目录

[镜像命令 3](#_Toc21980)

[容器命令 3](#_Toc22887)

[Docker 镜像详解 5](#_Toc12992)

[Docker容器数据卷 5](#_Toc8567)

[Docker数据卷用 -v 命令添加 5](#_Toc12539)

[Docker容器数据卷用 DockerFile添加 6](#_Toc19365)

[容器数据卷 7](#_Toc19321)

**Docker学习笔记**

Docker : 可以将 一些应用 软件(mysql ) ,打包成镜像,其他服务器系统,若安装了docker,则可以 将这个镜像环境迁移过来

Docker 可以解决 开发 环境和部署环境 不统一的问题.可以将 开发的环境(配置,数据等等) 一起 搬过去

Docker的三要素**: 仓库 镜像 容器**, docker 是一项 虚拟化技术

Docker 镜像就是一个可读的模板,镜像可以用来创建docker容器,一个镜像可以创建多个docker容器

Docker 仓库 是 集中 存放镜像的场所 , 阿里云就是一个 国内的docker仓库

从仓库 下载镜像 到本地, 镜像在容器内运行 ,一个镜像 可以生成多个容器

#### 阿里云镜像加速

(因为 默认连接的是 外网 下载镜像, 配置之后就可以 从阿里云 下载镜像)

<https://cr.console.aliyun.com/cn-hangzhou/new> :

1 到该网址注册一个自己的阿里云账户(可复用淘宝账号)

2 获得加速器地址 https://aatmuhx4.mirror.aliyuncs.com



3 配置本机Docker运行镜像加速器地址

centOS 7 配置文件: (centOS 7 一下 不是这个配置文件)

/etc/docker/daemon.json



5 检查配置是否成功

Docker 底层运行原理:

Docker 虚拟化技术. 干掉了 虚拟的硬件系统, 由docker 引擎替代

<https://hub.docker.com> : 在官网上 搜所 所需要的 镜像

Linux 系统内核 版本 须是 3.10 以上版本,检查内核版本: uname -r

yum install docker : 安装docker

CentOS 7.0 以上的安装 命令

systemctl start docker: 启动docker

docker -v :查看 docker版本 docker --version

docker version : 查看docker 版本相关信息

docker info : 查看docker的 详细信息

systemctl enabledocker: 开机docker

systemctl stop docker : 停止docker

yum remove docker: 卸载docker

### 镜像命令

docker search mysql : 搜索 镜像 ,默认 去 docker hub 官网搜索

docker pull mysql : 下载镜像 ( 将镜像 从远程仓库 下载到本地)

docker pull mysql : tag : 下载指定版本的镜像 (指定标签)

<https://hub.docker.com/> 该官网查询指定版本的镜像

docker images: 查看docker 有多少镜像

docker images -q : 返回全部镜像Id

docker rmi 镜像id : 删除 指定 id 的本地镜像

docker rmi hello-world:latest -f : 删除指定镜像 -f : 强制删除

docker rmi tomcat mysql -f : 删除多个镜像 (默认删除最新版镜像)

docker rmi $(docker images -q) : 组合命令, 删除本机所有镜像

启动镜像:mytomcat : 别名 -d: 后台运行

tomcat:latest:镜像名称以及版本(不写版本 默认就是latest)

### 容器命令

run 命令的选项: -i : 以交互模式运行容器,通常与 -t 同时使用

-t : 为容器重新分配一个伪输入终端,通常与 -i 同时使用

-d : 后台运行容器,并返回容器ID, 即启动守护式容器

docker run -it centos : 运行docker里面的centos容器,并生成终端进行交互(-it)

