# Docker

## 一、Docker 概述

### (一)、Docker 为什么出现？

1、一款产品： 开发–上线 两套环境！应用环境，应用配置！

2、开发 — 运维。 问题：我在我的电脑上可以允许！版本更新，导致服务不可用！对于运维来说考验十分大?

3、环境配置是十分的麻烦，每一个机器都要部署环境(集群 Redis、ES、Hadoop...) !费事费力。

4、发布一个项目( jar + (Redis MySQL JDK ES) )，项目能不能带上环境安装打包！

5、之前在服务器配置一个应用的环境 Redis、MySQL、JDK、ES、Hadoop 配置超麻烦了，不能够跨平台。开发环境 Windows，最后发布到 Linux！

6、传统：开发 jar，运维来做！

6、现在：开发打包部署上线，一套流程做完！

7、安卓流程：java — apk —发布（应用商店）—- 张三使用 apk 一安装即可用！

8、docker 流程： java-jar（环境） — 打包项目带上环境（镜像） —（ Docker 仓库：商店）———-下载我们发布的镜像 —- 直接运行即可！

8、Docker 给以上的问题，提出了解决方案！

9、Docker 的思想就来自于集装箱！

10、JRE – 多个应用(端口冲突) – 原来都是交叉的！

11、隔离：Docker 核心思想！打包装箱！每个箱子是互相隔离的。

12、Docker 通过隔离机制，可以将服务器利用到极致！

13、本质：所有的技术都是因为出现了一些问题，我们需要去解决，才去学习！

## (二)、Dcoker 的历史

1、2010 年，几个搞 IT 的年轻人，就在美国成立了一家公司 dotcloud

2、做一些 pass 的云计算服务！LXC（Linux Container 容器）有关的容器技术！

3、Linux Container 容器是一种内核虚拟化技术，可以提供轻量级的虚拟化，以便隔离进程和资源。

4、他们将自己的技术（容器化技术）命名就是 Docker。

4、Docker 刚刚延生的时候，没有引起行业的注意！dotCloud，就活不下去！

5、2013 年，Docker 开源！

6、越来越多的人发现 docker 的优点！火了。Docker 每个月都会更新一个版本！

7、2014 年 4 月 9 日，Docker1.0 发布！

**(三)、**docker 为什么这么火?

1、在容器技术出来之前，我们都是使用虚拟机技术！

2、虚拟机：在 window 中装一个 VMware，通过这个软件我们可以虚拟出来一台或者多台电脑！笨重！

3、虚拟机也属于虚拟化技术，Docker 容器技术，也是一种虚拟化技术！

4、vm : linux centos 原生镜像（一个电脑！） 隔离、需要开启多个虚拟机！ 几个 G 几分钟

5、docker: 隔离，镜像（最核心的环境 4m + jdk + mysql）十分的小巧，运行镜像就可以了！小巧！

几个 M 秒级启动！

6、Docker 基于 Go 语言开发的！开源项目！

7、docker 官网：https://www.docker.com/

8、文档：https://docs.docker.com/ Docker 的文档是超级详细的！

9、仓库：https://hub.docker.com/ 相当于 github

**(四)、Docker 能做什么？**

比较 Docker 和虚拟机技术的不同：

1、传统虚拟机,虚拟出一条硬件，运行一个完整的操作系统，然后在这个系统上安装和运行软件

容器内的应用直接运行在宿主机的内容，容器是没有自己的内核的，也没有虚拟我们的硬件，所以

2、就轻便了,每个容器间是互相隔离，每个容器内都有一个属于自己的文件系统，互不影响

**(五)、Docker 的基本组成**

- 镜像（image)：

  docker 镜像就好比是一个目标，可以通过这个目标来创建容器服务，tomcat 镜像===>run===>容器（提供服务器），通过这个镜像可以创建多个容器（最终服务运行或者项目运行就是在容器中的）。

- 容器(container)：

  Docker 利用容器技术，独立运行一个或者一组应用，通过镜像来创建的.

  启动，停止，删除，基本命令

  目前就可以把这个容器理解为就是一个简易的 Linux 系统。

- 仓库(repository)：

  仓库就是存放镜像的地方！

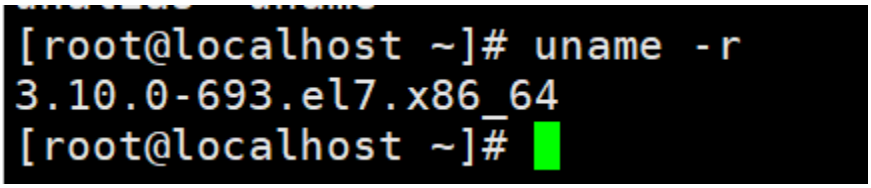
仓库分为公有仓库和私有仓库。(很类似 git)

Docker Hub 是国外的。

阿里云...都有容器服务器(配置镜像加速!)

## 二、安装

### (一)、Linux 的内核 3.10 以上




### (二)、清除杂的环境

```
yum remove docker \
    docker-client \
    docker-client-latest \
    docker-common \
    docker-latest \
    docker-latest-logrotate \
    docker-logrotate \
    docker-engine
```

```
[root@localhost ~]# yum remove docker \
>     docker-client \
>     docker-client-latest \
>     docker-common \
>     docker-latest \
>     docker-latest-logrotate \
>     docker-logrotate \
>     docker-engine
Loaded plugins: fastestmirror, langpacks
No Match for argument: docker
No Match for argument: docker-client
No Match for argument: docker-client-latest
No Match for argument: docker-common
No Match for argument: docker-latest
No Match for argument: docker-latest-logrotate
No Match for argument: docker-logrotate
No Match for argument: docker-engine
No Packages marked for removal
[root@localhost ~]#
```

## (三)、虚拟机联网安装应用

yum –y install yum-utils

```
[root@localhost ~]# yum install -y yum-utils
Loaded plugins: fastestmirror, langpacks
```

```
Complete!
[root@localhost ~]#
```

## (四)、设置镜像仓库

```
yum-config-manager \
    --add-repo \
```

https://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

```
[root@localhost ~]# yum-config-manager \
>     --add-repo \
>     https://mirrors.aliyun.com/docker-ce/linux/cento
Loaded plugins: fastestmirror, langpacks
adding repo from: https://mirrors.aliyun.com/docker-ce
grabbing file https://mirrors.aliyun.com/docker-ce/lin
repo saved to /etc/yum.repos.d/docker-ce.repo
[root@localhost ~]#
```

