比赛涉及Docker命令操作：

比赛使用的是乌班图Ubuntu系统，所以Docker是安装在乌班图下的，平时练习的时候可以直接在Centos下安装Docker

1. 安装Doker
   1. 前置条件

1）在安装 Docker 之前，先说一下配置，我这里是Centos7 Linux 内核：官方建议 3.10 以上，3.8以上貌似也可。可以通过 uname -r 查看内核版本：

2） 用 root 权限更新 yum 包（特别说明：这个命令不是必须执行的，看个人情况，后面出现不兼容的情况的话就必须update了）

yum -y update

如果这一步报错则说明以前安装docker时出现过错误（我出现了这个问题），如果有则执行下列命令，没有则无需执行：

3） 卸载旧版本（如果之前安装过的话，没有安装过则跳过）

yum remove docker docker-common docker-selinux docker-engine

* 1. 安装Doker

1、安装需要的软件包， yum-util 提供yum-config-manager功能，另两个是devicemapper驱动依赖

yum install -y yum-utils device-mapper-persistent-data lvm2

1. 设置 yum 源（下面两个都可以，选择中央仓库）
2. 第一个（中央仓库）

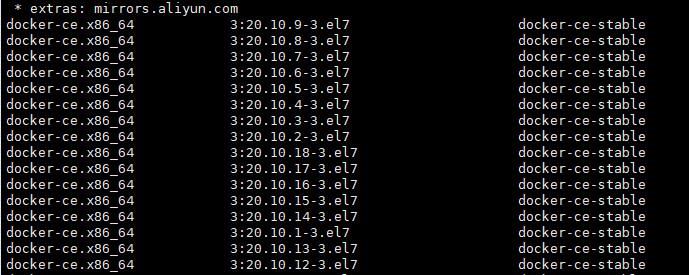
yum-config-manager --add-repo http://download.docker.com/linux/centos/docker-ce.repo

1. 第二个（阿里云仓库）

yum-config-manager --add-repo http://mirrors.aliyu.com/docker-ce/linux/centos/docker-ce.repo

1. 选择docker版本并安装 （1）查看可用版本有哪些

yum list docker-ce --showduplicates | sort -r



如果出现什么 header selection错误，删除/etc/yum.repos.d/下的 docker-ce文件

删除原有的文件 rm -f /etc/yum.repos.d/CentOS-Base.repo

重新下载，wget -O /etc/yum.repos.d/CentOS-Base.repo http://mirrors.aliyun.com/repo/Centos-7.repo

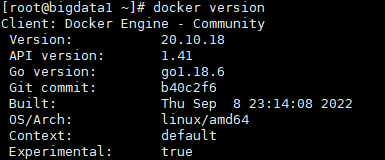
清理缓存 yum clean all

1. 选择一个版本并安装：yum install docker-ce-版本号（可以选择版本安装），这里我们**直接默认安装最新版本**：

yum install -y docker-ce docker-ce-cli containerd.io

1. 查看版本号

docker version



1. 启动docker服务

systemctl start docker

1. 停止服务

systemctl stop docker

1. 开机自启

systemctl enable docker

1. 查看docker状态

systemctl status docker

1. Docker使用

1、如果要登录仓库，登录仓库的命令是：

docker login -u 账户名 -p 密码 远程仓库的ip

搜索镜像：

docker search 镜像名

如：docker search centos7

2、从本地仓库拉去镜像（下面的命令不是绝对，根据比赛给予的文档参考）

docker pull 服务器ip/项目名/镜像名:Tag

拉取指定版本的 CentOS 镜像，这里我们安装指定版本为例(centos7)

如：docker pull bluedata/centos7

3、查看所有镜像：docker images



删除本地镜像（根据镜像的id）：

docker rmi -f 8652b9f0cb4c

4、创建容器前先查看配置网络：

1）查看docker网络：

docker network ls

当你运行一个容器的时候，可以使用 --network 参数来指定你的容器连接到哪一个网络。

查看指定的网络信息：

docker network inspect 网络配置名称

创建自定义网络：

docker network create -d bridge mynetwork2

自定义创建网络：e

docker network create -d bridge --subnet 172.30.0.0/24 --gateway 172.30.0.1 mynetwork2

