Docker

基本命令

```
# 查看日志
# docker logs 容器id
```

```
# 进入容器的两个命令
# docker exec -it 容器id /bin/bash
# docker attach 容器id

# ctrl + p + q 以运行方式退出
```

```
# 将容器中的内容拷贝到主机

# 进入容器内部
root@VM-8-2-debian:/home# docker attach 容器ID
[root@1950a6af833f /]# cd /home
[root@1950a6af833f home]# ls
[root@1950a6af833f home]# touch test.java
[root@1950a6af833f home]# ls
test.java
[root@1950a6af833f home]# exit
exit

# 将容器内文件拷贝到主机
root@VM-8-2-debian:/home# docker cp 1950a6af833f:/home/test.java /home
Successfully copied 1.536kB to /home
root@VM-8-2-debian:/home# ls
hello.java  lighthouse  test.java
```

```
root@VM-8-2-debian:/home#
```

```
# 创建一个nginx实例，并开发端口，其中3344为Linux主机的端口，80为容器的端口号
# docker run -d --name nginx01 -p 3344:80 nginx

# curl localhost:3344
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>

root@VM-8-2-debian:~# docker exec -it 82185ae5192f /bin/bash
root@82185ae5192f:/# whereis nginx
nginx: /usr/sbin/nginx /usr/lib/nginx /etc/nginx /usr/share/nginx
```

```
# 查看卷的情况
docker volume ls

# 匿名挂载
local     4f55a3b8d5c9c37be91cd8335dcbebbede402f24f59168e1b153d8b890aeeebf

# 具名挂载
```

```
root@VM-8-2-debian:/home# docker run -d -P --name mynginx -v test-nginx:/etc/nginx nginx
local     test-nginx

# 所有的docker容器内的卷，没有指定目录的情况下都在`/var/lib/docker/volumes/xxxxx/_data`
root@VM-8-2-debian:/home# docker inspect test-nginx
[
    {
        "CreatedAt": "2023-04-11T21:53:28+08:00",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/test-nginx/_data",
        "Name": "test-nginx",
        "Options": null,
        "Scope": "local"
    }
]

# 匿名挂载
-v 容器内目录

# 具名挂载
-v 名字：容器内目录

# 指定目录挂载
-v /主机目录：容器内目录


# 拓展：设置容器的权限，一旦设定，容器内部就有权限了
# ro:read only 只能通过宿主机来操作，容器内部无法操作
# rw:可读可写
docker run -d -P --name mynginx -v test-nginx:/etc/nginx:ro nginx
docker run -d -P --name mynginx -v test-nginx:/etc/nginx:rw nginx
```

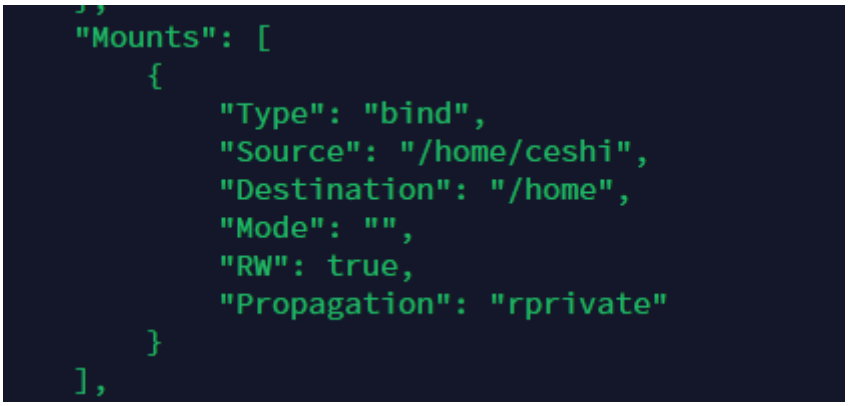
Commit 镜像

```
docker commit 提交容器成为一个新的镜像副本

docker commit -m="提交的描述信息" -a="作者" 容器id 目标镜像名:[TAG]
```

容器数据卷

方式一：直接使用命令来挂载 -v

```
docker run -it -v 主机目录: 容器内目录

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