**(五)、更新 yum 包索引**

yum makecache fast

```
[root@localhost ~]# yum makecache fast
Loaded plugins: fastestmirror, langpacks
base
docker-ce-stable
extras
updates
(1/2): docker-ce-stable/7/x86_64/primary_db
(2/2): docker-ce-stable/7/x86_64/updateinfo
Loading mirror speeds from cached hostfile
 * base: mirrors.bupt.edu.cn
 * extras: mirrors.bupt.edu.cn
 * updates: mirrors.bupt.edu.cn
Metadata Cache Created
[root@localhost ~]#
```

**(六)、安装 doker-ce(社区办) dockerEE 是企业版**

yum -y install docker-ce docker-ce-cli containerd.io

```
[root@localhost ~]# yum install docker-ce docker-ce-
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.bupt.edu.cn
 * extras: mirrors.bupt.edu.cn
 * updates: mirrors.bupt.edu.cn
```

```
Dependency Updated:
  libselinux.x86_64 0:2.5-15.el7
  libselinux-utils.x86_64 0:2.5-15.el7
  libsemanage-python.x86_64 0:2.5-14.el7
  policycoreutils.x86_64 0:2.5-34.el7
  selinux-policy.noarch 0:3.13.1-268.el7_9.2
  setools-libs.x86_64 0:3.3.8-4.el7

Complete!
[root@localhost ~]#
```

**(七)、启动 docker**

systemctl start docker

```
[root@localhost ~]# systemctl start docker
[root@localhost ~]#
```

(八)、查看 docker 版本

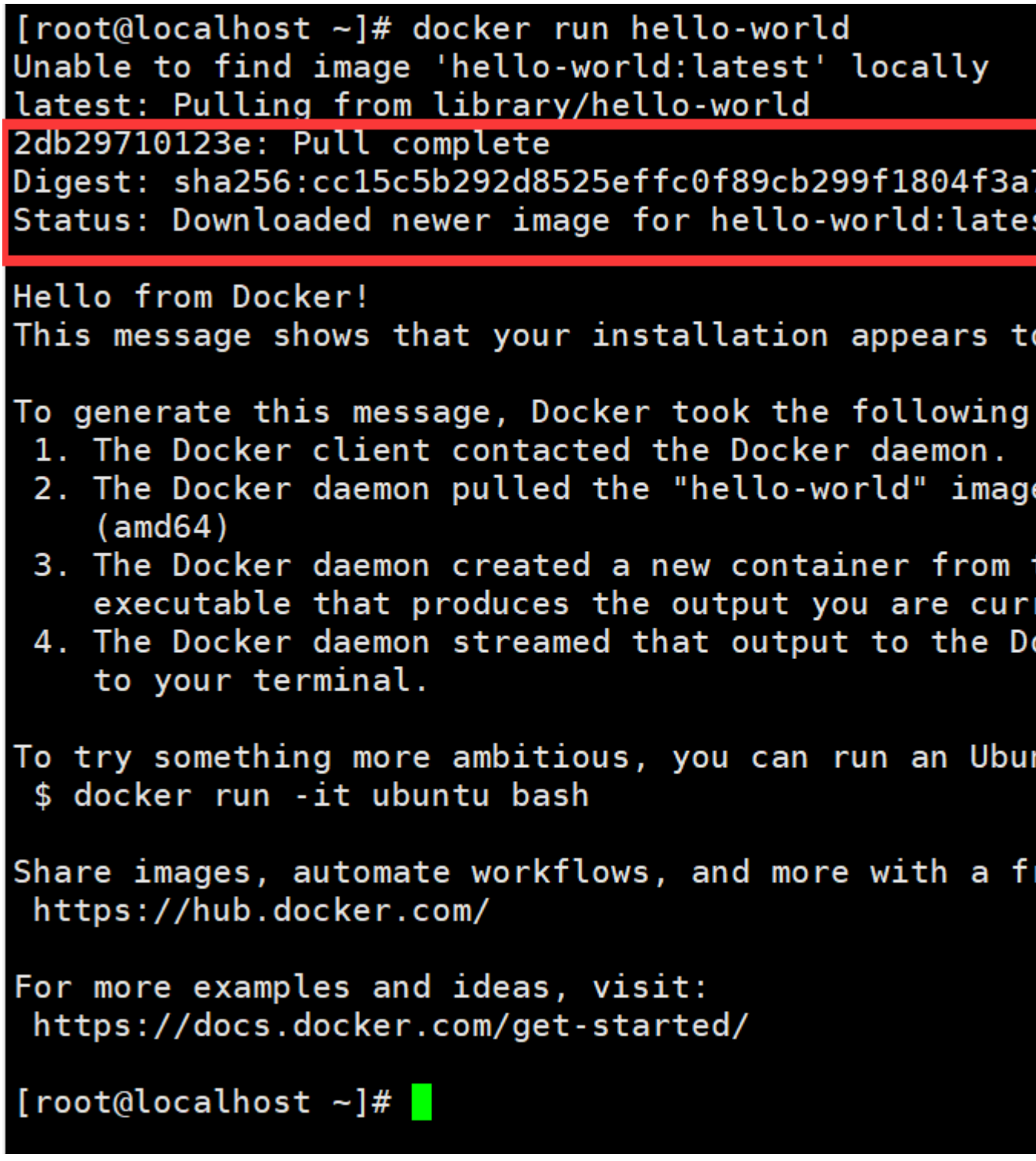
docker verson

```
[root@localhost ~]# docker version
Client: Docker Engine - Community
 Version:           20.10.11
 API version:       1.41
 Go version:        go1.16.9
 Git commit:        dea9396
 Built:             Thu Nov 18 00:38:53 2021
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true

Server: Docker Engine - Community
 Engine:
  Version:          20.10.11
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.16.9
  Git commit:       847da18
  Built:            Thu Nov 18 00:37:17 2021
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.4.12
  GitCommit:        7b11cfaabd73bb80907dd23182b9347b42
 runc:
  Version:          1.0.2
  GitCommit:        v1.0.2-0-g52b36a2
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
[root@localhost ~]#
```

