# 什么是容器

一种虚拟化的方案，操作系统级别的虚拟化，只能运行相同或相似内核的操作系统。

容器就是将软件打包成标准化单元，以用于开发、交付和部署。容器是一种轻量级、可移植、自包含的软件打包技术，使应用程序可以在几乎任何地方以相同的方式运行，开发人员在自己笔记本上创建并测试好的容器，无须任何修改就能够在生产系统的虚拟机、物理服务器或公有云主机上运行。

容器镜像是轻量的、可执行的独立软件包 ，包含软件运行所需的所有内容：代码、运行时环境、系统工具、系统库和设置。容器化软件适用于基于Linux和Windows的应用，在任何环境中都能够始终如一地运行。

容器赋予了软件独立性，使其免受外在环境差异（例如，开发和预演环境的差异）的影响，从而有助于减少团队间在相同基础设施上运行不同软件时的冲突。如果需要通俗的描述容器的话，我觉得容器就是一个存放东西的地方，就像书包可以装各种文具、衣柜可以放各种衣服、鞋架可以放各种鞋子一样。我们现在所说的容器存放的东西可能更偏向于应用比如网站、程序甚至是系统环境。



# 容器的优势

对于开发人员，Build Once,Run Anywhere，容器意味者环境隔离和可重复性，开发人员只需为应用创建一次运行环境，然后打包成容器便可在其他机器上运行，另外，容器环境与所在的Host环境是隔离的，就像虚拟机一样，但是更快更简单。

对于运维人员，Configure Once,Run Anything

# Linux容器技术和虚拟机的区别



容器虚拟化的是操作系统而不是硬件，容器之间是共享同一套操作系统资源的。虚拟机技术是虚拟出一套硬件后，在其上运行一个完整操作系统，因此容器的隔离级别会稍低一些。

# Docker的概念

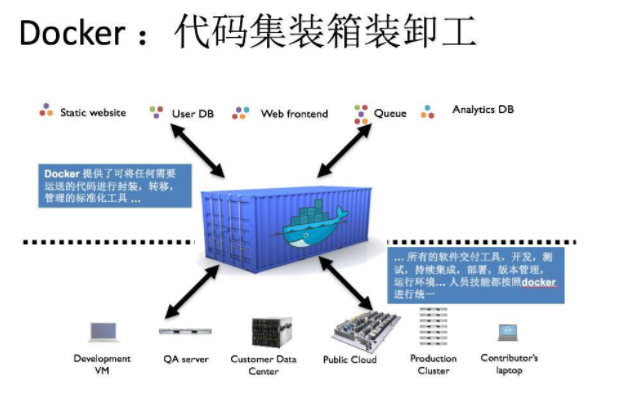
将应用程序自动部署到容器，基于Apache 2.0 开源授权协议发行。

（1）Docker是世界领先的软件容器平台。

（2）Docker使用Google公司推出的Go语言进行开发实现，基于Linux内核的cgroup，namespace，以及AUFS类的UnionFS等技术，对进程进行封装隔离，属于操作系统层面的虚拟化技术。由于隔离的进程独立于宿主和其它的隔离的进程，因此也称其为容器。Docke最初实现是基于LXC（linux containers）。

（3）Docker能够自动执行重复性任务，例如搭建和配置开发环境，从而解放了开发人员以便他们专注在真正重要的事情上：构建杰出的软件。

（4）用户可以方便地创建和使用容器，把自己的应用放入容器。容器还可以进行版本管理、复制、分享、修改，就像管理普通的代码一样。



# Docker的思想（重点）

（1）集装箱

（2）标准化：运输方式、存储方式、API接口

（3）隔离

# Docker容器的特点

## （1）轻量

在一台机器上运行的多个Docker容器可以共享这台机器的操作系统内核，它们能够迅速启动，只需占用很少的计算和内存资源。镜像是通过文件系统层进行构造的，并共享一些公共文件，这样就能尽量降低磁盘用量，并能更快地下载镜像。

## （2）标准

Docker容器基于开放式标准，能够在所有主流Linux版本、Microsoft Windows以及包括VM、裸机服务器和云在内的任何基础设施上运行。

## （3）安全

Docker赋予应用的隔离性不仅限于彼此隔离，还独立于底层的基础设施。Docker默认提供最强的隔离，因此应用出现问题，也只是单个容器的问题，而不会波及到整台机器。

# 为什么要使用Docker

（1）Docker的镜像提供了除内核外完整的运行时环境，确保了应用运行环境一致性，从而不会再出现“这段代码在我机器上没问题啊”这类问题；——一致的运行环境。

（2）可以做到秒级、甚至毫秒级的启动时间。大大的节约了开发、测试、部署的时间。——更快速的启动时间。

（3）避免公用的服务器，资源会容易受到其他用户的影响。——隔离性

（4）善于处理集中爆发的服务器使用压力；——弹性伸缩，快速扩展

（5）可以很轻易的在一个平台上运行的应用，迁移到另一个平台上，而不用担心运行环境的变化导致应用无法正常运行的情况。——迁移方便

（6）使用Docker可以通过定制应用镜像来实现持续集成、持续交付、部署。——持续交付和部署

# Docker目标（重点）

（1）提供简单轻量的建模方式

（2）职责的逻辑分离，使用Docker开发人员只需要关注Docker中的应用程序，运维人员只需要关注如何管理容器。

（3）快速高效的开发生命周期

缩短项目从开发、测试、运行、上线的生命周期。

（4）鼓励使用面向服务的架构

# Docker的使用场景（重点）

（1）使用Docker容器开发、测试、部署服务

（2）创建隔离的运行环境

（3）搭建测试环境

（4）构建多用户的平台即服务（PaaS）基础设施

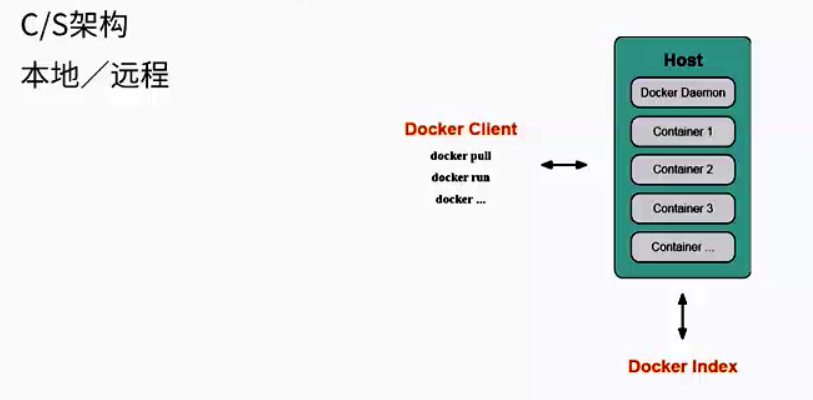
（5）提供软件即服务（SaaS）应用程序

（6）高性能、超大规模的宿主机部署。

# Docker的基本组成

## （1）Docker Client 客户端

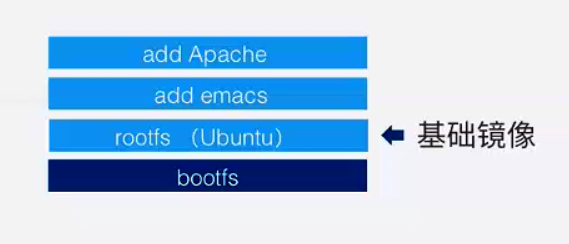
Docker是C/S架构，客户端发出请求，守护进程接收请求，然后进行处理，并返回给客户端。



## （2）Docker Daemon守护进程

## （3）Docker Image 镜像

容器的基石，层叠的只读文件系统，联合加载（union mount），一个镜像上面可以放多个镜像。



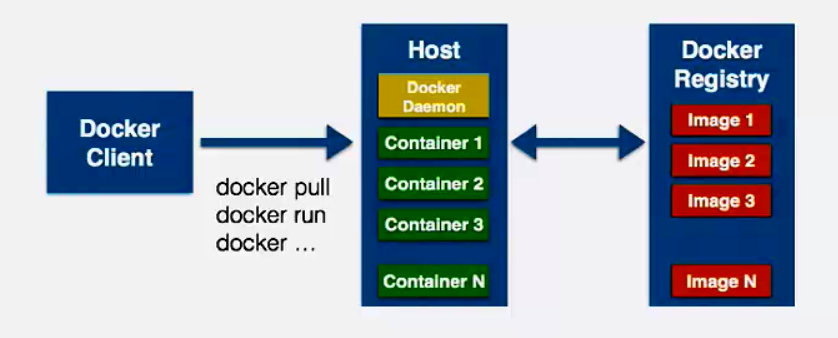
## （4）Docker Container 容器

通过镜像启动容器，容器位于启动和执行阶段，写时复制（copy on write）



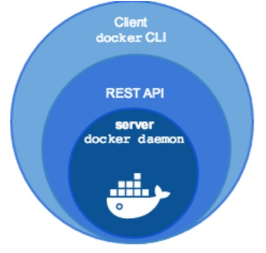
## （5）Docker Repository仓库

Docker用仓库去存储用户构建的镜像，分为公有和私有仓库两种。公有仓库Docker Hub，可以注册账号，分析和保存自己的镜像，我们也可以构建自己私有的仓库。



## Docker Engine

它是一个 client-server application，如下图：



Docker Engine 由三个部分组成：

Docker 进程（Docker Daemon）

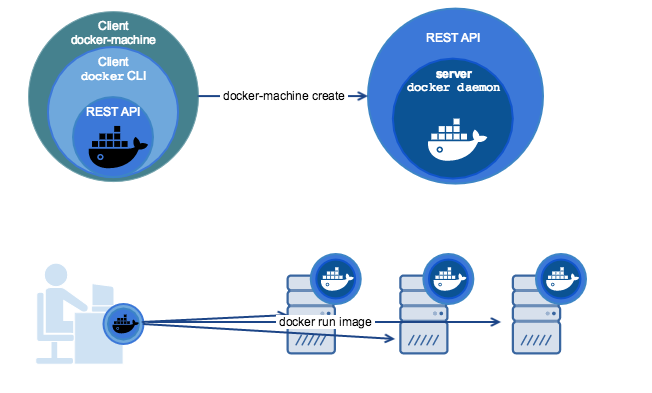
REST API：指定了和进程交互的接口

CLI(command line interface)：通过 REST API 和 daemon 通信，诸如：docker run <image>, docker ps

