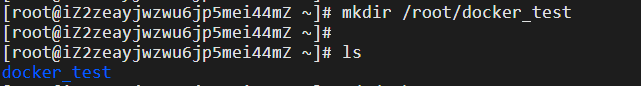
------------------------**SpringBoot 部署 docker 打包镜像**------------------------

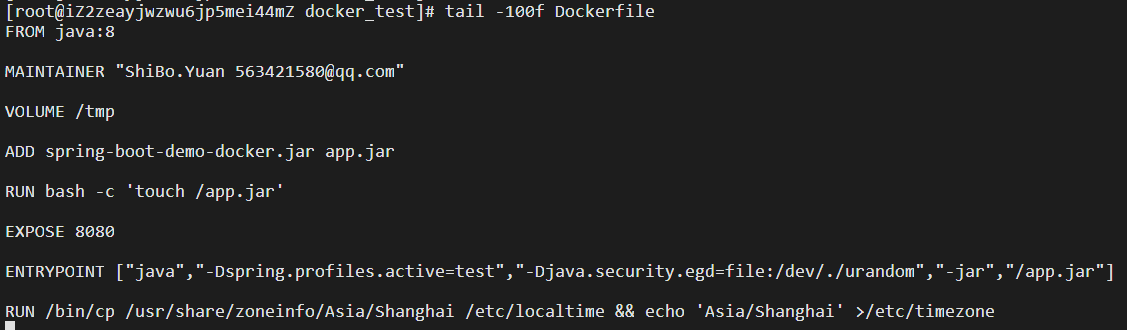
01)、创建测试文件夹



02)、将dockerfile文件和生成好的jar 使用ftp工具上传到linux服务器 随便找个文件夹 放进去 jar和dockerfile在同级目录下



03)、查阅dockerfile



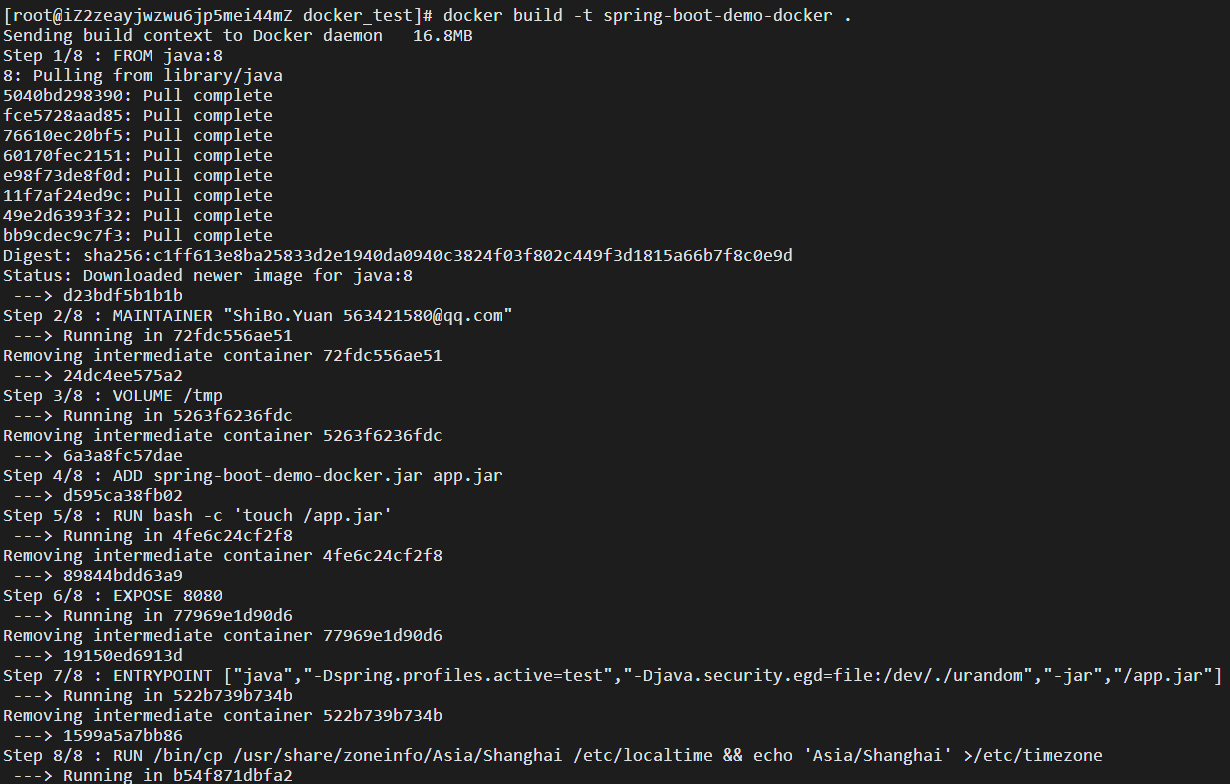
04)、将jar打包为镜像

[root@localhost ~]# docker build -t spring-boot-demo-docker .

命令参数：

-t：指定新镜像名

.：表示Dockfile在当前路径





05)、打包之后可以使用docker images命令查看镜像

[root@localhost ~]# docker images | grep spring-boot-demo-docker



06)、启动容器

[root@localhost ~]# docker run -d -p 8080:8080 --name myDemoDocker spring-boot-demo-docker:latest