```
# docker官方文档
# docker run --name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag


root@VM-8-2-debian:~# docker run -d -p 3310:3306 -v /home/mysql/conf:/etc/mysql/
```

删除容器，挂载到主机的数据卷依旧没有丢失

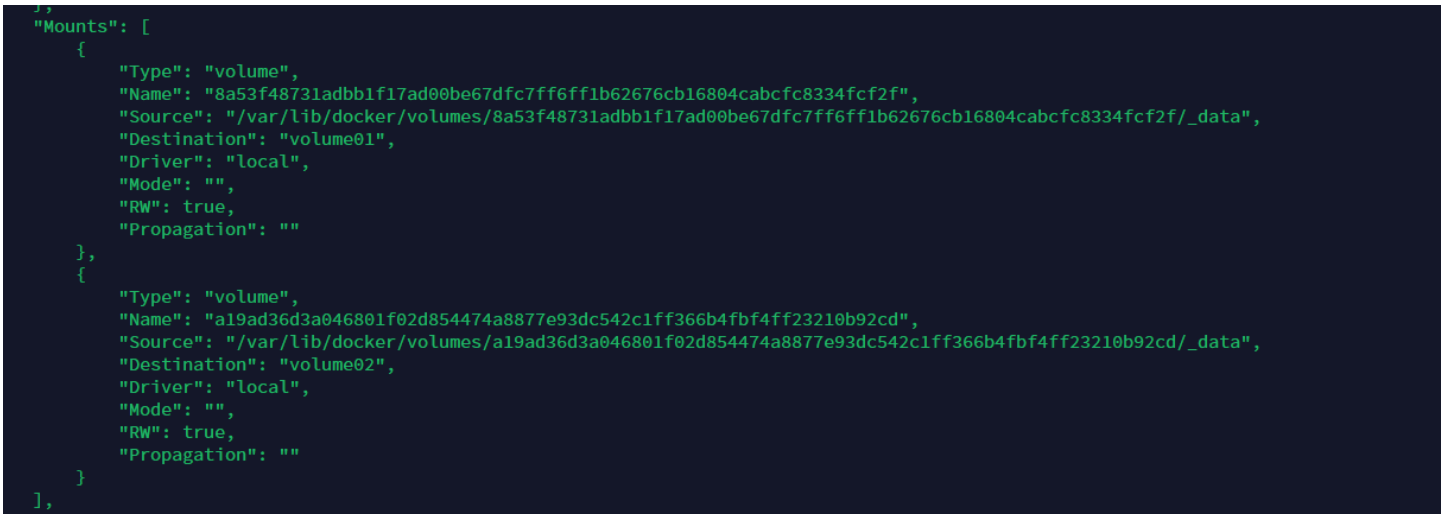
数据卷之DockerFile

```
# 创建dockerfile文件

root@VM-8-2-debian:~/docker-test-volume# cat dockerfile1
FROM centos

VOLUME ["volume01","volume02"]

CMD echo "-----end-----"
```

```
CMD /bin/bash
```

数据卷容器

```
# 构建容器
docker build -f dockerfile1 -t test/centos:v1.0 .

root@VM-8-2-debian:~/docker-test-volume# docker build -f dockerfile1 -t test/cen
[+] Building 0.2s (5/5) FINISHED
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load build definition from dockerfile1
 => => transferring dockerfile: 123B
 => [internal] load metadata for docker.io/library/centos:latest
 => [1/1] FROM docker.io/library/centos
 => exporting to image
 => => exporting layers
 => => writing image sha256:c5e35df12b062924cf4bf3406b7a35be11efba9e5693846a7ede
 => => naming to docker.io/test/centos:v1.0
```




```
# 创建第一个镜像
root@VM-8-2-debian:~# docker run -it --name docker01 c5e35df12b06

# 创建第二个镜像，挂载到第一个镜像
root@VM-8-2-debian:/# docker run -it --name docker02 --volumes-from docker01 c5e

```

结论：

数据卷容器的生命周期持续到没有容器使用为止，但持久化到本地就不会删除

DockerFile

构建步骤：

1、编写dockerfile文件

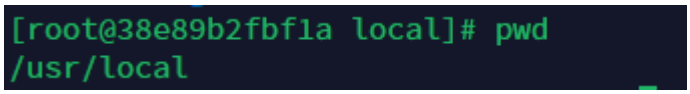
2、build 构建镜像

3、run 运行容器

4、push 发布镜像

DockerFile构建

```
1、编写docerfile文件

root@VM-8-2-debian:/dockerfile# cat mydockerfile-centos
FROM centos
MAINTAINER xiao<1341300669@qq.com>