**(八)、测试**

docker run hello-world




**(九)、查看镜像**

docker images

```
[root@localhost ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED
hello-world   latest    feb5d9fea6a5   2 months ago
[root@localhost ~]#
```

## 二、卸载 Docker

```
yum remove docker-ce docker-ce-cli containerd.io

rm -rf /var/lib/docker

rm -rf /var/lib/containerd
```

## 三、常用命令

### (一)、帮助命令

docker verson　　#显示 docker 的版本信息
docker info　　#显示 docker 的系统信息,包括镜像和容器数量
docker 命令 –help　#帮助命令

### (二)、镜像命令

#### 1、查看本机所有镜像

docker images

```
[root@localhost ~]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    feb5d9fea6a5   2 months ago   13.3kB
[root@localhost ~]#
```

**1.1、解释:**

REPOSITORY 镜像的仓库源
TAG 镜像的标签
image ID 镜像的 ID
CREATED 镜像的创建时间
SIZE 镜像的大小

**1.2、可选项**

-q, --all　　#列出所有的镜像
-q,--quiet　　# 只显示镜像的 id

**2、搜索镜像**

docker search 搜索的镜像名

```
[root@localhost ~]# docker search mysql
NAME                              DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
mysql                             MySQL is a widely used, open-source relation…   11741     [OK]
mariadb                           MariaDB Server is a high performing open sou…   4476      [OK]
mysql/mysql-server                Optimized MySQL Server Docker images. Create…   875                  [OK]
percona                           Percona Server is a fork of the MySQL relati…   564       [OK]
phpmyadmin                        phpMyAdmin - A web interface for MySQL and M…   383       [OK]
centos/mysql-57-centos7           MySQL 5.7 SQL database server                   92
mysql/mysql-cluster               Experimental MySQL Cluster Docker images. Cr…   89
centurylink/mysql                 Image containing mysql. Optimized to be link…   59                   [OK]
```

**2.1、可选项**

--filter=选项名=什么
例:--filter=stars=3000　　# 搜索出来镜像大于 3000 的

```
[root@localhost ~]# docker search mysql --filter=STARS=3000
NAME      DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
mysql     MySQL is a widely used, open-source relation…   11741     [OK]
mariadb   MariaDB Server is a high performing open sou…   4476      [OK]
[root@localhost ~]#
```

## 3、下载镜像

### 3.1、最新版下载

docker pull 镜像名(默认最新版)

```
[root@localhost ~]# docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
a10c77af2613: Pull complete
b76a7eb51ffd: Pull complete
258223f927e4: Pull complete
2d2c75386df9: Pull complete
63e92e4046c9: Pull complete
f5845c731544: Pull complete
bd0401123a9b: Pull complete
3ef07ec35f1a: Pull complete
c93a31315089: Pull complete
3349ed800d44: Pull complete
6d01857ca4c1: Pull complete
4cc13890eda8: Pull complete
Digest: sha256:aeecae58035f3868bf4f00e5fc623630d8b438db9d05f4d8c6538deb14d4c31b
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
[root@localhost ~]#
```

### 3.2、指定版本下载

docker pull mysql:5.7

```
[root@localhost ~]# docker pull mysql:5.7
5.7: Pulling from library/mysql
a10c77af2613: Already exists
b76a7eb51ffd: Already exists
258223f927e4: Already exists
2d2c75386df9: Already exists
63e92e4046c9: Already exists
f5845c731544: Already exists
bd0401123a9b: Already exists
2724b2da64fd: Pull complete
d10a7e9e325c: Pull complete
1c5fd9c3683d: Pull complete
2e35f83a12e9: Pull complete
Digest: sha256:7a3a7b7a29e6fbff433c339fc52245435fa2c308586481f2f92ab1df239d6a29
Status: Downloaded newer image for mysql:5.7
docker.io/library/mysql:5.7
[root@localhost ~]#
```

因为刚才安装了最新版,现在安装5.7版本,中间有一些冲突的
docker采用多文件方式,等下载冲突的文件的时候不再下载,而是
用本地的文件

## 4、删除镜像

### 4.1、删除单个镜像

docker rmi –f 镜像 id

```
[root@localhost ~]# docker rmi -f 8b43c6af2ad0
Untagged: mysql:5.7
Untagged: mysql@sha256:7a3a7b7a29e6fbff433c339fc52245435fa2c30858648
1f2f92ab1df239d6a29
Deleted: sha256:8b43c6af2ad08d95cdcb415d245446909a6cbc1875604c48c432
5972e5b00442
Deleted: sha256:aad43f4d2f66438acd2d156216cd544a728851238714975c38d9
a690f68afc57
Deleted: sha256:7b9addbc002c1e828aee7ec5c2679b04a591b6fa2b96002701dd
ee9d4ed54395
Deleted: sha256:b00f8e4e6ce8920fb563615503f232799ab380b338c3f2cbb5e8
6a2d762a6e80
Deleted: sha256:8fbabb17fd7b46a59cc15301741bf73a527b862f59cc6e84fae1
5b4dd5c425c0
[root@localhost ~]#
```