命令参数：

-d：后台运行

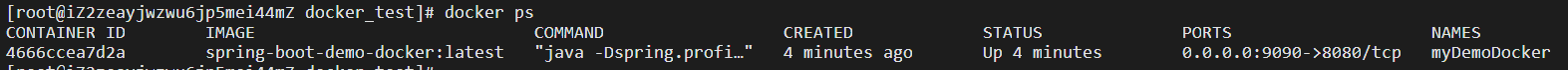
-p：公开指定端口号

--name：给容器命名



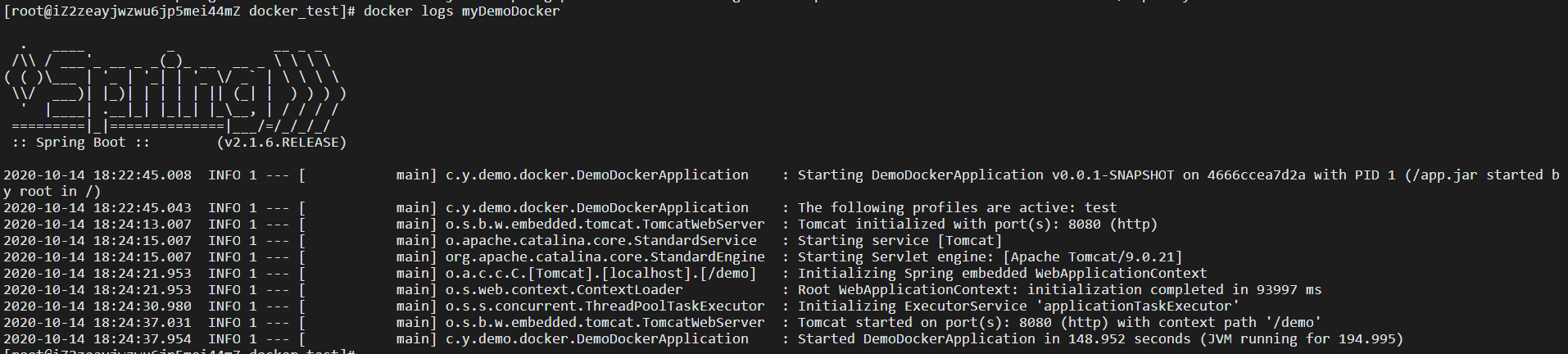
启动后可通过docker ps查看正在运行的容器：

[root@localhost ~]# docker ps



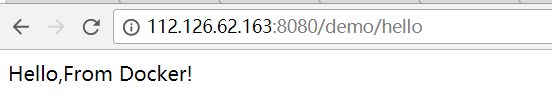
07)、查看容器启动日志：可以通过 docker logs 查看指定容器的日志

[root@localhost ~]# docker logs myDemoDocker



08)、遇到了无法访问的问题，参考后面的【docker 镜像启动成功但是无法访问】

09)、进行访问 http://112.126.62.163:8080/demo/hello

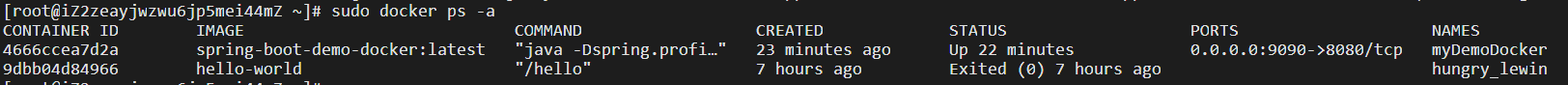


------------------------**docker容器的启动与停止**------------------------

可以去掉sudo直接使用（报错时可以尝试）。

01)、查看本地docker容器的情况：

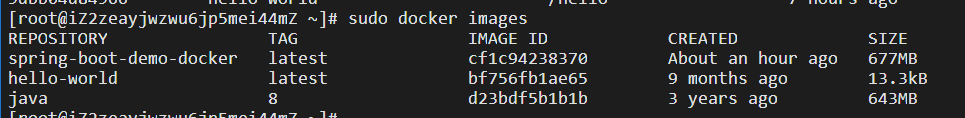
[root@localhost ~]# sudo docker ps –a



02)、使用本地的镜像启动一个新的容器

先执行: sudo docker images 查看本地的镜像；

[root@localhost ~]# sudo docker images



再根据镜像启动一个容器：sudo docker run 启动参数 镜像id/镜像名称+Tag

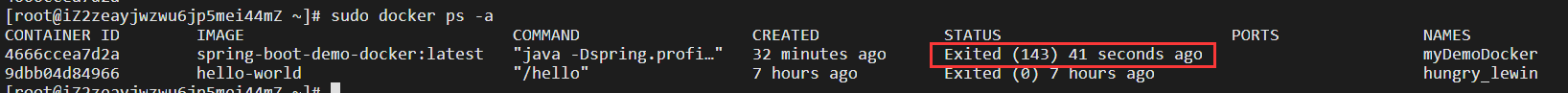