ENV MYPATH /usr/local
WORKDIR $MYPATH

RUN yum -y install vim
RUN yum -y install net-tools

EXPOSE 80

CMD echo $MYPATH
CMD echo "---end---"
CMD /bin/bash

2、构建镜像
root@VM-8-2-debian:/dockerfile# docker build -f mydockerfile-centos -t mycentos
```



```
[root@38e89b2fbf1a local]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.4  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:ac:11:00:04  txqueuelen 0  (Ethernet)
        RX packets 10  bytes 796 (796.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

```
root@VM-8-2-debian:/dockerfile# docker history 84180daba616
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
84180daba616   11 minutes ago   CMD ["/bin/sh" "-c" "/bin/bash"]                0B        buildkit.dockerfile.v0
<missing>      11 minutes ago   CMD ["/bin/sh" "-c" "echo \"---end---\""]       0B        buildkit.dockerfile.v0
<missing>      11 minutes ago   CMD ["/bin/sh" "-c" "echo $MYPATH"]             0B        buildkit.dockerfile.v0
<missing>      11 minutes ago   EXPOSE map[80/tcp:{}]                           0B        buildkit.dockerfile.v0
<missing>      11 minutes ago   RUN /bin/sh -c yum -y install net-tools # bu…   177MB     buildkit.dockerfile.v0
<missing>      11 minutes ago   RUN /bin/sh -c yum -y install vim # buildkit    259MB     buildkit.dockerfile.v0
<missing>      13 minutes ago   WORKDIR /usr/local                              0B        buildkit.dockerfile.v0
<missing>      13 minutes ago   ENV MYPATH=/usr/local                           0B        buildkit.dockerfile.v0
<missing>      13 minutes ago   MAINTAINER xiao<1341300669@qq.com>              0B        buildkit.dockerfile.v0
<missing>      19 months ago    /bin/sh -c #(nop)  CMD ["/bin/bash"]            0B
<missing>      19 months ago    /bin/sh -c #(nop)  LABEL org.label-schema.sc…   0B
<missing>      19 months ago    /bin/sh -c #(nop) ADD file:b3ebbe8bd304723d4…   204MB
```

实战Tomcat

Dockerfile，官方默认大写D，build的时候不需要加 -f

```
root@VM-8-2-debian:~/tomcat# ls
apache-tomcat-10.1.7.tar.gz  jdk-20_linux-x64_bin.tar.gz
root@VM-8-2-debian:~/tomcat# ll
total 198928
-rw-r--r-- 1 root root  12143798 Apr 12 13:46 apache-tomcat-10.1.7.tar.gz
-rw-r--r-- 1 root root 191554581 Apr 12 11:23 jdk-20_linux-x64_bin.tar.gz
root@VM-8-2-debian:~/tomcat# touch readme.txt
root@VM-8-2-debian:~/tomcat# ll
total 198928
-rw-r--r-- 1 root root  12143798 Apr 12 13:46 apache-tomcat-10.1.7.tar.gz
-rw-r--r-- 1 root root 191554581 Apr 12 11:23 jdk-20_linux-x64_bin.tar.gz
-rw-r--r-- 1 root root         0 Apr 12 13:48 readme.txt
root@VM-8-2-debian:~/tomcat# vim Dockerfile
```

1、编写Dockerfile

```
root@VM-8-2-debian:~/tomcat# cat Dockerfile
FROM centos:7

MAINTAINER xiao<1341300669@qq.com>

COPY readme.txt /usr/local/readme.txt

ADD jdk-8u202-linux-x64.tar.gz /usr/local/
ADD apache-tomcat-10.1.7.tar.gz /usr/local/

RUN yum -y install vim

ENV MYPATH /usr/local
WORKDIR $MYPATH

ENV JAVA_HOME /usr/local/jdk1.8.0_202
ENV CLASSPATH $JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
ENV CATALINA_HOME /usr/local/apache-tomcat-10.1.7
ENV CATALINA_BASH /usr/local/apache-tomcat-10.1.7
ENV PATH $PATH:$JAVA_HOME/bin:$CATALINA_HOME/lib:$CATALINA_HOME/bin

EXPOSE 8080

CMD /usr/local/apache-tomcat-10.1.7/bin/startup.sh && tail -F /usr/local/apache-
```