退出容器 exit : 退出 并关闭容器 ctrl + p +q : 退出但不关闭容器

docker exec -it 容器ID : 再次进入启动容器

docker exec -it 容器ID ls : 不进入容器,直接在宿主机(本机 非docker 的容器)拿到结果

docker attach 容器ID: 重新进入容器

(总之 exec比 attach 功能更加强大)

docker run 镜像ID : 运行镜像 生成容器

docker run --name mytomcat -d tomcat:latest

这样启动 的tomcat 是容器的,所以 外面访问不到,要想访问需要将其端口映射到 主机端口

docker run --name mytomcat -d -p 8888:8080 tomcat:latest 启动镜像容器

docker run -d -p 8888:8080 tomcat

docker run -d -p 8887:8080 tomcat ..... 一个镜像可以启动多个容器

docker run -it -P tomcat : 交互式运行 tomcat (打印一大堆启动日志)

访问 不了 可能是防火墙的原因

-p : 端口映射 8888:主机端口 8080:容器端口 , 之后 就能在浏览器访问

-P : 随机生成映射主机端口

IP 用 服务器 IP

docker ps : 查看正在运行的 镜像 容器

docker ps -l : 查看上一次运行的容器 (无论容器是否停止)

docker ps -n 3 : 查看最近 3 次运行的容器 (无论容器是否停止)

docker ps -a: 查看所有的镜像

docker ps -q : 只显示容器 ID

TIM截图20180927181632

docker stop CONTAINER ID/ names : 停止运行中的容器

docker stop $(docker ps -a -q) : 停止所有容器

docker kill 容器ID/names : 强制 停止 运行中容器

docker start CONTAINER ID: 启动一个容器 (不是镜像ID)

docker restart 容器ID : 重启容器

docker rm CONTAINER ID /names: 删除已经停止的容器 (-f 强制删除没停止容器)

docker rm -f $(docker ps -a -q) :删除所有容器

docker top 容器ID : 查看容器内运行的进程

docker cp 容器ID : /tmp/yum.log /root :

将容器(centos) 中 的 yum.log文件 ,拷贝到 宿主机(vmvare上的主机)

**防火墙**

service firewalld status : 查看防火墙状态

service firewalld stop :关闭 防火墙

docker logs 容器名/容器id: 查看 容器启动日志

Mysql 的启动: -e 传递参数

docker run --name mysql01 -e MYSQL\_ROOT\_PASSWORD=159786 -d mysql:5.5

docker run -p 3306:3306 -e MYSQL\_ROOT\_PASSWORD=159786 -d mysql:5.5

指定 端口 映射 那么 在 自己的windows主机上就可以访问该数据库了

Mysql 的一些高级命令 可以 去 <https://hub.docker.com/> 上查找

docker-cn.com : docker 在国内的官网

docker inspect 容器ID : 查看启动容器的json串

commit 命令:

docker commit -m=”描述信息” -a=”作者” 容器ID **目标镜像名:标签** :

创建自己的镜像 (原来的镜像自己做了修改,需要生成新的镜像)

目标镜像名:标签 : 这是自己定义的镜像名和自定义tag

### Docker 镜像详解

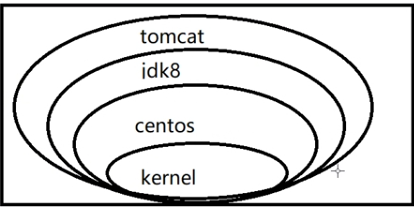
镜像采用UnionFS(联合文件系统)

镜像加载原理

分层的镜像 : 多个镜像共用宿主机的linux内核,不同系统的linux内核是大同小异的

采用分层结构的好处 : 资源共享

镜像内层是只读的



## Docker容器数据卷

容器卷是本质上是一个目录,但是 这个目录(容器中的目录,比如centos容器) 可以和宿主机进行数据交互.