03)、退出容器

[root@localhost ~]# sudo docker stop 容器id

[root@localhost ~]# sudo docker stop 4666ccea7d2a



再查看容器：sudo docker ps -a，可以看到本地的容器情况。



删除容器：docker rm （只退出时可以不执行该命令）

[root@localhost ~]# sudo docker rm 4666ccea7d2a

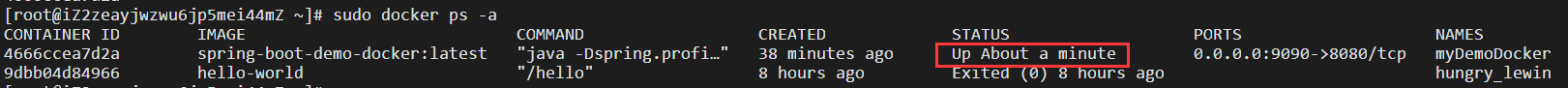
04)、启动一个停止的镜像

[root@localhost ~]# sudo docker start容器id

[root@localhost ~]# sudo docker start 4666ccea7d2a



再查看容器：sudo docker ps -a，可以看到本地的容器情况。



------------------------ **docker 镜像启动成功但是无法访问**------------------------

01)、

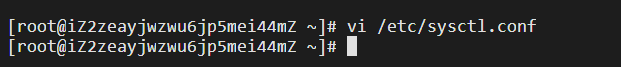
[root@localhost ~]# vi /etc/sysctl.conf

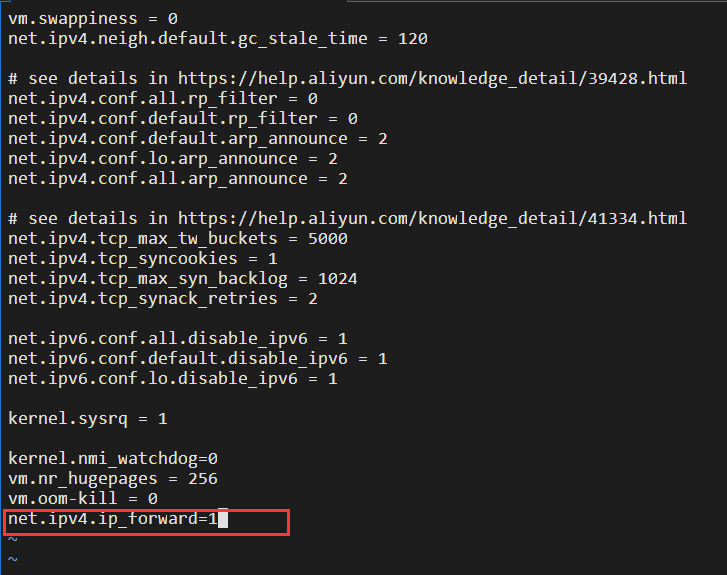
或者

[root@localhost ~]# vi /usr/lib/sysctl.d/00-system.conf

添加如下代码：

net.ipv4.ip\_forward=1





02)、重启network服务

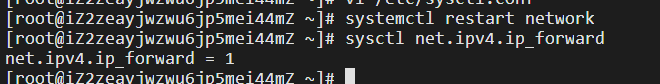
[root@localhost ~]# systemctl restart network