## Docker Machine

Docker Machine 是一种提供管理主机的工具。常规你会安装 Docker Machine在你的本地机器上。Docker Machine 有自己的命令client：docker-machine，Docker Engine 则有client：docker

我们可以使用 Docker Machine 来安装 Docker Engine 在一个或者多个虚拟系统上，这些虚拟系统可以是本地的（比如Virtualbox里），也可以是远程的（云）。



# Docker容器相关技术

Docker依赖的Linux内核特性

## （1）Namespaces 命名空间

编程语言

封装—代码隔离

操作系统

系统资源的隔离：包括进程、网络、文件系统

PID（Process ID）：进程隔离

NET（Network）：管理网络接口

IPC（InterProcess Communication） 管理跨进程通信的访问

MNT（Mount） 管理挂载点

UTS（Unix Timesharing System） 隔离内核和版本标识

## （2）Control groups (Cgroups) 控制组

用来分配资源，来源于google，为了容器而产生的。

提供的功能：

资源限制

优先级设定

资源计量

资源控制

Docker容器的能力

（1）文件系统隔离：每个容器都有自己的root文件系统。

（2）进程隔离：每个容器都运行在自己的进程环境中

（3）网络隔离：容器间的虚拟网络接口和IP地址都是分开的。

（4）资源隔离和分组：使用Cgroups将CPU和内存之类的资源独立分配给每个Docker容器。

# docker官网

https://www.docker.com/

# docker快速入门网址

https://docs.docker.com/get-started/#docker-concepts

# docker中文社区网址

http://www.docker.org.cn/

# docker安装方法的地址

https://docs.docker.com/install/

# 在linux系统中安装docker

## OS需求

要安装Docker Engine - Community，您需要CentOS 7的维护版本。不支持或测试存档版本。

必须启用centos-extras存储库。 默认情况下，此存储库已启用，但如果已将其禁用，则需要重新启用它。

建议使用overlay2存储驱动程序。

## 卸载旧的版本

较旧版本的Docker被称为docker或docker-engine。 如果已安装这些，请卸载它们以及相关的依赖项。



如果yum报告没有安装这些软件包，那就没关系。

## 安装Docker Engine - Community

您可以根据需要以不同方式安装Docker Engine - 社区：

1、大多数用户设置Docker的存储库并从中进行安装，以便于安装和升级任务。 这是推荐的方法。

2、有些用户下载RPM软件包并手动安装并完全手动管理升级。 这在诸如在没有访问互联网的气隙系统上安装Docker的情况下非常有用。

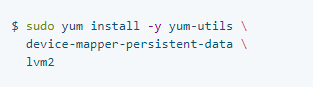
3、在测试和开发环境中，一些用户选择使用自动便捷脚本来安装Docker。

## 使用repository安装docker

在新主机上首次安装Docker Engine - Community之前，需要设置Docker存储库。 之后，您可以从存储库安装和更新Docker。

### 建立repository

1、安装所需的包。 yum-utils提供yum-config-manager实用程序，devicemapper存储驱动程序需要device-mapper-persistent-data和lvm2。



2、使用以下命令设置稳定存储库



### 安装docker engine community

1、安装最新版本的Docker Engine - 社区和容器，或者转到下一步安装特定版本：



如果提示接受GPG密钥，请确认指纹符合060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35，如果符合，则接受该指纹。

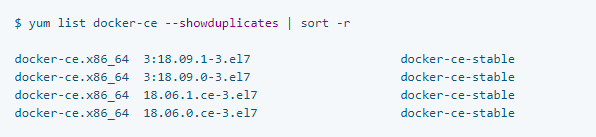
有多个Docker存储库？

如果您启用了多个Docker存储库，则无需在yum install或yum update命令中指定版本进行安装或更新，则始终安装尽可能高的版本，这可能不适合您的稳定性需求。

Docker已安装但尚未启动。 已创建docker组，但未向该组添加任何用户。

2、要安装特定版本的Docker Engine - Community，请列出repo中的可用版本，然后选择并安装：

a. 列出并对您的仓库中可用的版本进行排序。此示例按版本号对结果进行排序，从最高到最低，并被截断：



返回的列表取决于启用的存储库，并且特定于您的CentOS版本（在此示例中以.el7后缀表示）。

b. 通过其完全限定的包名称安装特定版本，包名称（docker-ce）加上从第一个冒号（:)开始的版本字符串（第2列），直到第一个连字符，用连字符分隔（ - ）。示例：docker-ce-18.09.1



Docker已安装但尚未启动。 已创建docker组，但未向该组添加任何用户。

3.开启docker



4. 通过运行hello-world映像验证Docker Engine - 社区是否已正确安装。



此命令下载测试映像并在容器中运行它。 当容器运行时，它会打印一条信息性消息并退出。

Docker Engine - 社区已安装并正在运行。 您需要使用sudo来运行Docker命令。 继续Linux postinstall以允许非特权用户运行Docker命令和其他可选配置步骤。

### 升级docker engine community

要升级Docker Engine Community，请按照安装说明选择要安装的新版本。

# 在windows7中安装Docker

# 在Windows10中安装Docker（重点）

## 参考资料

https://blog.csdn.net/hunan961/article/details/79484098

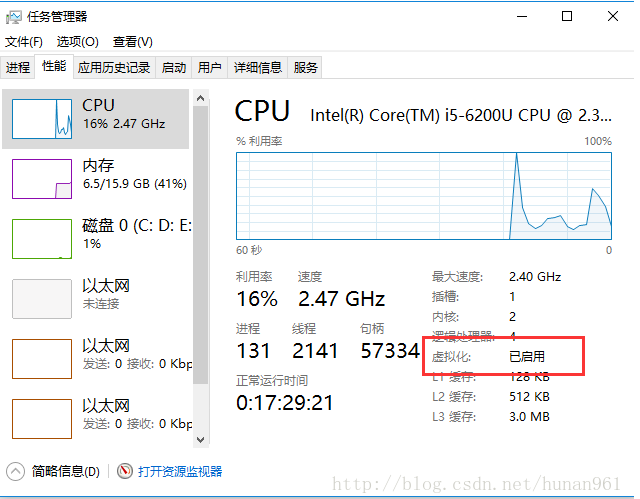
## 环境准备

Docker for Windows是一个Docker Community Edition（CE）应用程序。Docker for Windows安装包包含了在Windows系统上运行Docker所需的一切。如果你不想装虚拟机，想直接在你的Windows操作系统中安装与学习使用docker，那么你首先得查看你的系统是否满足Docker for Windows的安装与使用要求。

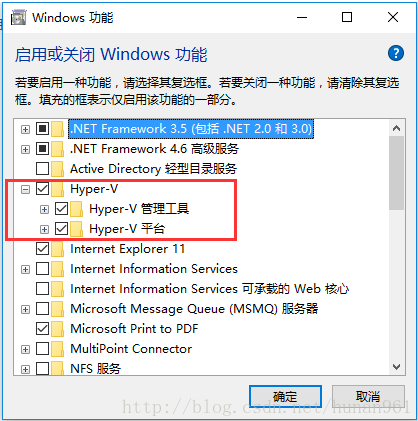
（1）Docker for Windows的当前版本运行在64位Windows 10 Pro，专业版、企业版和教育版（1607年纪念更新，版本14393或更高版本）上。Ps:家庭版是不行的，如果你是家庭版，那么一是升级到专业版，二是安装Docker Toolbox。

（2）如果你满足Docker for Windows的环境条件了，那么首先检查电脑的虚拟化开启了没有：进入任务管理器（ctrl+alt+delete），点击性能->cpu ,查看虚拟化是否已启用，如果虚拟化是已禁用，那么你需要重启电脑进入bios开启虚拟化（我们的笔记本cpu都是支持虚拟化的，重启时进入bios按esc -> 再按f12 -> 去开启虚拟化）

（3）开启虚拟化，重启后，进入任务管理器看虚拟化是否已启用



（4）然后再是进入电脑的控制面板->程序->启用或关闭Windows功能->把Hyper-v勾上，启用后电脑会重启，后面就可以下载并安装Docker for Windows了。



## 下载安装

1、进入网址<https://docs.docker.com/docker-for-windows/install/#download-docker-for-windows> 下载并安装。我安装的是稳定版。安装过程没什么要注意的。



2、启动以后会出现在桌面的右下角区域，鼠标放上去以后显示Docker is running表示启动成功，第一次安装启用好像是会弹出个Docker Cloud登录界面，去注册然后登录，使用和git有点类似。