Docker容器一关闭,数据都没了.有时,需要做持久化. 就像内存中的重要数据,一断电,就没了,需要将内存中数据保存的数据库一样

功能: 1 数据卷可在容器之间共享或者重用数据

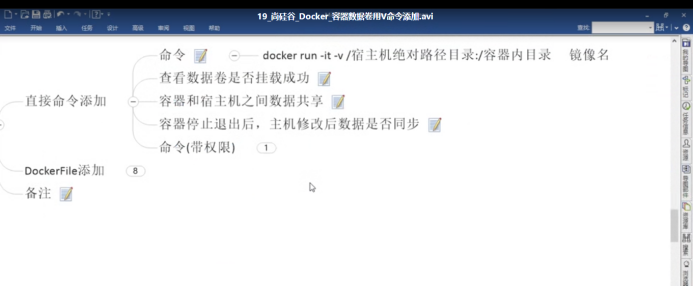
2 卷中的更改可以直接生效

3 数据卷中的更改不会包含在镜像的更新中

4 数据卷的生命周期一直持续到没有容器使用它为止

总之,容器的持久化,和 容器间继承 + 数据共享 ,宿主机和容器之间的数据共享

### Docker数据卷用 -v 命令添加



宿主机和容器间的数据共享: (下面这条命令应该在容器端使用)

**docker run -it -v /myDataVolume:/dataContainerVolume centos :**

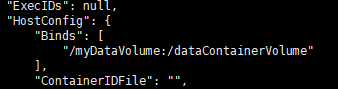
/myDataVolume : 宿主机根目录下的文件夹 -v 会创建该目录

/dataContainerVolume : centos 容器根目录下的文件夹 (-v 自动创建)

centos : 镜像名

之后会进入centos容器的 命令交互模式 (启动了 centos容器)

**docker inspect 容器ID** 命令看到类似下面的记录,说明宿主机和容器间数据实现了共享



之后 在这两个目录下创建文件啥的,都会在另一个目录下创建相应的文件

### Docker容器数据卷用 DockerFile添加

DockerFile 是镜像 的描述文件, DockerFile 是一个可以运行的脚本. (内容很多)

DockerFile 是一个文本文件,内容是一条条的指令,每一条指令构建一层. 用DockerFile定制 镜像.

可以在 <https://hub.docker.com/> 官网上查看镜像的dockerFile

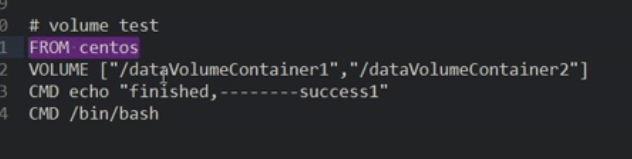


添加步骤

1 在宿主机根目录下(随便哪个目录)新建 mydocker(随便) 文件夹并 进入

2 :File构建 (dockerFile文件)

在 该目录下 创建 dockerFile文件(touch) 并并编辑该文件 添加下图内容



这是一个可执行文件,VOLUME 是一个命令,dataVolumeContainer1 和 dataVolumeContainer2 是 将在容器内生成的数据卷,可以指定多个

**FROM** : 该指令指定基础镜像,定制镜像肯定是以一个镜像为基础.

FROM 指令是DockerFile文件的必备指令,并且是第一条指令

3: build : 将DockerFile文件 构建成新的镜像

（-f 文件在当前路径，且名称为DockeFile的话，不用-f，也可以build文件）

docker build -f /mydocker/dockerFile -t duan/centos .

duan/centos : duan 类似于 java类中的包名,centos是镜像名

4 : run 镜像 生成容器 ,这个自定义的镜像生成的容器 在根目录下会自带

2个数据卷目录,用来和 宿主机进行数据交互(数据共享)

5: docker inspect 容器ID , 来查看 容器数据卷和宿主机目录的对应关系

默认 会在宿主机内 创建一个目录 与容器中的数据卷对应

### 容器数据卷

数据在容器之间共享

步骤

1: 用DockerFile 自定义的镜像 启动一个容器 (加入names 设置为 c1)

