<https://yeasy.gitbooks.io/docker_practice/content/image/pull.html>

<http://c.biancheng.net/view/3134.html>

Ubutun安装docker: sudo apt install docker.io

Docker的官方仓库: https://hub.docker.com

仓库

Dockerfile

镜像/image

Tar文件

容器

Pull

Push

Build

Load

Save

Run

Commit

仓库: 远端git保存

镜像: 集合了环境的文件,用于创建 Docker 容器的模板

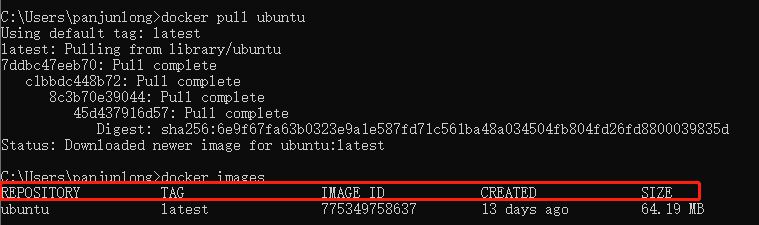
容器: 沙盒

Dockerfile: 用来创建一个自定义的image

Tar文件: 将image打包,复制给他人使用

一/从云端仓库拉取镜像

docker pull ubuntu



哪个仓库 镜像标记 id号(唯一) 创建时间 大小

通过 TAG 信息来区分发行版本，例如 10.04、12.04、12.10、13.04、14.04 等。

如果pull的时候不指定tag,那么会默认pull latest标签,而这个标签命名为default可能更准确

要拉取全部,用 docker pull -a ubuntu

二/运行本地镜像(镜像--->沙盒/容器)



Run:运行容器.eg: docker run -i -t ubuntu

\*\*\*注意:网上相当多的文章是 docker run -i -t ubuntu:15.10 /bin/bash 是错误的

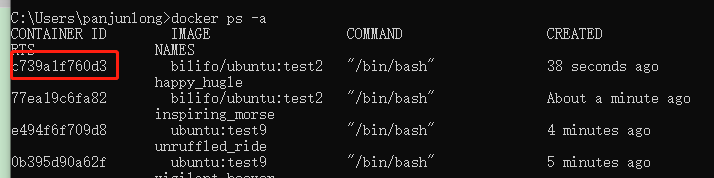
三/保存容器/沙盒的修改(容器/沙盒--->镜像)

\*\*\*注意:因为docker的特性,每次执行这个命令回产生一个新的容器,是种层层堆叠的形式,如果在沙盒运行中做了修改,想要保存下来,就必须用到commit,不然下次启动你的这些操作无法保留,会重置

docker commit -m “描述” -a “作者” 容器名称/容器id 要保存成的镜像名称:tag



\*\*\*注意:红框内的可以是id,该id并非 docker images时查到的id,而是使用 docker ps -l 查到的最近修改id



小细节:ps时,后面的name好像是docker随意打的

这里产生了一个新的疑问,每次运行后都必须 commit 一下,是不是太麻烦了?

解决方式:如果涉及到纯数据操作，其实最终都是落到文件系统的变动，这个可以用volumn ,就是把image里的文件夹映射到host里的文件夹，而不是在image里频繁修改。  
image里保留的只是配置信息，就是一个“壳”.

四/用dockerfile来建立镜像(dockerfile--->容器/沙盒)