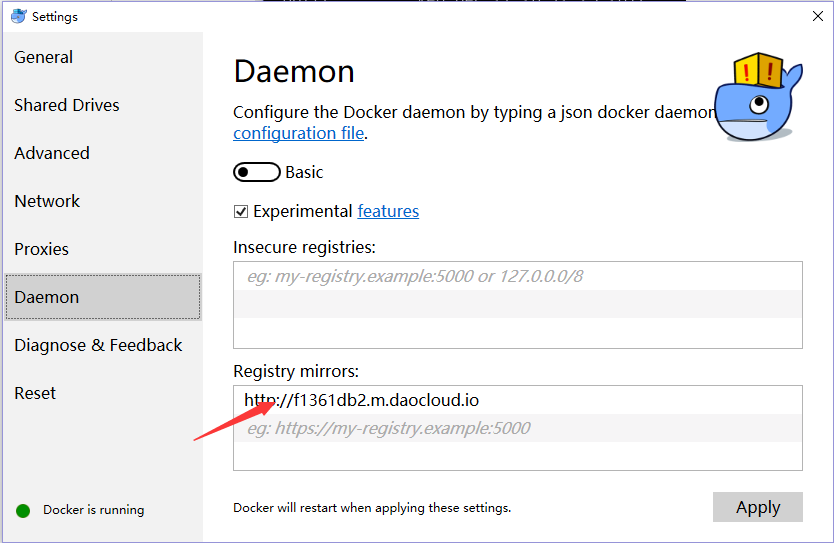


## 配置docker加速器

http://guide.daocloud.io/dcs/daocloud-9153151.html

我们配置daocloud加速器，在桌面右下角状态栏中右键 docker 图标，修改Docker Daemon 标签页中的信息，把下面的地址:

http://f1361db2.m.daocloud.io写上，如图：



点击apply，docker将会重启。

# docker入门

## 快速入门的网址

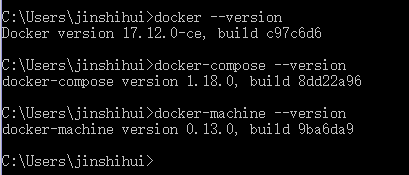
https://docs.docker.com/get-started/

## 检查Docker，Compose和Machine的版本

docker --version

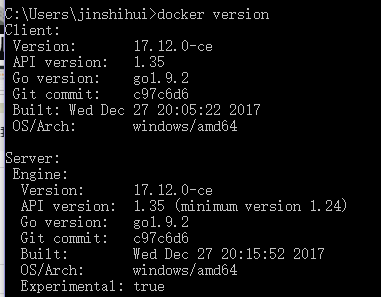
docker-compose --version

docker-machine --version

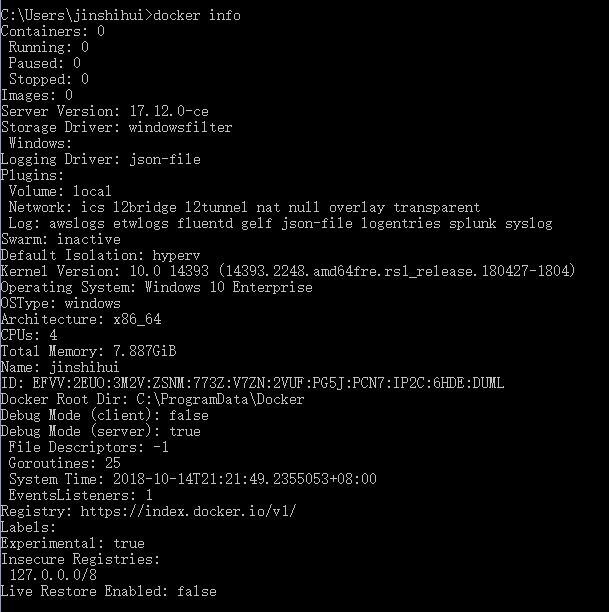


## 检查版本信息，并确保docker命令正常工作

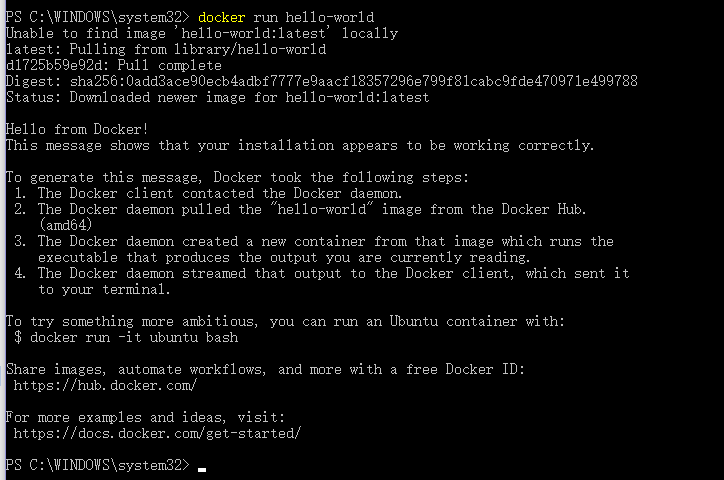
### docker version



### docker info



## 运行docker run hello-world以测试从Docker Hub中拉取镜像并启动容器



## 使用命令docker run -it ubuntu bash运行一个Ubuntu容器，我之前已经拉取过这个容器了，大概是几十兆吧，所以直接启用了，输入exit命令停止容器



说明：ubuntu和daocloud.io/library/ubuntu这是两个不同的镜像。

## 运行命令docker run -d -p 80:80 --name webserver nginx 启动一个Dockerized webserver 会下载nginx容器镜像并启动它，然后再打开浏览器键入[http://localhost](http://localhost/)



## 运行docker ps 命令，检查容器的详细信息



## 停止或移除容器和镜像

如果你想停止网络服务器，输入：docker stop webserver然后重新启动docker start webserver，要使用单个命令停止并删除正在运行的容器，请键入： docker rm -f webserver，这将删除容器，但不是 nginx镜像。您可以列出本镜像docker images。你可能想要保留一些镜像，这样你就不必再从Docker Hub中取出它们了。要删除不再需要的镜像，请使用docker rmi后跟镜像ID或镜像名称。例如docker rmi nginx

### 停止web server

docker stop webserver

### 启动web server

docker start webserver

### 列出所有的镜像

docker images

### 停止并删除正在运行的容器

docker rm –f webserver

### 检查容器的详细信息

docker ps

### 删除镜像

docker rmi nginx

# docker的常用配置

## 在PowerShell中设置 tab键自动补全（其实用的都是cmd.exe）

1、启动一个PowerShell（即以管理员身份运行）。搜索PowerShell，右键单击，然后选择以管理员身份运行。在PowerShell提示符下键入：

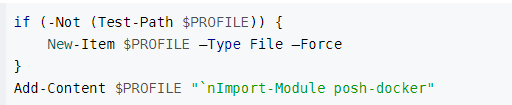
Set-ExecutionPolicy RemoteSigned

2、检查策略设置是否正确，运行：get-executionpolicy   
应该返回RemoteSigned

3、安装posh-dockerPowerShell模块以自动完成Docker命令，键入：Install-Module posh-docker或者，要仅为当前用户安装模块，键入：   
Install-Module -Scope CurrentUser posh-docker

4、安装完成后，只能为当前PowerShell启用自动完成功能，输入：Import-Module posh-docker

5、为了在所有PowerShell会话中保持Tab完成状态$PROFILE，请在PowerShell提示符处输入



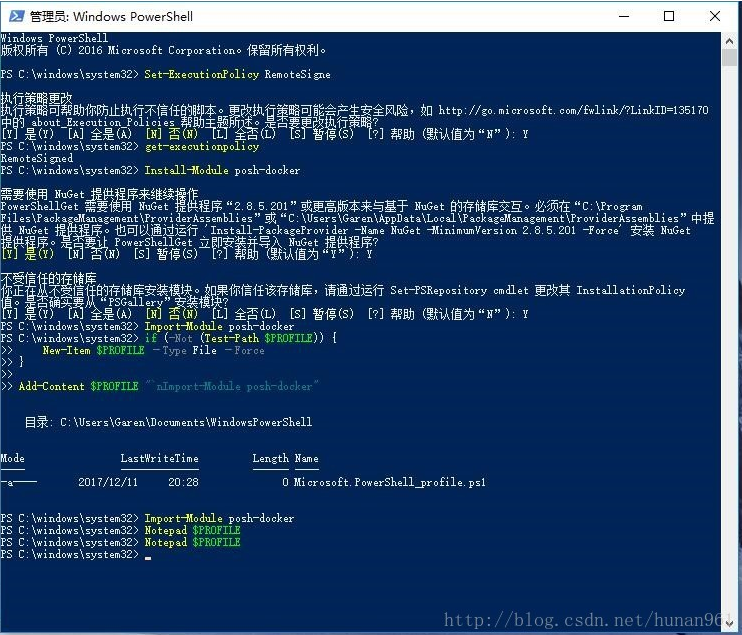
这将创建一个$PROFILE如果不存在，并将此行添加到文件中

Import-Module posh-docker

要检查文件是否已正确创建，或只需手动编辑，请在PowerShell中键入以下内容

Notepad $PROFILE

打开一个新的PowerShell会话。现在，当你键入的前几个字母后按Tab键，Docker命令（如开始，停止，运行及其选项）以及容器和映像名称现在都应该自动完成。

