Docker容器

体积小、启动速度快、docker是系统层面的隔离。

Docker 是一个开源的应用容器引擎，让开发者可以打包他们的应用以及依赖包到一个可移植的镜像(images)中，然后发布到任何流行的 Linux或Windows操作系统的机器上，也可以实现虚拟化。

容器(container)是完全使用沙箱(sandbox)机制，相互之间不会有任何接口。

1. 容器与镜像

镜像：（Image），应用程序以及其所需要的依赖、环境等等打包在一起的称为镜像。

容器：（Container）镜像的应用程序运行之后形成的进程就是容器，Docker让其透明不可见并且做了隔离。

1.1 镜像地址

Docker官方镜像地址： Docker Hub Container Image Library | App Containerization

1.2 Ubuntu系统的docker镜像加速

下面的代码在终端执行，其中内容是要一行一行写入

```
sudo mkdir -p /etc/docker

sudo tee /etc/docker/daemon.json <<-'EOF'

{
"registry-mirrors": ["https://6c3kptqz.mirror.aliyuncs.com"]
}

EOF

sudo systemctl daemon-reload

sudo systemctl restart docker
```

2. 镜像操作

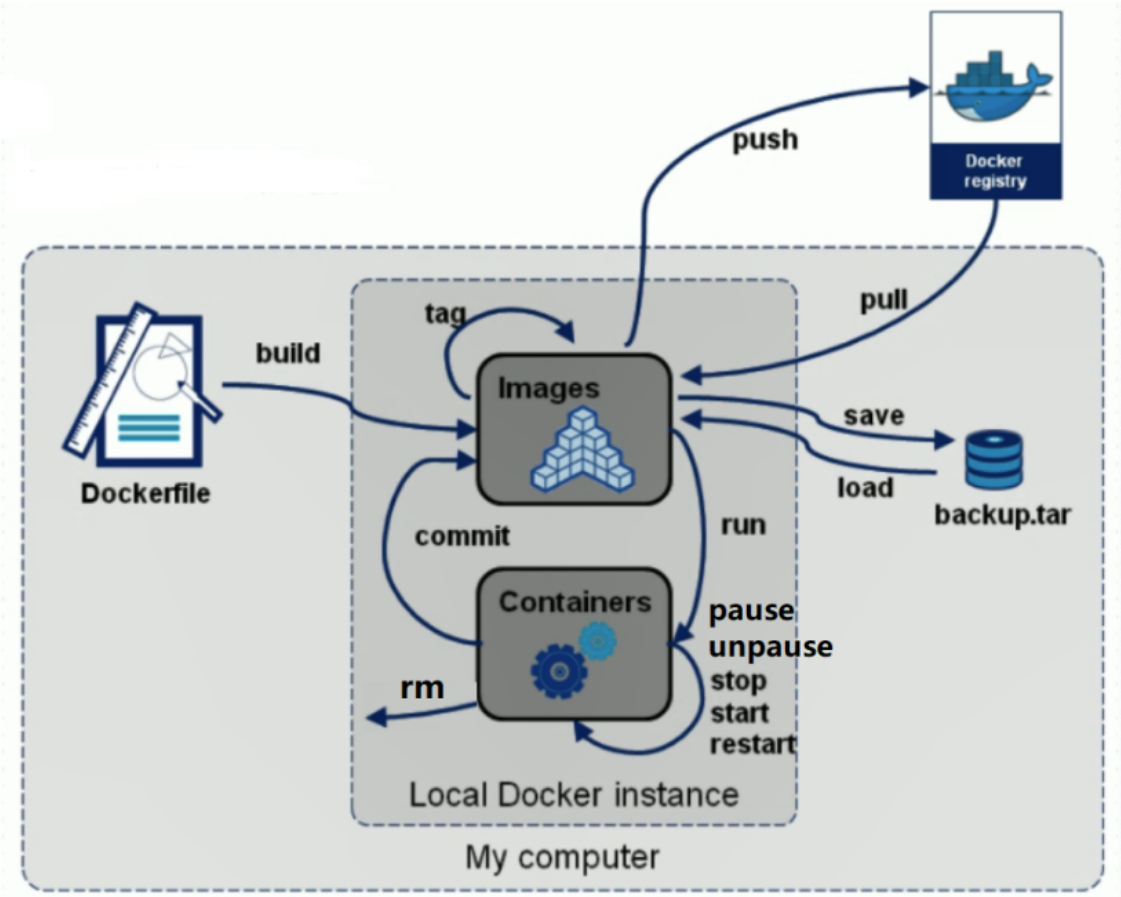
2.1 镜像名称组成

镜像名称由两部分组成：名称:版本号 ===>> [repository]:[tag]