### 4.2、删除多个镜像

docker rmi -f 镜像 id 镜像 id 镜像 id 镜像 id

### 4.3、删除全部镜像

docker rmi -f $(docker images -aq)

```
[root@localhost ~]# docker rmi -f $(docker images -aq)
Untagged: mysql:latest
Untagged: mysql@sha256:aeecae58035f3868bf4f00e5fc623630d8b438db9d05f4d8c6538deb14d4c31b
Deleted: sha256:b05128b000ddbafb0a0d2713086c6a1cc23280dee3529d37f03c98c97c8cf1ed
Deleted: sha256:2920230e18d6833c32c9f851905df9d3e2958a43b771c84908234ac031b25a45
Deleted: sha256:a790dd6a368bc9aa7d1b251b46ac2fc718ebae5a38ed51ff89ff99955dadaa35
Deleted: sha256:cd87c1db4b159f37f092e73a52c10d5ccb837ed7bfcdc3b008038540390454a4
Deleted: sha256:7f92300b04af4aef96d5ef6fab1e27456cef354eca04733d396ad74478bee7d8
Deleted: sha256:6a59f55eb4945598b4889ea269d79f8b99563c96e97ba2373e19712732d20352
Deleted: sha256:87030c256d8077b4d969e5819f5da01ed08f29e115eaec61b58b3f3134175e1e
Deleted: sha256:b1694d0bb0b1be63e940478b93aa34f46e18f8371539ccee3b5d580cbf576812
Deleted: sha256:f323fd0baccb89f82a5711fa6291f3b4c977b85c3bbba59b1080205b498133b1
Deleted: sha256:47a2799e90faa9d9aaaa4b63457390dcbf5b26ce67f0926821c50b982d741e32
Deleted: sha256:156f55d34ef3e567ef39380f8d86f7c946927a099a43205de8721e60bfef526e
Deleted: sha256:bb282bb84eb90a6040504a46f462ebe55cb9623df13219fc39f434a53ccd1687
Deleted: sha256:77b323d4ec74aad770337f99a60e862a64ccc53f4775b5f4945df0e606f78b90
Untagged: hello-world:latest
Untagged: hello-world@sha256:cc15c5b292d8525effc0f89cb299f1804f3a725c8d05e158653a563f15e4f
Deleted: sha256:feb5d9fea6a5e9606aa995e879d862b825965ba48de054caab5ef356dc6b3412
[root@localhost ~]#
```

```
[root@localhost ~]# docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
[root@localhost ~]#
```

## (三)、容器操作

### 1、下载镜像(有镜像才能生成容器)

```
docker pull centos
```

```
[root@localhost ~]# docker pull centos
Using default tag: latest
latest: Pulling from library/centos
a1d0c7532777: Pull complete
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177
Status: Downloaded newer image for centos:latest
docker.io/library/centos:latest
[root@localhost ~]#
```

### 2、容器命令总结

```
docker run 镜像 id 新建容器并启动
docker ps 列出所有运行的容器 docker container list
docker rm 容器 id 删除指定容器
docker start 容器 id #启动容器
docker restart 容器 id #重启容器
docker stop 容器 id #停止当前正在运行的容器
docker kill 容器 id #强制停止当前容器
```

### 3、新建容器并启动

docker run [参数] image

参数:

--name=”new Name” # 给容器起一个名字

--d # 后台方式运行

--it # 交互方式运行

--p # 制定容器端口

-p ip:主机端口:容器端口 # 容器端口和主机端口映射

-p　　主机端口:容器端口

-p　　容器端口

--P　　# 随机制定端口

docker run –it centos /bin/bash

```
[root@localhost ~]# docker run -it centos /bin/bash
[root@25de5a9a0e0a /]#
```

4、容器返回主机

exit　　# 结束容器运行并返回主机

```
[root@25de5a9a0e0a /]# exit
exit
[root@localhost ~]#
```

ctrl+p+q　　# 退出不停止容器

6、查看容器

**6.1** 查看正在运行的容器

```
docker ps
```

```
[root@localhost ~]# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[root@localhost ~]#
```

**6.2**、查看运行过的容器

```
docker ps -a
```

```
[root@localhost ~]# docker ps -a
CONTAINER ID   IMAGE          COMMAND       CREATED         STATUS                     PORTS     NAMES
25de5a9a0e0a   centos         "/bin/bash"   5 minutes ago   Exited (0) 3 minutes ago             magical_sanderson
04a07245c95a   feb5d9fea6a5   "/hello"      15 hours ago    Exited (0) 15 hours ago              friendly_colden
[root@localhost ~]#
```

## 7、启动&停止&强制停止

### 7.1、启动容器

docker start 容器 id

```
[root@localhost ~]# docker start 25de5a9a0e0a
25de5a9a0e0a
[root@localhost ~]#
```

### 7.2、停止容器

docker stop 容器 id

```
[root@localhost ~]# docker stop 25de5a9a0e0a
25de5a9a0e0a
[root@localhost ~]#
```

### 7.3、强制停止

docker kill 容器 id

```
[root@localhost ~]# docker kill 25de5a9a0e0a
25de5a9a0e0a
[root@localhost ~]#
```

## 5、删除容器

### 5.1、删除指定容器,不能删除运行中的容器

docker rm 容器 id

```
[root@localhost ~]# docker rm 25de5a9a0e0a
25de5a9a0e0a
[root@localhost ~]#
```

### 5.2、删除所有容器

docker rm -f $(docker ps -aq)
或
docker ps -a -qlxargs docker rm

```
[root@localhost ~]# rm -f docker rm -f $(docker ps -aq)
[root@localhost ~]#
```

## (四)、其他命令

### 1、用后台方式启动容器

```
docker run -d centos
docker  容器使用后台运行，就必须要有一个前台进程，
docker 发现没有应用，就会自动停止
```

### 2、查看日志

### 2.1、首先使用后台启动启动一个容器

```
docker run -d centos /bin/sh -c "while true;do echo
6666;sleep 1;done"
```

```
[root@localhost ~]# docker run -d centos /bin/sh -c "while true;do echo 6666;sleep 1;done"
WARNING: IPv4 forwarding is disabled. Networking will not work.
b573ababbe5d1a18b554fd76513d877f8e12882f0e3e205517a6aec99c9d7810
[root@localhost ~]#
```

### 2.2、查看容器日志信息

```
docker logs -tf --tail 查看多少条 容器id
```

```
[root@localhost ~]# docker logs -tf --tail 10 b573ababbe5d
2021-11-29T11:25:31.814273381Z 6666
2021-11-29T11:25:32.819539097Z 6666
2021-11-29T11:25:33.821493481Z 6666
2021-11-29T11:25:34.823532549Z 6666
2021-11-29T11:25:35.825199504Z 6666
2021-11-29T11:25:36.829500703Z 6666
2021-11-29T11:25:37.833339841Z 6666
2021-11-29T11:25:38.837234394Z 6666
2021-11-29T11:25:39.839852185Z 6666
2021-11-29T11:25:40.842121828Z 6666
2021-11-29T11:25:41.845143977Z 6666
2021-11-29T11:25:42.847552074Z 6666
2021-11-29T11:25:43.851946711Z 6666
```

### 3、查看容器中进程的信息

```
docker top 容器id
```

```
[root@localhost ~]# docker top b573ababbe5d
UID                 PID                 PPID                C                   STIME               TTY                 TIME
    CMD
root                15673               15654               0                   03:19               ?                   00:00:00
    /bin/sh -c while true;do echo 6666;sleep 1;done
root                16946               15673               0                   03:33               ?                   00:00:00
    /usr/bin/coreutils --coreutils-prog-shebang=sleep /usr/bin/sleep 1
[root@localhost ~]#
```

