# 快速入门

目录

[快速入门 1](#_Toc410068601)

[目录 1](#_Toc410068602)

[搜索镜像 1](#_Toc410068603)

[下载相应镜像 2](#_Toc410068604)

[列出本地所有有镜像 2](#_Toc410068605)

[启动一个容器 2](#_Toc410068606)

[列出当前运行的容器 3](#_Toc410068607)

[停止、重启、杀死、删除容器 3](#_Toc410068608)

[删除所有未运行的容器 3](#_Toc410068609)

[查看容器运行日志 4](#_Toc410068610)

[检查容器配置 4](#_Toc410068611)

[提交一个修改的镜像 4](#_Toc410068612)

[根据Dockerfile创建镜像 5](#_Toc410068613)

[导出和导入镜像 6](#_Toc410068614)

[给镜像另外取一个标记 6](#_Toc410068615)

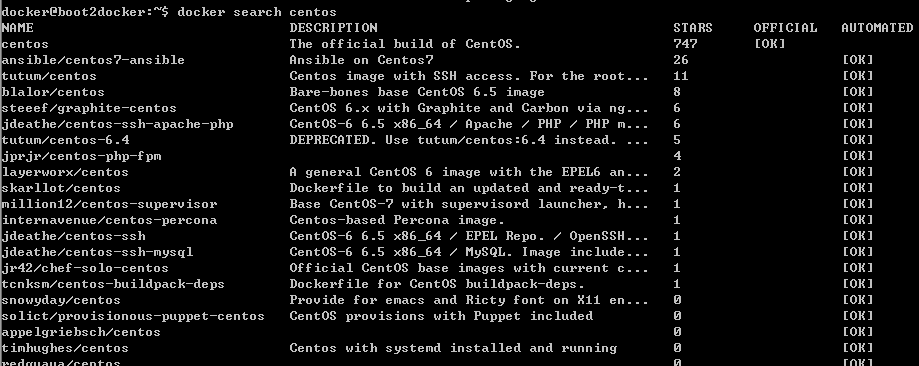
[给容器命名 7](#_Toc410068616)

[导出和导入容器快照 7](#_Toc410068617)

[挂载本地目录 8](#_Toc410068618)

## 搜索镜像

docker search 镜像仓库



## 下载相应镜像

Docker pull centos:latest

## 列出本地所有有镜像

Sudo docker images



来自于哪个仓库，比如 centos

镜像的标记，比如 V1

它的 ID 号（唯一）

创建时间

镜像大小

## 启动一个容器

Sudo docker run –it



-d Detached 或者 daemon mode，后台运行。

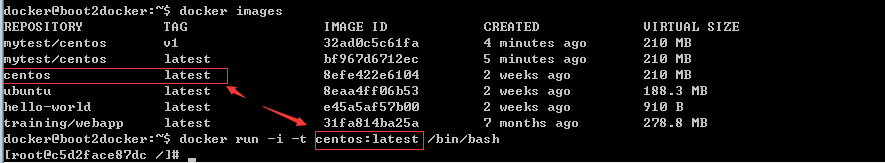
-i -t 开一个 tty 终端，保持交互模式，这两个一般共同使用。

-e 设置环境变量参数，参考 Install GitLab With Docker 。

-p [host\_port]:[container\_port] 映射 HOST 端口到容器，方便外部访问容器内服务，host\_port 可以省略，省略表示把 container\_port 映射到一个动态端口。