2、构建成功

```
root@VM-8-2-debian:~/tomcat# docker build -t diytomcat .
[+] Building 29.0s (11/11) FINISHED
 => [internal] load build definition from Dockerfile                       0.0s
 => => transferring dockerfile: 677B                                       0.0s
 => [internal] load .dockerignore                                          0.0s
 => => transferring context: 2B                                            0.0s
 => [internal] load metadata for docker.io/library/centos:7                0.4s
 => CACHED [1/6] FROM docker.io/library/centos:7@sha256:9d4bcbbb213dfd745b58b  0.0s
 => [internal] load build context                                          2.2s
 => => transferring context: 206.24MB                                      2.2s
 => [2/6] COPY readme.txt /usr/local/readme.txt                            0.1s
 => [3/6] ADD jdk-8u202-linux-x64.tar.gz /usr/local/                       3.3s
 => [4/6] ADD apache-tomcat-10.1.7.tar.gz /usr/local/                      0.4s
 => [5/6] RUN yum -y install vim                                          19.1s
 => [6/6] WORKDIR /usr/local                                               0.1s
 => exporting to image                                                     3.4s
 => => exporting layers                                                    3.4s
 => => writing image sha256:5acea14640b5c28d00943b2d021a587e5855cd88fe4ab4fdd  0.0s
 => => naming to docker.io/library/diytomcat                               0.0s
```

🏄 3、启动容器

```
root@VM-8-2-debian:~/tomcat# docker run -d -p 9090:8080 --name xiaotomcat -v /tomcat/test:/usr/local/apache-tomcat-10.1.7/webapps/test -v/tomcat/tomcatlogs/:/usr/local/apache-tomcat-10.1.7/logs diytomcat
7a449cd714808c34146519f070d8eaea4b440e0cabc86f910ddfb5d1edba4095
```

Docker 网络

服务器ip

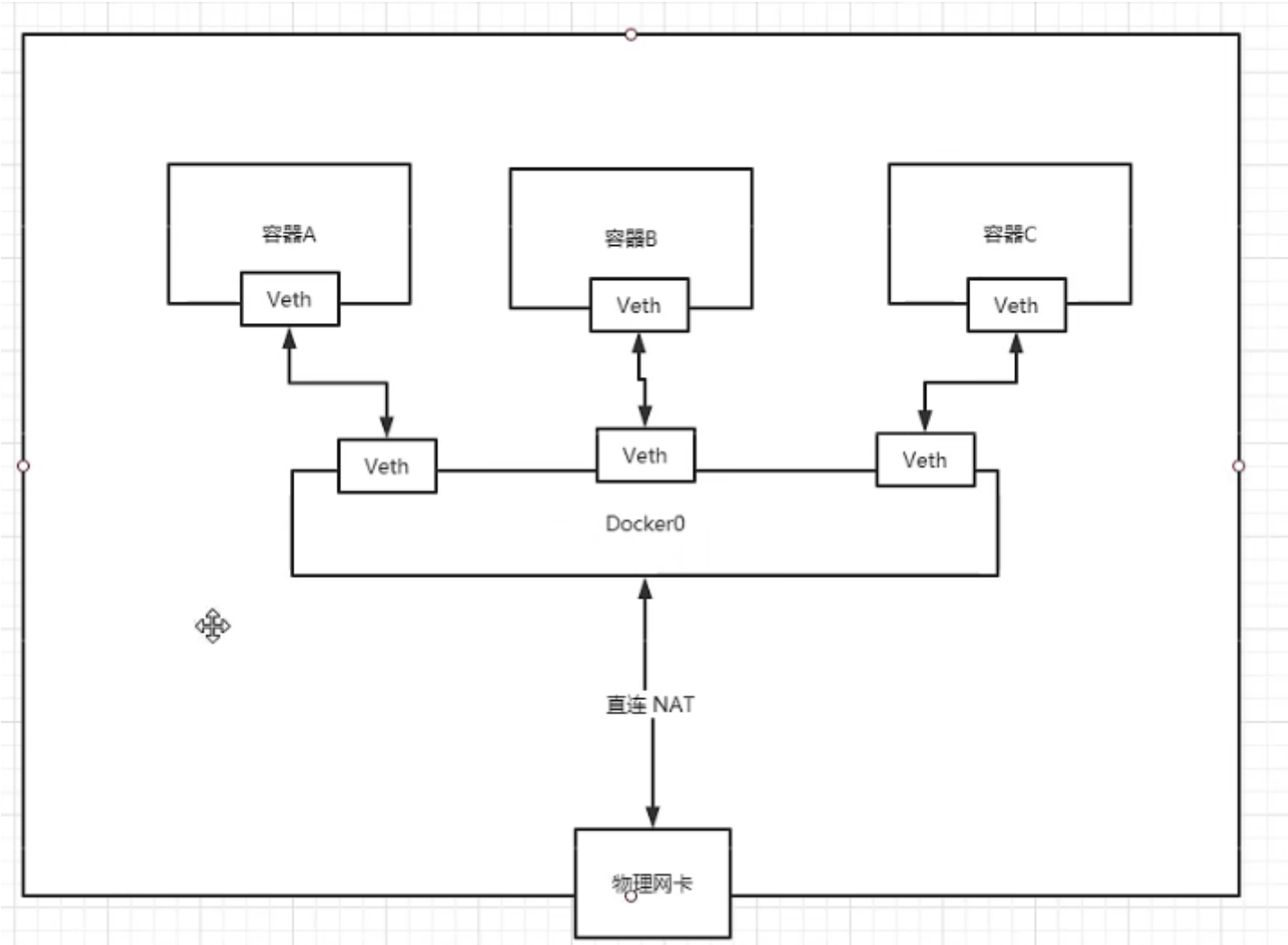
```
root@VM-8-2-debian:~/tomcat# ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 52:54:00:a2:2d:c4 brd ff:ff:ff:ff:ff:ff
    inet 10.0.8.2/22 brd 10.0.11.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:fea2:2dc4/64 scope link
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:9b:1a:a3:9a 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::42:9bff:fe1a:a39a/64 scope link
       valid_lft forever preferred_lft forever
```

```
# 容器带来的网卡都是一对一对的，
# evth-pair充当一个桥梁，连接各种虚拟设备
# tomcat01和tomcat02可以相互ping通
```

