Docker入门

Docker 基础命令

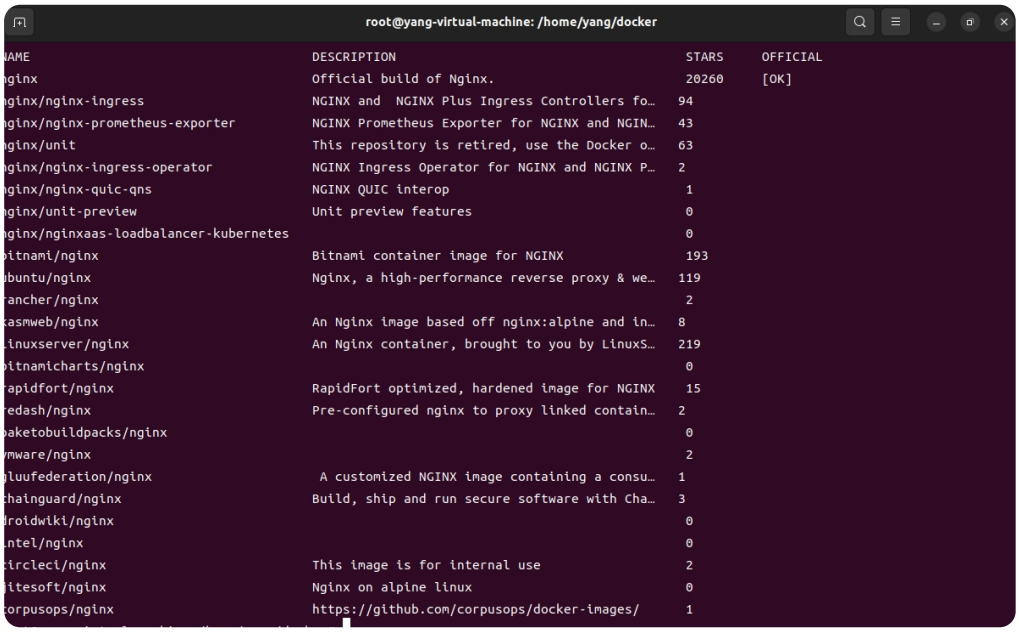
实验：启动一个nginx，将它的页面改为自己的页面并发布

步骤：下载镜像->启动容器->修改页面->保存镜像->分享社区

下载镜像

检索:docker search

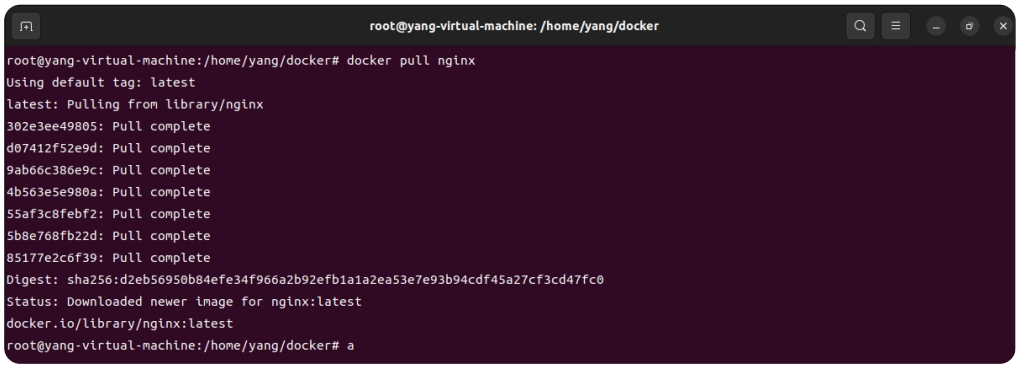
使用docker search命令 搜索nginx镜像



搜索结果中，NAME表示镜像名，DESCRIPTION是镜像的描述,OFFICIAL表示镜像是官方的

下载：docker pull

使用docker pull命令 可以下载对应镜像



> docker pull nginx 其实等价于 docker pull nginx:latest 下载最新版本
>
> 如果像下载特殊版本的镜像，推荐不使用search命令，而是到docker hub 官网搜索镜像名 查看对应版本号

列表: docker images

使用docker images命令，可以列出已经下载的镜像列表

```
root@yang-virtual-machine:/home/yang/docker# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
nginx         latest    7f553e8bbc89   9 days ago      192MB
hello-world   latest    d2c94e258dcb   17 months ago   13.3kB
```

REPOSITROTY：镜像名

TAG：镜像版本

IMAGE ID：镜像ID

CREATED：镜像创建时间

SIZE：镜像的大小

删除：docker rmi

使用docker rmi命令可以删除镜像

比如

```
1  docker rmi nginx:latest
```

启动容器

运行：docker run

```
1  Usage: docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

[]表示可选内容，一般COMMAND 和ARGS...可以忽略，除非想要修改镜像的初始启动方法

[OPTIONS] 填写启动镜像的一些参数设置