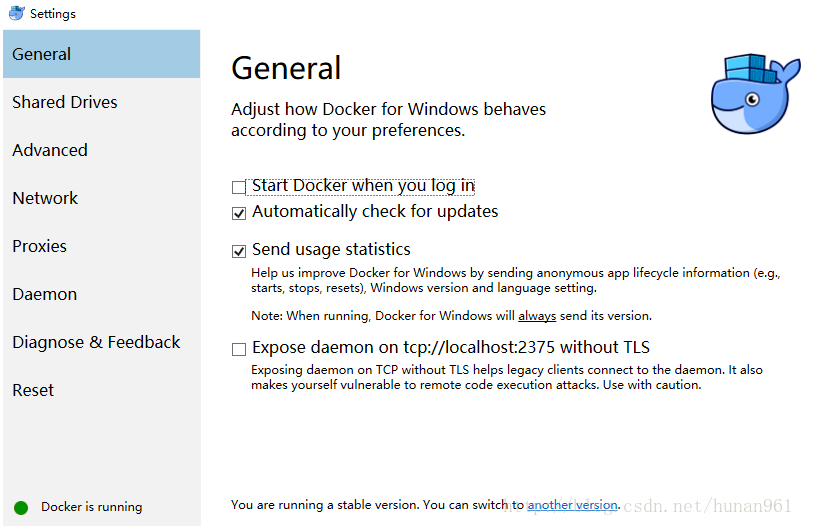


## 设置

找到右下角的docker图标，右击选择settings进去

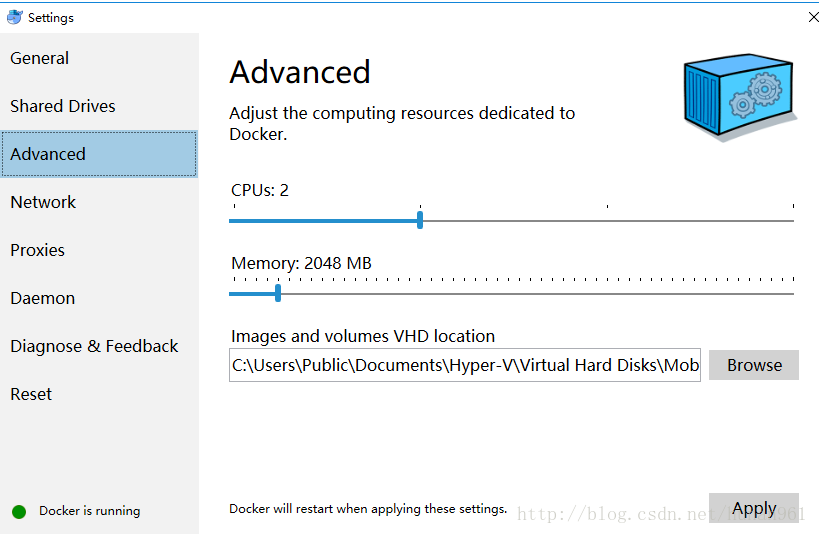
### General

这里是设置docker开机自启，应用程序启动时检查更新，发布使用情况统计信息



### Advanced

分配cpu数量与内存量

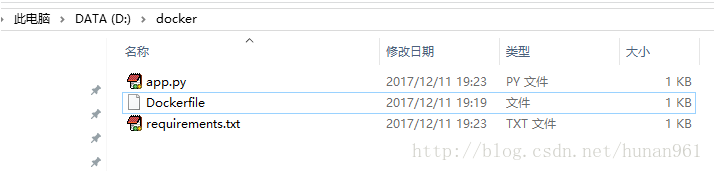


# 用Dockerfile定义一个镜像（重点）

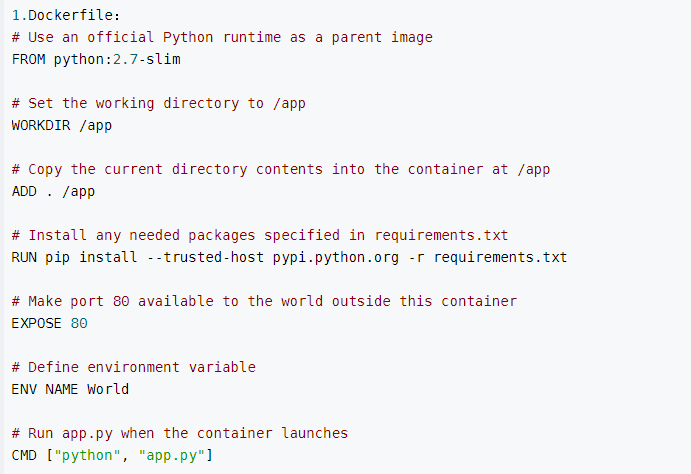
使用Docker，你可以将一个可移植的Python运行库作为一个镜像，不需要安装。然后，您的构建可以将基础Python镜像与应用程序代码一起包括在内，确保您的应用程序，依赖项和运行时都一起运行。 这些可移植的镜像是由一个叫做Dockerfile的东西来定义的。

## 新建目录文件

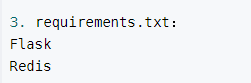
创建一个空目录，我取名叫docker，然后分别在里面新建三个文件：Dockerfile，app.py，requirements.txt



三个文件中的内容分别设计为如下：





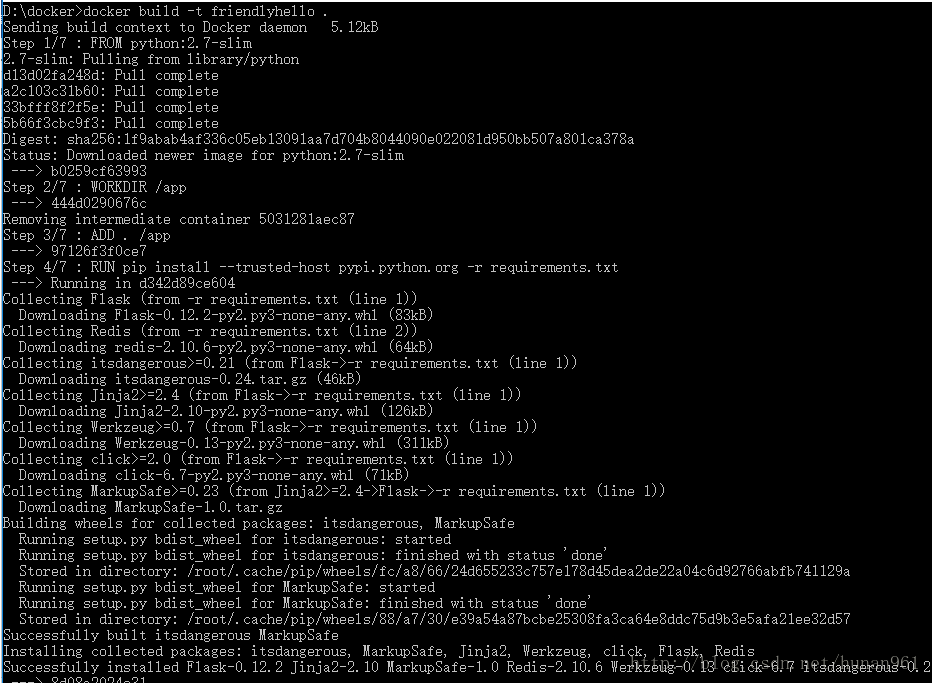


## 构建镜像

在docker目录下打开cmd.exe运行命令（确保能够找到Dockerfile文件，镜像取名叫friendlyhello）

docker build -t friendlyhello .

说明：千万不要落了上面那行后面的那个点，命令包括后面的那个点。



然后再执行命令：

docker images

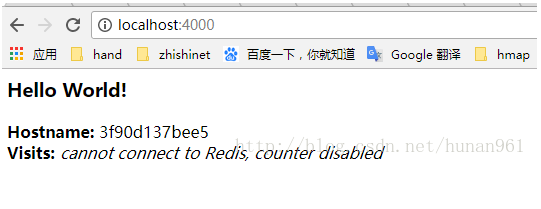


运行镜像程序

docker run -p 4000:80 friendlyhello



可以看到Python正在为应用程序提供消息的[http://0.0.0.0:80](http://0.0.0.0/)。但是，这个消息来自容器内部，它不知道我们将该容器的端口80映射到4000，从而打开URL： [http://localhost:4000](http://localhost:4000/)



## 停止容器运行

首先在接着上面的操作步骤后按下ctrl+c在终端退出，这并不意味着镜像停止运行了。

键入docker container ls 列出正在运行的容器





运行命令：   
docker container stop <Container NAME or ID>   
停止容器。否则在下一步中重新运行容器时，将会收到错误响应



