# Docker

Docker官方网址: <https://docs.docker.com/> 英文地址

Docker中文网址: <http://www.docker.org.cn/> 中文地址

Docker是基于Go语言实现的云开源项目，诞生于2013年初，最初发起者是dotClouw公司。Docker 自开源后受到广泛的关注和讨论，目前已有多个相关项目，逐断形成了围Docker的生态体系。dotCloud 公司后来也改名为Docker Ine。

Docker是一个开源的容器引擎，它有助于更快地交付应用。 Docker可将应用程序和基础设施层隔离，并且能将基础设施当作程序一样进行管理。使用 Docker可更快地打包、测试以及部署应用程序，并可以缩短从编写到部署运行代码的周期。

## Docker优点

### 简化程序

Docker 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容器中，然后发布到任何流行的 Linux 机器上，便可以实现虚拟化。Docker改变了虚拟化的方式，使开发者可以直接将自己的成果放入Docker中进行管理。方便快捷已经是 Docker的最大优势，过去需要用数天乃至数周的 任务，在Docker容器的处理下，只需要数秒就能完成。

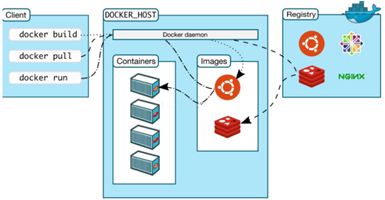
### 避免选择恐惧症

如果你有选择恐惧症，还是资深患者。Docker 帮你 打包你的纠结！比如 Docker 镜像；Docker 镜像中包含了运行环境和配置，所以 Docker 可以简化部署多种应用实例工作。比如 Web 应用、后台应用、数据库应用、大数据应用比如 Hadoop 集群、消息队列等等都可以打包成一个镜像部署。

### 节省开支

一方面，云计算时代到来，使开发者不必为了追求效果而配置高额的硬件，Docker 改变了高性能必然高价格的思维定势。Docker 与云的结合，让云空间得到更充分的利用。不仅解决了硬件管理的问题，也改变了虚拟化的方式。

## Docker架构



客户端→Docker\_Host(宿主机)→仓库

容器→镜像→仓库

Maven

PC电脑端如何查找软件安，360软件管家

Docker 理解为应用市场管家

核心简化运维安装软件的发杂环境

Docker daemon（ Docker守护进程）

Docker daemon是一个运行在宿主机（ DOCKER-HOST）的后台进程。可通过 Docker客户端与之通信。

Client（ Docker客户端）

Docker客户端是 Docker的用户界面，它可以接受用户命令和配置标识，并与 Docker daemon通信。图中， docker build等都是 Docker的相关命令。

Images（ Docker镜像）

Docker镜像是一个只读模板，它包含创建 Docker容器的说明。它和系统安装光盘有点像，使用系统安装光盘可以安装系统，同理，使用Docker镜像可以运行 Docker镜像中的程序。

Container（容器）

容器是镜像的可运行实例。镜像和容器的关系有点类似于面向对象中，类和对象的关系。可通过 Docker API或者 CLI命令来启停、移动、删除容器。

Registry

Docker Registry是一个集中存储与分发镜像的服务。构建完 Docker镜像后，就可在当前宿主机上运行。但如果想要在其他机器上运行这个镜像，就需要手动复制。此时可借助 Docker Registry来避免镜像的手动复制。

一个 Docker Registry可包含多个 Docker仓库，每个仓库可包含多个镜像标签，每个标签对应一个 Docker镜像。这跟 Maven的仓库有点类似，如果把 Docker Registry比作 Maven仓库的话，那么 Docker仓库就可理解为某jar包的路径，而镜像标签则可理解为jar包的版本号。

