



链滴

SpringCloud Alibaba 微服务实战十五 - Sp ringCloud 容器化部署

作者: [jianzh5](#)

原文链接: <https://ld246.com/article/1585219736834>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

SpringCloud Alibaba系列文章已经写了16篇了，基本框架大体完成，业务相关的逻辑还需要根据项本身的业务进行梳理改造。

今天将是本系列的最后一篇 - SpringCloud容器化部署（理论上最后一篇，不排除后面会对相关组件行升级）

看在写了这么多的份上是不是应该点个在看呢？

每次都是：下次一定！
都多少个下次了？



开启Docker远程访问

由于我是在windows上进行开发没有安装docker，所以需要找一台安装好docker的服务器并开启远访问。使用mac的同学请忽略。

- 打开docker配置文件

```
vi /lib/systemd/system/docker.service
```

- 开放2376端口

找到ExecStart=/usr/bin/dockerd所在行，在后面追加-H tcp://0.0.0.0:2376 -H unix:///var/run/docker.sock，修改完成的效果如下：

```
ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock -H tcp://0.0.0.0:2376 -H unix:///var/run/docker.sock
```

- 重启docker服务

```
systemctl daemon-reload  
systemctl restart docker.service
```

- 使用netstat查看端口

```
netstat -nptl
```

```
[root@bingo-172 system]# netstat -nptl  
Active Internet connections (only servers)  
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name  
tcp        0      0 127.0.0.1:25            0.0.0.0:*                LISTEN     1619/master  
tcp        0      0 0.0.0.0:22              0.0.0.0:*                LISTEN     1686/sshd  
tcp6       0      0 :::1:25                  :::*                    LISTEN     1619/master  
tcp6       0      0 :::9600                  :::*                    LISTEN     21620/docker-proxy  
tcp6       0      0 :::5601                  :::*                    LISTEN     21592/docker-proxy  
tcp6       0      0 :::2376                  :::*                    LISTEN     21415/dockerd  
tcp6       0      0 :::9100                  :::*                    LISTEN     21683/docker-proxy  
tcp6       0      0 :::9200                  :::*                    LISTEN     21578/docker-proxy  
tcp6       0      0 :::5044                  :::*                    LISTEN     21637/docker-proxy  
tcp6       0      0 :::9300                  :::*                    LISTEN     21566/docker-proxy  
tcp6       0      0 :::22                    :::*                    LISTEN     1686/sshd
```

- 访问/info，确定端口正常开放

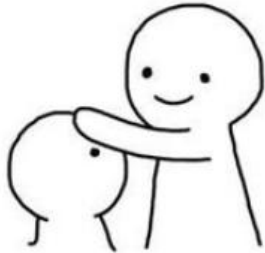
curl http://127.0.0.1:2376/info

```
[root@bings-172 ~]# curl http://127.0.0.1:2376/info
{"ID": "5560209c-0d1e-331a-52f8-5166", "WP": "CVC4-E3B8", "WB": "322A-PQR", "Containers": 5, "ContainersRunning": 4, "ContainersPaused": 0, "ContainersStopped": 1, "Images": 4, "Driver": "overlay2", "DriverStatus": [{"Backing Filesystem": "xfs"}, {"Supports_d_type": "true"}, {"Native Overlay Diff": "true"}], "SystemStatus": null, "Plugins": [{"Volume": "local"}, {"Network": "bridge"}, {"Host": "ipfs"}, {"Storage": "nfs"}, {"Image": "overlay2"}, {"Authz": "ratelimit"}, {"Log": "journald"}, {"JSONFile": "local"}, {"LogDriver": "syslog"}, {"Memory": "mem"}, {"Swap": "zfs"}, {"KernelMemory": "true"}, {"KernelMemoryTCP": "true"}, {"CPU": "cfs"}, {"CPUShares": "true"}, {"CPUSets": "true"}, {"PIDLimit": "true"}, {"IPv4Forwarding": "true"}, {"BridgeNfIptables": "true"}, {"BridgeNfIp6tables": "true"}, {"Debug": "false"}, {"NFd": "155"}, {"OOMKillDisable": "true"}, {"MiscRoutines": "63"}, {"SystemTime": "2020-03-20T11:22:52.47289528+08:00"}, {"LoggingDriver": "json-file"}, {"GroupDriver": "cgroupfs"}, {"EventsListener": "0"}, {"KernelVersion": "3.10.0-1062.9.1.el7.x86_64"}, {"OperatingSystem": "CentOS Linux 7 (Core)"}, {"OSType": "linux"}, {"Architecture": "x86_64"}, {"IndexServerAddress": "https://index.docker.io/v1/"}, {"RegistryConfig": {"AllowInsecureArtifacts": true}, {"AllowInsecureArtifactsCIDRs": []}, {"AllowInsecureArtifactsHostnames": []}, {"InsecureRegistryCIDRs": ["127.0.0.0/8"]}, {"IndexConfigs": {"docker.io": {"Name": "docker.io", "Mirrors": ["https://hub-mirror.c.163.com", "https://mirror.ccs.tencent.com", "https://reg-mirror.qiniu.co/"], "Secure": true, "Official": true}}, {"Mirrors": ["https://hub-mirror.c.163.com", "https://mirror.ccs.tencent.com", "https://reg-mirror.qiniu.co/"], "MCP": {"MCPID": "16655960784", "GenericResources": null}, {"DockerRootDir": "/var/lib/docker"}, {"HttpProxy": ""}, {"HttpsProxy": ""}, {"NoProxy": ""}, {"Name": "bings-172.31.0.207.fai.no.01.novalocal"}, {"Labels": {"ExperimentalBuild": "false"}, {"ServerVersion": "19.03.5"}, {"ClusterStore": ""}, {"ClusterAdvertise": ""}, {"Runtimes": {"runc": {"path": "runc"}, {"DefaultRuntime": "runc"}, {"Swarm": {"NoBuild": true}, {"NodeAddress": "local", {"NodeState": "inactive"}, {"ControlAvailable": "false"}, {"Error": ""}, {"NetworkManagers": null}, {"LiveRestoreEnabled": "false"}, {"Isolation": "cgroup"}, {"InitBinary": "docker-init"}, {"ContainerdCommit": {"ID": "b3455c34f1e51051e91910307917664f7e80269", "Expected": "133455c34f1e51051e91910307917664f7e80269"}, {"Expected": "133455c34f1e51051e91910307917664f7e80269"}, {"InitCommit": {"ID": "fec3683", "Expected": "fec3683"}, {"SecurityOptions": ["name=seccomp,profile=default"]}, {"Warnings": {"WARNING: API is accessible on http://0.0.0.0:2376 without encryption via Access to the remote API is equivalent to root access on the host. Refer to the 'Docker daemon attack surface' section in the documentation for more information: https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface"}}}
```

