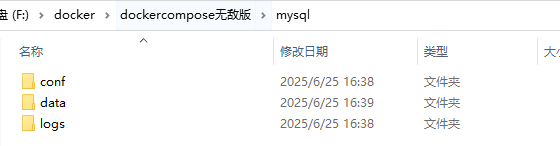
# 写在前面

本教程使用前默认你已经安装好了docker,docker-compose。本教程的目的是简化docker-compose的配置，笔者上一个版本的教程，个人觉得还是比较繁琐的，所以这个版本的旨在简化docker-compose编排过程。

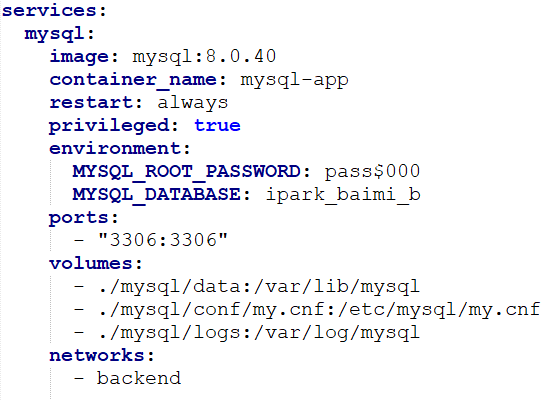


上图所示的就是常用项目所需要的的部署配置。依次是mysql、nginx、redis、service(springboot) ，接下来将依次介绍各个模块的配置。

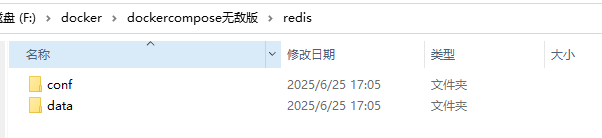
# Mysql



数据库模块的配置较为简单，他的目录如上图所示。Conf对应数据库的配置文件目录，目前包含my.cnf。对应docker-compose的配置如下图所示：



# Redis



Redis模块也是比较简单，整体结构如上图所示。conf对应数据库的配置文件目录，包含redis.conf。对应docker-compose的配置如下图所示：

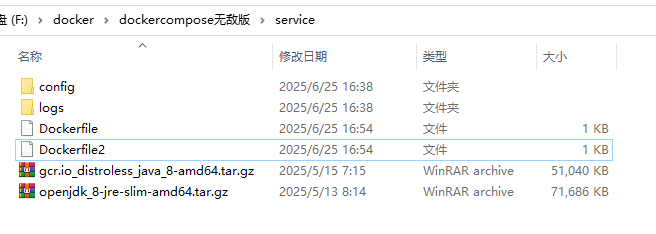


# Service

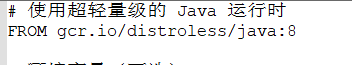
Service 模块相对来说比较复杂些，本教程在service模块的配置基于java8版本。我们可以在dockerfile中看到使用的jre镜像如下图所示。



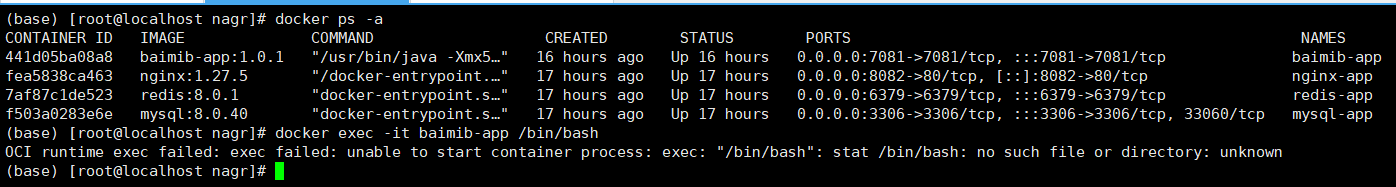
Service模块主要的配置如下图所示：config目录就是springboot服务启动所依赖的配置文件，Dockerfile 就是构建springboot 镜像的核心文件。Springboot的配置文件中，数据库就用容器名就行了,不需要写 ip，因为在同一个network。



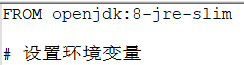
Dockerfile和Dockerfile2的区别就是Dockerfile用的是超轻量级的 Java 运行时，



这个镜像的有点就是体积小，占用空间少。缺点就是不能使用 docker exec 命令进入容器，适合在生产环境部署。



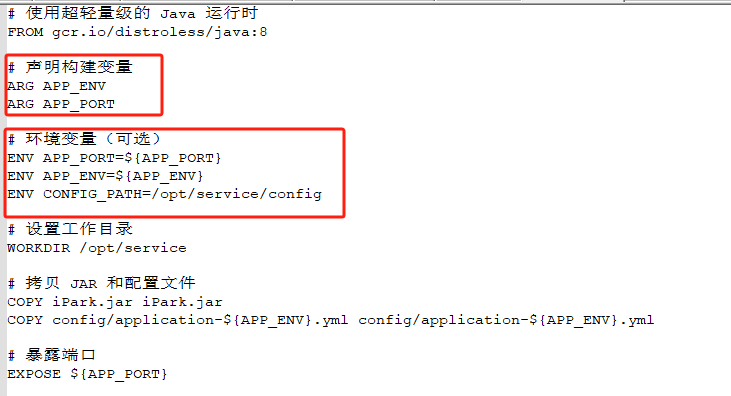
Dockerfile2使用的是openjdk:8-jre-slim。



Service模块对应docker-compose的配置如下图所示：



Service容器的构建和运行过程的参数（数据库端口，启动端口等）都配置在docker-compose中，这样避免多处配置。Dockerfile中通过如下配置接收参数：