2 : docker run -it --name c2 --volumes-from c1 镜像ID/镜像的REPOSITORY

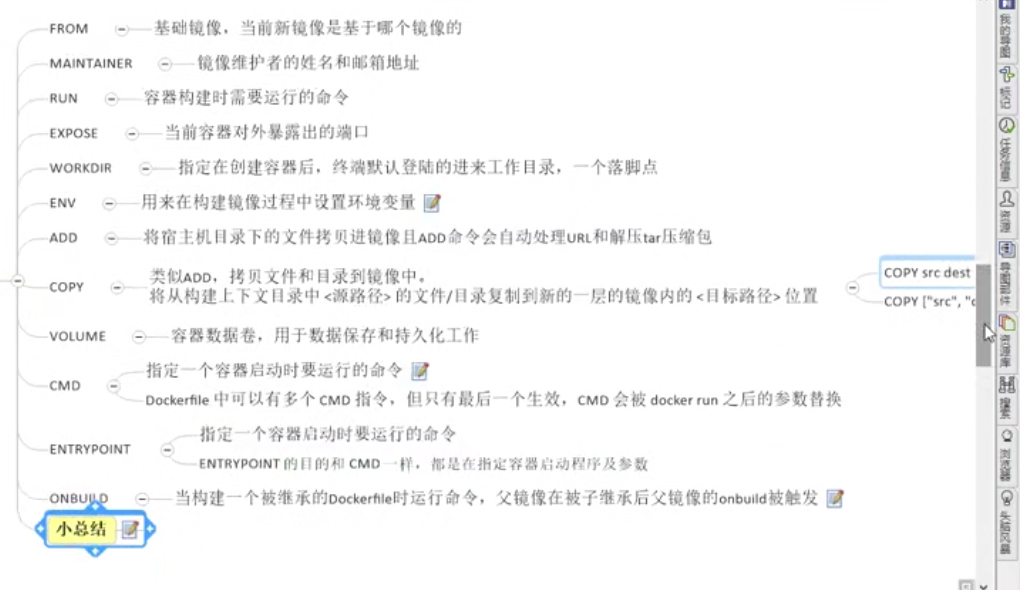
则c2这个容器 继承自 c1, 无论 哪个容器的数据卷 内容有变动,都会更新到另一个容器的数据卷 , 就算删除了父容器,其他的容器 数据卷也是共享的

### DockerFile

DockerFile 是镜像 的描述文件, DockerFile 是一个可以运行的脚本. (内容很多)

DockerFile 是一个文本文件,内容是一条条的指令,每一条指令构建一层. 用DockerFile定制 镜像. DockerFile 就是Docker镜像的构建文件

#### DockerFile 保留字指令



用这些保留字,编写的DockerFile文件,可以构建出来自定义的镜像

RUN: 构建镜像时需要执行的linux命令

CMD : 在dockerFile文件中, 会被最后的CMD 命令覆盖. docker run 后面加的参数,启动容器时,会覆盖 DockerFile 中的CMD保留字指令

ENTRYPOINT 指令和 CMD 一样,都是运行命令 ,但它不会被覆盖

CMD 和RUN 的区别:

RUN 是在 镜像构建时 运行 linux命令

CMD 是在容器启动时 运行 linux命令

ONBUILD : 子类镜像构建时 触发 父类镜像ONBUILD 命令

比如 在一个dockerFile文件有这么一句:

ONBUILD RUN ehco “father onbuild run--------”

这个文件 构建成 的镜像名称 为 father

有 另外一个镜像文件 dockerFile2

内容 FROM father

构建dockerFile2 镜像时 会 触发 father镜像 的 ONBUILD

一个自定义的小的 DockerFile文件

FROM centos

ENV HOME /tmp ENV : 设置变量 HOME的值 为 /tmp

RUN yum -y install vim RUN: 容器构建时 安装vim命令,不然不能用vim

WORKDIR $HOME WORKDIR: 启动镜像容器的落脚点 : $HOME 引用变量

EXPOSE 8080 EXPOSE : 对外暴露的端口

CMD ehco "------------" ehco :打印提示信息

CMD /bin/bash

构建镜像:

docker build -f /mydocker/dockerFile -t duan/centos:tag .

/mydocker/dockerFile: dockerFile 文件所在路径

duan/centos:tag 镜像名称以及版本 . 代表当前目录

docker history 镜像ID: 查看镜像的构建历史

自定义tomcat