03)、查看是否修改成功

[root@localhost ~]# sysctl net.ipv4.ip\_forward

如果返回为“net.ipv4.ip\_forward = 1”则表示成功了



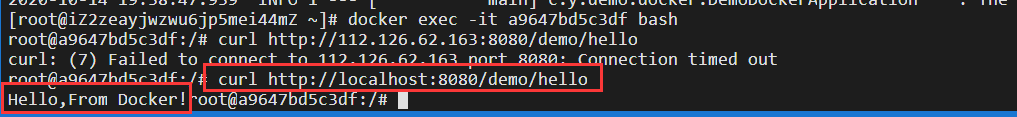
------------------------执行完还是不能访问------------------------



**因为测试原因，删除了原有的容器及修改了端口号，所以容器id做了改变。**

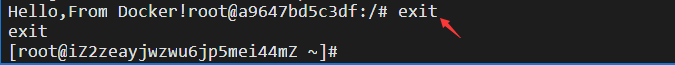
01)、进入容器，执行 docker exec -it [容器ID] bash 访问http://localhost:8080/demo/hello

[root@localhost ~]# docker exec -it a9647bd5c3df bash



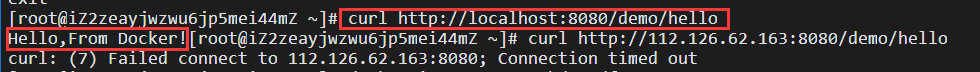
发现容器里面是可以访问成功的。

退出容器：exit



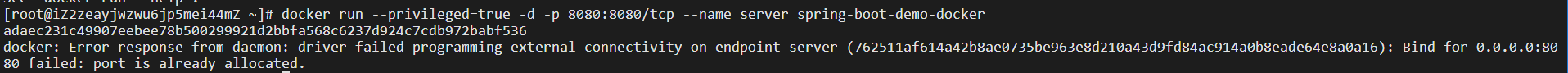
02)、在容器外访问http://localhost:8080/demo/hello

[root@localhost ~]# curl http://localhost:8080/demo/hello



03)、对8080端口开放访问权限

[root@localhost ~]# docker run --privileged=true -d -p 8080:8080/tcp --name server spring-boot-demo-docker



**使用阿里云应该设置安全组。**

**阿里云设置安全组：**

