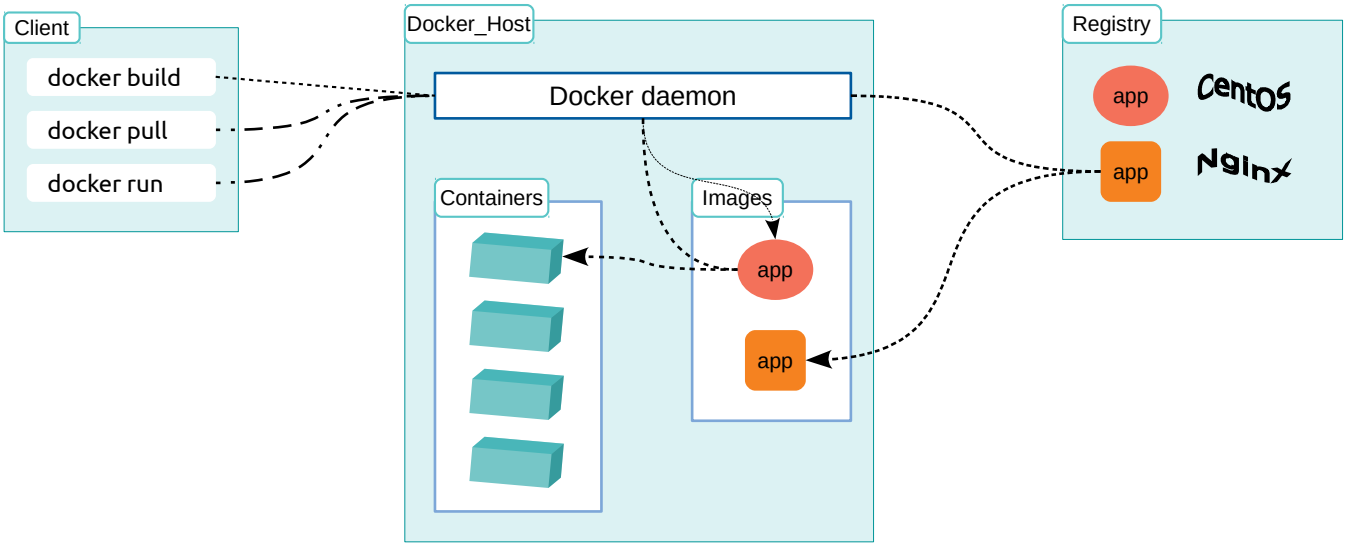
Docker 运行原理图




```
$ docker run ubuntu:15.10 /bin/echo "Hello world"
```

运行一个容器 镜像名 版本号 执行命令

如果本地仓库没有该镜像，那么 docker 会去远程镜像库下载。

```
runoob@runoob:~$ docker run -i -t ubuntu:15.10 /bin/bash
root@0123ce188bd8:/#
```

