# Docker入门

1. 什么是docker
2. docker的思想

集装箱：将我们的程序放到集装箱，然后通过集装箱整体部署项目，减少重复配置

标准化：运输方式（开发环境->码头->测试或服务器）

存储方式（不需要关心应用存储的地址，通过命令就可以启动和停止服务）

API接口(通过接口控制应用)

隔 离：与实体机环境隔离(进程、资源)

1. 解决了什么问题

我本地运行没问题呀：Docker打包时，会将应用所依赖的基础环境一起打包，这样解决了应用移动时由于环境原因导致应用启动不了

系统好卡：docker在启动应用的时候就会限定应用所需要的资源（cpu、内存等），即使我们的程序存在问题，也不会影响其他应用的正常运行

服务器撑不住了：可以快速扩展应用

3．镜像：集装箱

仓库：码头

容器：运行程序的地方

Docker运行就是去仓库把镜像拉到本地，然后用命令把镜像运行起来，变成容器

1. 安装Docker
2. yum install docker
3. docker version
4. 启动Docker服务

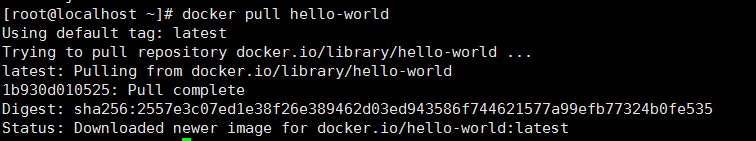