```
root@yang-virtual-machine:/home/yang/docker# docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
nginx         latest    7f553e8bbc89   9 days ago      192MB
hello-world   latest    d2c94e258dcb   17 months ago   13.3kB
root@yang-virtual-machine:/home/yang/docker# docker run nginx:latest
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/10/12 02:36:40 [notice] 1#1: using the "epoll" event method
2024/10/12 02:36:40 [notice] 1#1: nginx/1.27.2
2024/10/12 02:36:40 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2024/10/12 02:36:40 [notice] 1#1: OS: Linux 6.8.0-45-generic
2024/10/12 02:36:40 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/10/12 02:36:40 [notice] 1#1: start worker processes
2024/10/12 02:36:40 [notice] 1#1: start worker process 29
2024/10/12 02:36:40 [notice] 1#1: start worker process 30
```

这个命令会阻塞控制台

查看：docker ps

```
root@yang-virtual-machine:/home/yang/桌面# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS     NAMES
7d7cdd581719   nginx:latest   "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   80/tcp    bold_lamport
root@yang-virtual-machine:/home/yang/桌面#
```

docker ps命令可以查看运行中的应用

CONTAINER ID: 运行中镜像的唯一ID

IMAGE:镜像名

COMMAND: 启动命令

CREATED: 多久之前启动的

STATUS：Up代表上线成功

PORTS: 代表应用所占的端口为80端口

NAMES:应用容器的名字

使用docker ps -a可以查看所有容器，包括停止的容器

```
root@yang-virtual-machine:/home/yang/docker# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS                     PORTS     NAMES
7d7cdd581719   nginx:latest   "/docker-entrypoint.…"   6 minutes ago   Exited (0) 32 seconds ago            bold_lamport
d8afc2f526e7   hello-world    "/hello"                 2 weeks ago     Exited (0) 2 weeks ago               modest_dijkstra
27159fd437ed   hello-world    "/hello"                 2 weeks ago     Exited (0) 2 weeks ago               youthful_meitner
```

停止:docker stop

docker stop可以停止应用运行

```
root@yang-virtual-machine: /home/yang/docker
root@yang-virtual-machine:/home/yang/docker# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS              PORTS     NAMES
7d7cdd581719   nginx:latest   "/docker-entrypoint.…"   9 minutes ago   Up About a minute   80/tcp    bold_lamport
root@yang-virtual-machine:/home/yang/docker# docker stop bold_lamport
bold_lamport
root@yang-virtual-machine:/home/yang/docker# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS                     PORTS     NAMES
7d7cdd581719   nginx:latest   "/docker-entrypoint.…"   9 minutes ago   Exited (0) 5 seconds ago             bold_lamport
d8afc2f526e7   hello-world    "/hello"                 2 weeks ago     Exited (0) 2 weeks ago               modest_dijkstra
27159fd437ed   hello-world    "/hello"                 2 weeks ago     Exited (0) 2 weeks ago               youthful_meitner
```

后面的参数可以是CONTAINER ID也可以是NAMES

启动:docker start

使用docker start命令，可以启动停止的容器

```
root@yang-virtual-machine:/home/yang/docker# docker start 7d7cdd581719
7d7cdd581719
root@yang-virtual-machine:/home/yang/docker# docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS     NAMES
7d7cdd581719   nginx:latest   "/docker-entrypoint.…"   7 minutes ago   Up 22 seconds  80/tcp    bold_lamport
root@yang-virtual-machine:/home/yang/docker#
```

后面的参数可以是CONTAINER ID也可以是NAMES

重启:docker restart

docker restart命令可以重启应用

状态:docker stats

docker stats可以展示应用所占的资源信息

```
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT    MEM %     NET I/O       BLOCK I/O    PIDS
7d7cdd581719   bold_lamport   0.00%     3.07MiB / 3.778GiB   0.08%     3.15kB / 0B   0B / 4.1kB   3
```

日志:docker logs

docker logs可以查看某个应用运行中产生的日志

删除:docker rm

使用docker rm 可以删除停止运行的应用

```
root@yang-virtual-machine:/home/yang/桌面# docker rm 7d7
7d7
```

docker run 的细节

```
root@yang-virtual-machine:/home/yang/桌面# docker run -d --name mynginx nginx
868c797acfcb56ee1953a200760f9b92b3eaf0aa04eb97984aa0f86d7e2010fa
root@yang-virtual-machine:/home/yang/桌面# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS
  PORTS     NAMES
868c797acfcb   nginx     "/docker-entrypoint.…"   8 seconds ago   Up 7 seconds
 80/tcp    mynginx
```

参数解析:

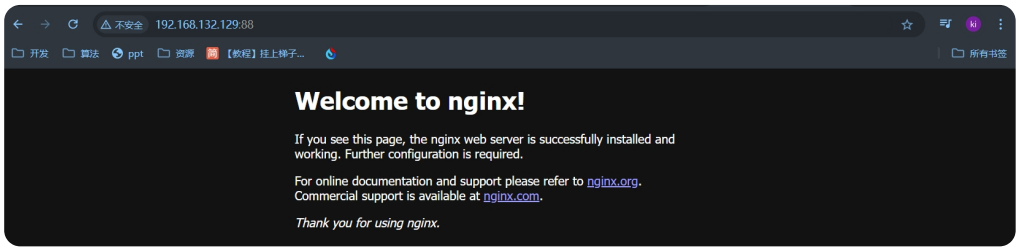
-d 后台启动容器

--name 给容器名字