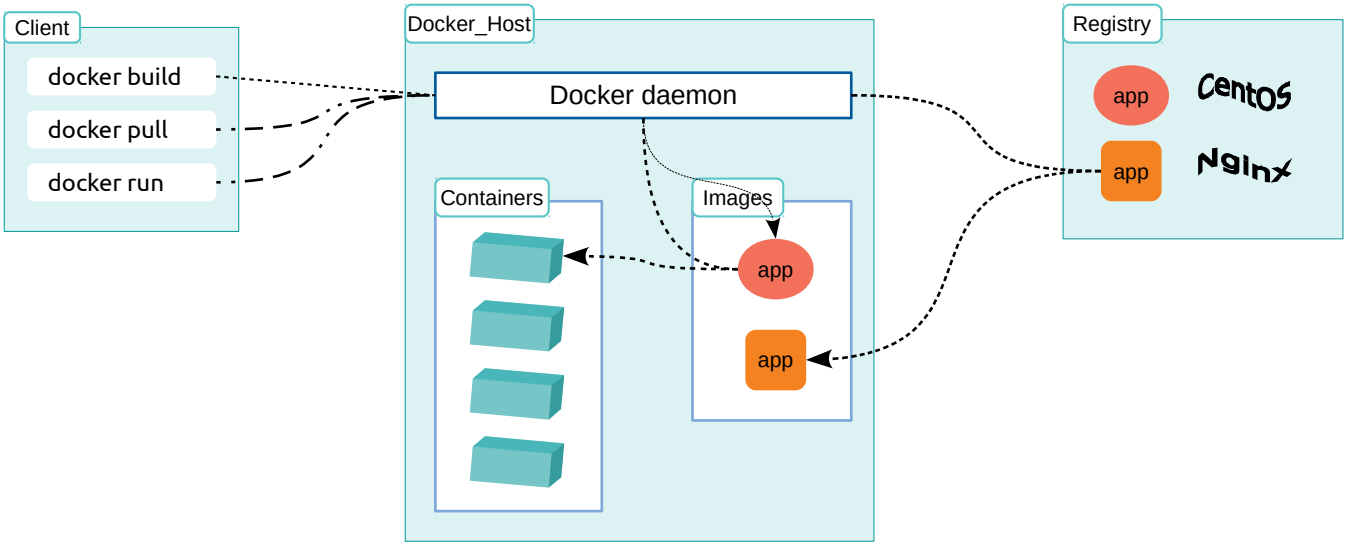
运行并进入容器中，实现与容器对话。

-t：在新的容器中指定一个终端 -i：允许对容器的标准输入（STDIN）进行交互 可以连写为：**-it**

```
runoob@runoob:~$ docker run -d ubuntu:15.10 /bin/sh -c "while true; do echo hello world; sleep 1; done"
2b1b7a428627c51ab8810d541d759f072b4fc75487eed05812646b8534a2fe63
```

-d：在后台运行。

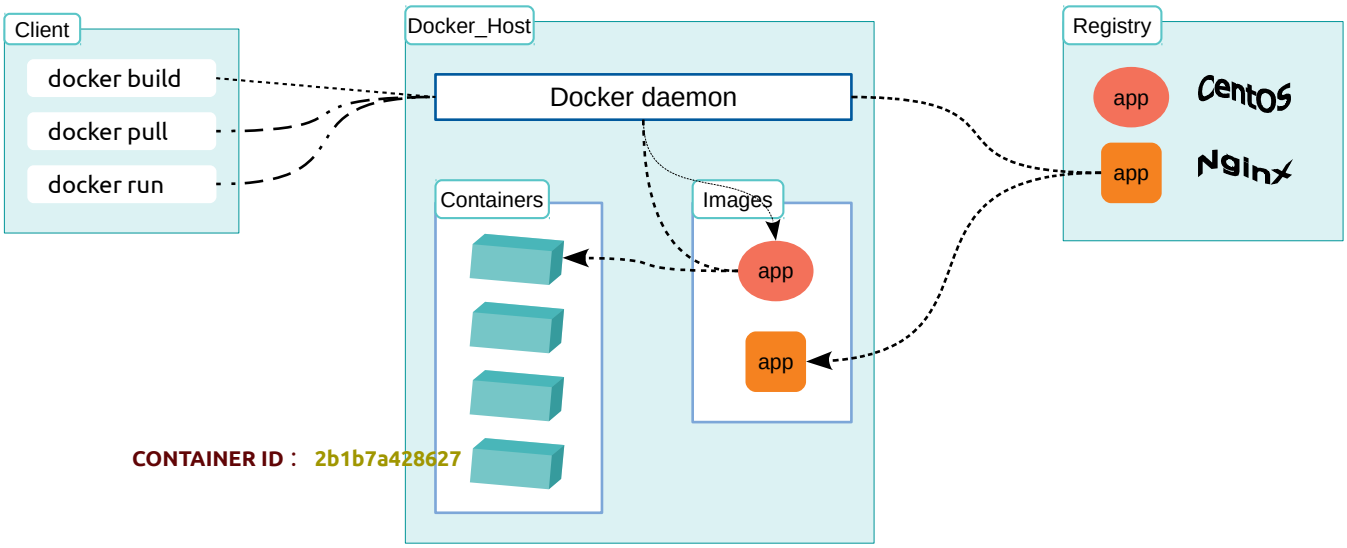
Docker ps 查看容器详情



CONTAINER ID: 容器 ID。
IMAGE: 使用的镜像。
COMMAND: 启动容器时运行的命令。
CREATED: 容器的创建时间。
STATUS: 容器状态。
PORTS: 容器的端口信息和使用的连接类型（tcp\udp）。
NAMES: 自动分配的容器名称。

状态有 7 种：
created （已创建）
restarting （重启中）
running （运行中）
removing （迁移中）
paused （暂停）
exited （停止）
dead （死亡）

Docker logs 查看容器内部输出



runoob@runoob:~$ docker logs 2b1b7a428627

Docker stop 停止容器

runoob@runoob:~$ docker stop 2b1b7a428627

Docker restart 重启容器

runoob@runoob:~$ docker restart 2b1b7a428627

进入容器

runoob@runoob:~$ docker attach

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
1e560fca3906        ubuntu              "/bin/bash"         16 minutes ago      Up 5 minutes                            ubuntu-test

LINSHAODONG@HTJT-LSDONG MINGW64 /e/project/swoft-master
$ docker attach 1e560fca3906
root@1e560fca3906:/# exit
exit

LINSHAODONG@HTJT-LSDONG MINGW64 /e/project/swoft-master
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES

LINSHAODONG@HTJT-LSDONG MINGW64 /e/project/swoft-master
```

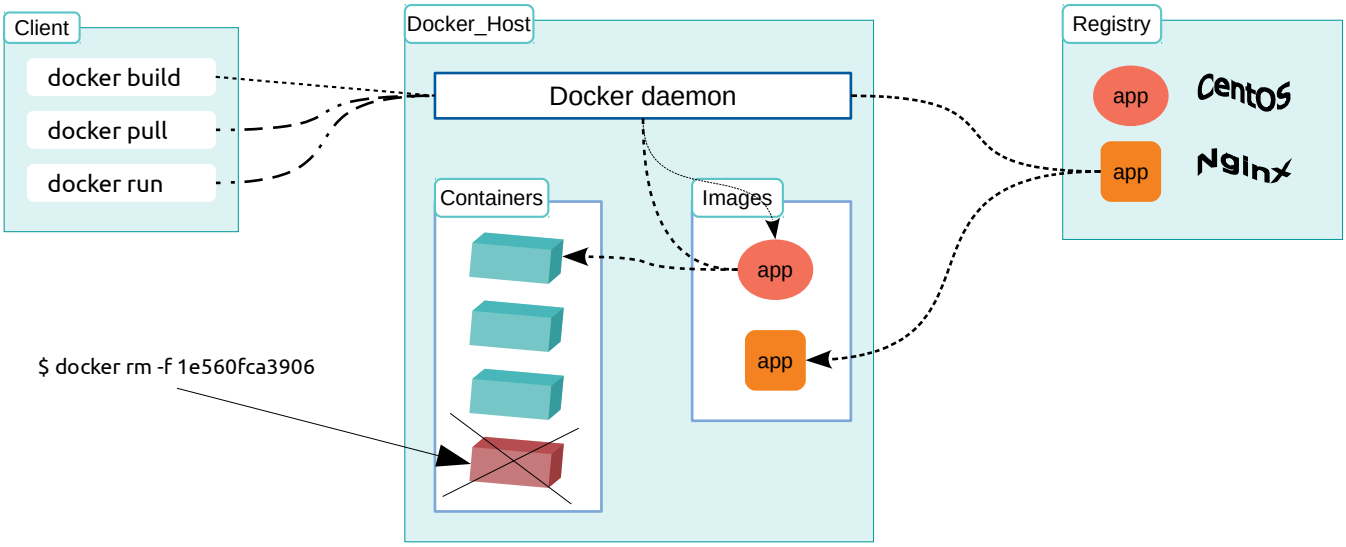
runoob@runoob:~$ docker exec：推荐使用。因为此法退出容器终端，不会导致容器停止。

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
1e560fca3906        ubuntu              "/bin/bash"         19 minutes ago      Up 3 seconds                            ubuntu-test

LINSHAODONG@HTJT-LSDONG MINGW64 /e/project/swoft-master
$ docker exec -it 1e560fca3906 /bin/bash
root@1e560fca3906:/# exit
exit

LINSHAODONG@HTJT-LSDONG MINGW64 /e/project/swoft-master
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
1e560fca3906        ubuntu              "/bin/bash"         20 minutes ago      Up 34 seconds                           ubuntu-test
```

Docker 删除容器



镜像操作

$ docker search 镜像名

$ docker images # 列出所有的镜像

$ docker pull 镜像名 # 获取新的镜像

$ docker rmi 镜像名 # 删除镜像

DockerFile 构建镜像

```
runoob@runoob:~$ cat Dockerfile
FROM    centos:6.7
MAINTAINER      Fisher "fisher@sudops.com"

RUN     /bin/echo 'root:123456' |chpasswd
RUN     useradd runoob
RUN     /bin/echo 'runoob:123456' |chpasswd
RUN     /bin/echo -e "LANG=\"en_US.UTF-8\"" >/etc/default/local
EXPOSE  22
EXPOSE  80
CMD     /usr/sbin/sshd -D
```

Dockerfile 文件所在目录，可以指定 Dockerfile 的绝对路径

指定要创建的目标镜像名

```
runoob@runoob:~$ docker build -t runoob/centos:6.7 .  # 构建镜像 < - - -
```

容器连接

```
runoob@runoob:~$ docker run -d -P training/webapp python app.py
fce072cc88cee71b1cdceb57c2821d054a4a59f67da6b416fceb5593f059fc6d
```

-P （大写） ： 是容器内部端口随机映射到主机的高端口。

-p （小写） ： 是容器内部端口绑定到指定的主机端口。

```
runoob@runoob:~$ docker run -d -p 5000:5000 training/webapp python app.py
33e4523d30aaf0258915c368e66e03b49535de0ef20317d3f639d40222ba6bc0
```

容器命名

当我们创建一个容器的时候， docker 会自动对它进行命名。另外，我们也可以使用 --name 标识来命名容器