docker pull 镜像 //从仓库源下载镜像

docker pull cenos:6 //从镜像源仓库下载指定版本的镜像

docker images //查看所有镜像

REPOSTITORY：表示镜像的仓库源

TAG：镜像的标签（同一个仓库源有多个TAG，代表这个仓库源的不同版本。我们使用REPOSTITORY:TAG来定义不同的镜像。） latest表示最新的版本

IMAGE ID：镜像ID

CREATED：镜像的创建时间

SIZE：镜像的大小

docker run -t -i centos:latest /bin/bash //交互式启动（如果不指定版本号默认是最新的版本）

docker search httpd //查找仓库源镜像

NAME：镜像仓库源的名称

DESCRIPTION：镜像的描述

OFFICIAL：是否是官网发布的镜像（标识是OK）

docker commit -m="qwe" -a="zxt" 08b62d3c93c5 centos:7.5 //从已经创建的容器中更新镜像，并提交镜像

-m：是描述信息

-a：镜像的作者

08b62d3c93c5：镜像的id

centos:7.5：定义了镜像名字以及版本号

docker tag centos:7.5 xxx:1.1 //更改镜像的名字以及版本号

docker rmi 1eb0ae1150a7 -f //删除镜像 -f：强制

如果不用-f的话必须先用rm删除依赖于这个镜像的所有容器

docker rm 容器 //删除容器

--rm：当进入容器后，退出时将该容器删除

docker run -it --rm centos bash //当使用docker run创建容器时使用 --rm进入到交互式界面后，退出后自动删除容器。

docker rmi httpd: latest //删除镜像 也可以直接用镜像名删除或镜像名：版本号的方式删除

docker run --name mynginx -d nginx //使用镜像创建容器

--name：指定容器名

-d：后台运行

docker ps //查看当前正在运行的容器

docker run --name younginx -d -p 8000:80 nginx //使用镜像创建容器并将容器的80端口映射到宿主机8000端口上

-P：指定端口映射

-P（大）：随机端口

docker logs younginx //查看容器内部的标准输出（id或者是名字都可以）

-f：可以实时查看容器内部的标准输出

docker run --name 1nginx -d -p 9000:80 -v /var/html/www:/var/html/www（:ro） nginx

//创建容器映射端口号并将容器的var目录映射到宿主机的目录

-v：指定容器的文件夹映射

:ro //表示只读不加时表示可读可写

docker run -d -p 192.168.135.134::80 nginx //两个冒号代表把容器80端口，映射到宿主机的随机端口

docker port nginx1 //查看镜像映射的本地端口

docker exec -it ebc44a6376e8 bash //进入容器

docker stop 1nginx //暂停容器

docker stop registry -t 20 //20秒后停止该容器

docker ps -a //查看所有容器（包含停止的容器）

docker start 1nginx //启动容器

docker rm 1nginx //删除容器（前提容器的状态是停止的）

-f：可以强制删除容器（包括正在运行的容器）

docker save -o /root/mysql56.tar ebc44a6376e8 //导出镜像到指定目录

docker export 6ceaa5990ca8 > /root/mysql-cont.tar //导出镜像到指定目录

docker export -o /root/mysql2-cont.tar 6ceaa5990ca8 //导出镜像到指定目录

docker import /root/mysql-cont.tar mysql1232 //导入指定镜像

docker load -i /root/mysql56.tar //导入指定镜像

docker history ebc44a6376e8 //显示镜像有几个层级

docker inspect ebc44a6376e8 //显示镜像详细信息

docker stats //查看docker系统占用资源

docker pause 6ceaa5990ca8 //暂停容器的运行、

docker diff 6ceaa5990ca8 //查看容器那些文件发生了变更

docker cp /etc/fstab 6 :/tmp // 将/etc/fstab 导入到6号容器的/tmp下

docker port 6 //查看6号容器暴露出来的端口

docker rename 6ceaa5990ca8 mysql57 //更改容器名

which ip //查找命令所在的目录下

rpm -qf /usr/sbin/ip //查看命令所依赖的软件包

docker的四种网络模式：host、container、none、bridge

安装docker创建了三种网络，用docker network ls 命令列出这些网络。

1. host模式，使用docker run时，用--net=host指定，这时docker使用的网路实际上和宿主机一样，在容器内看到的网卡ip就是宿主机上的ip。

docker run -it --net=host centos bash

1. container模式：使用 --net=container：已有的容器id / 名字 //多个容器使用一个网络，ip地址都是一样的。

docker run -it --name centos1 centos bash

docker run -it --net=container:centos1 --name centos2 centos bash

1. none模式：使用 --net=none 指定，这种模式下不会配置任何网路。

docker run -it --net=none --rm centos bash

1. bridge模式：使用 --net=bridge指定，默认模式，不用指定，这种模式会为每一个容器分配一个独立的Network Namespace。类似于vmware的net模式，同一个宿主机上的所有容器都会在同一个网段下，相互之间可以通信。

dockerfile 定制镜像

mkdir zxxx

cd zxxx/

vim dockerfile

FROM httpd //FROM指定基础镜像为httpd

RUN echo 'hello word' > /var/www/html/index.html //RUN指定执行命令行命令

docker build -t myhttpd:2 .

-t：指定镜像名：版本号

FROM //镜像名

MAINTAINER Asher woo //作者名

LABEL description=”A basic Apache container on RHEL7.3”

ADD training.repo /etc/yum.repo.d/training.repo

RUN yum update -y && yum install httpd -y && yue clean all

EXPOSE 80

CMD [“httpd”,”-D”,”FOREGROUND”]

创建私有仓库

mkdir /registry

vim /etc/docker/daemon.json

{

"insecure-registries":["192.168.135.134:5000"]

}

systemctl restart docker

docker run -d -p 5000:5000 --name registry1 --restart=always --privileged=true -v /registry:/var/lib/registry registry

docker tag docker.io/centos 192.168.135.134:5000/centos

docker push 192.168.135.134:5000/centos

curl -XGET <http://192.168.135.134:5000/v2/_catalog> //查看私有镜像

ADD\_REGISTRY='--add-registry C:\Users\17316\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.png192.168.150.150:5000'  
BLOCK\_REGISTRY='--block-registry C:\Users\17316\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pngdocker.io'  
INSECURE\_REGISTRY='--insecure-registry C:\Users\17316\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.png192.168.150.150:5000'

docker-machine

<https://github.com/docker/machine/releases>

curl -L https://github.com/docker/machine/releases/download/v0.16.1/docker-machine-`uname -s`-`uname -m` > /usr/local/bin/docker-machine

chmod +x docker-machine

docker-machine -v //查看machine版本号