### 4、查看镜像元数据

```
docker inspect 容器id
```

```
[root@localhost ~]# docker inspect 5d0da3dc9764
[
    {
        "Id": "sha256:5d0da3dc976460b72c77d94c8a1ad043720b0416bfc16c52c45d4847e53fadb6",
        "RepoTags": [
            "centos:latest"
        ],
        "RepoDigests": [
            "centos@sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b177"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2021-09-15T18:20:05.184694267Z",
        "Container": "9bf8a9e2ddff4c0d76a587c40239679f29c863a967f23abf7a5babb6c2121bf1",
        "ContainerConfig": {
            "Hostname": "9bf8a9e2ddff",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
```

5、进入后台容器(方式一)

```
docker exec -it 容器id /bin/bash
```

```
[root@localhost ~]# docker exec -it b573ababbe5d /bin/bash
[root@b573ababbe5d /]#
```

6、进入后台容器(方式二)

```
docker attach 容器id
```

```
[root@localhost ~]# docker attach b573ababbe5d
6666
6666
6666
6666
```

7、2 种进入后台命令的区别

exec　　# 开启一个新的终端,可以在里面操作 exit 不终止容器

attach　# 进入容器正在执行的终端,exit 退出终止容器

8、从容器中拷贝到主机

**8.1、进入容器并创建文件**

docker run –it –name=”cpTest” centos /bin/bash

```
[root@localhost ~]# docker run -it --name="cpTest" centos /bin/bash
WARNING: IPv4 forwarding is disabled. Networking will not work.
[root@f1f4b98d24f6 /]# ls
bin  dev  etc  home  lib  lib64  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
[root@f1f4b98d24f6 /]# cd home/
[root@f1f4b98d24f6 home]# touch test.java
[root@f1f4b98d24f6 home]#
```

**8.2、退出容器不要终止**

ctrl+p+q

```
[root@f1f4b98d24f6 home]# [root@localhost ~]#
```

**8.3、开始复制**

```
docker cp 容器id:容器文件位置 复制到本地的目录
```

```
[root@localhost ~]# docker cp f1f4b98d24f6:/home/test.java /home
[root@localhost ~]# ls /home/
111  test.java
[root@localhost ~]#
```

# 四、安装服务

## (一)、安装 nginx

1、搜索镜像

docker search nginx

```
[root@localhost ~]# docker search nginx
NAME                              DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
nginx                             Official build of Nginx.                        15883     [OK]
jwilder/nginx-proxy               Automated Nginx reverse proxy for docker con…   2098                 [OK]
richarvey/nginx-php-fpm           Container running Nginx + PHP-FPM capable of…   819                  [OK]
jc21/nginx-proxy-manager          Docker container for managing Nginx proxy ho…   283
linuxserver/nginx                 An Nginx container, brought to you by LinuxS…   160
tiangolo/nginx-rtmp               Docker image with Nginx using the nginx-rtmp…   146                  [OK]
jlesage/nginx-proxy-manager       Docker container for Nginx Proxy Manager        144                  [OK]
```

## 2、拉取镜像

docker pull nginx

```
[root@localhost ~]# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
eff15d958d66: Pull complete
1e5351450a59: Pull complete
2df63e6ce2be: Pull complete
9171c7ae368c: Pull complete
020f975acd28: Pull complete
266f639b35ad: Pull complete
Digest: sha256:097c3a0913d7e3a5b01b6c685a60c03632fc7a2b50bc8e35bcaa3691d788226e
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest
[root@localhost ~]#
```

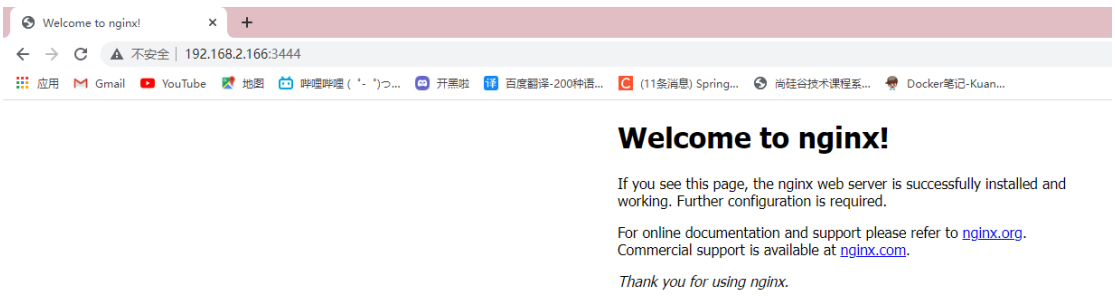
## 3、创建容器

docker run –it 名字 映射本地端口为 3344 容器中端口为 80 使用 nginx 镜像

docker run –it –name=”one nginx test” –p 3344:80 nginx

```
[root@localhost ~]# docker run  -d --name one nginx test –p 3344:80 nginx
WARNING: IPv4 forwarding is disabled. Networking will not work.
976ad1d2657da239874587880bb052d1339f6114bac03953c0fdcdfc8b02f876
[root@localhost ~]#
```

## 4、测试



## (二)、安装 tomcat

### 1、搜索镜像

docker search tomcat

```
[root@localhost ~]# docker search tomcat
NAME                          DESCRIPTION                                     STARS     OFFICIAL   AUTOMATED
tomcat                        Apache Tomcat is an open source implementati…   3184      [OK]
tomee                         Apache TomEE is an all-Apache Java EE certif…   94        [OK]
dordoka/tomcat                Ubuntu 14.04, Oracle JDK 8 and Tomcat 8 base…   58                   [OK]
kubeguide/tomcat-app          Tomcat image for Chapter 1                      31
consol/tomcat-7.0             Tomcat 7.0.57, 8080, "admin/admin"              18                   [OK]
cloudesire/tomcat             Tomcat server, 6/7/8                            15                   [OK]
aallam/tomcat-mysql           Debian, Oracle JDK, Tomcat & MySQL              13                   [OK]
arm32v7/tomcat                Apache Tomcat is an open source implementati…   11
```

