose 工具安装](#)

[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#)网站首页。

- 【推荐】50万行VC++源码: 大型组态工控、电力仿真CAD与GIS源码库
- 【推荐】用1%的研发投入，搭载3倍性能的网易视频云技术
- 【推荐】融云发布 App 社交化白皮书 IM 提升活跃超 8 倍



最新IT新闻:

- 雷军：企业家都批评小米卖得太便宜！
  - 89元！公牛苹果安卓二合一数据线开卖：2.4A快充
  - 外卖哥因3个差评吃老鼠药欲自杀 朋友圈告别
  - 中外20家实验室发表文章 声称无法重复韩春雨实验
  - 乐视真正的商业模式与囚徒困境
- » 更多新闻...



最新知识库文章:

- 循序渐进地代码重构
  - 技术的正宗与野路子
  - 陈皓：什么是工程师文化？
  - 没那么难，谈CSS的设计模式
  - 程序猿媳妇儿注意事项
- » 更多知识库文章...