```
root@yang-virtual-machine:/home/yang/桌面# docker run -d --name mynginx -p 88:80 nginx
077ab6136bf44d604735a8afa871ce9a1b80530ae58102dc24fa2ae1452ba0fa
root@yang-virtual-machine:/home/yang/桌面# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS         PORTS                                   NAMES
077ab6136bf4   nginx     "/docker-entrypoint.…"   3 seconds ago   Up 2 seconds   0.0.0.0:88->80/tcp, [::]:88->80/tcp   mynginx
root@yang-virtual-machine:/home/yang/桌面#
```

-p 端口映射

此时可以通过主机浏览器访问虚拟机88端口来访问容器



进入:docker exec

```
root@yang-virtual-machine:/home/yang/桌面# docker exec -it mynginx /bin/bash
root@077ab6136bf4:/#
```

-it 交互模式

/bin/bash 使用控制台模式交互

```
root@077ab6136bf4:/# cd /usr/share/nginx/html/
root@077ab6136bf4:/usr/share/nginx/html# ls
50x.html  index.html
root@077ab6136bf4:/usr/share/nginx/html# echo "<h1>Hello Docker I Want Change Index.html</h1>" > index.html
root@077ab6136bf4:/usr/share/nginx/html# cat index.html
<h1>Hello Docker I Want Change Index.html</h1>
root@077ab6136bf4:/usr/share/nginx/html#
```

我们进入容器后，cd到指定文件目录(可以到官网查)，修改nginx的index.html

由于容器及其轻量级，以至于没有vi命令

我们使用echo 插入一个h1标签到index.html中

修改后效果

Hello Docker I Want Change Index.html

exit可以退出该容器

保存镜像

提交: docker commit

可以从容器的改变里创建一个新的镜像

```
Usage:  docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]
```

Options:

-a 作者

-c 有哪些改变的列表

-m 本次改变的消息

-p 提交期间暂停容器运行

```
root@yang-virtual-machine:/home/yang/桌面# docker commit -m "update index.html"
mynginx mynginx:v1.0
sha256:785d25781fde026280b8bc5177cf7ae3211d7faf5c397ed293552a019098c703
root@yang-virtual-machine:/home/yang/桌面# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
mynginx       v1.0      785d25781fde   7 seconds ago    192MB
nginx         latest    7f553e8bbc89   9 days ago       192MB
root@yang-virtual-machine:/home/yang/桌面#
```

提交后，使用docker images命令，可以看到已经有一个mynginx

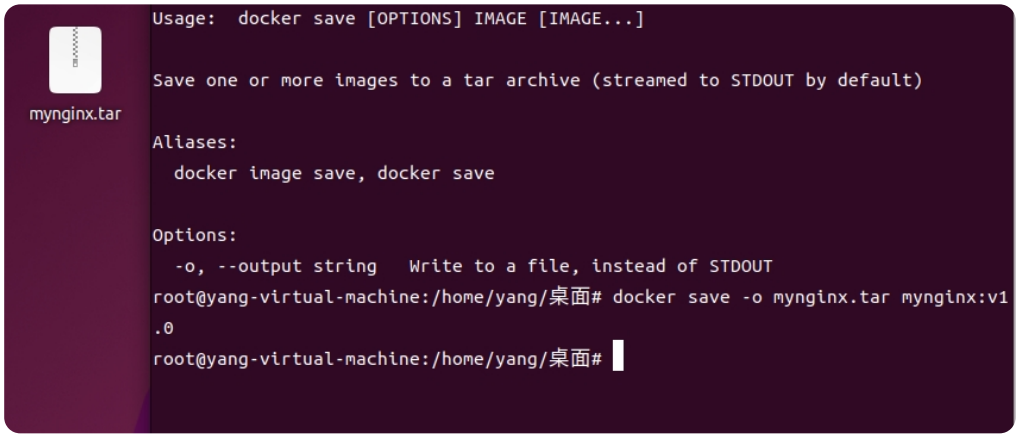
保存:docker save

保存一个或多个镜像为tar文件

```
1  Usage:  docker save [OPTIONS] IMAGE [IMAGE...]
```

Options:

-o 文件名称



加载:docker load

从一个tar包或者STDIN中加载一个镜像

```
1  Usage:  docker load [OPTIONS]
```

Options:

-i 指定压缩包

-q

```
root@yang-virtual-machine:/home/yang/桌面# docker load -i mynginx.tar
8d853c8add5d: Loading layer  77.83MB/77.83MB
756474215d29: Loading layer  117.9MB/117.9MB
3d07a4a7eb2a: Loading layer  3.584kB/3.584kB
6b133b4de5e6: Loading layer  4.608kB/4.608kB
1c1f11fd65d6: Loading layer   2.56kB/2.56kB
7619c0ba3c92: Loading layer   5.12kB/5.12kB
825fb68b6033: Loading layer  7.168kB/7.168kB
7ab0dfc85263: Loading layer  14.85kB/14.85kB
Loaded image: mynginx:v1.0
```

load后使用docker run 命令即可运行容器

```
root@yang-virtual-machine:/home/yang/桌面# docker run -d --name app01 -p 88:80 m
ynginx:v1.0
84ec9230a2268be94241ff61d0a0063e72429b1f925be4f35d82dea6cb17f41c
```

分享社区

登录：docker login

注册一个docker hub账号，并在命令行使用docker login 命令登录

```
root@yang-virtual-machine:/home/yang/桌面# docker login
Failed to start web-based login - falling back to command line login...

