Docker虚拟化容器应用与实战

第3章 Docker容器数据卷

本章目标

- commit镜像
- 什么是容器数据卷
- 使用数据卷
- 实战安装Mysql
- 具名挂载和匿名挂载

commit镜像

- docker commit 提交容器成为一个新的副本
- docker commit -m="描述信息" -a="作者" 容器id 目标镜像名:[TAG]

实战测试

1、启动一个默认的centos
- docker run -it --name centos1 centos /bin/bash

2、发现这个默认的centos 是没有vim和ifconfig命令的，进入镜像安装命令
#安装vim和net-tools
yum install -y net-tools
yum install -y vim
3、将操作过的容器通过commit为一个镜像！我们以后就使用我们修改过的镜像即可，这就是我们自己 的一个修改的镜像。
- docker commit -m="描述信息" -a="作者" 容器id 目标镜像名:[TAG]

- docker commit -a="zhangsan" -m="add command" 容器id mycentos:1

实战测试

1、启动一个默认的tomcat
- docker run -d -p 8080:8080 tomcat

2、发现这个默认的tomcat 是没有webapps应用，官方的镜像默认webapps下面是没有文件的!
- docker exec -it 容器id

3、拷贝文件进去

4、将操作过的容器通过commit调教为一个镜像！我们以后就使用我们修改过的镜像即可，这就是我们自己 的一个修改的镜像。
- docker commit -m="描述信息" -a="作者" 容器id 目标镜像名:[TAG]

- docker commit -a="zhangsan" -m="add webapps app" 容器id tomcat02:1

第3章 Docker容器数据卷

什么是容器数据卷

- 如果数据都在容器中，那么我们将容器删除，数据就会丢失！
- 需求：数据可以持久化
- MySQL，容器删除了，删库跑路！
- 需求：MySQL数据可以存储在本地！
- 容器之间可以有一个数据共享的技术！Docker容器中产生的数据，同步到本地！
- 这就是卷技术！目录的挂载，将我们容器内的目录，挂载到Linux上面
- **总结一句话：容器的持久化和同步操作！容器间也是可以数据共享的！**

使用容器数据卷

直接使用命令挂载 -v （数据同步到本地）

测试文件的同步

- -v, 给容器挂载存储卷，挂载到容器的某个目录
- docker run -it -v 宿主机目录:容器内目录 -p 主机端口:容器内端口
- docker run -it -v /home/ceshi:/home centos /bin/bash

#通过 docker inspect 容器id 查看

测试文件同步

再来测试！

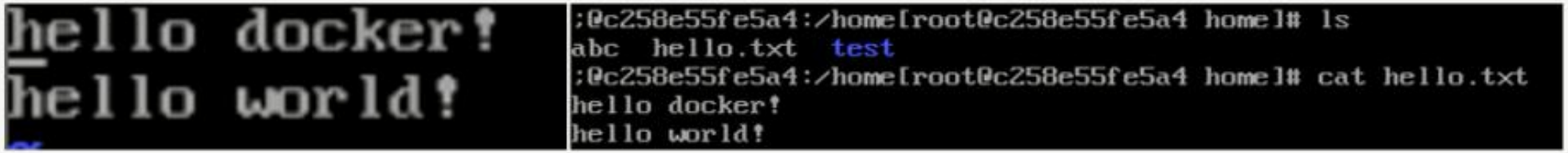
1、停止容器

2、宿主机修改文件

3、启动容器

4、容器内的数据依旧是同步的

好处：我们以后修改只需要在本地修改即可，容器内会自动同步！



实战测试

```
 docker pull mysql:5.7
# 获取mysql镜像

# 运行容器,需要做数据挂载

#安装启动mysql，需要配置密码的，这需要注意！

#启动我们 -d 后台运行 -p 端口映射 -v 卷挂载 -e 环境配置 -- name 容器名字

 docker run -d -p 3306:3306 -v /home/mysql/conf:/etc/mysql/conf.d -v
/home/mysql/data:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=123456 --name
mysql01 mysql:5.7
```

第3章 Docker容器数据卷

本章目标

- commit镜像
- 什么是容器数据卷
- 使用数据卷
- 实战安装Mysql
- 具名挂载和匿名挂载

数据卷容器

使用 --volumes-from挂载（容器间数据同步）

- --volumes-from list：从指定容器中装载卷

实战测试(容器间数据同步)

docker run -it -v /volumes:/home --name centos01 centos /bin/bash

docker run -it --name centos02 --volumes-from centos01 centos /bin/bash

docker run -it --name centos03 --volumes-from centos01 centos /bin/bash

在centos1的/home目录下建立hello.txt文档，查看centos02和centos03的/home文件夹，然后停止centos02容器，在centos01容器的/home下建立abc-test文件，查看centos03的/home文件夹，然后启动centos02容器，再查看centos02的/home文件夹

实战测试

假设我们将容器删除：

```
[root@localhost data]# docker rm -f mysql02
mysql02
[root@localhost data]# docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[root@localhost data]# _
```

发现，我们挂载到本地的数据卷依旧没有丢失，这就实现了容器数据持久化功能。

具名和匿名挂载

- 所有的docker容器内的卷，没有指定目录的情况下都是在/var/lib/docker/volumes/xxxx/_data 下

```
➜ ~ docker volume inspect juming-nginx
[
    {
        "CreatedAt": "2020-05-16T11:32:01+08:00",
        "Driver": "local",
        "Labels": null,
        "Mountpoint": "/var/lib/docker/volumes/juming-nginx/_data",
        "Name": "juming-nginx",
        "Options": null,
        "Scope": "local"
    }
]
```

具名和匿名挂载

匿名挂载

- -v 容器内路径!
- docker run -d -P --name nginx01 -v /etc/nginx nginx
- docker volume ls

这里发现，这种就是匿名挂载，我们在 -v后只写了容器内的路径，没有写容器外的路径!

具名挂载

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

通过 -v 卷名：容器内路径
查看一下这个卷
- docker volume ls

具名和匿名挂载

```
→ ~ docker run -d -P --name nginx05 -v /home/nginx05:/etc/nginx/conf.d nginx   启动挂载到具体目录
863196e8441ce2fd443c8dd8a04eaba053bef007cf31845671cda1ab87cbe793
→ ~ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
           NAMES
863196e8441c        nginx               "nginx -g 'daemon of…"   4 seconds ago       Up 3 seconds        0.0.0.0:3277
3->80/tcp   nginx05
bc2cc955f71d        nginx               "nginx -g 'daemon of…"   3 minutes ago       Up 3 minutes        0.0.0.0:3277
2->80/tcp   nginx04
dbb8319f1788        nginx               "nginx -g 'daemon of…"   9 minutes ago       Up 9 minutes        0.0.0.0:3277
0->80/tcp   nginx02
→ ~ docker volume ls 查看不到
DRIVER              VOLUME NAME
local               juming-nginx
→ ~
```

三种挂载： 匿名挂载、具名挂载、指定路径挂载

#指定路径挂载

- -v 宿主机路径：容器内路径

#匿名挂载

- -v 卷名：容器内路径

#具名挂载

- -v 宿主机内卷名：容器内路径

#指定路径挂载 docker volume ls 是查看不到的

小结

- commit镜像
- 什么是容器数据卷
- 使用数据卷
- 实战安装Mysql
- 具名挂载和匿名挂载

谢谢观赏