1.1 容器存储

1.1.1 volume

步骤 1　创建一个卷，并挂载个一个 httpd 容器。

```
[root@localhost ~]# docker run -d -p 8080:80 -v /usr/local/apache2/htdocs httpd
```

```
[root@localhost ~]# docker run -d -p 8080:80 -v /usr/local/apache2/htdocs httpd
0314232b0a05b4c2abf7affbdc16ac0aa651b825c1073b5f5d08ca468657030d
```

步骤 2　查看 volume 信息。

```
[root@localhost ~]# docker volume ls
```

```
[root@localhost ~]# docker volume ls
DRIVER              VOLUME NAME
local               3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc
```

步骤 3　查看容器的 volume 挂载信息，同时得到 volume 路径。Type=volume。

```
docker inspect 0314232b0a05
```

```
"Mounts": [
    {
        "Type": "volume",
        "Name": "3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc",
        "Source": "/var/lib/docker/volumes/3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc/_data",
        "Destination": "/usr/local/apache2/htdocs",
        "Driver": "local",
        "Mode": "",
        "RW": true,
        "Propagation": ""
    }
],
```

步骤 4　查看 volume 中的数据。

```
[root@localhost ~]# cd /var/lib/docker/volumes/3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc/_data
[root@localhost _data]# ls
index.html
[root@localhost _data]# cat index.html
<html><body><h1>It works!</h1></body></html>
```

步骤 5　查看容器中相应的数据。结果：容器中的数据=volume 中数据。

```
curl 127.0.0.1:8080
```

```
[root@localhost ~]# curl 127.0.0.1:8080
<html><body><h1>It works!</h1></body></html>
```

步骤 6　进入容器，更新 index.html 文件内容。

```
echo “update the index” > index.html
```

```
[root@localhost ~]# docker exec -it 0314232b0a05 bash
root@0314232b0a05:/usr/local/apache2# cd htdocs
root@0314232b0a05:/usr/local/apache2/htdocs# echo "update the index" > index.html
```

步骤 7　再次查看 volume 中的内容，已同步跟新。

```
[root@localhost ~]# cat /var/lib/docker/volumes/3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc/_data/index.html
update the index
```

由此，实现容器和 Host 的数据共享。

步骤 8 强制删除容器。

```
[root@localhost ~]# docker rm -f 0314232b0a05
[root@localhost ~]# docker rm -f 0314232b0a05
0314232b0a05
```

步骤 9 查看 volume 中的数据，依旧存在。

```
[root@localhost ~]# cat /var/lib/docker/volumes/3d5603513b758599b8695834235b3dc174379552dda002944bbbf013db2d05fc/_data/index.html
update the index
```

1.1.2 bind mount

步骤 1 在宿主机上创建/root/htdocs 目录，以只读的方式挂载给一个名为 httpd1 的 httpd 容器，映射端口 8081。

```
[root@localhost ~]# docker run --name httpd1 -d -p 8081:80 -v
/root/htdocs:/usr/local/apache2/htdocs:ro httpd
[root@localhost ~]# docker run --name httpd1 -d -p 8081:80 -v /root/htdocs:/usr/local/apache2/htdocs:ro httpd
ee885d10fea155fe31311262219030b20b4eea6e2f2f2ca28fe0b1ad5ff4c91a
```

步骤 2 查看容器挂载信息。Type=bind。

```
[root@localhost ~]# docker inspect httpd1
"Mounts": [
    {
        "Type": "bind",
        "Source": "/root/htdocs",
        "Destination": "/usr/local/apache2/htdocs",
        "Mode": "ro",
        "RW": false,
        "Propagation": "rprivate"
    }
],
```

步骤 3 在宿主机上更新 index.html 文件数据，发现 httpd1 容器中的数据也一起更新。

```
[root@localhost ~]# echo "bind mount" > /root/htdocs/index.html
[root@localhost ~]# echo "bind mount" > /root/htdocs/index.html
[root@localhost ~]# curl 127.0.0.1:8081
bind mount
```