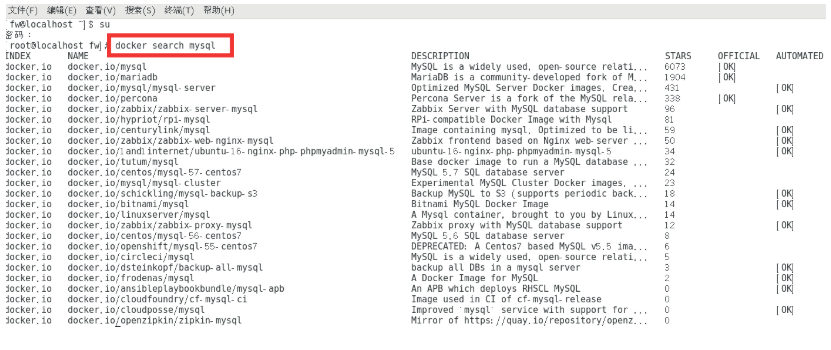
# 常用的docker命令（重点）

## dokcer镜像命令

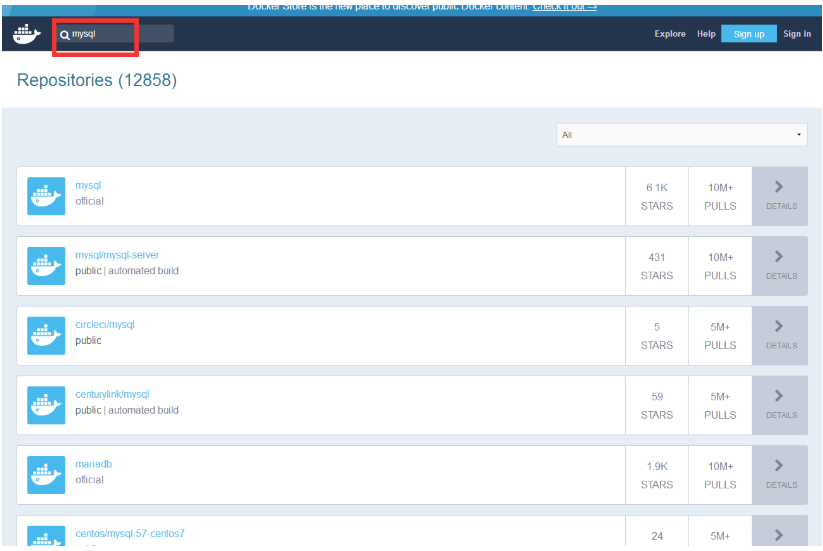
### 镜像检索

docker search 镜像名

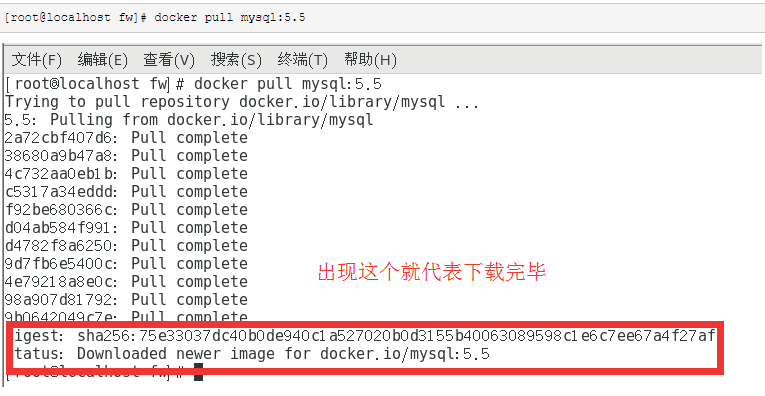
示例：docker search mysql



上面查询到的列表其实就是 https://hub.docker.com/explore/ 里面的



使用命令查看下载 mysql 5.5 版本（默认下载最新的，通过 tag 可以下载指定版本）



### 镜像下载

docker pull 镜像名

### 下载指定版本的镜像

docker pull 镜像名:tag

说明：:tag是可选的，tag表示标签，多为软件的版本，默认是latest

### 镜像列表

docker images

### 删除指定的镜像

docker rmi 镜像id

### 删除所有的镜像

docker rmi ${docker images -q}

### 将修改的镜像提交

docker commit 镜像ID 起一个新的名字

示例

docker commit 镜像ID learn/ping

//相当于创建了一个新的镜像

### 安装ping命令

docker run 镜像名 apt –get install –y ping

### 将镜像保存到Docker hub上

docker push 镜像名

## 容器命令

### 运行镜像为容器（重点）

docker run --name 容器名 –d 镜像名

示例：

运行web服务器：

docker run -d -p 80:80 httpd

### 查看运行中的容器

docker ps

### 查看运行中的容器

docker inspect 镜像ID(或者镜像名称)

//返回docker对象的低级信息



### 查看所有的容器（运行的和停止的）

docker ps –a

### 停止容器

docker stop 容器名称/容器id

### 启动容器

docker start 容器名称/容器id

### 删除指定的容器

docker rm 容器Id

### 删除所有容器

docker rm ${docker ps –a -q}

### 查看当前容器日志

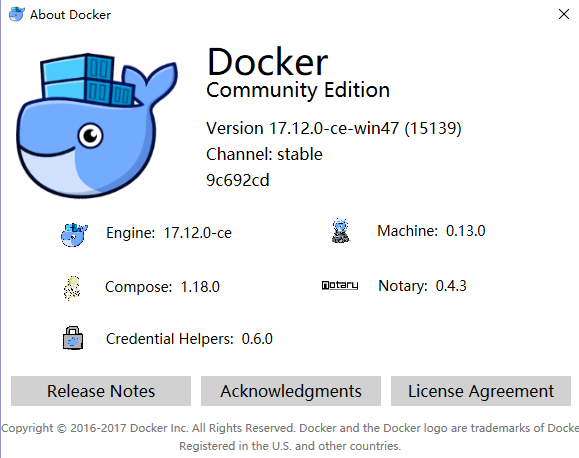
docker logs容器名称/容器id

### 登录容器

docker exec –it容器名称/容器id bash

## 查看docker的信息（图形化界面）

右键docker的图标，选择about docker



## 重启Docker daemon（docker服务器）

systemctl restart docker.service

# docker镜像地址（重点）

https://hub.daocloud.io/

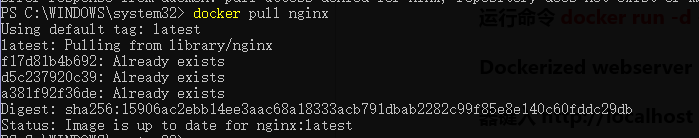
//这是daocloud的镜像市场

https://hub.docker.com/

//这是docker官方的镜像地址

# 常用的精选镜像

## nginx



## mysql

## redis

## java

## centos

## tomcat

docker pull tomcat

## ubuntu

## RabbitMQ

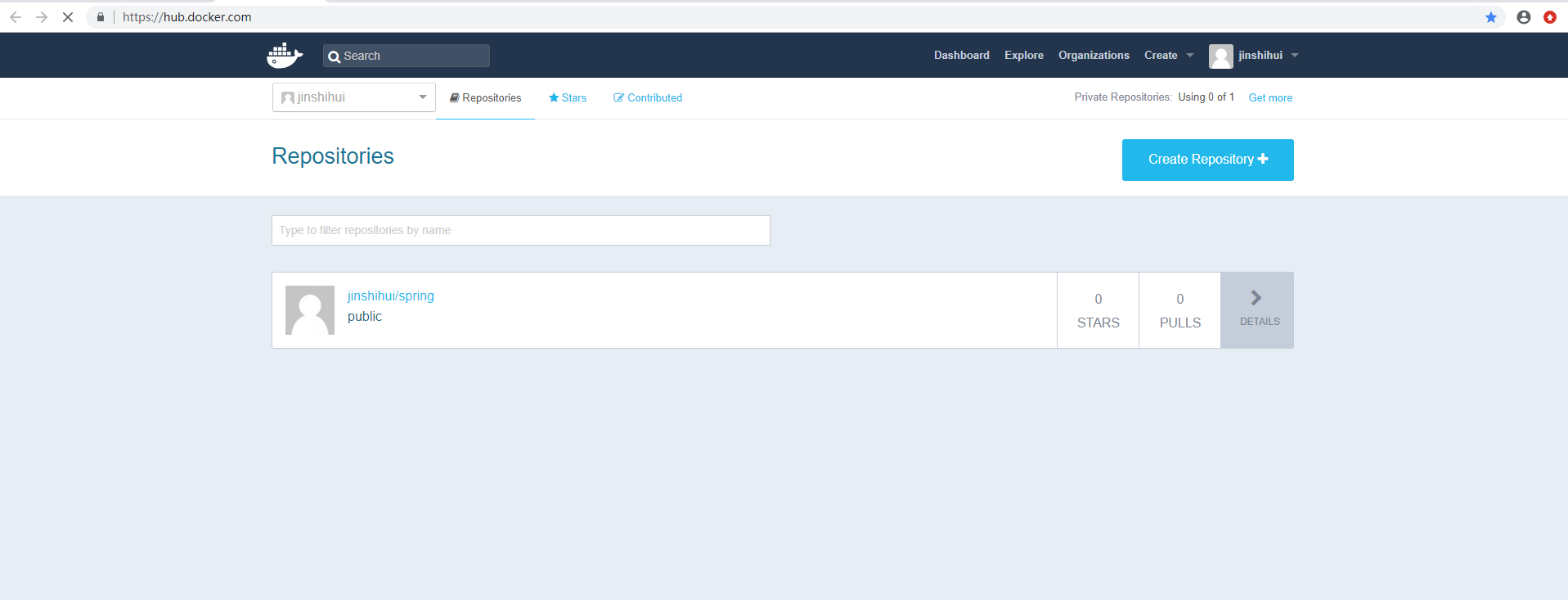
## Docker Registry

## Mongdb

# Docker Hub的常用操作（重点）

## 登录到Docker Hub

docker和git的操作有类似之处，所以docker也有远程仓库，如果前面已经注册过并登录了docker cloud，那么访问网址：https://hub.docker.com 在里面创建存储库，否则先注册。