service docker start

systemctl start docker.service

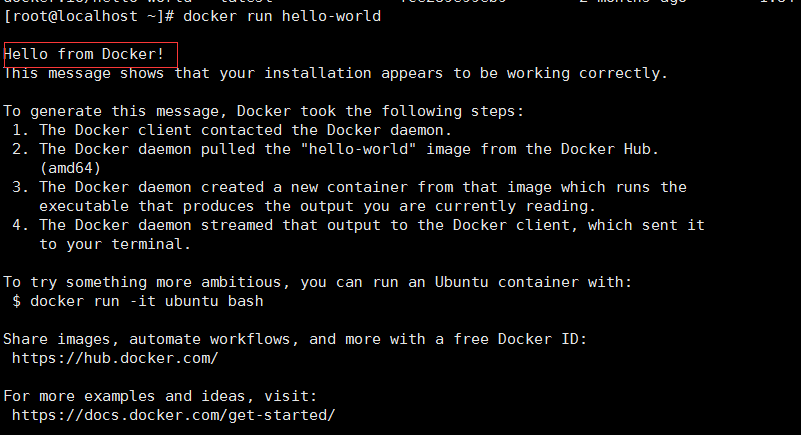
1. docker images 查看docker镜像



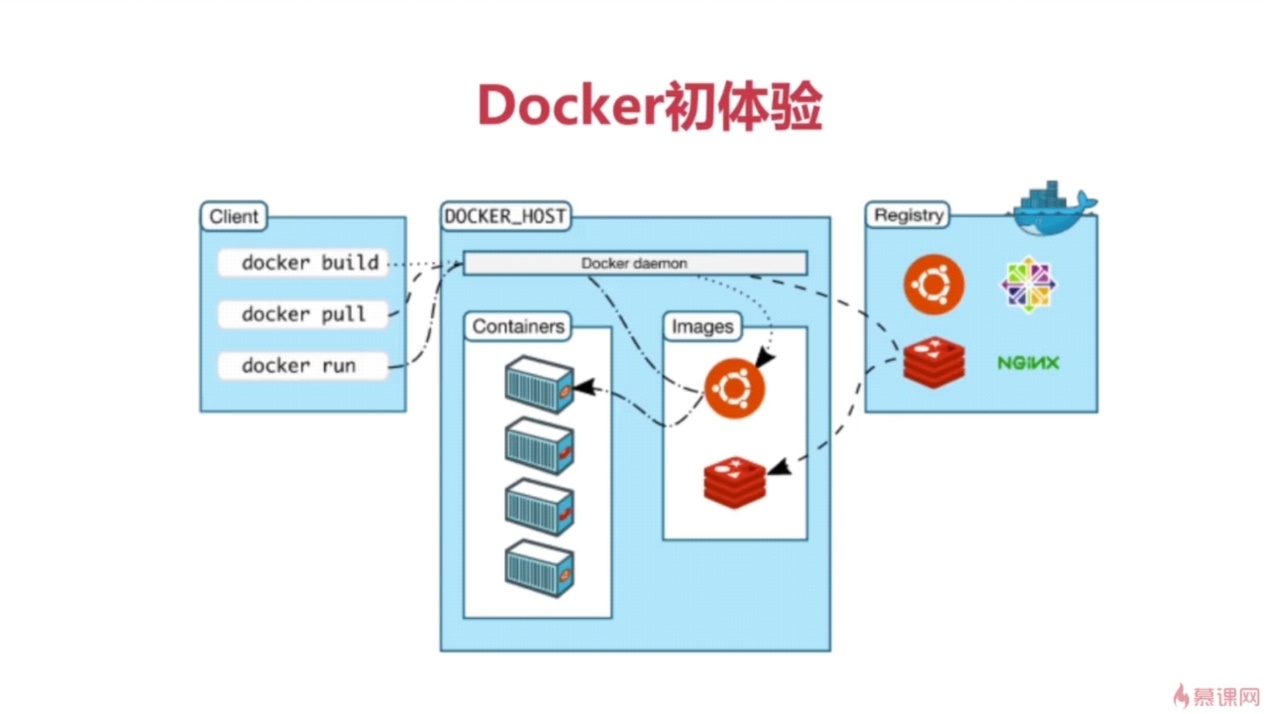
docker pull hello-world 拉去hello-world镜像（默认到hub.docker.com仓库下载）



docker run [OPTIONS] IMAGE[:TAG] [COMMAND] [ARG…] 运行docker镜像

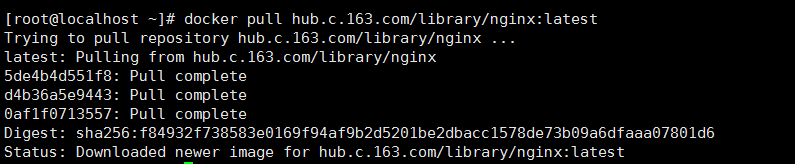


docker命令执行过程:

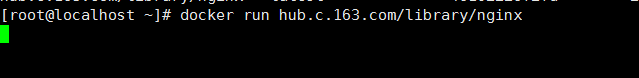


1. Docker运行Nginx
2. 到网易蜂巢找到Nginx镜像下载

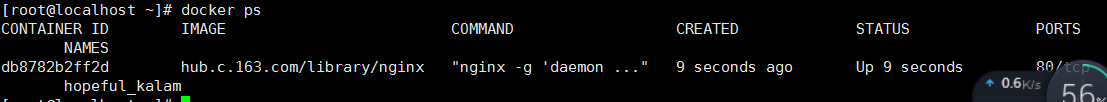
docker pull hub.c.163.com/library/nginx:latest