由此，实现容器和 Host 的数据共享。

步骤 4 进入 httpd1 容器中更新 index.html 文件数据，提示 Read-only。

```
[root@localhost ~]# docker exec -it httpd1 bash
root@ee885d10fea1:/usr/local/apache2# echo "read only ?" > /usr/local/apache2/htdocs/index.html
bash: /usr/local/apache2/htdocs/index.html: Read-only file system
```

步骤 5　将宿主机/root/htdocs 挂载给名为 httpd2 的 httpd 容器，映射端口 8082，不设置 ro。

```
[root@localhost ~]# docker run --name httpd2 -d -p 8082:80 -v
/root/htdocs:/usr/local/apache2/htdocs httpd
```

```
[root@localhost ~]# docker run --name httpd2 -d -p 8082:80 -v /root/htdocs:/usr/local/apache2/htdocs httpd
8c4caa765582a3ccbdb7380a397986436458def89a805b46a6bed6b336f689c4
```

步骤 6　进入 httpd2 容器中更新 index.html 文件数据。

```
[root@localhost ~]# docker exec -it httpd2 bash
```

```
[root@localhost ~]# docker exec -it httpd2 bash
root@8c4caa765582:/usr/local/apache2# echo "i can do this" > /usr/local/apache2/htdocs/index.html
root@8c4caa765582:/usr/local/apache2# exit
```

步骤 7　分别查看宿主机、httpd1 容器、httpd2 容器中数据。三者数据一致。

```
[root@localhost ~]# cat /root/htdocs/index.html
i can do this
[root@localhost ~]# curl 127.0.0.1:8081
i can do this
[root@localhost ~]# curl 127.0.0.1:8082
i can do this
```

由此实现容器间的数据共享。

步骤 8　为方便后续实验，删除本小节中的容器。

```
docker kill
docker rm
```

1.1.3 volume container

步骤 1　创建一个名为 myvolume1 的空卷。

```
[root@localhost ~]# docker volume create myvolume1
```

```
[root@localhost ~]# docker volume create myvolume1
myvolume1
[root@localhost ~]# docker volume ls
DRIVER              VOLUME NAME
local               myvolume1
```

步骤 2　将 myvolume1 挂载一个名为 vc 的容器。

```
[root@localhost ~]# docker create --name vc -v
myvolume1:/usr/local/apache2/htdocs httpd
```

```
[root@localhost ~]# docker create --name vc -v myvolume1:/usr/local/apache2/htdocs httpd
8a084c7dd8032165beee5df8c831a7bb31569a44eab459d558457d004c40ab3f
```

步骤 3　运行两个 httpd 容器，都引用容器 vc 中的数据，分别命名为 httpd3 和 httpd4。

```
[root@localhost ~]# docker run --name httpd3 -d -p 1003:80 --volumes-from vc
httpd
[root@localhost ~]# docker run --name httpd4 -d -p 1004:80 --volumes-from vc
httpd
```

```
[root@localhost ~]# docker run --name httpd3 -d -p 1003:80 --volumes-from vc httpd
4c7cb65680aba437393f9130201375b21b72965688366ac1843a58c356e894fe
[root@localhost ~]# docker run --name httpd4 -d -p 1004:80 --volumes-from vc httpd
49c258d3fb68c26094d82462d2bfbf576a0999069f1cfb61c4806b1fa67eae61
```

步骤 4　进入 httpd3 中，更新 index.html 数据。

```
[root@localhost ~]# docker exec -it httpd3 bash
```

```
[root@localhost ~]# docker exec -it httpd3 bash
root@4c7cb65680ab:/usr/local/apache2# echo "Sharing with httpd4" > /usr/local/apache2/htdocs/index.html
root@4c7cb65680ab:/usr/local/apache2# exit
exit
```

步骤 5　查看 httpd4 容器中数据，已同步跟新。

```
[root@localhost ~]# curl 127.0.0.1:1004
```

```
[root@localhost ~]# curl 127.0.0.1:1004
Sharing with httpd4
```

步骤 6　为方便后续实验，删除本小节中的容器。

```
docker kill
docker rm
```