docker-maven-plugin 构建 docker镜像

在开始打包之前一定要先在**bom**和**common**模块执行**mvn clean install** 命令，否则打包不成功。

看过本系列文章的人一定也知道docker 和 docker-compose的相关指令了，这里就不再说明。



优秀的同学，
绝对是优秀的同学

- 在需要构建组件的模块引入docker-maven-plugin插件

```
<plugin>
<groupId>com.spotify</groupId>
<artifactId>docker-maven-plugin</artifactId>
<version>1.2.2</version>
<configuration>
<imageName>${project.artifactId}</imageName>
<imageTags>
<imageTag>latest</imageTag>
</imageTags>
<!--指定Dockerfile路径-->
<dockerDirectory>${project.basedir}/src/main/docker</dockerDirectory>
<dockerHost>http://xxx.xx.xx.xx:2376</dockerHost>
<resources>
<resource>
<targetPath>/</targetPath>
<!--${project.basedir}/target-->
<directory>${project.build.directory}</directory>
<!--${project.artifactId}-${project.version}-->
<include>${project.build.finalName}.jar</include>
</resource>
</resources>
</configuration>
</plugin>
```

这里端口与前面开放的端口保持一致。

- 在模块的 **src/main/docker** 目录下建立Dockerfile文件

```
FROM openjdk:8-jdk-alpine
VOLUME /tmp
ADD cloud-gateway-1.0.0.jar app.jar
RUN sh -c 'touch /app.jar'
ENTRYPOINT ["java", "-Djava.security.egd=file:/dev/./urandom", "-jar", "/app.jar"]
```

每个模块需要自行修改 **ADD** 指令

- 在模块下执行如下命令构建docker镜像

```
mvn clean package docker:build -DskipTests
```

执行效果如下:

```
[INFO] Building image auth-service
Step 1/5 : FROM openjdk:8-jdk-alpine

Pulling from library/openjdk
e7c96db7181b: Pulling fs layer
f910a506b6cb: Pulling fs layer
c2274a1a0e27: Pulling fs layer
f910a506b6cb: Downloading [=====] 238B/238B
f910a506b6cb: Verifying Checksum
f910a506b6cb: Download complete
e7c96db7181b: Downloading [>] 32.63kB/2.757MB
e7c96db7181b: Downloading [=====] 534.2kB/2.757MB
e7c96db7181b: Downloading [=====] 566.1kB/2.757MB
e7c96db7181b: Downloading [=====] 600.8kB/2.757MB
e7c96db7181b: Downloading [=====] 705.1kB/2.757MB
e7c96db7181b: Downloading [=====] 736.9kB/2.757MB
c2274a1a0e27: Downloading [>] 528.5kB/70.73MB
e7c96db7181b: Downloading [=====] 768.8kB/2.757MB
e7c96db7181b: Downloading [=====] 835.4kB/2.757MB
e7c96db7181b: Downloading [=====] 865.8kB/2.757MB
e7c96db7181b: Downloading [=====] 957kB/2.757MB
e7c96db7181b: Downloading [=====] 987.4kB/2.757MB
e7c96db7181b: Downloading [=====] 1.09MB/2.757MB
e7c96db7181b: Downloading [=====] 1.119MB/2.757MB
```