当不指定`tag`时，默认是`latest`也就是最新版本的镜像。

2.2 镜像命令

常见的镜像操作命令如图：



常用的镜像命令

`docker images` 查看镜像

`docker rmi` 删除镜像

`docker push` 推送镜像到服务器

`docker pull` 从服务器拉取镜像

`docker save` 保存镜像为tar压缩包

`docker load` 加载tar压缩包为镜像

拉取、查看镜像

1. 首先查看docker中已经安装的镜像

```
docker images
```

2.根据查看到的镜像名称，拉取自己需要的镜像

```
# 例如拉取nginx
docker pull nginx
```

3.通过命令：docker images 查看拉取到的镜像

```
docker images
```

保存、导入镜像

利用docker xx --help命令查看docker save和docker load的语法

```
docker save --help
```

使用`docker save`导出镜像到磁盘

命令格式：

```
docker save -o [保存的目标文件名称.压缩格式] [镜像名称:版本号]

# 例如
docker save -o nginx.tar nginx:latest
```

使用`docker load`加载镜像

1.本地有镜像先删除镜像

```
# 例如
docker rmi nginx:latest
```

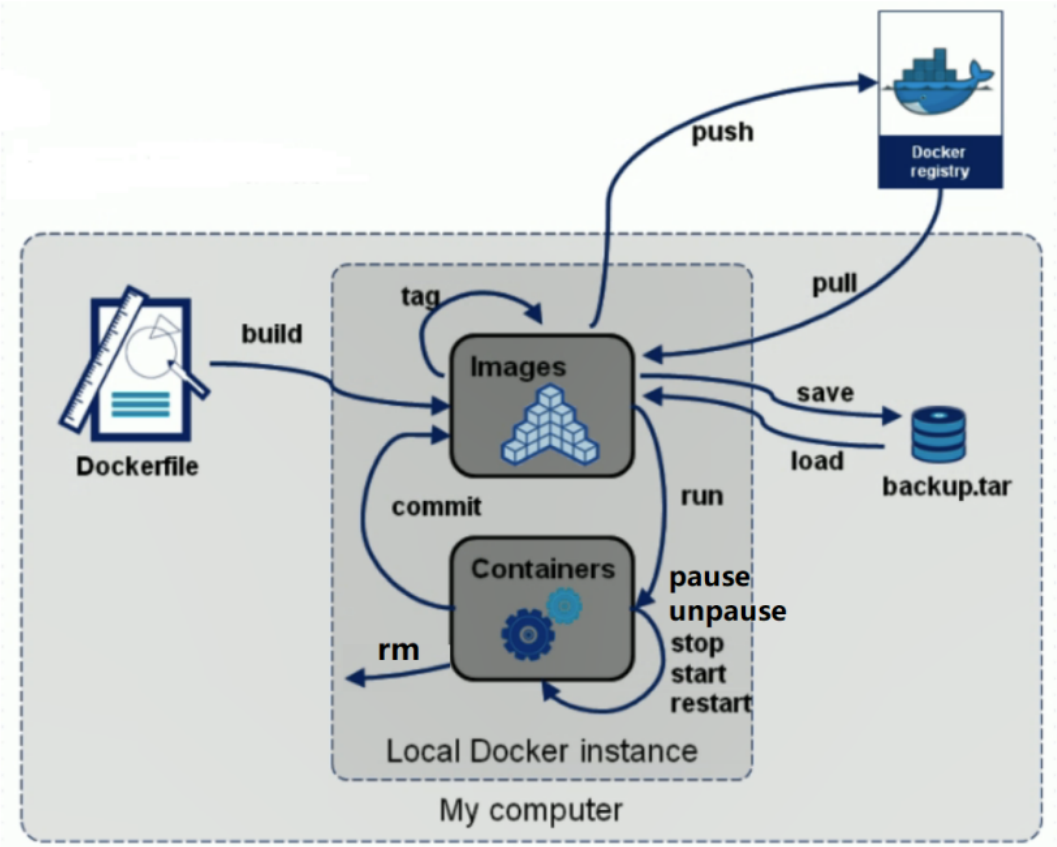
2.然后运行命令，加载本地文件

```
# 例如
docker load -i nginx.tar
```

3. 容器操作

3.1 容器相关命令

容器操作的命令如图：



容器保护三个状态：

运行：进程正常运行

暂停：进程暂停，CPU不再运行，并不释放内存

停止：进程终止，回收进程占用的内存、CPU等资源

其中的操作命令：

`docker run` 创建并运行一个容器，处于运行状态

`docker stop` 停止一个运行的容器

`docker start` 让一个停止的容器再次运行

`docker restart` 重新启动容器

`docker rm` 删除一个容器

`docker pause` 让一个运行的容器暂停

`docker unpause` 让一个容器从暂停状态恢复运行

查看容器状态：

1. `docker ps` 查看运行的容器
2. `docker ps -a` 查看所有容器，包括已经停止的

查看容器日志的命令：

1. `docker logs 创建的容器名称` 查看docker容器中的某一个容器的访问日志
2. `docker logs -f 创建的容器名称` 持续查看docker容器中的某一个容器的访问日志、停止持续显示访问日志：Ctrl + C

