免费电子书

学习

Docker

#docker

1: Docker

Docker。 。

DockerLinuxMac OSXWindowslinuxdocker。 VirtualBoxDocker Toolbox docker。 Linuxdocker。

17.05.0	201754
17.04.0	201745
17.03.0	201731
1.13.1	201628
1.12.0	2016728
1.11.2	2016413
1.10.3	201624
1.9.1	2015113
1.8.3	2015811
1.7.1	2015616
1.6.2	201547
1.5.0	2015210

Examples

Mac OS XDocker

OS X 10.8“Mountain Lion”Docker。

dockerMac OS XLinux。

1.12.0

1.12VMDockerOSX`Hypervisor.framework`Linux。

docker

1. Docker for Mac
2. 。
3.

。

。

1.11.2

1.11Linux VMDocker ToolboxDockerVirtualBoxLinux。

docker

1. Docker Toolbox
2. Mac。
3. 。

`/usr/local/bin`DockerVirtual Box。 。

1.12.0

1. Applications`Docker.app` 。 。

1.11.2

1. `Docker Quickstart Terminal` Docker。

2.
   ```
   $ docker run hello-world
   ```
3. 。

WindowsDocker

64Windows 7。

dockerWindowsLinux。

1.12.0

1.12VMDockerWindowsHyper-VLinux。

docker

1. Docker for Windows
2. 。
3. 。

。

1.11.2

1.11Linux VMDocker ToolboxDockerVirtualBoxLinux。

docker

1. Docker Toolbox
2. Windows。
3. 。

DockerVirtual Box。 。

1.12.0

1. `Docker`“”。 `cmd`PowerShell

1.11.2

1. Docker Toolbox。 Docker Toolbox。
2.
   ```
   docker run hello-world
   ```
3. 。

Ubuntu Linuxdocker

*64*Ubuntu LinuxDocker

- Ubuntu Xenial 16.04LTS
- Ubuntu Wily 15.10
- Ubuntu Trusty 14.04LTS
- Ubuntu Precise 12.04LTS

  DockerDocker。 `Ubuntu-managed`Ubuntu。

  Ubuntu Utopic 14.1015.04DockerAPT。

- Docker64Linux。
- DockerLinux3.10 `Ubuntu Precise 12.04`3.13。 3.10Docker。 `uname -r`。 `Ubuntu Precise (12.04 LTS)`。 WikiHowUbuntu。

APT

Docker。

1. `sudo`root。
2. 。
3. APThttpsCA。

```
$ sudo apt-get update
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
```

```
    curl \
    software-properties-common
```

4. DockerGPG

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

   9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88 。

   ```
   $ sudo apt-key fingerprint 0EBFCD88
   ```

```
    pub   4096R/0EBFCD88 2017-02-22
          Key fingerprint = 9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
    uid                  Docker Release (CE deb) <docker@docker.com>
    sub   4096R/F273FCD8 2017-02-22
```

5. Ubuntu。 APTDocker。 UbuntuLTS。

Ubuntu	
12.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-precise main`
Trusty 14.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-trusty main`
15.10	`deb https://apt.dockerproject.org/repo ubuntu-wily main`
Xenial 16.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-xenial main`

   Docker。 `https://master.dockerproject.org`。 docker `[arch=...]`。 Debian Multiarch wiki
   。

6. `<REPO>`。

   $ echo“”| sudo tee /etc/apt/sources.list.d/docker.list

7. `sudo apt-get update`APT。

8. `APT`。

Docker。 URL `https://apt.dockerproject.org/repo/` 。 `***`。

```
$ apt-cache policy docker-engine

  docker-engine:
    Installed: 1.12.2-0~trusty
    Candidate: 1.12.2-0~trusty
    Version table:
   *** 1.12.2-0~trusty 0
          500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
          100 /var/lib/dpkg/status
```

```
      1.12.1-0~trusty 0
         500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
      1.12.0-0~trusty 0
         500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
```

`apt-get upgrade` APT。

Ubuntu

Ubuntu Trusty14.04Wily15.10Xenial16.04`linux-image-extra-*`aufs。

`linux-image-extra-*`

1. Ubuntu。
2. `sudo apt-get update`。
3. 。

   ```
   $ sudo apt-get install linux-image-extra-$(uname -r) linux-image-extra-virtual
   ```