### 2、安装 tomcat

docker pull tomcat:9.0

### 3、创建 tomcat 容器

docker run -d --name onTomcat -p3999:80 tomcat

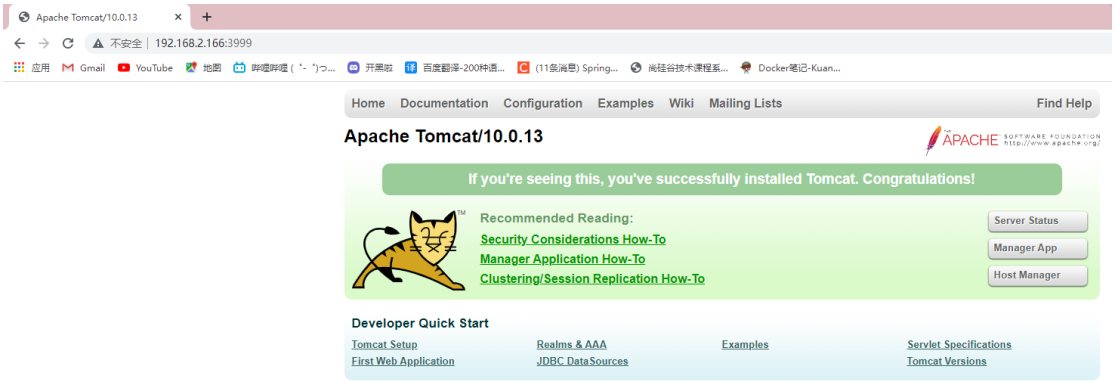
### 4、进入容器

docker exec  -it  onTomcat /bin/bash

### 5、复制 tomcat 模板到主目录

cp webapps.dist/* webapps

## 6、访问




# (三)、Docker 可视化

## 1、安装

```
docker run -d -p 8080:9000 \
--restart=always -v
/var/run/docker.sock:/var/run/docker.sock --
privileged=true portainer/portainer
```

```
[root@localhost ~]# docker run -d -p 8080:9000 \
> --restart=always -v /var/run/docker.sock:/var/run/docker.sock --privil
eged=true portainer/portainer
Unable to find image 'portainer/portainer:latest' locally
latest: Pulling from portainer/portainer
94cfa856b2b1: Pull complete
49d59ee0881a: Pull complete
a2300fd28637: Pull complete
Digest: sha256:fb45b43738646048a0a0cc74fcee2865b69efde857e710126084ee5de9be0f3f
Status: Downloaded newer image for portainer/portainer:latest
WARNING: IPv4 forwarding is disabled. Networking will not work.
f5f9fbcf3634dc708c25c6ff92ccb9ca0ec665028ad340c77dcb0e2f5e64a506
[root@localhost ~]#
```

2、测试

```
[root@localhost ~]# curl http://192.168.2.166:8080
<!DOCTYPE html
><html lang="en" ng-app="portainer">
  <head>
    <meta charset="utf-8" />
    <title>Portainer</title>
    <meta name="description" content="" />
    <meta name="author" content="Portainer.io" />

    <!-- HTML5 shim, for IE6-8 support of HTML5 elements -->
    <!--[if lt IE 9]>
      <script src="//html5shim.googlecode.com/svn/trunk/html5.js"></script>
    <![endif]-->

    <!-- Fav and touch icons -->
    <link rel="apple-touch-icon" sizes="180x180" href="dc4d092847be46242d8c013d1bc7c494.png" />
    <link rel="icon" type="image/png" sizes="32x32" href="5ba13dcb526292ae707310a54e103cd1.png" />
    <link rel="icon" type="image/png" sizes="16x16" href="f9508a64a1beb81be174e194573f7450.png" />
    <link rel="mask-icon" href="07745d55b001c85826eedd479285cdbb.svg" color="#5bbad5" />
```

# 五、打包镜像

## (一)、命令

docker commit 提交容器为一个镜像
docker commit –m=”描述信息” –a=”作者” 容器 id 目标镜像名:版本

## (二)、打包镜像说明

一开始的 tomcat 的 webapp 下没有东西,webapp 的东西在 webapp.dict 中,我们创建一个官方的 tomcat 容器,进去把 webapp.dict 的内容复制一份到 webapp 中,并打包成为自己的 tomcat.

## (三)、开始打包

### 1、使用 tomcat 进入容器

docker run -it --name oneTomcat -p 3666:8080 tomcat /bin/bash

```
[root@localhost ~]# docker run -it --name oneTomcat -p 3666:8080 tomcat /bin/bash
WARNING: IPv4 forwarding is disabled. Networking will not work.
root@e1671f809e32:/usr/local/tomcat#
```

2、复制 webapp.dict 的文件到 webapp 中

```
root@e1671f809e32:/usr/local/tomcat# cp -r webapps.dist/* webapps
root@e1671f809e32:/usr/local/tomcat#
```

3、退出但不终止容器

ctrl+p+q

4、打包镜像

docker commit -m="my one image" -a="ma" e1671f809e32 onetomcatimage:1.0

```
[root@localhost ~]# docker commit -m="my one image" -a="ma" e1671f809e32 onetomcatimage:1.0
sha256:4cf53e8288f99ee2aaef2ef1151f5ffad619c8e5c7df95c9160c9fc26de513a2
[root@localhost ~]# docker images
REPOSITORY            TAG       IMAGE ID       CREATED         SIZE
onetomcatimage        1.0       4cf53e8288f9   4 seconds ago   684MB
tomcat                9.0       76206e3ba4b1   11 days ago     680MB
tomcat                latest    904a98253fbf   11 days ago     680MB
nginx                 latest    ea335eea17ab   13 days ago     141MB
centos                latest    5d0da3dc9764   2 months ago    231MB
portainer/portainer   latest    580c0e4e98b0   8 months ago    79.1MB
[root@localhost ~]#
```

## 六、容器数据卷

### (一)、什么是容器卷

当我们使用容器的时候在容器中创建文件,如果不小心删除了容器就没有了,这时候就需要存储到本地,这就用到了容器卷,使容器中的文件和本地文件相对应

即使容器停止或删除,本地文件还是有的

## (二)、使用数据卷命令

```
docker run -it -v 主机目录:容器内目录
```

## (三)、使用 Centos 容器做例子

1、使 Centos 容器中的/home 目录和本地的/home/ceshi 绑定

```
docker run -it -v /home/ceshi:/home centos /bin/bash
```

```
[root@localhost home]# docker run -it -v /home/ceshi:/home centos /bin/bash
WARNING: IPv4 forwarding is disabled. Networking will not work.
[root@34c92f544a5b /]#
```