## Docker与虚拟机比较

作为一种轻量级的虚拟化方式，Docker在运行应用上跟传统的虚拟机方式相比具有显著优势：

Docker容器很快，启动和停止可以在秒级实现，这相比传统的虚拟机方式要快得多。

Docker容器对系统资源需求很少，一台主机上可以同时运行数千个Docker容器。

Docker通过类似Git的操作来方便用户获取、分发和更新应用镜像，指令简明，学习成本较低。

Docker通过Dockerfile配置文件来支持灵活的自动化创建和部署机制，提高工作效率。



## Docker安装

Docker 是一个开源的商业产品，有两个版本：社区版（Community Edition，缩写为 CE）和企业版（Enterprise Edition，缩写为 EE）。企业版包含了一些收费服务，个人开发者一般用不到

Docker 要求 CentOS 系统的内核版本在 3.10以上 ，查看本页面的前提条件来验证你的CentOS 版本是否支持 Docker 。

1、通过 uname -r 命令查看你当前的内核版本

|  |
| --- |
| uname -r |

1. 使用 root 权限登录 Centos。确保 yum 包更新到最新。

|  |
| --- |
| yum -y update |

1. 卸载旧版本(如果安装过旧版本的话)

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

1. 安装需要的软件包， yum-util 提供yum-config-manager功能，另外两个是devicemapper驱动依赖的

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

1. 设置yum源

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

1. 可以查看所有仓库中所有docker版本，并选择特定版本安装

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

1. 安装docker

|  |
| --- |
| sudo yum install -y docker-ce #由于repo中默认只开启stable仓库，故这里安装的是最新稳定版18.03.1 |

1. 启动并加入开机启动

|  |
| --- |
| systemctl start docker  systemctl enable docker |

1. 验证安装是否成功(有client和service两部分表示docker安装启动都成功了)

|  |
| --- |
| docker version |

## ****镜像相关命令****

1、搜索镜像 # docker search java

可使用 docker search命令搜索存放在 [Docker Hub](https://hub.docker.com/)(这是docker官方提供的存放所有docker镜像软件的地方，类似maven的中央仓库)中的镜像。执行该命令后， Docker就会在Docker Hub中搜索含有 java这个关键词的镜像仓库。

[Docker Hub](https://hub.docker.com/) 官网https://hub.docker.com/search?q=java&type=image

2、下载镜像docker pull java:8

使用命令docker pull命令即可从 Docker Registry上下载镜像，执行该命令后，Docker会从 Docker Hub中的 java仓库下载最新版本的 Java镜像。如果要下载指定版本则在java后面加冒号指定版本

3、列出镜像使用 docker images命令即可列出已下载的镜像

4、删除镜像 使用 docker rmi java 命令即可删除指定镜像

docker rmi d23bdf5b1b1b

## ****阿里镜像加速****

详细参考: https://cr.console.aliyun.com/cn-hangzhou/mirrors

## ****容器相关命令****

启动容器 docker run -d -p 81:80 nginx

在本例中，为 docker run添加了两个参数，含义如下：

-d 后台运行

-p 宿主机端口:容器端口 #开放容器端口到宿主机端口

访问 http://Docker宿主机 IP:81/，将会看到nginx的主界面如下：

需要注意的是，使用 docker run命令创建容器时，会先检查本地是否存在指定镜像。如果本地不存在该名称的镜像， Docker就会自动从 Docker Hub下载镜像并启动一个 Docker容器。

2. 列出容器用 docker ps命令即可列出运行中的容器

3. 查看容器的信息 docker inspect 3af5513d208e

## **构建自己的docker镜像**

1、将jar包上传linux服务器/usr/local/**docker**app目录，在jar包所在目录创建名为Dockerfile的文件

2、在Dockerfile中添加以下内容

|  |
| --- |
| ###指定java8环境镜像  FROM java:8  ###复制文件到容器app-springboot  ADD docker-springboot-0.0.1.jar /app-springboot.jar  ###声明启动端口号  EXPOSE 8080  ###配置容器启动后执行的命令  ENTRYPOINT ["java","-jar","/app-springboot.jar"] |

使用docker build命令构建镜像

docker build -t docker-springboot-0.0.1 .

# 格式： docker build -t 镜像名称:标签 Dockerfile的相对位置

docker run -p 8080:8080 docker-springboot-0.0.1 .

重启systemctl restart docker

关闭防火墙  systemctl stop firewalld

192.168.212.215:8080