容器内部ip

```
root@bec9f4a98b48:/etc/apt# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
12: eth0@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group defaul
t
    link/ether 02:42:ac:11:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.17.0.2/16 brd 172.17.255.255 scope global eth0
       valid_lft forever preferred_lft forever
```

🎉 Docker中所有端口都是虚拟的，转发效率高



自定义网络

```
root@VM-8-2-debian:~/tomcat# docker network create --driver bridge --subnet 192.
9ba99a9c6cec07ec08fa713ac9e878dac26920893e4c9ffdc2eb2528b8ecaa1d
root@VM-8-2-debian:~/tomcat# docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
2243d3cc628c   bridge    bridge    local
```

```
6 689cadb14a0f   host      host      local
7 9ba99a9c6cec   mynet     bridge    local
8 94506ee32c1e   none      null      local
```

```
root@VM-8-2-debian:~/tomcat# docker network inspect mynet
[
    {
        "Name": "mynet",
        "Id": "9ba99a9c6cec07ec08fa713ac9e878dac26920893e4c9ffdc2eb2528b8ecaa1d",
        "Created": "2023-04-12T17:08:18.334098687+08:00",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "192.168.0.0/16",
                    "Gateway": "192.168.0.1"
                }
            ]
        },
        "Internal": false,
        "Attachable": false,
        "Ingress": false,
        "ConfigFrom": {
            "Network": ""
        },
        "ConfigOnly": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

网络连通

```
root@VM-8-2-debian:~/tomcat# docker network connect --help

Usage:  docker network connect [OPTIONS] NETWORK CONTAINER

Connect a container to a network

Options:
      --alias strings           Add network-scoped alias for the container
      --driver-opt strings      driver options for the network
      --ip string               IPv4 address (e.g., "172.30.100.104")
      --ip6 string              IPv6 address (e.g., "2001:db8::33")
      --link list               Add link to another container
      --link-local-ip strings   Add a link-local address for the container
```

一个容器两个ip

Redis实战

```
root@VM-8-2-debian:~# for port in $(seq 1 6); \
> do \
> mkdir -p /mydata/redis/node-${port}/conf
> touch /mydata/redis/node-${port}/conf/redis.conf
> cat << EOF >/mydata/redis/node-${port}/conf/redis.conf
> port 6379
> bind 0.0.0.0
> cluster-enabled yes
> cluster-config-file nodes.conf
> cluster-node-timeout 5000
> cluster-announce-ip 172.38.0.1${port}
> cluster-announce-port 6379
> cluster-announce-bus-port 16379
> appendonly yes
> EOF
> done
root@VM-8-2-debian:~# cd /mydata/
root@VM-8-2-debian:/mydata# ls
redis
root@VM-8-2-debian:/mydata# cd redis/
root@VM-8-2-debian:/mydata/redis# ls
node-1  node-2  node-3  node-4  node-5  node-6
```

