

# Docker学习

## 1. 基本命令

### 1.1 Docker 客户端

我们可以直接输入docker命令来查看到Docker客户端的所有命令选项。

```
1 | runoob@runoob:~# docker
```

可以通过**docker command --help**更深入的了解指定的Docker命令使用方法。

```
1 | runoob@runoob:~# docker run --help
```

### 1.2容器使用

#### 1.2.1获取容器

如果我们本地没有ubuntu镜像的话，我们可以使用docker pull命令进入该容器：

```
1 | $ docker pull ubuntu(后面也可以添加相应的版本，具体可以到下面的网站去查看，即公开的 registry)
```

<https://hub.docker.com>

```
1 | $ docker pull [option] name [:tag|@digest]
2 | //你可以这样用
3 | $ docker pull ubuntu:14.04
4 | //也可以这样用
5 | $ docker pull
   ubuntu@sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb
   2
6 | //当我们通过摘要digest拉取镜像的方式的话，我们就可以获取一个固定版本的镜像，要想获得镜像的
   摘要信息，需要我们先pull下一个镜像文件，随后docker会自动帮我们打
7 | //印出该镜像的摘要信息
```

#### 1.2.2启动容器

以下命令使用Ubuntu镜像启动一个容器，参数位以命令行模式进入该模式：

该命令在使用的时候需要注意，当我们的docker中没有相关的镜像的时候，我们使用该命令，docker会现在本地文件中寻找相关的docker文件，当找不到的时候，会去官网上去下载相关的镜像文件，最后创建容器。我们每执行依次该命令的时候，就会新创建一个容器，和之前一样的容器。

所以说，该命令一般是对一个镜像使用一次就可以了！

```
1 | $ docker run -it ubuntu /bin/bash
```

参数说明：

- **-i**:交互式操作;
  - **-t**:终端;
  - **ubuntu**:ubuntu镜像;
  - **/bin/bash**:放在镜像名字后面的是命令, 这里我们希望有一个交互式shell, 因此使用的是/bin/bash.
- 如果说要退出终端, 直接输入exit即可。

### 1.2.3 启动已经停止运行的容器

查看所有的容器的命令是:

```
1 | $ docker ps -a
```

```
kirito@kirito-virtual-machine:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
8de9b40eb61a	ubuntu	"/bin/bash"	13 minutes ago	Exited (0) 13 minutes ago		funny_feynman
f03ab6476abd	centos:centos7	"/bin/bash"	58 minutes ago	Exited (0) 58 minutes ago		zealous_diffie
4ee50550f2f8	ubuntu:14.04	"/bin/bash"	3 hours ago	Exited (0) 2 hours ago		interesting_elion
c1ec5f88572e	busybox	"echo hello world"	3 hours ago	Exited (0) 3 hours ago		adoring_hawking

使用docker start启动一个已经停止的容器:

```
1 | $ docker start 8de9b40eb61a
```

```
kirito@kirito-virtual-machine:~$ docker start 3de9b40eb61a
3de9b40eb61a
```

### 1.2.4 后台运行

在大部分的场景下, 我们希望docker的服务是在后台运行的, 我们可以通过-d指定容器的运行模式:

```
1 | $ docker run -itd --name ubuntu-test ubuntu /bin/bash
```

```
kirito@kirito-virtual-machine:~$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3de9b40eb61a	ubuntu	"/bin/bash"	13 minutes ago	Exited (0) 13 minutes ago		funny_feynman
f03ab6476abd	centos:centos7	"/bin/bash"	58 minutes ago	Exited (0) 58 minutes ago		zealous_diffie
4ee50550f2f8	ubuntu:14.04	"/bin/bash"	3 hours ago	Exited (0) 2 hours ago		interesting_elion
c1ec5f88572e	busybox	"echo hello world"	3 hours ago	Exited (0) 3 hours ago		adoring_hawking

```
kirito@kirito-virtual-machine:~$ docker run -itd --name funny_feyman ubuntu /bin/bash
7e6e47c21ba905e098711b3f4e7c77c977a8f3a6946fa12b06e73a216f74ac50
kirito@kirito-virtual-machine:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7e6e47c21ba9	ubuntu	"/bin/bash"	9 seconds ago	Up 8 seconds		funny_feyman

```
kirito@kirito-virtual-machine:~$ docker stop 7e6e47c21ba9
7e6e47c21ba9
kirito@kirito-virtual-machine:~$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

### 1.2.5 停止一个容器

停止一个容器的命令是:

```
1 | $ docker stop <容器 ID>
```

停止的容器可以通过docker restart重启:

```
1 | $ docker restart <容器 ID>
```

## 1.2.6 进入容器

在使用参数 `-d` 的时候，容器启动后会进入后台，此时想要进入容器，可以通过以下指令进入：

```
docker attach
```

`docker exec`：推荐大家使用 `docker exec` 命令，因为此命令会退出容器终端，但不会导致容器的停止。

`attach` 命令

下面演示使用 `docker attach` 命令：

```
1 | $ docker attach <容器 ID>
```

**ps:**从该容器退出的话，会导致容器的停止。

下面演示使用 `docker exec` 命令：

```
1 | docker exec -it <容器 ID> /bin/bash
```

**ps:** 如果从这个容器退出的话，容器不会停止

## 1.2.7 导出和导入容器

### 1) 导出容器

使用命令 `docker export`

```
1 | $ docker export <容器 ID> > ubuntu.tar
2 | //导出容器快照到本地文件ubuntu.tar
3 | //快照就是对数据一个快速的复制
4 | //执行以上命令之后，会在当前文件夹生成ubuntu.tar文件，我们可以将文件拷贝到其他机器上去，通过导入命令实现容器的迁移
```

### 2) 导入容器快照到镜像文件中：

```
1 | $ cat ubuntu.tar | docker import - test/ubuntu:v1
2 | //上述命令会创建一个名字为test/ubuntu的容器以及版本位v1
```

然后我们再通过镜像来创建一个新的容器，这样一个一模一样的容器就创建完毕！

此外，我们也可以通过指定URL或者某一个目录来导入，例如：

```
1 | $ docker import http://example.com/exampleimage.tgz example/imagerepo
```

## 1.2.8 删除容器

删除容器使用 `docker rm` 命令：

```
1 | $ docker rm -f <容器 ID>
```

清除掉所有处于终止状态的容器：

```
$ docker container prune
```

..... (更多参考<https://www.runoob.com/docker/docker-container-usage.html>)

## 1.3 docker小小的配置

我们最开始安装完docker后，执行相关的命令会出现

```
"Got permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get
http://%2Fvar%2Frun%2Fdocker.sock/v1.26/images/json: dial unix /var/run/docker.sock:
connect: permission denied"
```

大概的意思就是：docker进程使用Unix Socket而不是TCP端口。而默认情况下，Unix socket属于root用户，需要root权限才能访问。

### 解决方法1

使用sudo获取管理员权限，运行docker命令

### 解决方法2

docker守护进程启动的时候，会默认赋予名字为docker的用户组读写Unix socket的权限，因此只要创建docker用户组，并将当前用户加入到docker用户组中，那么当前用户就有权限访问Unix socket了，进而也就可以执行docker相关命令：

```
1 | sudo groupadd docker      #添加docker用户组
2 | sudo gpasswd -a $USER docker    #将登陆用户加入到docker用户组中
3 | newgrp docker      #更新用户组
4 | docker ps      #测试docker命令是否可以使用sudo正常使用
```

docker pull kalilinux/kali-bleeding-edge:latest