

# 1. Docker

## 1.1 1. Docker 简介

Docker 是一个开源的应用容器引擎，基于 Go 语言并且遵从 Apache2.0 协议开源。

Docker 将应用程序以及应用程序的依赖打包在一起，运行在一个 Docker 容器中，这样你的应用程序在任何环境都会有一致的表现，这里程序运行的依赖也就是容器就好比集装箱，容器所处的操作系统环境就好比货船或港口，**程序的表现只和集装箱有关系( 容器 )，和集装箱放在哪个货船或者哪个港口( 操作系统 )没有关系。**

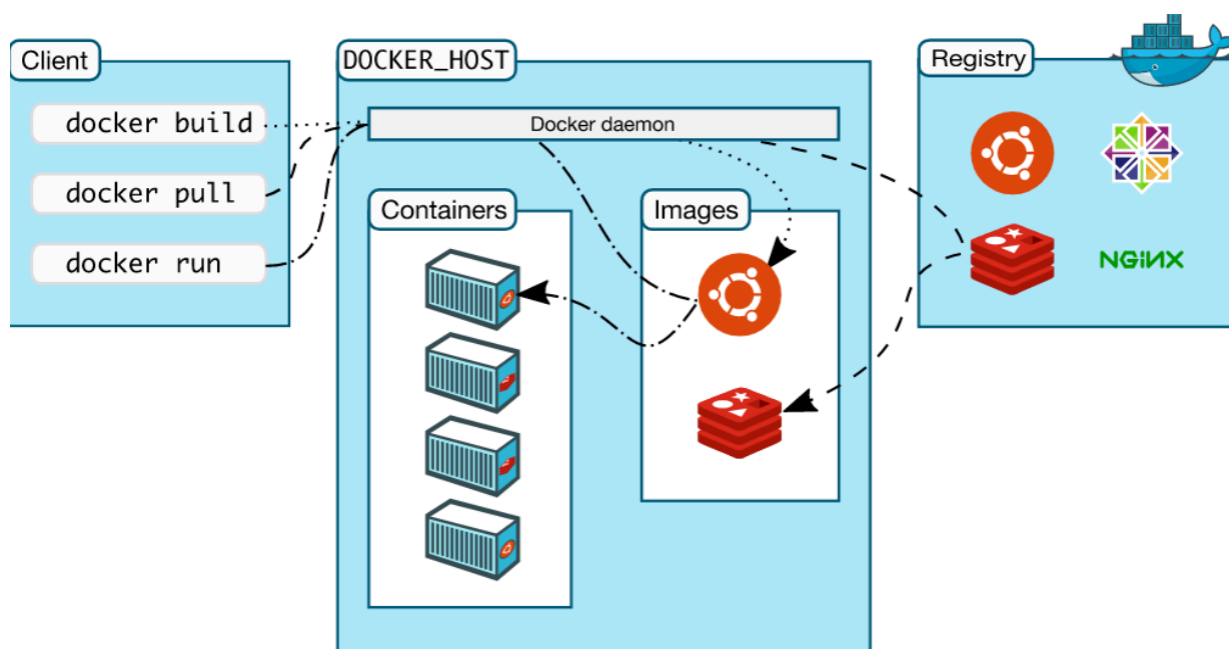
Docker 的口号：“build once, run everywhere”。一次构建，处处运行。

### 1.1.1 1.1 核心概念

Docker 包括三个基本概念：

- **镜像 (Image)**: Docker 镜像是一个特殊的文件系统，除了提供容器运行时所需的程序、库、资源、配置等文件外，还包含了一些为运行时准备的一些配置参数（如匿名卷、环境变量、用户等）。镜像不包含任何动态数据，其内容在构建之后也不会被改变。
- **容器 (Container)**: 镜像（Image）和容器（Container）的关系，就像是 **程序和进程** 一样，镜像是静态的定义，容器是镜像运行时的实体。容器可以被创建、启动、停止、删除、暂停等。
- **仓库 (Repository)**: 仓库（Repository）类似 Git 的远程仓库，集中存放镜像文件。

三者的关系，可以用下图表示：



## 1.2 2. Docker 安装

### 1. 卸载旧版本

```
1 | for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-docker  
  | containerd runc; do sudo apt-get remove $pkg; done
```

## 2. 设置 Docker 的 apt 仓库

```
1  # Add Docker's official GPG key:
2  sudo apt-get update
3  sudo apt-get install ca-certificates curl
4  sudo install -m 0755 -d /etc/apt/keyrings
5  sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o
   /etc/apt/keyrings/docker.asc
6  sudo chmod a+r /etc/apt/keyrings/docker.asc
7
8  # Add the repository to Apt sources:
9  echo \
10     "deb [arch=$(dpkg --print-architecture) signed-
   by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \
11     $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" |
   \
12     sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
13
14  sudo apt-get update
```

## 3. 安装 Docker

```
1  sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin \
2  docker-compose-plugin
```

## 4. 修改镜像源 (/etc/docker/daemon.json, 如果文件不存在, 则自己创建)

```
1  {
2      "registry-mirrors": [
3          "https://docker.m.daocloud.io/",
4          "https://dockerpull.com",
5          "https://docker-0.unsee.tech",
6          "https://docker-cf.registry.cyou",
7          "https://docker.lpanel.live"
8      ]
9  }
```