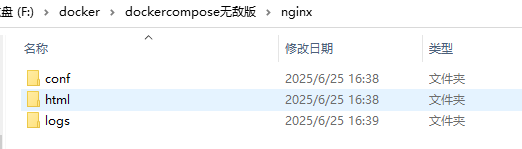
容器运行的yml配置文件中通过如下配置接收参数：



depends\_on标签表示service容器生成之前，需要等待mysql容器和redis容器都生成完成。

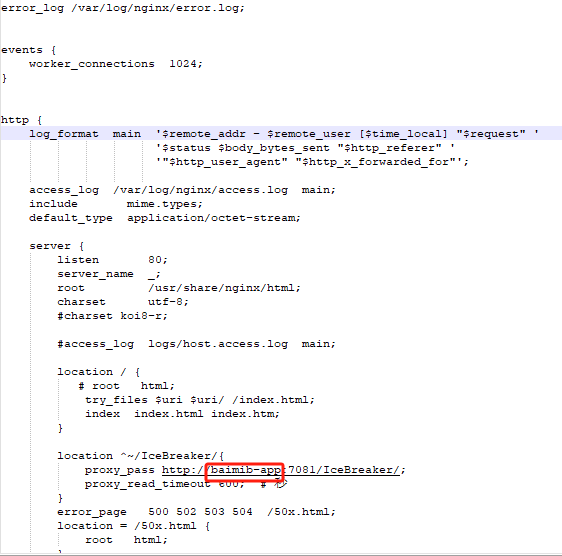
# nginx

nginx模块的整体结构如下图所示：

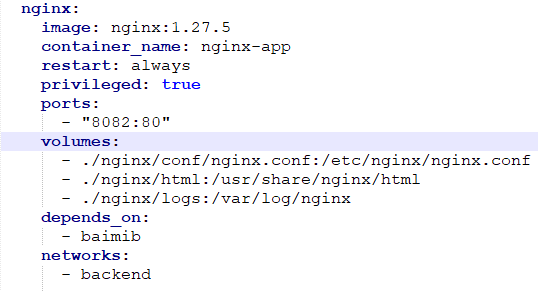


Conf存放nginx.conf，html存在前端页面，logs存在日志。

nginx.conf的结构如下：



如果跟service容器在同一个network，配置后端接口地址可以使用service的容器名。nginx模块对应docker-compose的配置如下图所示：

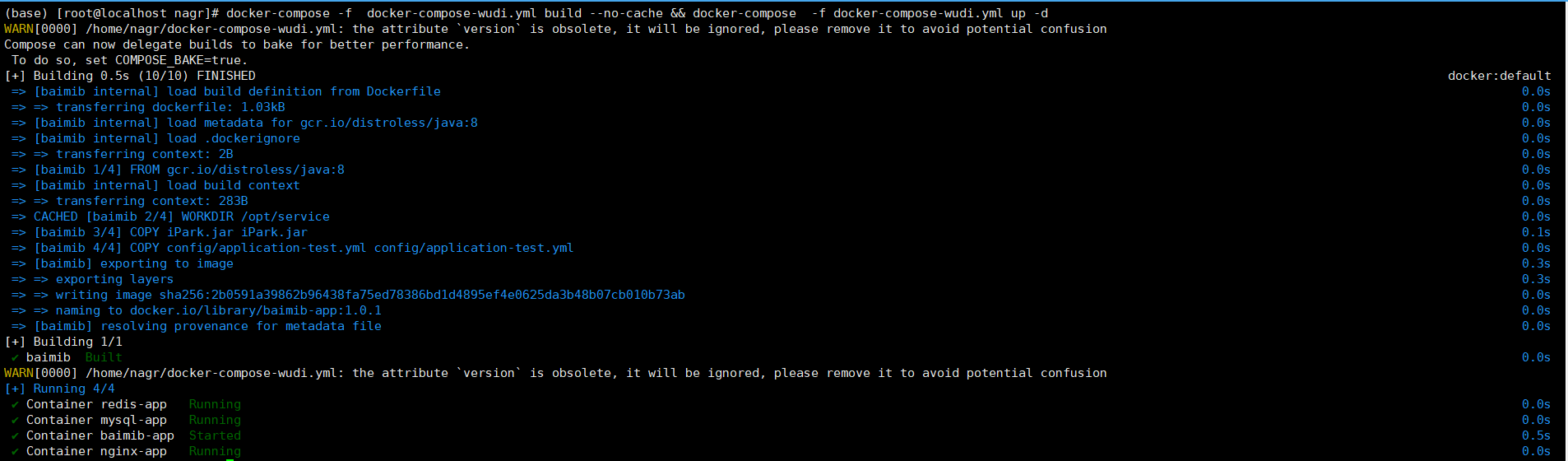


depends\_on标签表示nginx容器生成之前，需要等待后端容器生成完成。

# 总结

上述模块配置完之后，执行如下命令即可：

docker-compose -f docker-compose-wudi.yml build --no-cache && docker-compose -f docker-compose-wudi.yml up –d



如果在实际应用运用过程中需要添加其他模块，比如MQ，postgresql，按照docker-compose-wudi.yml的格式依次填写配置即可。