使用 docker\_makefiles\android-ubuntu 下的脚本,执行

docker build --tag=”用户/image名:tag名称” dockerfile所在的主目录 //-t用来标记tag

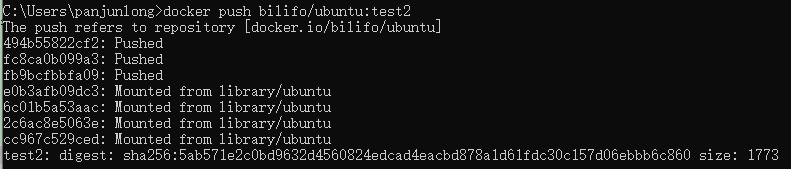


五/上传镜像到docker hub

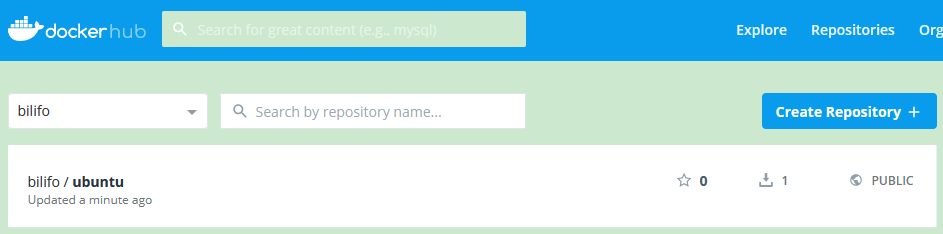
docker login



docker push 镜像名称:tag



让我们去dockerhub上看看,它上传成功了!



六/打包和解包

打包:docker save 镜像:tag > 路径/压缩包名称.tar



解包:docker load < 路径/压缩包名称.tar

七/多开命令窗口

在已经运行的容器,想单开一个命令窗口,用docker exec -it 通过ps查到的镜像的id /bin/bash



Docker 是服务器----客户端架构。命令行运行docker命令的时候，需要本机有 Docker 服务。如果这项服务没有启动，可以用下面的命令启动（[官方文档](https://docs.docker.com/config/daemon/systemd/" \t "http://www.ruanyifeng.com/blog/2018/02/_blank)）。

# service 命令的用法

$ sudo service docker start

# systemctl 命令的用法

$ sudo systemctl start docker

$ docker image ls

$ docker image rm [imageName] # 删除 image 文件

image 文件是通用的，一台机器的 image 文件拷贝到另一台机器，照样可以使用。一般来说，为了节省时间，我们应该尽量使用别人制作好的 image 文件，而不是自己制作。即使要定制，也应该基于别人的 image 文件进行加工，而不是从零开始制作。

Image的生成:

$ docker image build -t koa-demo .

$ docker image build -t koa-demo:0.0.1 .

-t参数用来指定 image 文件的名字，后面还可以用冒号指定标签。如果不指定，默认的标签就是latest。最后的那个点表示 Dockerfile 文件所在的路径，上例是当前路径，所以是一个点。

Dockerfile的编写:(文件名就是dockerfile，没有什么后缀，也不能是其他的名字)

FROM python

ADD test.py .

RUN python test.py

FROM

基于哪个镜像

ADD或COPY

将外部文件添加到容器中.ADD 目标文件路径 容器内路径

RUN

安装软件用

MAINTAINER

镜像创建者

CMD

container启动时执行的命令，但是一个Dockerfile中只能有一条CMD命令，多条则只执行最后一条CMD.

Image的移植

docker save -o 要保存的文件名.tar  要保存的镜像

docker load < 文件名.tar

使用docker search 名称，可以搜索远端仓库时候有该image

Start和run的差别：  
docker run 只在第一次运行时使用，将镜像放到容器中，以后再次启动这个容器时，只需要使用命令docker start 即可。  
docker run相当于执行了两步操作：将镜像放入容器中（docker create）,然后将容器启动，使之变成运行时容器（docker start）。

没有名字的image：

没有成功build的image或临时image。一般删掉，使用docker rmi 容器id来删除

或用

docker tag IMAGEID(镜像id) REPOSITORY:TAG（仓库：标签）

常用命令

docker ps -a 显示出所有容器，包括没有启动的

docker rename old\_name new\_name 重命名

错误集:

安装好后,输入指令报错

Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?

解决:

在/etc/daemon目录下建立daemon.json,输入以下内容:

{ "registry-mirrors":

["https://registry.docker-cn.com"]

}

在重启docker服务:

service docker restart

不行,原来是win10子系统下的ubuntu安装docker不能成功,要用不同的docker软件,不能直接用 sudo apt install docker.io,且注意,win10只能安装ubuntu18.04的那个子系统

# Install packages to allow apt to use a repository over HTTPS

$ sudo apt-get install apt-transport-https ca-certificates curl software-properties-common

# Add Docker's official GPG key

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# Set up the repository

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

# Update source lists

sudo apt-get update

# Install Docker

sudo apt-get install docker-ce

发现还是不行,原因是目前 wsl 是不支持 docker 的守护进程的,虽然也有办法,不过那种ubuntu子系统链接window桌面docker来控制的方式,完全没有必要,因为我就是桌面版下不下来才想用子系统的