Log in with your Docker ID or email address to push and pull images from Docker
Hub. If you don't have a Docker ID, head over to https://hub.docker.com/ to crea
te one.
You can log in with your password or a Personal Access Token (PAT). Using a limi
ted-scope PAT grants better security and is required for organizations using SSO
. Learn more at https://docs.docker.com/go/access-tokens/

Username: yang20031012
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
```

命名：docker tag

从源镜像创建一个新的目标镜像

```
1  Usage:  docker tag SOURCE_IMAGE[:TAG]
   TARGET_IMAGE[:TAG]
```

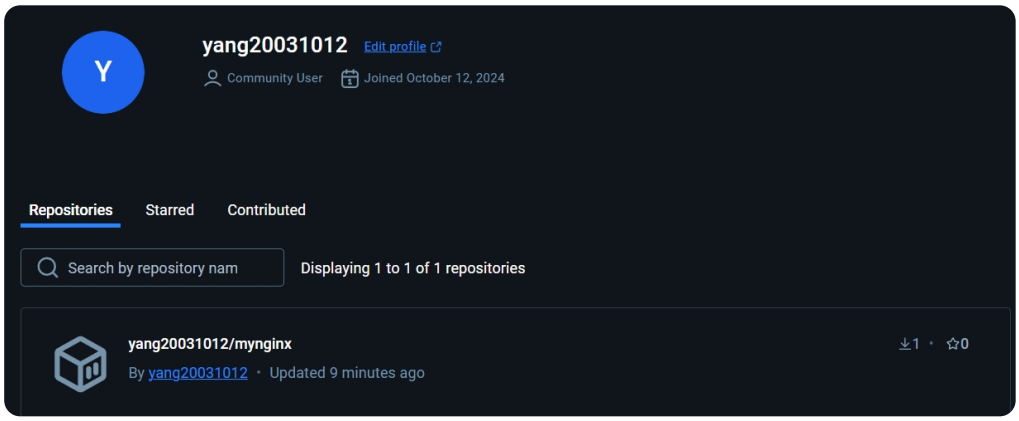
```
root@yang-virtual-machine:/home/yang/桌面# docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
mynginx       v1.0      785d25781fde   17 minutes ago   192MB
root@yang-virtual-machine:/home/yang/桌面# docker tag --help

Usage:  docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]

Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE

Aliases:
  docker image tag, docker tag
root@yang-virtual-machine:/home/yang/桌面# docker tag mynginx:v1.0 yang20031012/
mynginx:v1.0
root@yang-virtual-machine:/home/yang/桌面# docker images
REPOSITORY              TAG       IMAGE ID       CREATED          SIZE
mynginx                 v1.0      785d25781fde   20 minutes ago   192MB
yang20031012/mynginx    v1.0      785d25781fde   20 minutes ago   192MB
```

虽然镜像名和标签不同，但是镜像ID是相同的

推送：docker push

使用docker push命令 即可将镜像推送至docker hub 之后便可以使用docker pull命令拉取