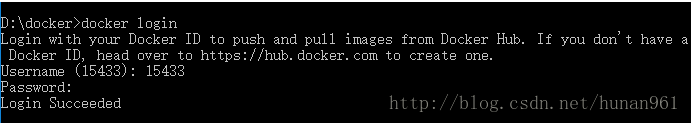


## push镜像

  前面在本地创建了一个friendlyhello的镜像，现在要把它push到   
自己的docker hub的存储库中去，首先：

### 1、登录docker hub

docker login



### 2、标记镜像

把镜像放入wangliguo存储库并标记为test

docker tag 镜像名 15433/wangliguo:test



### 3、查看镜像

docker images

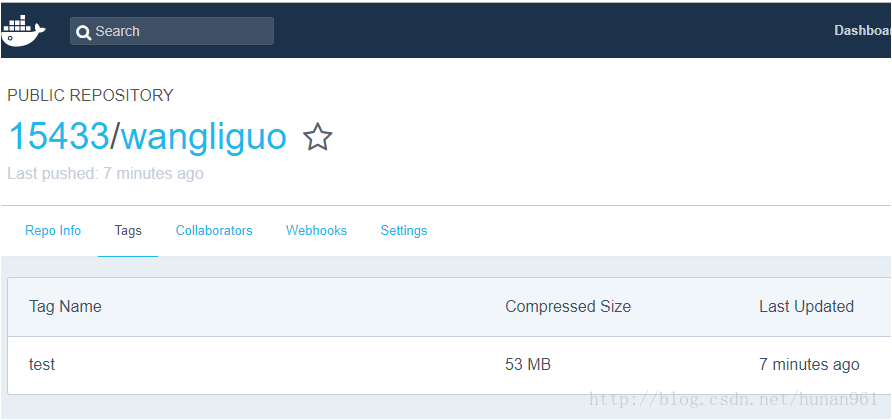


### 4、发布镜像

docker push 15433/wangliguo:test



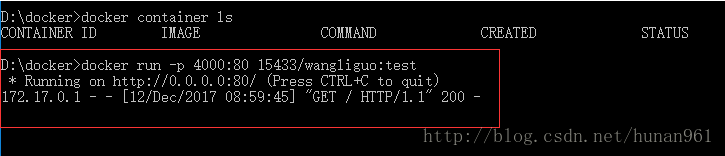
### 5、Docker Hub上查看镜像

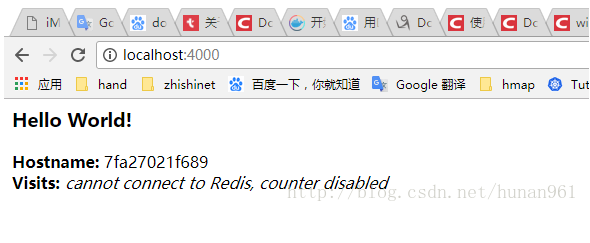


### 6、从远程存储库中提取并运行镜像

现在当远程存储库有了镜像后，就可以从远程存储库提取并运行了

docker run –p 4000:80 15433/wangliguo:test

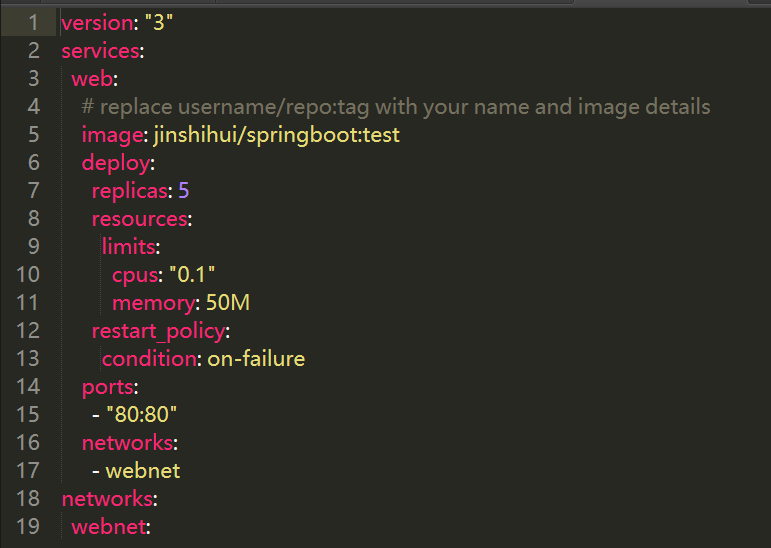




# 服务

在分布式应用程序中，应用程序的不同部分被称为“服务”。例如，想象一个视频共享站点，它可能包括用于将应用程序数据存储在数据库中的服务，用于用户上传东西的视频转码服务，为前端服务等等。 服务实际上只是“生产中的容器”。服务只运行一个镜像，但它编码镜像运行的方式，应该使用哪个端口，容器应该运行多少个副本，以便服务具有所需的容量，以及等等。使用Docker平台定义，运行和扩展服务非常简单，只需编写一个docker-compose.yml文件即可。

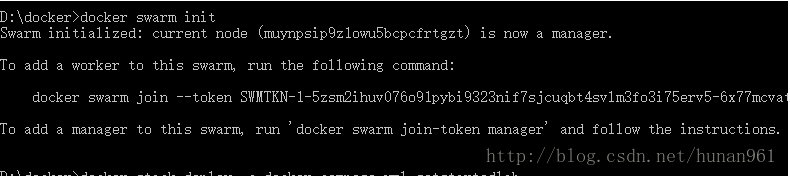
## 创建一个docker-compose.yml文件



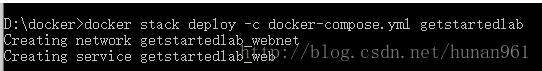
拉取的是上个步骤中的存储库的tag，运行该镜像的5个实例作为一个服务调用web，限制每个使用，最多10％的CPU（跨所有核心）和50MB的RAM。 如果一个失败，立即重新启动容器。 将主机上的端口80映射到web端口80。 指导web容器通过一个负载平衡的网络共享80端口webnet。（在内部，容器本身将web在临时端口上发布到 端口80）。 webnet使用默认设置（这是一个负载平衡覆盖网络）定义网络。

## 运行新的负载均衡应用程序

先运行命令：docker swarm init

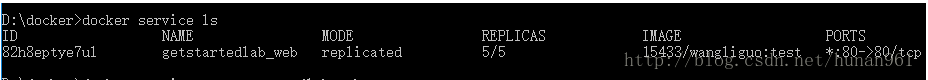


然后再运行命令：docker stack deploy -c docker-compose.yml getstartedlab   
给它取名叫getstartedlab

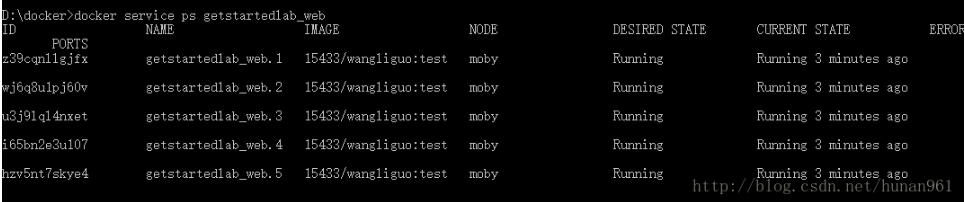


服务堆栈在这台主机上运行了5个部署镜像的容器实例

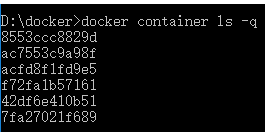
运行命令查看：docker service ls



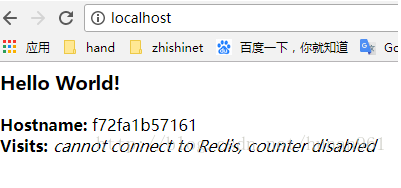
在服务中运行的单个容器称为任务，可以看到上面有getstartedlab\_web的服务 ，运行命令docker service ps getstartedlab\_web 查看此服务下的任务：



如果只列出系统中的所有容器，也会显示任务，但不会被服务过滤：   
运行命令：docker container ls –q



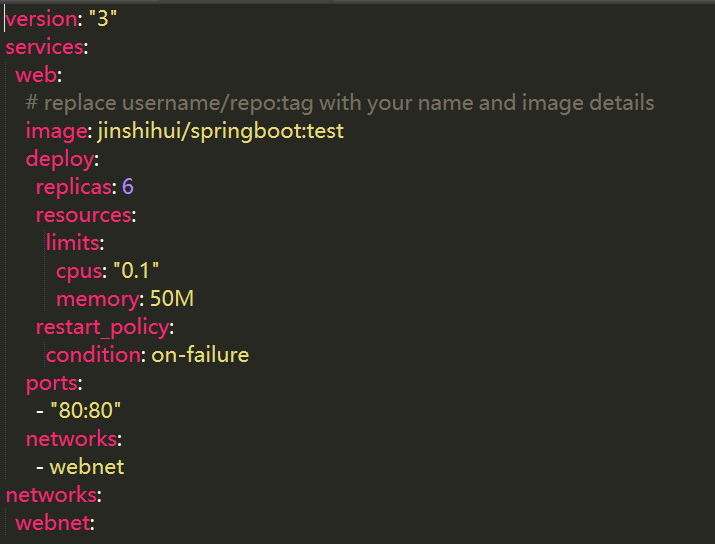
然后打开浏览器，键入[http://localhost](http://localhost/)



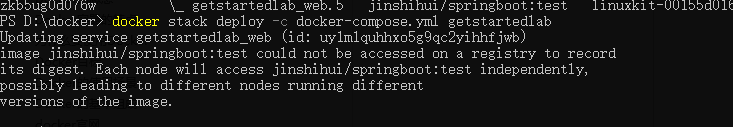
点击刷新多次，可以发现Hostname的更改，以循环方式选择5个任务中的一个来响应。容器ID将与前一个命令（docker container ls -q）的输出相匹配。

## 更改应用程序

比如更改docker-compose.yml中的replicas值，保存更改并重新运行docker stack deploy命令来更新应用程序。



运行命令：   
docker stack deploy -c docker-compose.yml getstartedlab



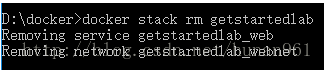
Docker会做一个就地更新，然后重新运行docker container ls -q以查看重新配置的已部署实例

可以看到之前是6个，现在是7个，刚好多了一个任务。   
ps:电脑中还运行这之前步骤中从docker hub中拉取并运行着的那个任务，所以会看到6个和7个

## 关闭应用程序和集群

### 关闭应用程序

docker stack rm getstartedlab



### 关闭集群

docker swarm leave –force



现在服务中的任务都关闭了以后再运行命令：   
docker container ls –q



说明当前容器中没有任务运行

# 集群

## 概念

swarm是运行Docker并加入到一个集群中的一组机器。但是现在它们将由群集管理器在群集上执行，群体中的机器可以是物理的或虚拟的。加入群体后，他们被称为节点。

Swarm管理人员可以使用多种策略来运行容器，比如“最空的节点”（emptiest node），它使用容器填充最少使用的机器。或“全局”，这确保了每台机器只能得到指定容器的一个实例。您可以指示swarm manager在Compose文件中使用这些策略。群体管理者是群体中唯一可以执行你的命令的机器，或者授权其他机器作为工作者加入群体。工人提供能力，并没有权力告诉任何其他机器可以做什么和不可以做什么。 到目前为止，之前都是在本地机器上以单主机模式使用Docker。但是Docker也可以切换到群集模式，这就是使用群集的原因。启用群模式使当前机器成为群管理器。则Docker将运行您正在管理的群集上执行的命令，而不仅仅是在当前的机器上。