1. 前台启动镜像 docker run hub.c.163.com/library/nginx

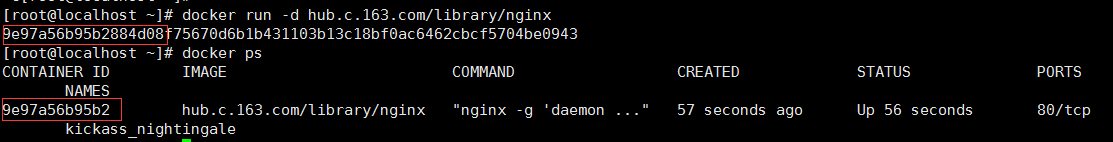


1. docker ps 查看当前运行的docker容器

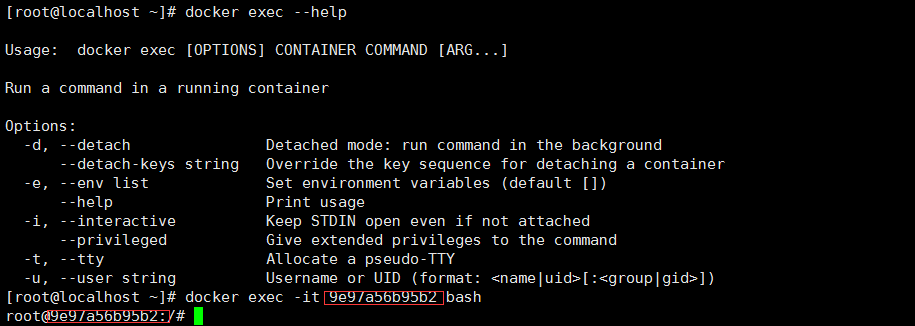


1. 后台运行 docker run -d hub.c.163.com/library/nginx

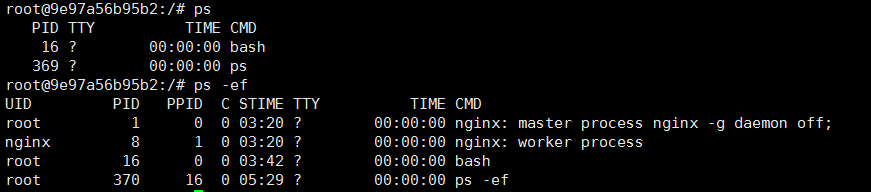




1. docker exec –it 9e97a56b95b2 bash 进入docker容器



1. 执行ps查看容器内的运行进程（apt-get update apt-get install procps）



1. exit 退出容器

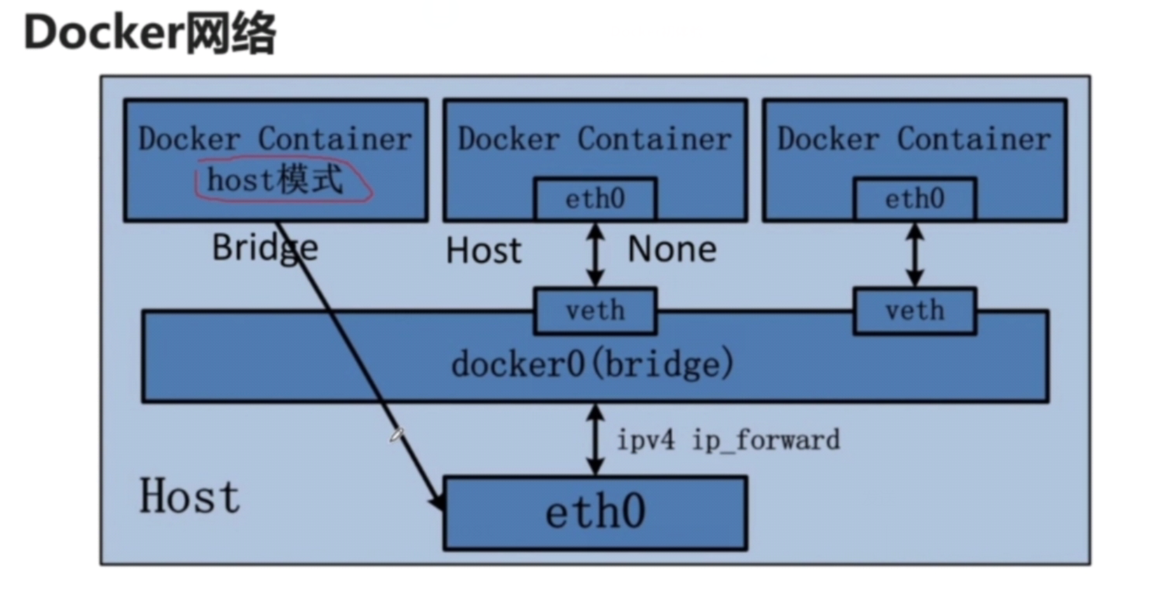


1. 停止docker容器



四．Docker的网络

网络类型：bridge(默认)、host、none



1. docker run –d –p 8080:80 hub.c.163.com/library/nginx(桥接模式 将容器80映射到主机的8080端口 小p是指定映射，大P是随机映射)