-v [host-path]:[container-path] 把 HOST 文件夹挂载到 Container 用以保存数据。

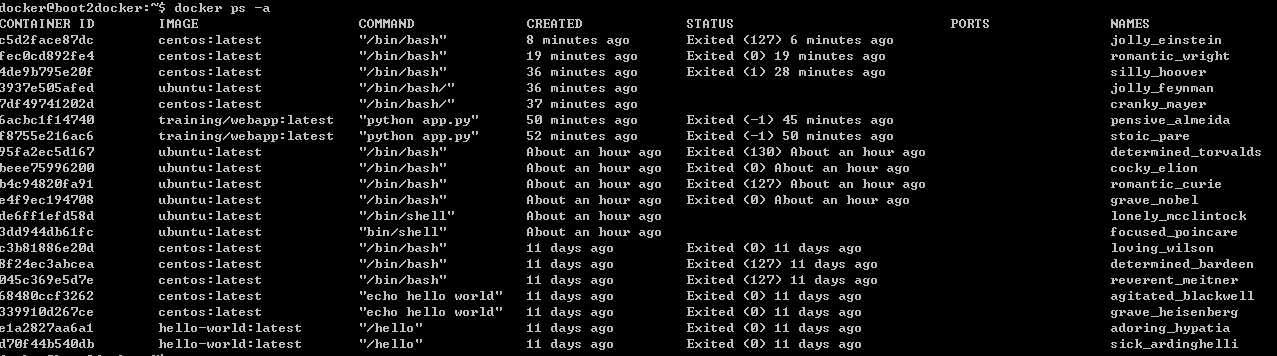
--rm 自动删除已运行存在的相同 IMAGE 的容器。



## 列出当前运行的容器

docker ps -a –q

列出当前运行的容器， -a 会列出所有，包括已停止的， -q 只列出容器 ID



## 停止、重启、杀死、删除容器

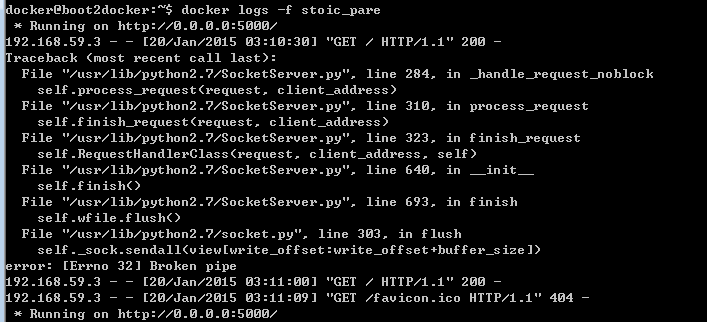
docker stop/start/restart/kill/rm CONTAINER

## 删除所有未运行的容器

docker ps -a -q | xargs docker rm

## 查看容器运行日志

docker logs -f CONTAINER



## 检查容器配置

docker inspect CONTAINER | grep IPAddress

## 提交一个修改的镜像

Docker commit【选项】容器名称[仓库【：标记】]

创建一个容器。在里面安装一些软件。如json。然后提交为一个新的镜像









提交为一个新的镜像



-a=“作者名称”

-m=“说明”



## 根据Dockerfile创建镜像

使用 # 来注释

FROM 指令告诉 Docker 使用哪个镜像作为基础

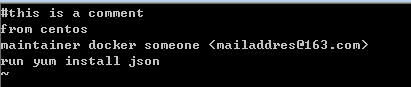
接着是维护者的信息

RUN 开头的指令会在创建中运行

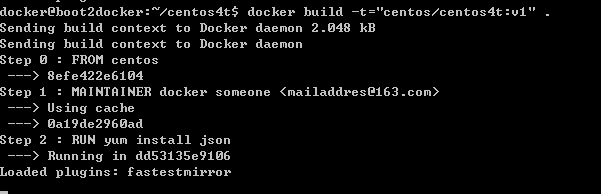
Mkdir centos4t

Cd centos 4t

Vi Dockerfile



现在版本最大层为127



-t 标记来添加 tag，指定新的镜像的信息。

“.” 是 Dockerfile 所在的路径（当前目录），也可以

替换为一个具体的 Dockerfile 的路径。

安装完成后查看镜像列表

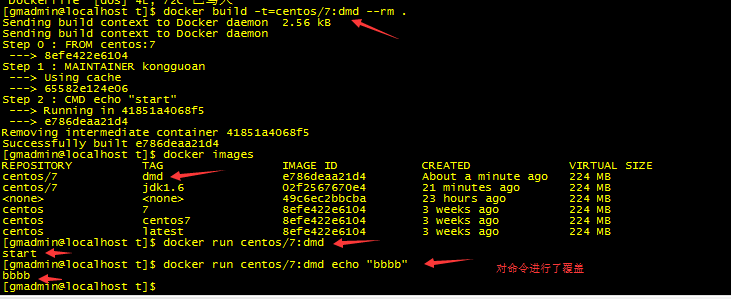


关于CMD和ENTRYPOINT区别

CMD和ENTRYPOINT都是类似与windows下的开机启动。但是cmd是可被覆盖的。比如

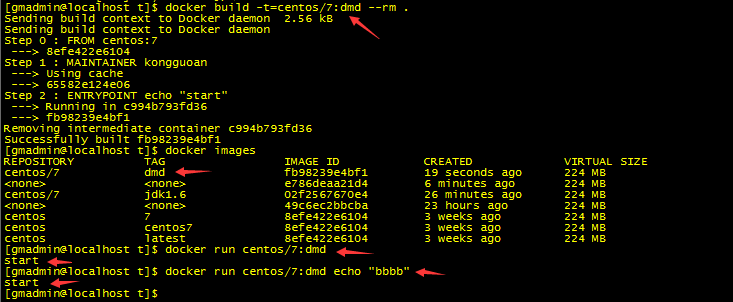
CMD echo “start”