## 创建一个集群

一个集群由多个节点组成，可以是物理机或者是虚拟机，基本的概念很简单，运行docker swarm init启用集群模式，使当前的机器成为集群管理器，然后docker swarm join在其他机器上运行，让它们作为工人加入群体，下面将使用虚拟机快速创建一个双机群集，并将其变成群集。方法如下：

1、以管理员运行cmd.exe 这里必须是管理员运行，不然后续操作权限不够

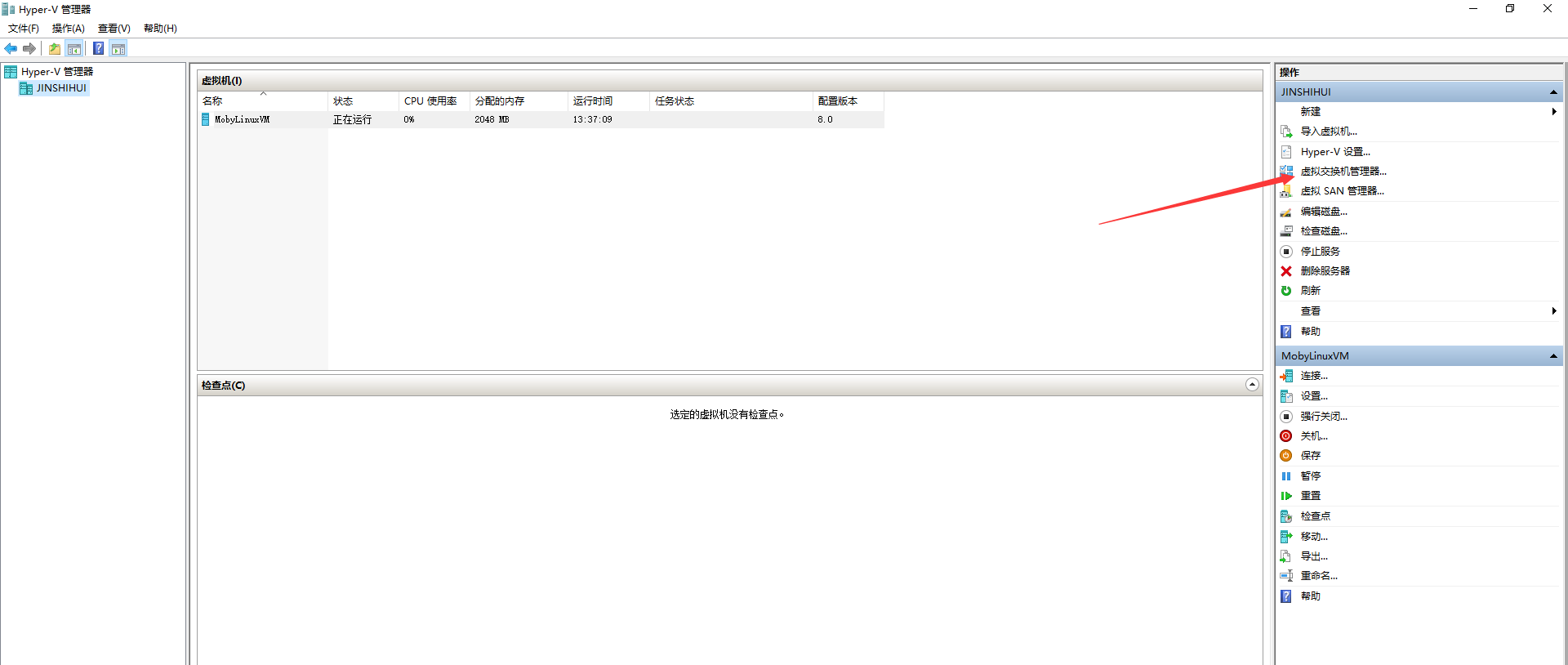


2、运行docker swarm init启用群模式，使当前的机器成为群管理器，然后docker swarm join在其他机器上运行 ，让它们作为工人加入群体。

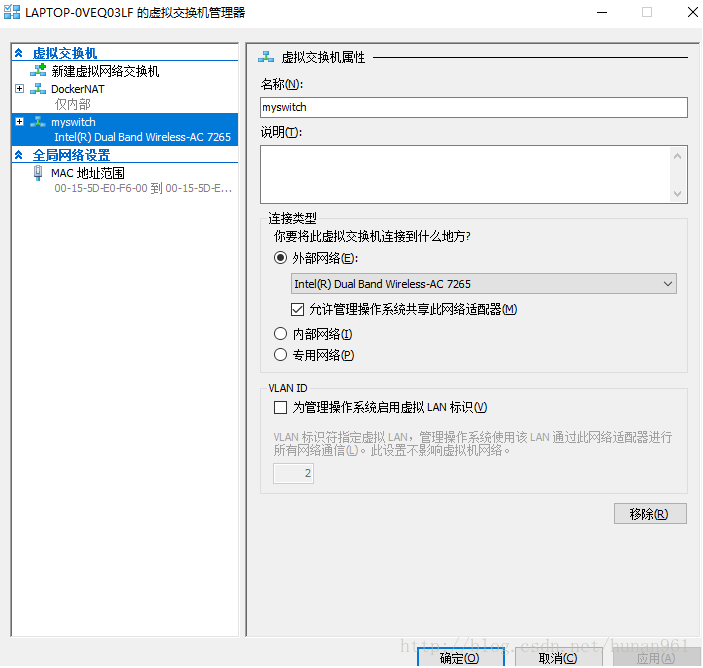
3、启动Hyper-V管理器

点击开始—windows管理工具—Hyper-V管理器

4、单击右侧菜单中的虚拟交换机管理器



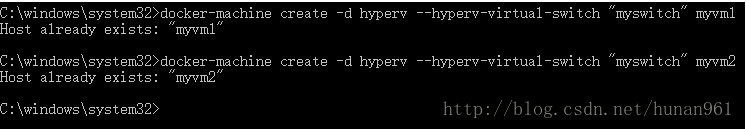
5、单击创建类型为外部网络的虚拟交换机，给它的名称myswitch，并检查框共享您的主机的活动网络适配器



6、使用节点管理工具创建几个虚拟机docker-machine：

docker-machine create -d hyperv –hyperv-virtual-switch “myswitch” myvm1

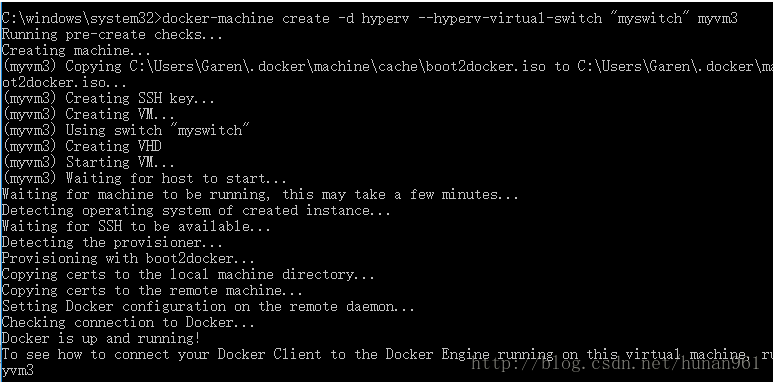
docker-machine create -d hyperv –hyperv-virtual-switch “myswitch” myvm2

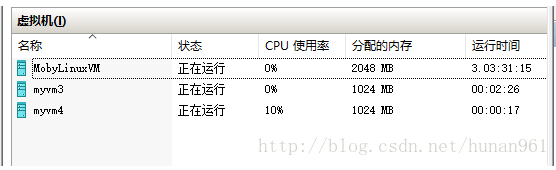


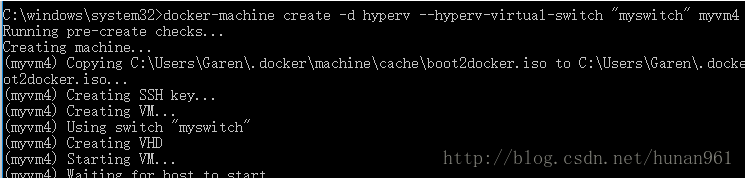
之前我已经创建过了，所以我现在再新建两个：myvm3 和 myvm4

docker-machine create -d hyperv –hyperv-virtual-switch “myswitch” myvm3

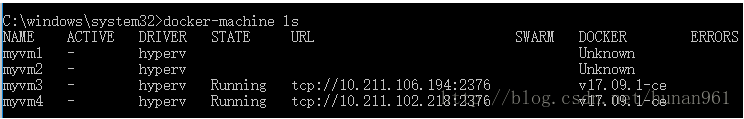
docker-machine create -d hyperv –hyperv-virtual-switch “myswitch” myvm4