特别说明：docker创建的网段不要和宿主机一个网段，否则发生冲突。

查看自定义网络：

docker inspect mynetwork2

删除网络：

docker network rm 网络名称

根据镜像创建容器：

docker run -itd --name 容器名字 --net mynetwork --ip 172.18.0.XX --privileged=true 镜像名:Tag /sbin/init

说明：--name后面是容器的名字

--net是网络配置

--ip是给容器一个固定的ip（必须是使用subnet创建的网络才能给ip）

--privileged=true给权限，不给的话在容器中无法启动容器

镜像名:Tag ，如果没有给Tag则默认使用Latest

最后的 /sbin/init 不能掉

如：

容器的名称是 master

网络ip地址是 172.18.0.101

5、查看所有容器：docker ps -a

ps只会显示正在运行Up状态的

-a是显示所有的

6、进入容器：docker exec -it 容器名 /bin/bash

或者：docker exec -it 容器名 bash

7、停止容器：docker stop 容器名

8、从乌班图宿主机拷贝内容到容器中：docker cp /xxx 容器名:/xxx

其他相关资料内容：

|  |
| --- |
| docker load -i 镜像tar文件 # 导入镜像至本地仓库  #### ------ 以下 XX:XX 必须要能够在 docker images查询到 -----####  docker tag XX:XX YY:YY # 镜像改名，修改tag（像是另存为）  docker pull 10.18.4.10/library/XX:XX # 从远程镜像仓库拉取镜像  docker rmi -f XX:XX #删除镜像，-f表示强制操作  docker save -o 本地路径/XX.tar XX:XX #将镜像存储在本地，-o指定导出的路径和文件名  docker push 10.18.4.10/library/XX:XX #上传镜像至远程仓库（镜像的tag必须带有远程地址和项目名）  docker inspect XX:XX # 查看镜像详细信息  docker import和docker load的区别  简单来说，就是带不带历史信息的区别，这点上可以分别使用他们两个后，用docker images查看出来的时间点的区别就可以看出  具体参照：<https://www.cnblogs.com/Cherry-Linux/p/8025777.html>  docker ps -a #查看容器，如果没有加-a选项，则只显示已启动的容器  docker container ps #查看容器  docker create -it --name 容器名 镜像 [容器命令] [命令参数] #创建容器  docker start 容器名或容器id #启动容器  docker run --rm --it --name 容器名 镜像 [容器命令] [命令参数] #创建新容器并启动，--rm是指容器退出后就自动删除该容器  docker run -d [--name 容器名] 镜像名称 #在后台启动一个容器  docker run和docker create的区别  docker run 命令同时完成了 create 及 start 操作  docker pause 容器名 #暂停容器  docker unpause 容器名 #取消暂停容器  docker restart [-t 数字] 容器名 #重启，-t表示多少秒后重启，默认10s  docker stop 容器名 #关闭容器  docker kill 容器名 #终止容器  docker rm [-f] 容器名 #删除容器，-f表示强制删除正在运行的容器  docker stop 和 docker pause的区别  一个是暂停容器中的进程（pause），而另外一个是暂停容器（stop）  docker exec -it 容器名 /bin/bash #进入一个运行中的容器  docker exec 容器名 命令 #在一个正在运行的容器中运行命令，登录出容器ctrl+d  docker logs 容器名 #查看容器的运行日志  docker inspect 容器名 #查看容器的详细信息  docker port 容器名 #查看容器占用的端口  docker rename 旧容器名 新容器名 #改容器名  docker commit -m '评论' -a '作者' 容器名 新镜像名:版本号 #基于容器创建镜像  docker cp 源 目标 #复制文件 容器名或id：路径 |

Docker操作案例：

|  |
| --- |
| 1、启动一个nginx容器  docker run --it -d --name web1 -p 8081:80 nginx:lastest  2、启动一个httpd服务器  docker run --it -d --name web2 -p 8082:80 nginx:lastest  3、修改每个服务器的默认网页文件  在本地准备好网页文件 vim index.html  4、分别复制至相应容器中：  docker cp index.html web1:/usr/share/nginx/html/  docker cp index.html web2:/var/www/html/ |

容器中修改root密码：passwd root

修改：vi /etc/ssh/sshd\_config  
将下面的配置改为yes，将配置PermitRootLogin no 注释掉，保留下面的  
PermitRootLogin yes  
然后重启sshd服务：systemctl restart sshd