1. netstat –na|grep 8080 查看8080端口状态

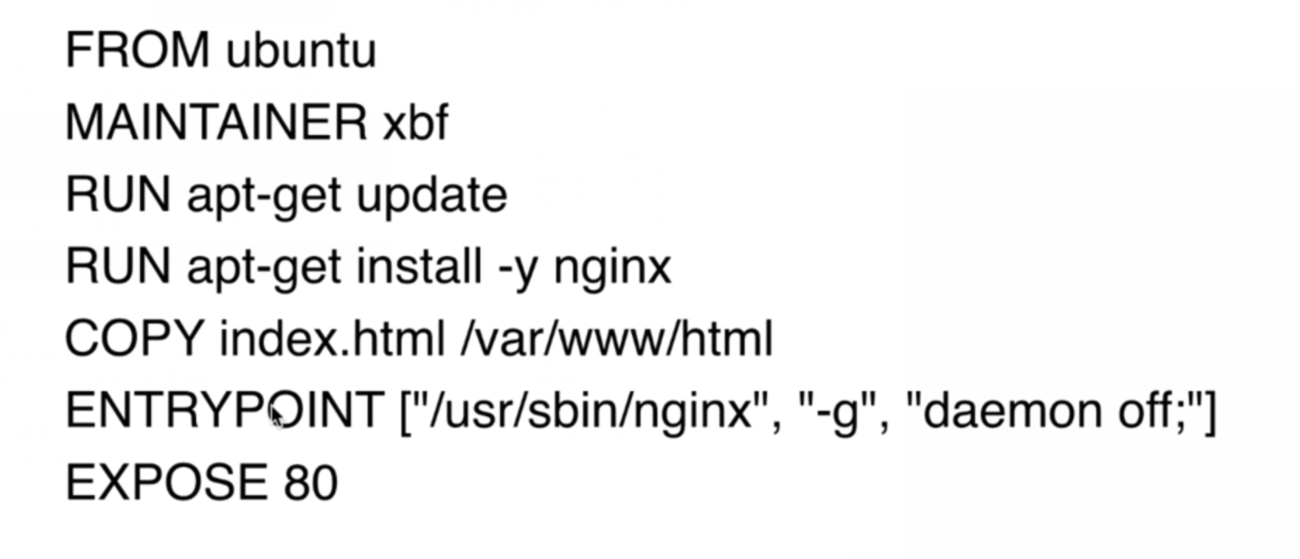


五．制作自己的镜像

Dockerfile:告诉docker怎么样去制作镜像

docker build 命令执行dockerfile构建镜像

制作自己的nginx



docker build –t gqm/nginx .

制作war包运行镜像

1. 下载tomcat镜像

docker pull hub.c.163.com/library/tomcat:latest

1. 制作Dockerfile，将war包复制到tomcat下
2. 制作镜像

docker build .

docker build –t gqm:latest .