3.2 创建并运行容器

命令语法：

```
docker run --name 创建的容器名称 -p 服务器的端口:容器使用的端口 -d 镜像名称:版本号
```

语法解析：

docker run ： 创建并运行一个容器

--name : 给容器起的名字

-p ： 将服务器端口与容器端口映射，冒号左侧是服务器端口，右侧是容器端口

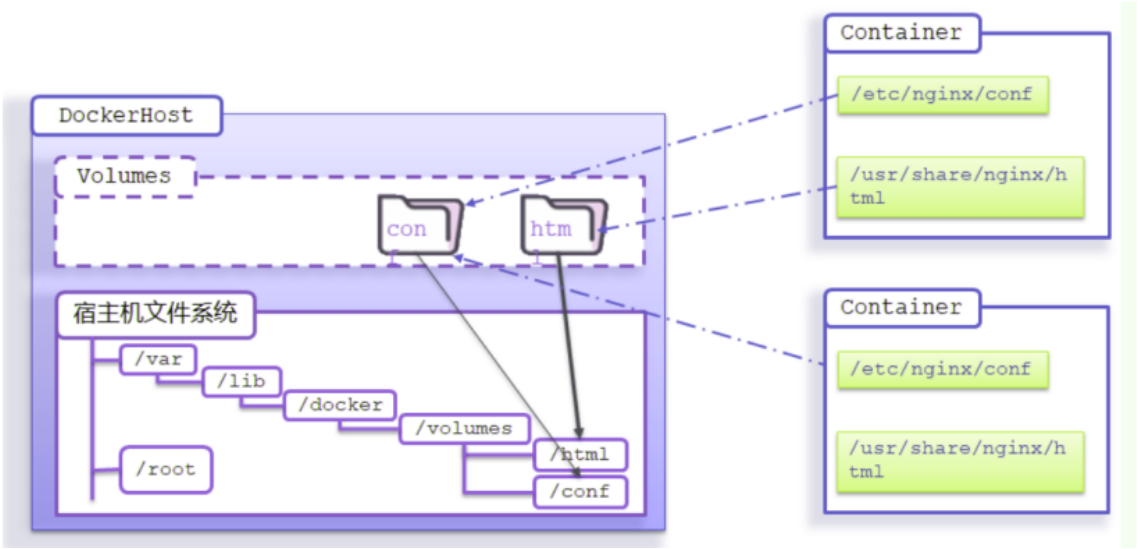
-d：后台运行这个容器

这里的 -p 参数，是将容器端口映射到宿主机端口

4 数据卷

正常情况下删除镜像后，数据也同时删除了，产生这种问题的原因是容器与数据（容器内文件）耦合带来的后果。要解决这个问题，必须将数据与容器解耦，这就要用到数据卷了。

数据卷（volume）是一个虚拟目录，指向宿主机文件系统中的某个目录。



一旦完成数据卷挂载，对容器的一切操作都会作用在数据卷对应的宿主机目录了。

这样，我们操作宿主机的/var/lib/docker/volumes/html目录，就等于操作容器内的/usr/share/nginx/html目录了

4.1 数据卷操作命令

数据卷操作的基本语法如下：

```
docker volume [COMMAND]
```

docker volume命令是数据卷操作，根据命令后跟随的command来确定下一步的操作：

`create` 创建一个volume

`inspect` 显示一个或多个volume的信息

`ls` 列出所有的volume

`prune` 删除未使用的volume

`rm` 删除一个或多个指定的volume

4.2 创建和查看数据卷

创建数据卷

```
docker volume create 自定义名称
```

查看所有数据卷

```
docker volume ls
```

查看数据卷详细信息卷

```
docker volume inspect 自定义名称
```

`docker volume create`：创建数据卷

`docker volume ls`：查看所有数据卷

`docker volume inspect`：查看数据卷详细信息，包括关联的宿主机目录位置

`docker volume rm`：删除指定数据卷

`docker volume prune`：删除所有未使用的数据卷

4.3 挂载数据卷

我们在创建容器时，可以通过 -v 参数来挂载一个数据卷到某个容器内目录，命令格式如下：

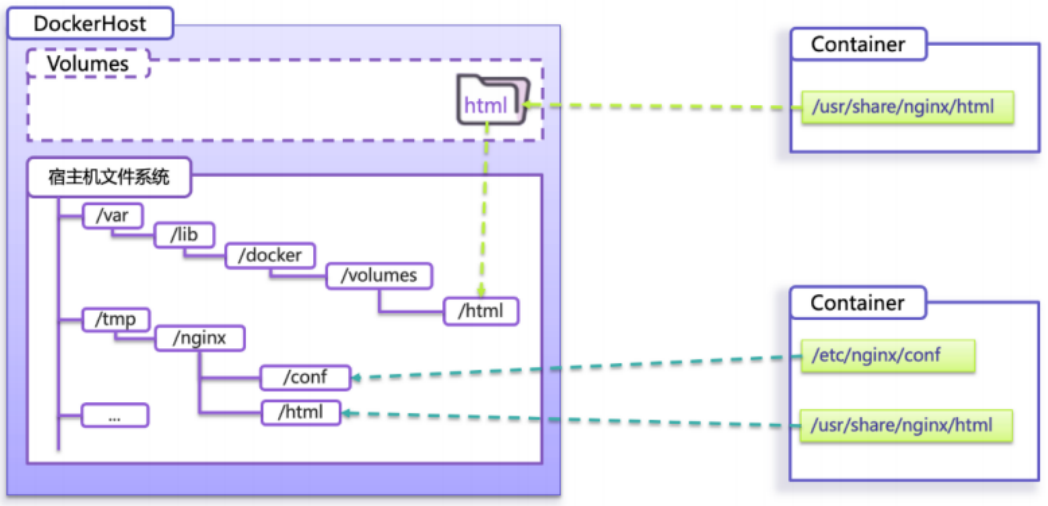
```
docker run --name 创建的容器名称 -v 自己的名称:/usr/自定义名称/去镜像官网看具体目录 -p 服务器的端口:容器使用的端口 -d 镜像名称:版本号
```

4.4 MySQL挂载本地目录

容器不仅仅可以挂载数据卷，也可以直接挂载到宿主机目录上。关联关系如下：

1. 带数据卷模式：宿主机目录 ---> 数据卷 ---> 容器内目录
2. 直接挂载模式：宿主机目录 ---> 容器内目录

如图：



语法：

目录挂载与数据卷挂载的语法是类似的：

1. -v [宿主机目录]:[容器内目录]