```
# 开启第一个node

root@VM-8-2-debian:/mydata/redis/node-1/conf# docker run -p 6371:6379 -p 16371:1
> -v /mydata/redis/node-1/data:/data \
> -v /mydata/redis/node-1/conf/redis.conf:/etc/redis/redis.conf \
> -d --net redis --ip 172.38.0.11 redis:5.0.9-alpine3.11 redis-server /etc/redis

# 开启6个node


docker run -p 6376:6379 -p 16376:16379 --name redis-6 \
-v /mydata/redis/node-6/data:/data \
-v /mydata/redis/node-6/conf/redis.conf:/etc/redis/redis.conf \
-d --net redis --ip 172.38.0.16 redis:5.0.9-alpine3.11 redis-server /etc/redis/r
```

```
# 集群创建配置

root@VM-8-2-debian:/mydata/redis/node-1/conf# docker exec -it redis-1 /bin/sh
/data # redis-cli --cluster create 172.38.0.11:6379 172.38.0.12:6379 172.38.0.13
>>> Performing hash slots allocation on 6 nodes...
Master[0] -> Slots 0 - 5460
Master[1] -> Slots 5461 - 10922
Master[2] -> Slots 10923 - 16383
Adding replica 172.38.0.15:6379 to 172.38.0.11:6379
Adding replica 172.38.0.16:6379 to 172.38.0.12:6379
Adding replica 172.38.0.14:6379 to 172.38.0.13:6379
M: f1c21e2f053cd5209f89a7dbe280b77ce59712f9 172.38.0.11:6379
   slots:[0-5460] (5461 slots) master
M: 0aaee2fd5501f7f34b6c157c24a0ff904651411c 172.38.0.12:6379
   slots:[5461-10922] (5462 slots) master
M: e93c8e7384a032298dad5625303b1f58e93b832a 172.38.0.13:6379
   slots:[10923-16383] (5461 slots) master
S: 1ee349482b9f192f53b739c31e520872c8f2207a 172.38.0.14:6379
   replicates e93c8e7384a032298dad5625303b1f58e93b832a
S: 0e64b815b23404c8434dfc684d7776f568af6ca1 172.38.0.15:6379
   replicates f1c21e2f053cd5209f89a7dbe280b77ce59712f9
S: 90d3427ef39c3356b4e4be47c72b6af8055c2a34 172.38.0.16:6379
   replicates 0aaee2fd5501f7f34b6c157c24a0ff904651411c
Can I set the above configuration? (type 'yes' to accept): yes
>>> Nodes configuration updated
>>> Assign a different config epoch to each node
>>> Sending CLUSTER MEET messages to join the cluster
Waiting for the cluster to join
...
>>> Performing Cluster Check (using node 172.38.0.11:6379)
M: f1c21e2f053cd5209f89a7dbe280b77ce59712f9 172.38.0.11:6379
   slots:[0-5460] (5461 slots) master
   1 additional replica(s)
M: 0aaee2fd5501f7f34b6c157c24a0ff904651411c 172.38.0.12:6379
   slots:[5461-10922] (5462 slots) master
   1 additional replica(s)
S: 1ee349482b9f192f53b739c31e520872c8f2207a 172.38.0.14:6379
   slots: (0 slots) slave
   replicates e93c8e7384a032298dad5625303b1f58e93b832a
S: 90d3427ef39c3356b4e4be47c72b6af8055c2a34 172.38.0.16:6379
   slots: (0 slots) slave
   replicates 0aaee2fd5501f7f34b6c157c24a0ff904651411c
M: e93c8e7384a032298dad5625303b1f58e93b832a 172.38.0.13:6379
   slots:[10923-16383] (5461 slots) master
   1 additional replica(s)
S: 0e64b815b23404c8434dfc684d7776f568af6ca1 172.38.0.15:6379
   slots: (0 slots) slave
```

```
   replicates f1c21e2f053cd5209f89a7dbe280b77ce59712f9
[OK] All nodes agree about slots configuration.
>>> Check for open slots...
>>> Check slots coverage...
[OK] All 16384 slots covered.
```