7、现在创建了两个虚拟机，分别命名为myvm3和myvm4。使用命令   
docker-machine ls 列出机器并获取其IP地址



## 初始化集群并添加节点

1、先使用命令docker-machine ssh myvm3

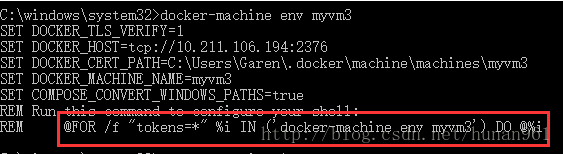


然后让myvm3 成为一个管理员：docker swarm init  
注意红框中的内容，这是后面的myvm4 加入集群要执行的命令

## 在集群上部署应用程序

### 1、docker-machine这个命令为swarm管理器配置一个shell

运行命令：docker-machine env myvm3

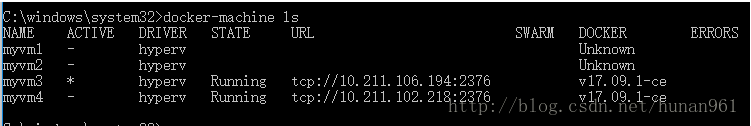


注意最后一行的内容：

我这里是@FOR /f "tokens=\*" %i IN ('docker-machine env myvm3') DO @%i   
复制粘贴运行它

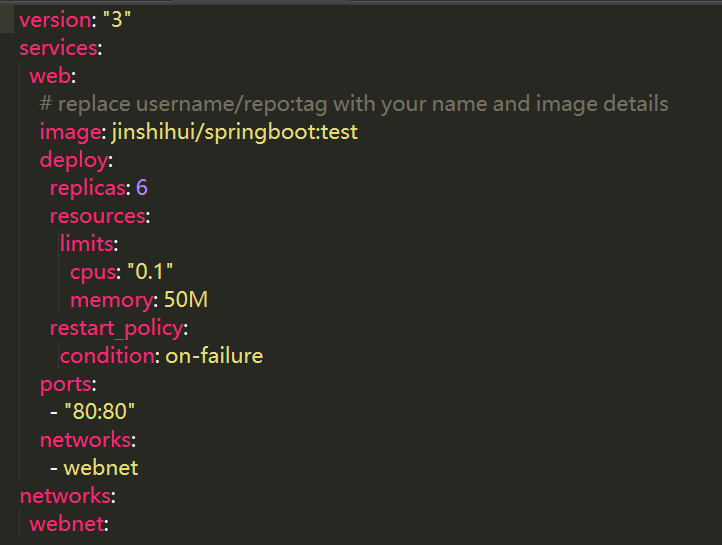


再运行docker-machine ls以验证它myvm3 是否为活动机器



### 2、在swarm管理器上部署应用程序

以部署我们之前docker-compose.yml服务为例



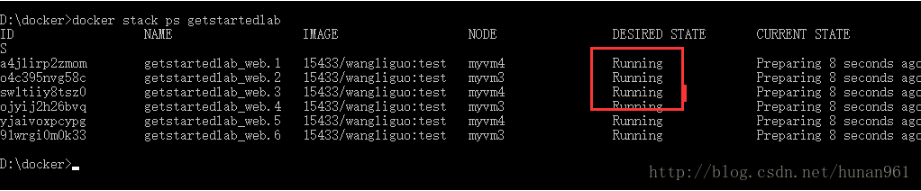
首先在这个以管理员身份打开的cmd中进入到docker-compose.yml文件的所在的目录中去，我的因为在D/docker中，如果不过来，那么执行命令：   
docker stack deploy -c docker-compose.yml getstartedlab会出现



进入目录后执行上面那条命令



再执行命令docker stack ps getstartedlab 查看服务详情



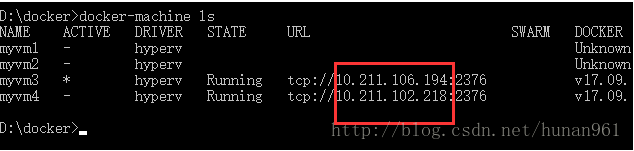
如果发现state存在shutdown的情况，那应该是你在之前运行过命令：

docker stack deploy -c docker-compose.yml getstartedlab 而没有把getstartedlab移除掉，这时应该执行命令：docker stack rm getstartedlab 把getstartedlab移除掉

说明：重新运行docker stack deploy -c docker-compose.yml getstartedlab 和 docker stack ps getstartedlab，命令就好了。

### 3、浏览器访问集群的网址

如图红框所示，浏览器上访问其中的任何一个，然后刷新，将看到6个可能的容器ID，它们都是随机循环的，展示了负载平衡。



### 4、其他的一些命令

（1）如果修改了docker-compose.yml文件后，执行命令：

docker stack deploy -c docker-compose.yml getstartedlab

再次运行部署这些更改

（2）移除应用程序

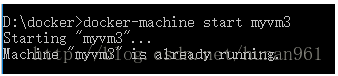
docker stack rm getstartedlab

（3）离开集群

docker swarm leave –force

（4）重新启动停止的虚拟机

docker-machine start <machine-name>



# PowerShell

## 参考资料

https://www.cnblogs.com/lavender000/p/6935589.html

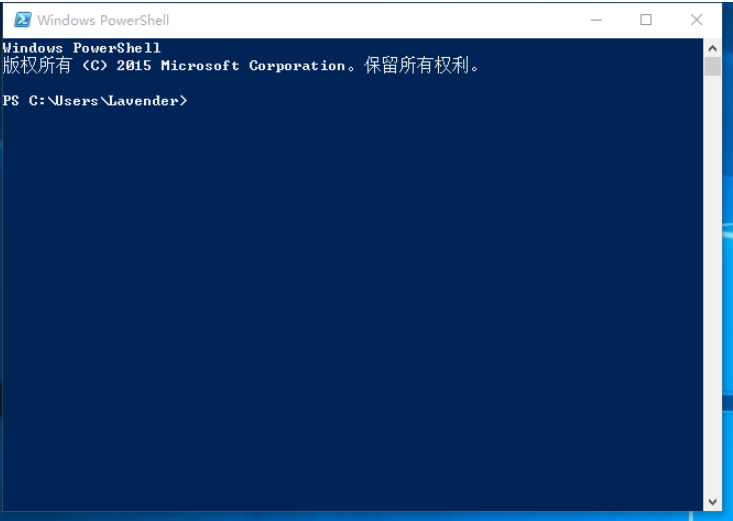
## 基本概念

Windows PowerShell是基于任务的命令行管理程序和脚本语言，专为进行系统管理而设计。在 .NET Framework 的基础上构建的 Windows PowerShell 可帮助 IT 专业人士和高级用户控制和自动执行 Windows 操作系统以及在 Windows 上运行的应用程序的管理。Windows PowerShell 的诞生就是要提供功能相当于 UNIX 系统 BASH 的命令行外壳程序，同时也内建脚本语言以及辅助脚本程序的工具。

## 启动PowerShell

现在的Windows系统中一般都集成了PowerShell，启动方法：

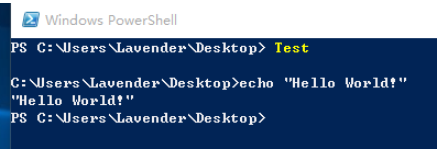
直接开始菜单 -> 所有程序 ->  Windows PowerShell -> Windows PowerShell（不同系统可能路径不一样）。启动后效果如下：



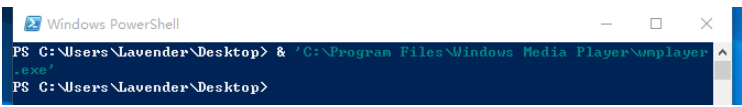
## 如何运行程序、脚本和已有的软件

如果以前经常使用一些别的工具进行开发，可能会有一些可执行文件、Perl脚本、批处理文件等，如果不想放弃这些，可以使用PowerShell直接运行它们。

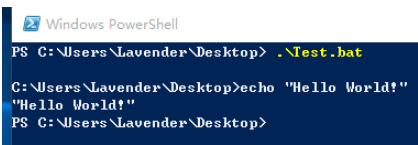
（1）在系统路径下运行程序、脚本、批处理文件或者其它可执行文件，可以直接输入文件名运行。比如我把一个Test.cmd文件放在了”C:\Windows\System32”下， 就可以如下方式运行它



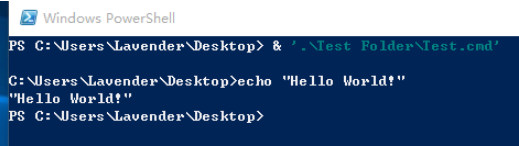
（2）如果文件名字或路径中有空格，需要用（'）引号将命令扩起来，同时在前面加上符号（&），这种叫做调用操作。



（3） 要运行当前目录下的命令，需要文件名前加.\



（4）要运行当前目录下的命令，而且命令名字或路径带有空格，需要同时加上符号（&）和（'）



## 运行PowerShell命令

# 遇到的问题

## 1、[Docker unauthorized: incorrect username or password](https://www.cnblogs.com/jaxer/p/7742186.html)

解决方法：命令行输入docker login，使用docker ID去登录。

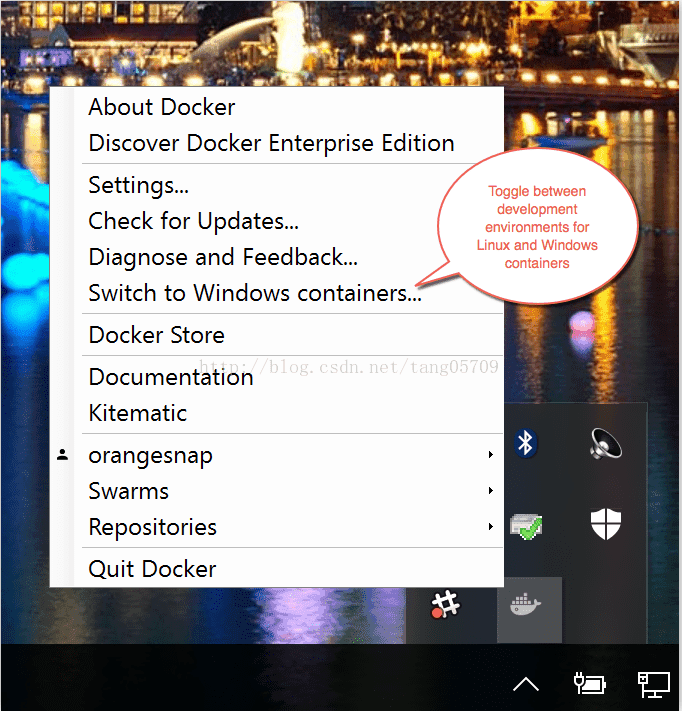


## 2、docker下拉镜像报error pulling image configuration错误

出现这个问题原因为国内网络问题，无法连接到 docker hub。 好在国内已经有  daocloud，docker指定该源即可。添加后重启docker。

## 3、image operating system "linux" cannot be used on this platform

你可以选择切换daemon（Linux or Windows）the Docker



4、