2、在 Centos 的/home 中创建文件

```
[root@34c92f544a5b /]# cd /home/
[root@34c92f544a5b home]# touch test.java
[root@34c92f544a5b home]#
```

3、退出查看本地的/home/ceshi 文件

```
[root@localhost home]# cd /home/ceshi/
[root@localhost ceshi]# ls
test.java
[root@localhost ceshi]#
```

## (四)、使用 Mysql 做例子

1、下载 mysql&开启容器&文件绑定

docker run -it -p 3344:3306 -v /home/mysql/conf:/etc/mysql/my.conf -v /home/mysql/data:/var/lib/mysql -e MYSQL

```
[root@localhost ~]# docker run -it -p 3344:3306 -v /home/mysql/conf:/etc/mysql/my.conf -v /home/mysql/data:/v
ar/lib/mysql -e MYSQL_ROOT_PASSWORD=123456 --name onemysql mysql:5.7 /bin/bash
Unable to find image 'mysql:5.7' locally
5.7: Pulling from library/mysql
a10c77af2613: Pull complete
b76a7eb51ffd: Pull complete
258223f927e4: Pull complete
2d2c75386df9: Pull complete
63e92e4046c9: Pull complete
f5845c731544: Pull complete
bd0401123a9b: Pull complete
2724b2da64fd: Pull complete
d10a7e9e325c: Pull complete
1c5fd9c3683d: Pull complete
2e35f83a12e9: Pull complete
Digest: sha256:7a3a7b7a29e6fbff433c339fc52245435fa2c308586481f2f92ab1df239d6a29
Status: Downloaded newer image for mysql:5.7
WARNING: IPv4 forwarding is disabled. Networking will not work.
root@c29bbac00e3f:/#
```

2、查看容器中/var/lib/mysql 并查看目录

```
root@c29bbac00e3f:/etc/mysql# cd /var/lib/mysql
root@c29bbac00e3f:/var/lib/mysql# ls
root@c29bbac00e3f:/var/lib/mysql#
```

3、创建文件

```
root@c29bbac00e3f:/var/lib/mysql# touch mysql.test
root@c29bbac00e3f:/var/lib/mysql# ls
mysql.test
root@c29bbac00e3f:/var/lib/mysql#
```

4、退出容器查看本地文件

```
[root@localhost ~]# cd /home/mysql/data/
[root@localhost data]# ls
mysql.test
[root@localhost data]#
```

## (五)、具名和匿名挂载

### 1、匿名挂载

docker run -d -P --name nginx01 -v /etc/nginx nginx

```
[root@localhost ~]# docker run -d -P --name nginx01 -v /etc/nginx nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
eff15d958d66: Pull complete
1e5351450a59: Pull complete
2df63e6ce2be: Pull complete
9171c7ae368c: Pull complete
020f975acd28: Pull complete
266f639b35ad: Pull complete
Digest: sha256:097c3a0913d7e3a5b01b6c685a60c03632fc7a2b50bc8e35bcaa3691d788226e
Status: Downloaded newer image for nginx:latest
WARNING: IPv4 forwarding is disabled. Networking will not work.
985e7f97b4592befab0c646b7c82aa798035ce33603c360a987f1e1e68f28b19
[root@localhost ~]#
[root@localhost ~]#
```

### 2、查看匿名挂载

```
[root@localhost ~]# docker volume ls
DRIVER    VOLUME NAME
local     b7fcb89046108317bbe7ebba1b1debf4ec3d8ea4895aac9eb8f763de38513148
[root@localhost ~]#
```

### 3、具名挂载

docker run -d -P --name nginx02 -v juming-nginx:/etc/nginx nginx

```
[root@localhost ~]# docker run -d -P --name nginx02 -v juming-nginx:/etc/nginx nginx
WARNING: IPv4 forwarding is disabled. Networking will not work.
8e3f9ca71dd02cf2b51653f25ac8531a22ebffd22450e39d91f9459b1c7afc45
[root@localhost ~]#
```

### 4、查看具名挂载

```
[root@localhost ~]# docker volume  ls
DRIVER    VOLUME NAME
local     b7fcb89046108317bbe7ebba1b1debf4ec3d8ea4895aac9eb8f763de38513148
local     juming-nginx
[root@localhost ~]#
```

5、说明

匿名挂载:volume 没有名字,会生成一个 uuid 作为卷的名字
具名挂载:volume 规定了名字,但没有指定本机的目录,默认是在 /var/lib/docker/volume 下


6、分类

-v 容器内路径 # 匿名挂载
-v 名字:容器内路径 # 具名挂载
-v 本地路径:容器内路径 # 指定路径挂载

## (六)、卷权限设置

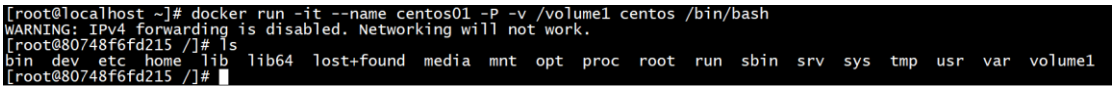
ro readonly # 只读
rw readwrite # 可读可写
docker run –d –P –name nginx01 –v name:/etc/nginx:ro nginx
docker run –d –P –name nginx01 –v name:/etc/nginx:rw nginx

ro 权限:
当卷设置了 ro 权限,只能通过本地主机修改文件,容器内对文件无法操作

## (七)、数据卷容器(容器数据同步)

1、首先创建一个用来同步卷的容器

docker run -it --name centos01 centos -v /volume1 /bin/bash



2、退出不终止

3、创建另一个容器实现同步机制

使用参数　--volumes-from 同步卷的容器

```
[root@localhost ~]# docker run -it --name contos02 -P --volumes-from centos01 centos
WARNING: IPv4 forwarding is disabled. Networking will not work.
[root@9ae318b7608f /]# ls
bin  dev  etc  home  lib  lib64  lost+found  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var  volume1
[root@9ae318b7608f /]#
```

4、进入 volume1 文件夹创建文件

```
[root@9ae318b7608f /]# cd volume1/
[root@9ae318b7608f volume1]# touch java.test
[root@9ae318b7608f volume1]# ls
java.test
[root@9ae318b7608f volume1]#
```

5、退出不终止,进入 centos1

```
[root@localhost ~]# docker attach centos01
[root@80748f6fd215 /]# cd volume1/
[root@80748f6fd215 volume1]# ls
java.test
[root@80748f6fd215 volume1]#
```

## 七、DockerFile(创建自己的镜像)

### (一)、介绍

dockerFile 是用来构建 docker 镜像的的文件,命令参数脚本!