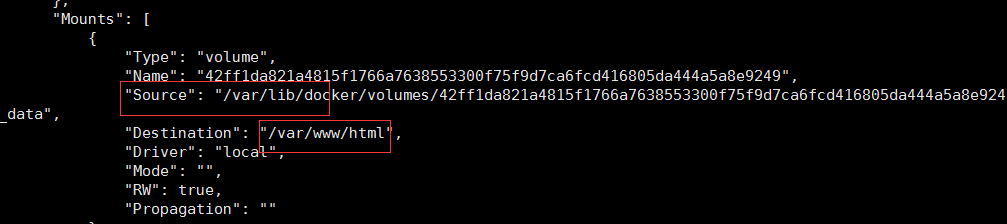
六．Volume持久化存储

如果容器在运行的时候，对容器进行操作比如在一个文件里进行写操作，当容器停止后这些操作都将消失。所以要用Volume进行持久化

1. docker run –d –v /usr/share/nginx/html nginx

将nginx网页地址/usr/share/nginx/html 挂接到数组机

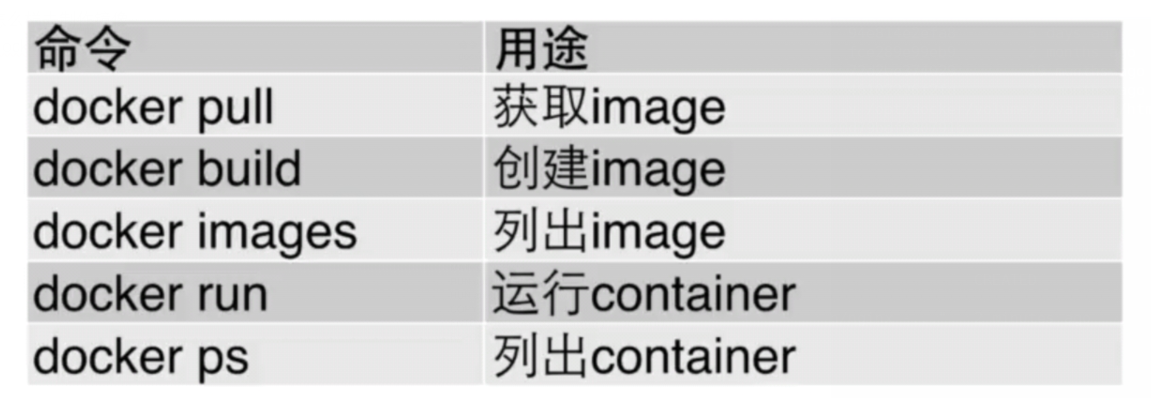
Docker inspect 86bd5e7bc9c5 查看容器信息，找到/usr/share/nginx/html在宿主机中的位置



这样就会将容器内的文件与宿主机的文件对应，就会保存容器对文件的操作，这个操作是双向的，但是每次启动都得挂接目录，且每次挂接到宿主机的目录是不一样的，所以启动容器后还是一个新的容器，以前的修改并没有影响

1. docker run –v $PWD/code:/var/www/html nginx

将本地目录映射到容器的相应目录（PWD是shell的一个命令，表示当前目录，后面可以跟/html）,即这种方式可以指定宿主机的哪个目录挂接到容相应的目录





七 .制作tomcat镜像

1. 编写Dockerfile

From docker.io/ubuntu

MAINTAINER gqm

Workdir /usr/local

run mkdir java

run mkdir tomcat-9

add jdk1.8.0\_201 /usr/local/java

add tomcat-9 /usr/local/tomcat-9

ENV JAVA\_HOME=/usr/local/java

ENV JAVA\_PATH=${JAVA\_HOME}/bin

ENV CLASSPATH=.:${JAVA\_HOME}/lib

ENV PATH=$PATH:${JAVA\_PATH}

EXPOSE 8080

ENTRYPOINT ["/usr/localtomcat-9/bin/catalina.sh","run"]

docker build -t mytomcat .

Dockerfile样板

FROM ccc7a11d65b1 （这串数字是我已经创建好一个ubuntu镜像的镜像id，在这里作为tomcat的基础镜像

MAINTAINER hmk

ENV REFRESHED\_AT 2018-03-10 （这个环境变量用来表名该镜像模板的最后更新时间）

#切换镜像目录，进入/usr目录

WORKDIR /usr

#在/usr/下创建jdk目录,用来存放jdk文件

RUN mkdir jdk

#在/usr/下创建tomcat目录，用来存放tomcat

RUN mkdir tomcat

#将宿主机的jdk目录下的文件拷至镜像的/usr/jdk目录下

ADD jdk1.8.0\_131 /usr/jdk/

#将宿主机的tomcat目录下的文件拷至镜像的/usr/tomcat目录下

ADD apache-tomcat-7.0.81 /usr/tomcat/

#设置环境变量

ENV JAVA\_HOME=/usr/jdk

ENV JRE\_HOME=$JAVA\_HOME/jre

ENV CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar:$JRE\_HOME/lib:$CLASSPATH

ENV PATH=/sbin:$JAVA\_HOME/bin:$PATH

#公开端口

EXPOSE 8080

#设置启动命令

ENTRYPOINT ["/usr/tomcat/bin/catalina.sh","run"]