2. -v [宿主机文件]:[容器内文件]

创建并运行一个MySQL容器，将宿主机目录直接挂载到容器

步骤：

- mysql.tar文件上传到服务器
- 创建目录/usr/mysql/data
- 创建目录/usr/mysql/conf
- 将提供的hmy.cnf文件上传到/usr/mysql/conf
- 挂载/usr/mysql/data到mysql容器内数据存储目录
- 挂载/usr/mysql/conf/hmy.cnf到mysql容器的配置文件
- 设置MySQL密码

实现过程如下：

1. 获取mysql镜像

2. 运行mysql容器

```
docker run \
--name 自定义的容器名称 \
-e MYSQL_ROOT_PASSWORD=root \
-p 服务器开放的端口:3306 \
-v /usr/自定义文件名称/conf/hmy.cnf:/etc/mysql/conf.d/hmy.cnf \
-v /usr/自定义文件名称/data:/var/lib/mysql \
-d \
mysql:版本号
```

3. hmy.cnf文件里面的内容

```
[mysqld]
skip-name-resolve
character_set_server=utf8
datadir=/var/lib/mysql
server-id=1000
```

`SqlYog`连接docker里的`mysql8.0`，还需要设置

- 服务器的安全组添加 `你开放的` 的端口
- ubuntu的防火墙要允许 `你开放的` 端口通信

```
sudo ufw status

sudo ufw allow 你开放的端口号
```

- 从linux进入到docker中的容器的命令是

```
docker exec -it 自定义的容器名称 bash
```

- 进入docker中创建的MySQL的容器中后重置mysql8.0密码

```
# 进入mysql
mysql -u root -p

# 先进入 mysql 数据库
use mysql;

# 指定加密方式重置密码
alter user root@'%' identified with mysql_native_password by 'root';
```

Linux的一些常用命令

```
```

```
#查看所有端口是否有监听
sudo netstat -tunlp | grep LISTEN

#查看防火墙状态
sudo ufw status

#关闭防火墙
sudo ufw disable

#打开防火墙
sudo ufw enable

#重启防火墙
sudo ufw reload

#开放端口（开启完成，需要重启防火墙生效）
sudo ufw allow 端口号

#查看端口信息
sudo netstat -tunlp | grep 端口号

#关闭端口
sudo ufw delete allow 端口号

#停止进程
kill PID

#强制停止进程
kill -9 PID

#启动mysql服务
service mysql start

#停止mysql服务
service mysql stop
```