```
root@yang-virtual-machine:/home/yang/桌面# docker push yang20031012/mynginx:v1.0
The push refers to repository [docker.io/yang20031012/mynginx]
7ab0dfc85263: Pushed
825fb68b6033: Mounted from library/nginx
7619c0ba3c92: Mounted from library/nginx
1c1f11fd65d6: Mounted from library/nginx
6b133b4de5e6: Mounted from library/nginx
3d07a4a7eb2a: Mounted from library/nginx
756474215d29: Mounted from library/nginx
8d853c8add5d: Mounted from library/nginx
v1.0: digest: sha256:a84ea7df916cc272dcf50d3a3d2518ebb019c5a6852d2b1bf9d4332a879
1fef0 size: 1986
```

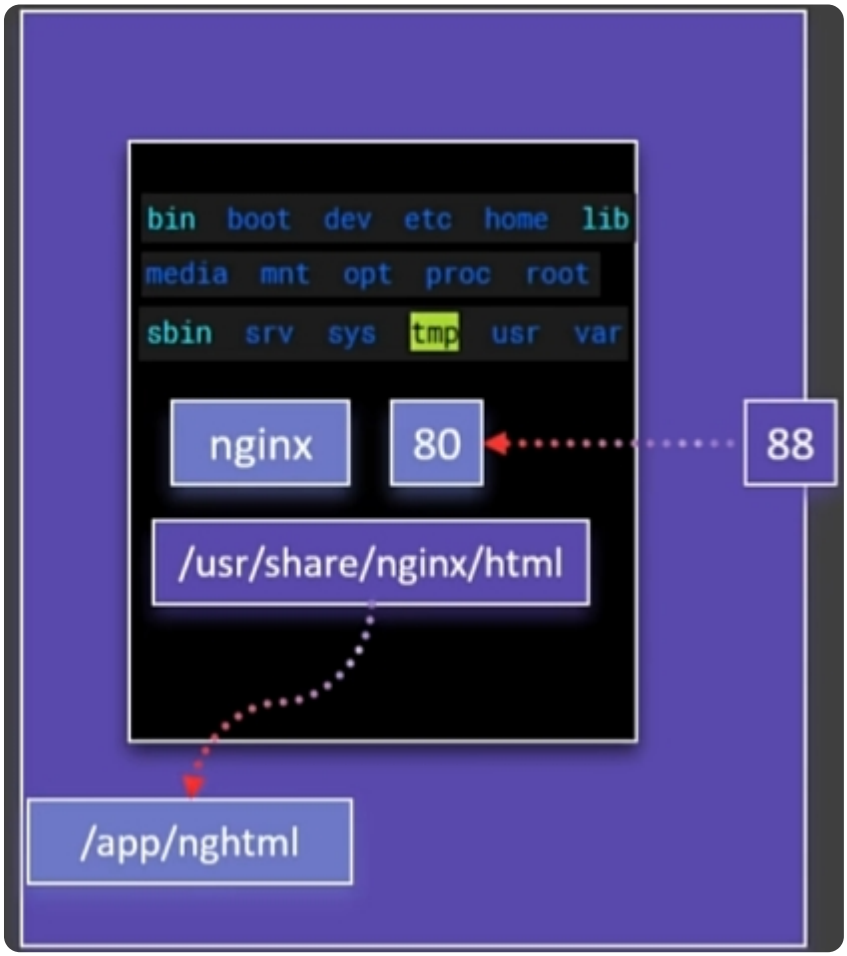


Docker 存储

熟悉目录挂载、数据卷，让容器数据不再丢失

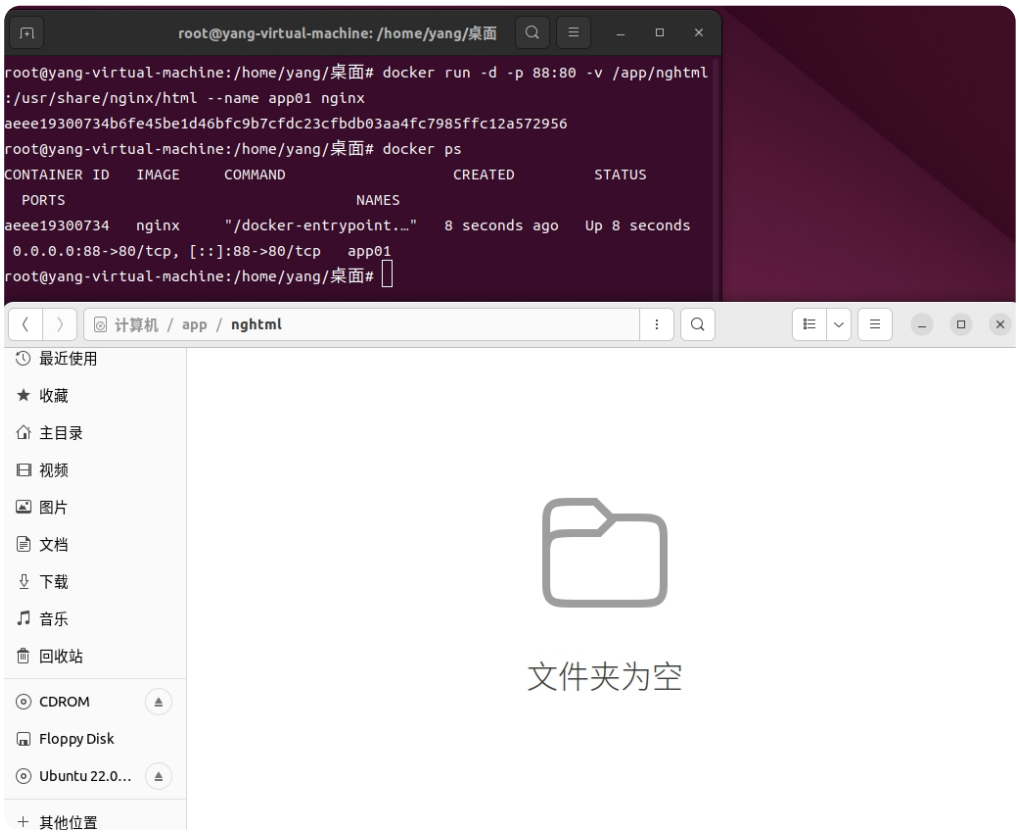
问题：直接进入docker容器中修改数据不容易，数据也容易丢失

目录挂载



使用-v命令进行目录挂载