**ENTRYPOINT echo "start"**





**完整例子**

**# 这个就是注释**

**#基本镜像**

**FROM centos:7**

**#维护者名称**

**MAINTAINER kongguoan**

**#安装json**

**run yum install -q -y json**

**#创建一个目录（run 来）**

**run mkdir dockerDir**

**#环境变量添加 DOCKEVERSION**

**EVN DOCKEVERSION 2015012801**

**#暴露出来的端口**

**EXPOSE 80**

**#本地目录拷贝（只能add dockerfile 所在目录下的）**

**add usr/local/jdk1.6.0\_31/ user/local/java/**

**#启动时候执行**

**ENTRYPOINT echo "ok ENTRYPOINT"**

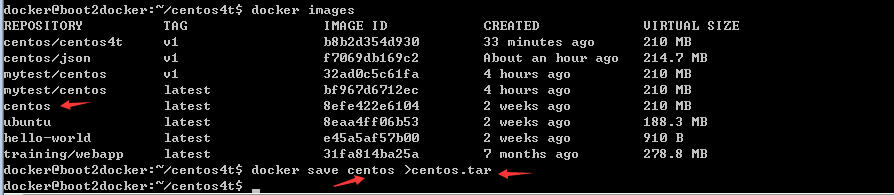
**CMD echo "ok cmd"**

## 导出和导入镜像

导出

找到要导出的镜像

Docker images



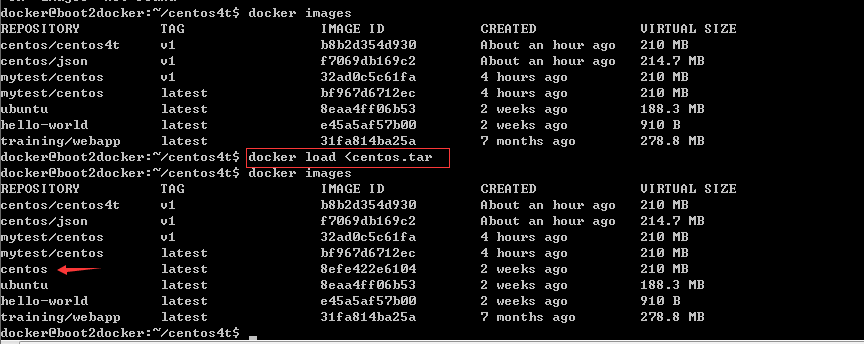
查看导出的镜像



导入

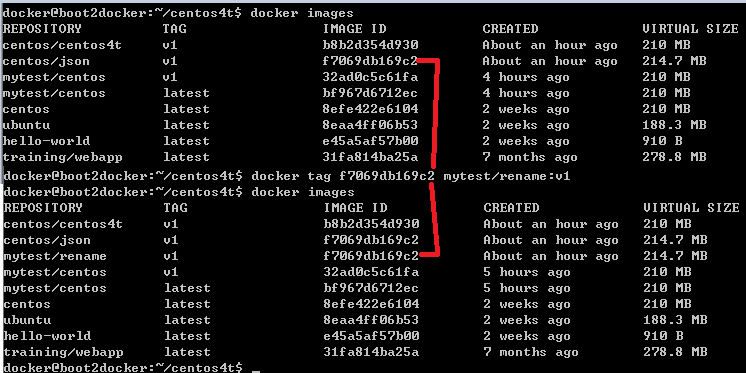
为了测试先删除centos 镜像然后在导入

Docker load<centos.tar



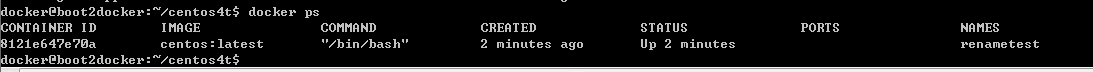
## 给镜像另外取一个标记

Doker tag f7069db169c2 mytest/rename:v1



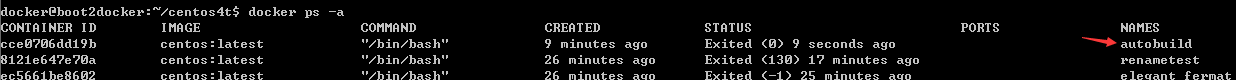
## 给容器命名

Docker run –d –i –t --name renametest centos /bin/bash



## 导出和导入容器快照

找出要导出的容器

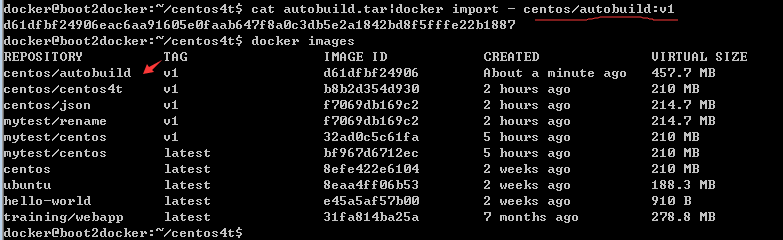


导出 Docker export autobuild>autobuild.tar



导入

cat autobuild.tar | docker import - centos/autobuild:v1.0



此外，也可以通过指定 URL 或者某个目录来导入，例如

docker import <http://docker.widuu.com/ubuntu.tar>

## 挂载本地目录

docker run -it -v /usr/local/jdk1.6.0\_31:/usr/local/jdk1.6.0\_31 centos:7 /bin/bash



通过-v参数，冒号前为宿主机目录，必须为绝对路径，冒号后为镜像内挂载的路径。