### (二)、构建步骤

1、编写一个 dockerfile 文件

2、docker build 构建一个镜像
3、docker run 运行一个镜像
4、docker push 发布镜像(dockerhub 或阿里云仓库)

**(三)、dockerfile 的规则**

1、每个关键字必须 大写
2、执行从上到下
3、#表示注释

**(四)、dockerfile 常用命令**

```
FROM                # 基础镜像，一切从这里开始构建
MAINTAINER          # 镜像是谁写的， 姓名+邮箱
RUN             # 镜像构建的时候需要运行的命令
ADD             # 步骤，tomcat 镜像，这个 tomcat 压缩包！添加内容 添加同目录
WORKDIR             # 镜像的工作目录
VOLUME              # 挂载的目录
EXPOSE              # 保留端口配置
CMD             # 指定这个容器启动的时候要运行的命令，只有最后一个会生效，可被替代。
ENTRYPOINT          # 指定这个容器启动的时候要运行的命令，可以追加命令
ONBUILD             # 当构建一个被继承 DockerFile 这个时候就会运行 ONBUILD 的指令，触发指令。
COPY                # 类似 ADD，将我们文件拷贝到镜像中
ENV             # 构建的时候设置环境变量！
```

## (五)、创建自己的 dockerfile

1、在/home 下创建一个 dockerfile 文件夹储存 dockerfile 文件

```
[root@localhost home]# mkdir dockfile
[root@localhost home]# cd dockfile/
[root@localhost dockfile]#
```

2、编写 dockerfile 文件

```
[root@localhost dockfile]# vi dockrfile1
```

```
FROM centos                    # 使用 centos 作为基础环境
MAINTAINER ma<2992455524@qq.com>          # 作者<邮箱>
ENV MYPATH /usr/local          # 环境变量
WORKDIR $MYPATH                # 刚进去的工作目录为/usr/local
RUN yum -y install vim         # 中间执行的命令
RUN yum -y install bind        # 下载 dns 服务
EXPOSE 80                      # 暴露端口为 80 端口
CMD echo $MYPATH               # 输出$MYPATH 环境变量
CMD /bin/bash                  # 以 bash 的方式运行
```

```
FROM centos      # 使用centos作为基础环境
MAINTAINER ma<2992455524@qq.com>          # 作者<邮箱>
ENV MYPATH /usr/local    # 环境变量
WORKDIR $MYPATH          # 刚进去的工作目录为/usr/local
RUN yum -y install vim   # 中间执行的命令
RUN yum -y install bind  # 下载dns服务
EXPOSE 80                # 暴露端口为80端口
CMD echo $MYPATH         # 输出$MYPATH环境变量
CMD /bin/bash            # 以bash的方式运行
```

3、构建这个镜像

docker build –f dockerfile 文件名 –t 镜像名:版本号

docker build-f dockerfile1 –t mycentos:1.0 . (不要忘了后面还有

一个.)

```
[root@localhost dockfile]# docker build -f dockrfile1 -t mycentos:1.0 .
Sending build context to Docker daemon  2.048kB
Step 1/8 : FROM centos
 ---> 5d0da3dc9764
Step 2/8 : MAINTAINER ma<2992455524@qq.com>
 ---> [Warning] IPv4 forwarding is disabled. Networking will not work.
 ---> Running in 5e260504637b
Removing intermediate container 5e260504637b
 ---> e90e58f86551
Step 3/8 : ENV MYPATH /usr/local
 ---> [Warning] IPv4 forwarding is disabled. Networking will not work.
 ---> Running in f5e96252a686
Removing intermediate container f5e96252a686
 ---> b5358001cbeb
Step 4/8 : WORKDIR $MYPATH
 ---> [Warning] IPv4 forwarding is disabled. Networking will not work.
 ---> Running in ddf2c75f1897
Removing intermediate container ddf2c75f1897
 ---> 209aaf446458
Step 5/8 : RUN ls
 ---> [Warning] IPv4 forwarding is disabled. Networking will not work.
 ---> Running in ffb79e851046
```

```
Successfully built 39762a3a9740
Successfully tagged mycentos:1.0
```

4、使用镜像

docker run –it mycentos:1.0 /bin/bash

```
[root@localhost dockfile]# docker run -it mycentos:1.0 /bin/bash
WARNING: IPv4 forwarding is disabled. Networking will not work.
[root@68b6116038f5 local]#
```

## (六)、制作 tomcat 镜像

1、准备 jdk 和 tomcat 压缩包

```
[root@localhost home]# ls
111  apache-tomcat-8.0.50-windows-x64.zip  Dockerfile  java-11-openjdk-11.0.11.9-1.windows.redhat.x86_64.zip
[root@localhost home]#
```

2、编写 dockerfile 文件

FROM centos
MAINTAINER ma<2992455524@qq.com>
ADD java-11-openjdk-11.0.11.9-1.windows.redhat.x86_64.zip /usr/local/

```
ADD apache-tomcat-8.0.50-windows-x64.zip /usr/local/
ENV MYPATH /usr/local
WORKDIR $MYPATH
ENV JAVA_HOME /usr/local/java-11-openjdk-11.0.11.9-1.windows.redhat.x86_64
ENV CATALINA_HOME /usr/local/apache-tomcat-8.0.50-windows-x64
ENV PATH $PATH:$JAVA_HOME/bin:$CATALINA_HOME/lib
EXPOSE 8080
CMD /usr/local/apache-tomcat-8.0.50-windows-x64/bin/startup.sh && /usr/local/apache-tomcat-8.0.50-windows-x64/logs/catalina.out
```