```
1  docker run -d -p 88:80 -v
   /app/nghtml:/usr/share/nginx/html --name app01 nginx
2  //将容器目录挂载到本地的/app/nghtml
```



创建index.html文件，就可以直接在主机目录下进行修改,也不用担心数据丢失

卷映射

使用目录挂载，我们开始时缺少初始文件，这时我们就需要用到卷映射

```
root@yang-virtual-machine:/home/yang/桌面# docker run -d -p 88:80 -v /app/nghtml
:/usr/share/nginx/html -v ngconf:/etc/nginx --name app03 nginx
0184c93d3b1c4765bc009b4f7c39ca51b5879dd5e18bcd239f0a58022d00ba15
```

```
1  docker run -d -p 88:80 -v
   /app/nghtml:/usr/share/nginx/html -v ngconf:/etc/nginx
   --name app03 nginx
```

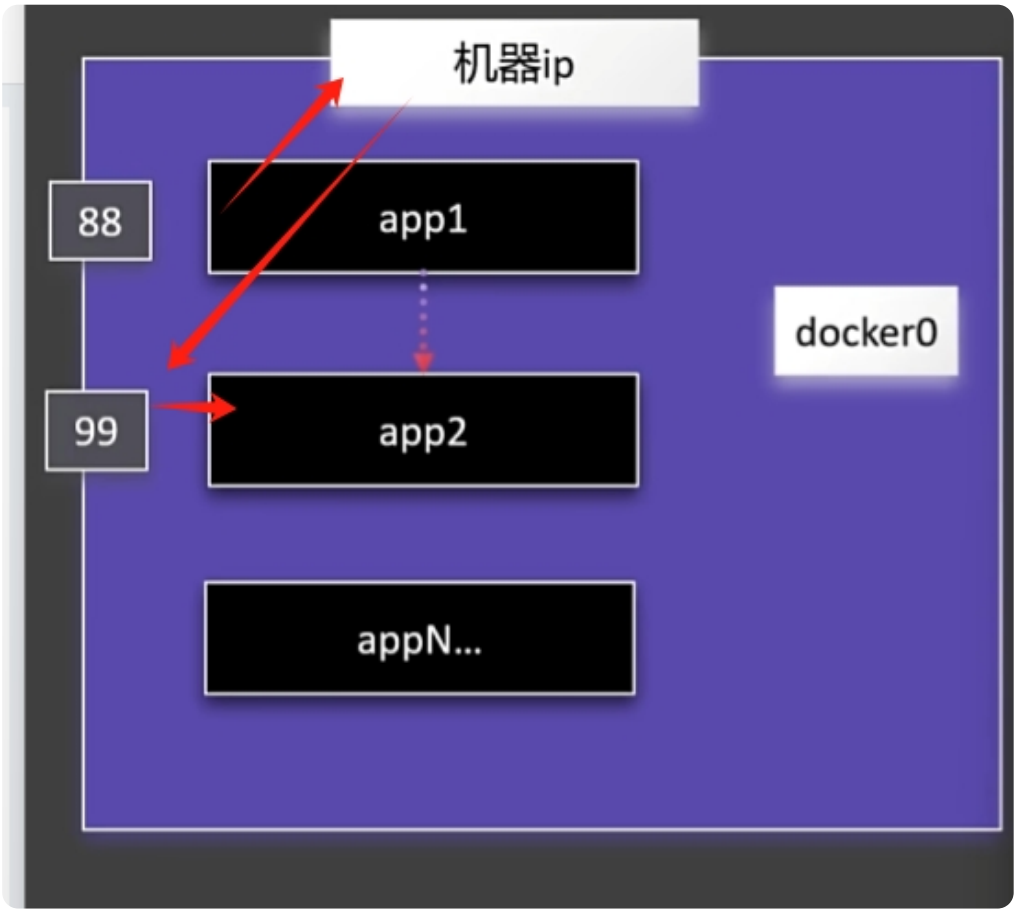
docker 将卷统一放在了 /var/lib/docker/volumes/目录下

Docker 网络

掌握网络机制，构建集群

自定义网络

docker多个容器间的互相访问，可以通过映射端口号来进行

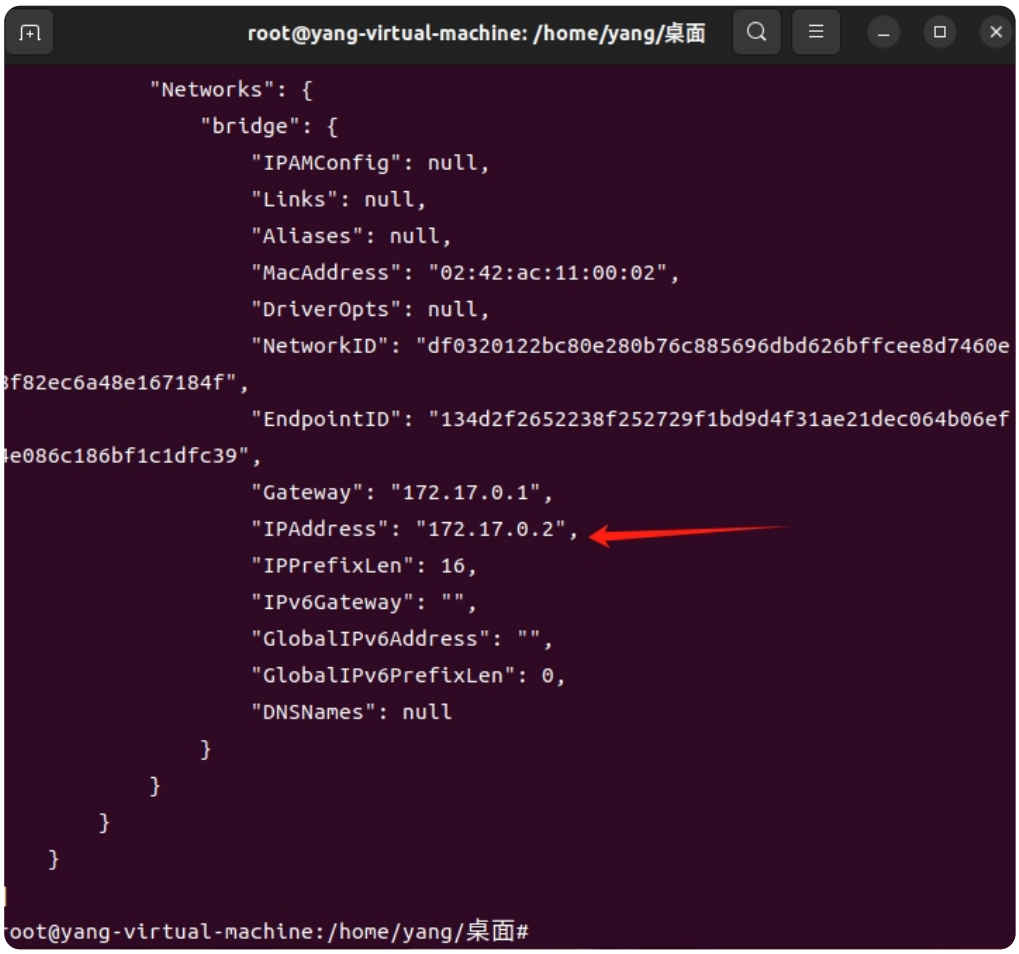


这显然不够方便快捷

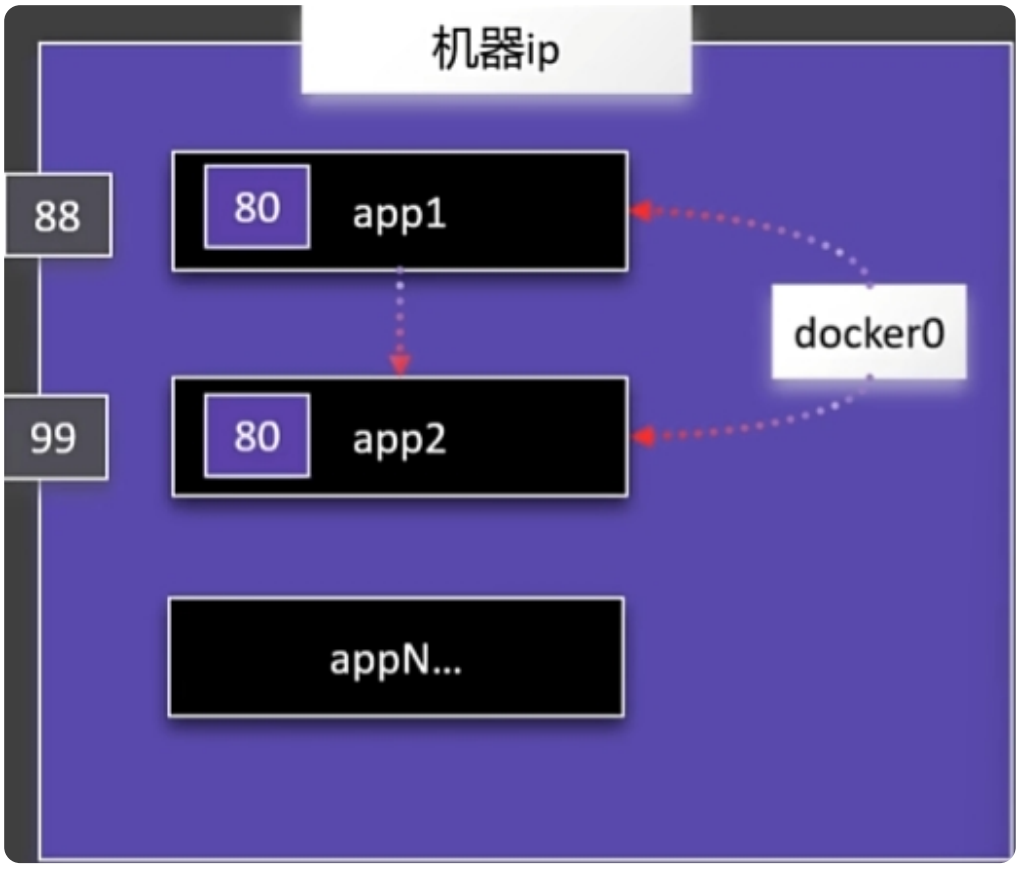
docker有一个docker0网络地址，我们可以通过ip addr 命令查看