## 5. 重启 Docker 服务, 并设置开机自启动

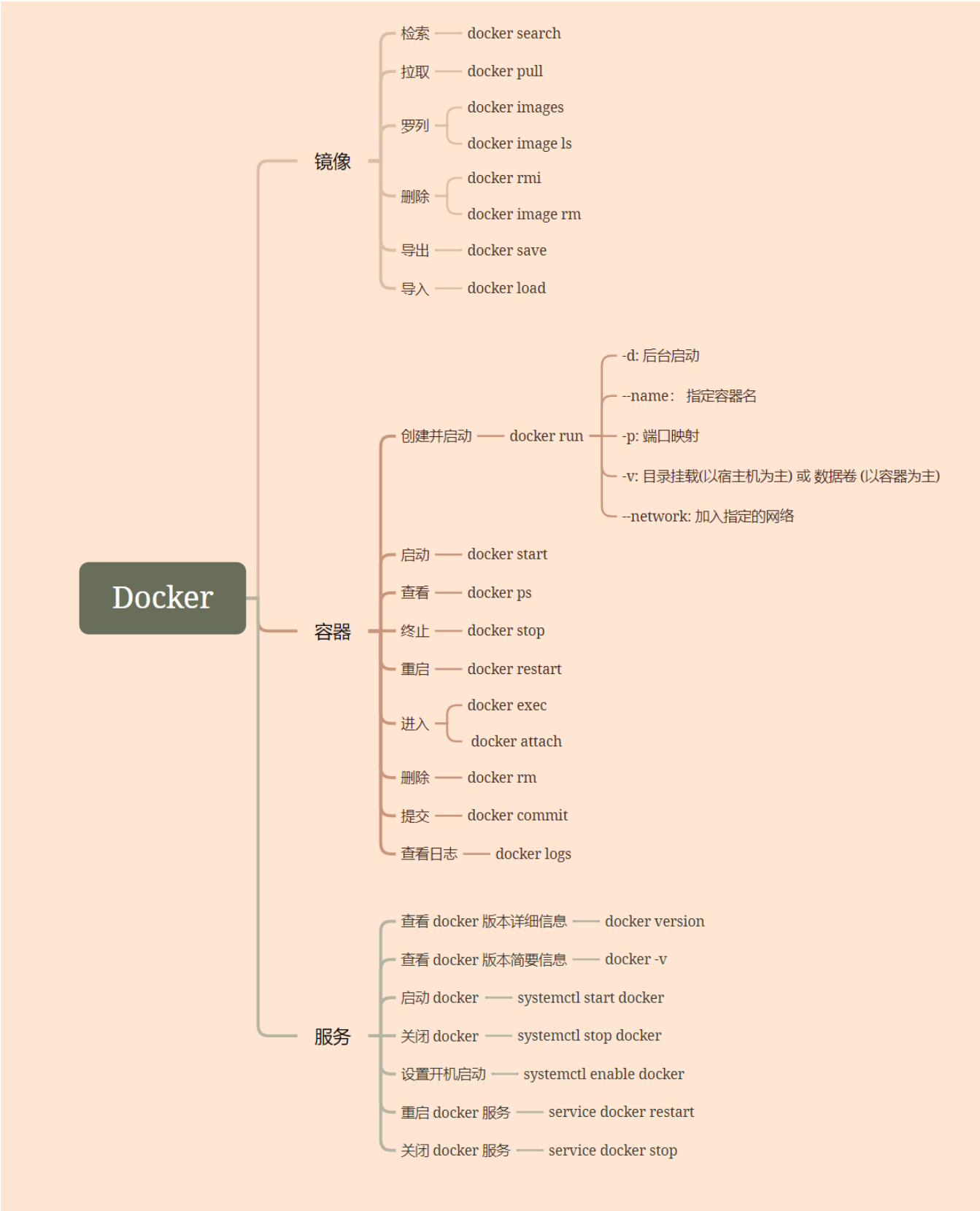
```
1  sudo systemctl daemon-reload
2  sudo systemctl restart docker
3  sudo systemctl enable docker
```

## 6. 验证

```
1  sudo docker run hello-world
```

### 1.3 3. Docker 命令

Docker 有很多很多命令，常用的有以下这些：



接下来，我们回通过一些具体的实验，让大家灵活运用这些命令：

**实验1：**启动一个nginx，并将它的首页改为自己的页面，发布到仓库，让所有人都能使用。

docker run:

- d: 后台启动
- name: 指定容器名
- p: 端口映射, 如 -p 88:80

进入容器: docker exec -it <容器> bash

保存镜像:

- docker commit
- docker save
- docker load

分享社区 (需要科学上网):

- docker login
- docker tag
- docker push

**实验2:** Docker 存储 (让容器内部的数据不再丢失)

- 目录挂载: -v /app/nginx/html(以它为准):/usr/share/nginx/html
- 数据卷: -v ngconf:/usr/share/nginx/html (以它为准)

**实验3:** Docker 网络

docker 0:

docker inspect (docker container inspect)

docker 为每个容器分配了唯一 ip, 使用容器ip+容器端口可以相互访问。

弊端: ip 由于各种原因可能发生变化, 给迁移容器带来了很大的问题; docker0 默认不支持域名的访问方式。

自定义网路:

容器名就是主机域名

docker network

curl <http://app2>