# Docker Compose

## 为什么使用Docker-Compose

当容器实例太多时，使用docker-Compose管理容器;类似于spring中appplicationContext.xml对bean对象统一集中管理；

## 什么是Docker-Compose

1.Docker-Compose是Docker官方的开源项目，负责实现对Docker容器集群的快速编排；

2.可以管理多个Docker容器组成一个应用，我们只需要定义一个Yaml格式的配置文件docker-compose.yml，写好读个容器之间的调用关系，然后，只要一个命令，就能同时启动或停止这些容器

## Docker-Compose能干什么

1.Docker建议每一个容器只运行一个服务，因为docker容器本身占用资源极少，所以最好是将每个服务器单独的分割开来但是这要面临一个问题，如果我们需要同时部署多个服务，每个服务单独编写dockerfile然后构建镜像，构建容器，工作量大;

所以docker的官方提供了docker-compose多服务部署工具；

2.Compose允许用户通过一个单独的docker-compose.yml模板文件（yml格式）来定义一组相关联的应用容器为一个项目；可以很容易地用一个配置文件定义一个多容器的应用，然后使用一条指令安装这个应用的所有依赖，完成构建。Docker-comose解决了容器与容器之间如何管理的问题。

# 安装Docker-Compose

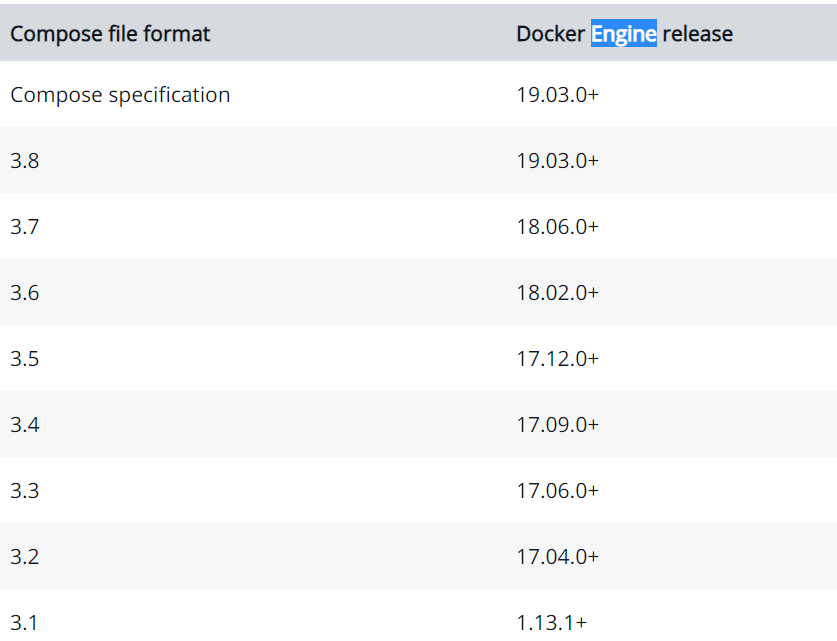
官网地址：<https://docs.docker.com/compose/compose-file/compose-file-v3/>

官网下载：<https://docs.docker.com/compose/install/>

**安装Compose3.8步骤：**

**要求：**

Docker-compose和docker 版本需要对应



1. 运行以下命令下载 Docker Compose 的当前稳定版本：

请替换3.8 为您要使用的 Compose 版本

$ sudo curl -L "https://github.com/docker/compose/releases/download/v3.3.8/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

2. 对二进制文件应用可执行权限：

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

3.查看版本信息

docker-compose --version

wget https://github.com/docker/compose/releases/download/ v3.8/docker-compose-$(uname -s)-$(uname -m) -O /usr/local/bin/docker-compose

**卸载步骤：**

**如果使用curl方式安装的Docker compose**

sudo rm /usr/local/bin/docker-compose

# Docker Compose使用

## 核心概念

**一个文件**：docker-compose.yml

**两个要素**：服务和工程

服务：一个个应用容器实例，如订单服务、mysq容器；

工程：由一组关联的应用容器组成的一个完整的业务单元,在docker-compose.yml文件中定义；

## 如何使用

Docker-compose使用步骤：

1. 编写Dockerfile定义各个微服务应用并构建出对应的镜像文件
2. 使用docker-compose.yml定义一个完整业务单元，安排好整体应用中的各个容器服务。最后，执行docker-compose up命令来启动并运行整个应用程序，完成一键部署上线；

## 常用命令

docker-compose -h #查看帮助

docker-compose up #启动所有docker-compose服务

docker-compose up -d #启动所有docker-compose服务并在后台运行

docker-compose down #停止并删除容器、网络、卷、镜像；

docker-compose exec yml里面的服务id #进入容器实例内部

docker-compose exec docker-compose.yml文件中写的服务id /bin/bash

docker-compose ps #展示当前docker-compose编排过的运行的所有容器

docker-compose top #展示当前docker-compose编排过的容器进程

docker-compose logs yml里面的服务id #查看容器输出日志

docker-compose config #检查配置

docker-compose config -q #检查配置，有问题才有输出

docker-compose restart #重启服务

docker-compose start #启动服务

docker-compose stop #停止服务

## 微服务编排

1. 不使用compose效果如何，有哪些问题？
2. 先后顺序要求固定，想mysql+redis才能启动微服务访问成功;
3. 需要单独运行容器(多次run)
4. 容器键的启停或宕机，有可能导致ip地址对应的容器实例变化，映射出错，要么生产环境IP写死(不推荐)，要么通过服务调用

**使用Compose 编排**

**1.编写java项目**

**2.编写docker-compose.yml文件**

**3.修改java项目配置信息**

# Swarm

# Kubernetes

## 特点：

1. 轻量级：消耗的资源比较少；
2. 开源
3. 弹性伸缩
4. 负载均衡

K8s集群

1.