- 构建完成后登陆服务器查看docker 镜像

docker images

```
[root@bingo-172 ~]# docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
cloud-gateway       latest         8f1cca46eddf   31 seconds ago 221MB
account-service     latest         36b9c0f967e2   3 minutes ago  245MB
auth-service        latest         c2ee694ae3aa   4 hours ago    159MB
```

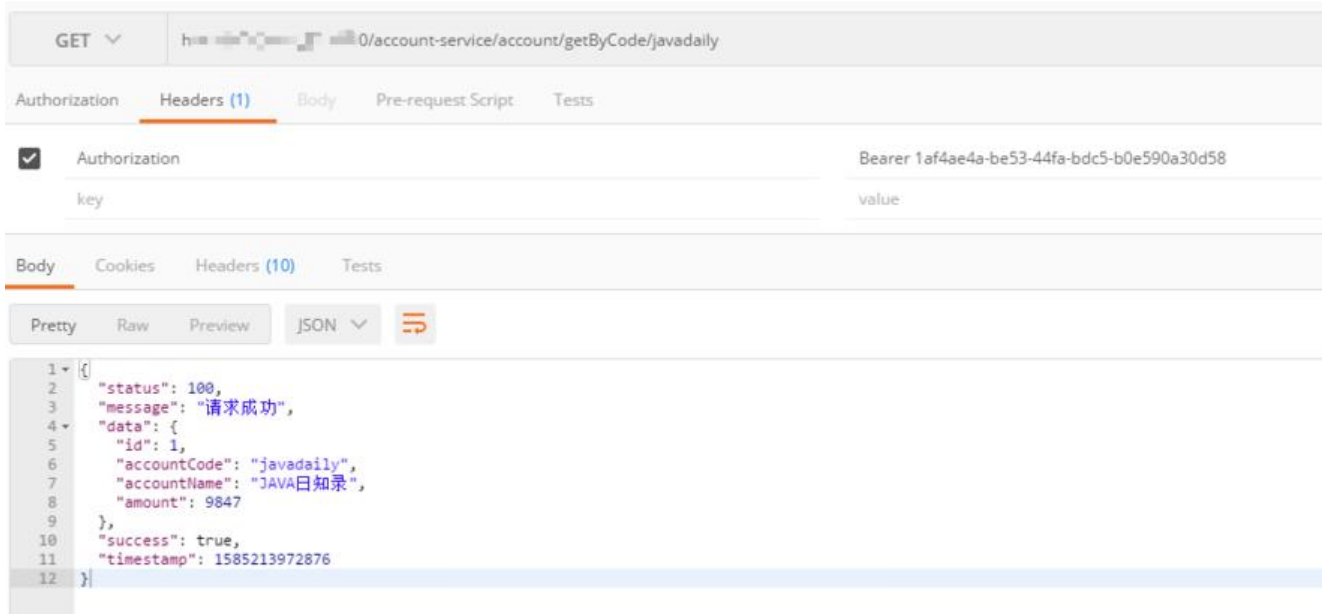
- 启动镜像, 带上 **--rm** 指令便于删除容器。

```
docker run -d -p 5000:5000 --rm auth-service
docker run -d -p 8010:8010 --rm account-service
docker run -d -p 8090:8090 --rm cloud-gateway
```

- 查看是否正常启动

```
[root@bingo-172 ~]# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS
6b3d9e795444   cloud-gateway  "java -Djava.securit..." 2 seconds ago Up 2 seconds 0.0.0.0:8090->8090/tcp
02be2efb4f64   account-service "java -Djava.securit..." 33 seconds ago Up 33 seconds 0.0.0.0:8010->8010/tcp
c1ddcc2c5b67   auth-service   "java -Djava.securit..." About a minute ago Up About a minute 0.0.0.0:5000->5000/tcp
Names:
cocky_elion
romantic_babbage
keen_noether
```

- 使用postman进行测试



服务正常响应!

使用docker-compose启动

- 建立 `cloud-service.yml` 文件, 编写服务编排镜像

```
version: "3"
services:
  auth-service:
    container_name: auth-service
    image: auth-service:latest
    ports:
      - "5000:5000"
    restart: always
```

```
cloud-gateway:
  container_name: cloud-gateway
  image: cloud-gateway:latest
  ports:
    - "8090:8090"
  restart: always
```

```
account-service:
  container_name: account-service
  image: account-service:latest
  ports:
    - "8010:8010"
  restart: always
```

- 将文件上传至服务器, 使用如下脚本启动

```
docker-compose -f cloud-service up
```