```
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP gr
oup default
    link/ether 02:42:29:9e:43:98 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
    inet6 fe80::42:29ff:fe9e:4398/64 scope link
       valid_lft forever preferred_lft forever
```

docker 为每个容器都分配了不同的ip



我们可以通过这个ip地址来访问



可以使用任意容器ip+容器端口相互访问

但是我们又有新的问题：ip地址改变怎么办

解决的方法：自定义网络

自定义网络中，容器名就是稳定域名

可以通过域名来访问容器

docker network

```
root@yang-virtual-machine:/home/yang/桌面# docker network --help

Usage:  docker network COMMAND

Manage networks

Commands:
  connect     Connect a container to a network
  create      Create a network
  disconnect  Disconnect a container from a network
  inspect     Display detailed information on one or more networks
  ls          List networks
  prune       Remove all unused networks
  rm          Remove one or more networks

Run 'docker network COMMAND --help' for more information on a command.
```

通过docker network指令，可以创建网络(create)，查看网络(ls),查看网络细节(inspect)等等

```
1  docker network create mynet
```

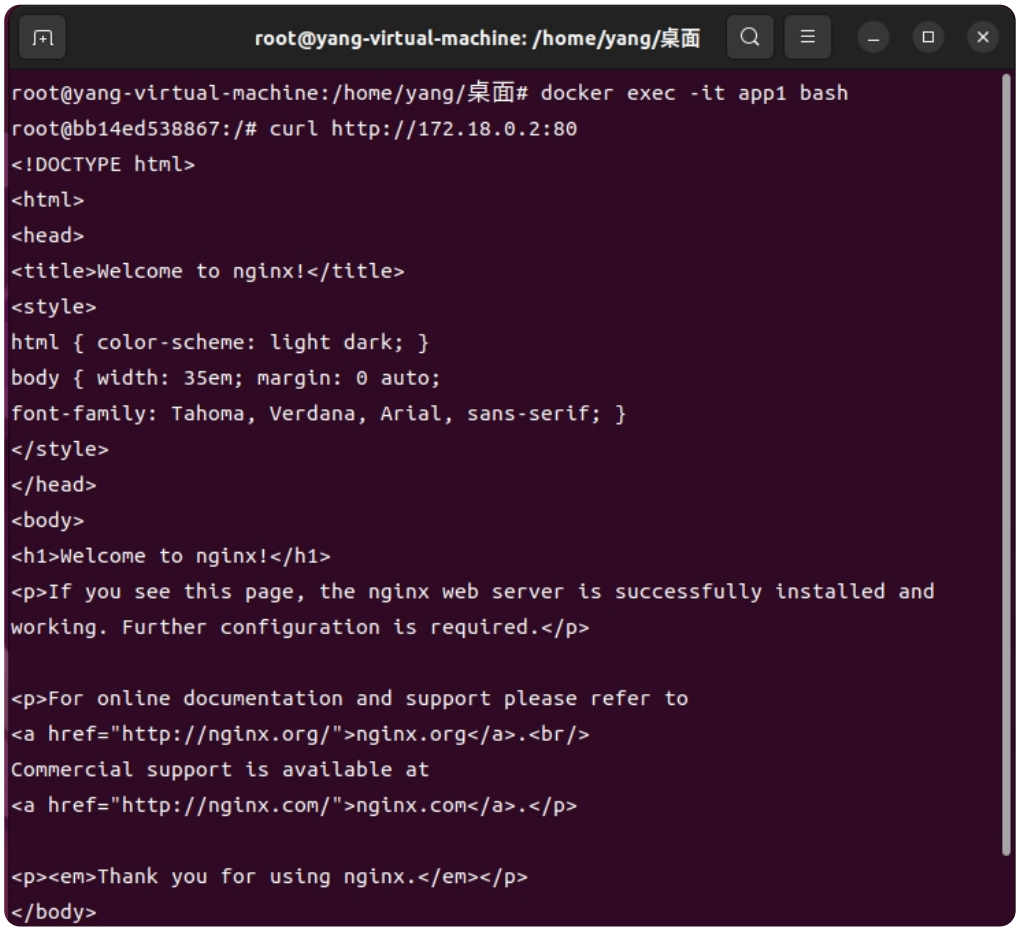
创建一个名为mynet的自定义网络

我们在创建容器时，使用--network参数即可使容器加入自定义网络

```
1  docker run -d -p 81:80 --name app1 --network mynet
   nginx
```

我们创建两个容器 app1 app2，并将两个容器都加入自定义网络

同时进入第二个容器访问第一个容器



也可以通过app2:80来访问

Docker Compose

批量管理容器

Docker Compose语法

顶级元素

name：名字

services：服务

networks：网络

volumes：卷

configs：配置

secrets：密钥

根据官方文档填写 yaml文件

使用 docker compose -f compose.yaml up -d可以后台批量启动容器

Dockerfile

构建自定义镜像