4. Docker

Ubuntu Precise12.04 LTSDocker3.13。 3.13。

`linux-image-generic-lts-trusty`	Linux。 AUFS 。 Docker。
`linux-headers-generic-lts-trusty`	ZFSVirtualBox guest additions。 trusty。 。
`xserver-xorg-lts-trusty`	Unity / Xorg。 Docker 。
`ligbl1-mesa-glx-lts-trusty`	backportedLTS Enablement Stack 。 5。

1. Ubuntu。
2. `sudo apt-get update`。
3. 。

   ```
   $ sudo apt-get install linux-image-generic-lts-trusty
   ```

4. 。
5. `sudo reboot`。
6. Docker。

。

Docker。 。

1. `sudo`Ubuntu。 `sudo -su` 。
2. `sudo apt-get update`APT。
3. `sudo apt-get install docker-ce`Docker Community Edition。
4. `sudo service docker start` docker。
5. `docker`Hello。

   ```
   $ sudo docker run hello-world
   ```

。 。

rootDocker

`sudo`sudo `docker`Unix。 `docker`dockerUnix/。

`docker`

1. `sudo`Ubuntu。
2. `sudo groupadd docker`docker。
3. `docker`。

   ```
   $ sudo usermod -aG docker $USER
   ```

4. 。
5. `sudo`docker。

   ```
   $ docker run hello-world
   ```

```
Cannot connect to the Docker daemon. Is 'docker daemon' running on this host?
```

shell`DOCKER_HOST`。

```
$ env | grep DOCKER_HOST
```

。 。

```
$ unset DOCKER_HOST
```

`~/.bashrc`~/.profile`DOCKER_HOST`。

UbuntuDocker

Docker3.10Linux。 64Ubuntu LinuxDocker

- Ubuntu Xenial 16.04LTS
- Ubuntu Wily 15.10
- Ubuntu Trusty 14.04LTS
- Ubuntu Precise 12.04LTS

UbuntuDockerDocker。

DockerDocker`curl`Docker

```
$ curl -sSL https://get.docker.com/ | sh
```

`wget`Docker

```
$ wget -qO- https://get.docker.com/ | sh
```

Docker。

Docker。

```
$ sudo apt-get update
$ sudo apt-get install apt-transport-https ca-certificates
```

GPG

```
$ sudo apt-key adv --keyserver hkp://p80.pool.sks-keyservers.net:80 \
  --recv-keys 58118E89F3A912897C070ADBF76221572C52609D
```

`/etc/apt/sources.list.d/docker.list`。 。 。

- Ubuntu Precise 12.04LTS

  ```
  deb https://apt.dockerproject.org/repo ubuntu-precise main
  ```

- Ubuntu Trusty 14.04LTS

  ```
  deb https://apt.dockerproject.org/repo ubuntu-trusty main
  ```

- Ubuntu Wily 15.10

  ```
  deb https://apt.dockerproject.org/repo ubuntu-wily main
  ```

- Ubuntu Xenial 16.04LTS

  ```
  deb https://apt.dockerproject.org/repo ubuntu-xenial main
  ```

Docker`apt`repo

```
$ sudo apt-get update
$ sudo apt-get purge lxc-docker
$ sudo apt-cache policy docker-engine
```

Ubuntu

- Ubuntu Xenial 16.04LTSUbuntu Wily 15.10Ubuntu Trusty 14.04LTS

  `sudo apt-get update && sudo apt-get install linux-image-extra-$(uname -r)`

- Ubuntu Precise 12.04LTS

  Ubuntu3.13。

  `linux-image-generic-lts-trusty`

  Linux。 AUFS。 Docker。

  `linux-headers-generic-lts-trusty`

  ZFSVirtualBox guest。 `trusty`。 。

  `xserver-xorg-lts-trusty`

  `libgl1-mesa-glx-lts-trusty`

  Unity / Xorg。 Docker。

  backportedLTS Enablement Stack - 5。

  `$ sudo apt-get install linux-image-generic-lts-trusty`

  `$ sudo reboot`

`apt`Docker

```
$ sudo apt-get update
$ sudo apt-get install docker-engine
```

```
$ sudo service docker start
```

docker

```
$ sudo docker run hello-world
```

。

Google Clouddocker

dockerdocker。 `gcloud` Google Cloud util

```
docker-machine create --driver google --google-project `your-project-name` google-machine-type
```

```
f1-large fm02
```

Google Cloud。 `f1-large`

UbuntuDocker

*64*Ubuntu LinuxDocker

- Ubuntu Xenial 16.04LTS
- Ubuntu Wily 15.10
- Ubuntu Trusty 14.04LTS
- Ubuntu Precise 12.04LTS

  DockerDocker。 `Ubuntu-managed`Ubuntu。

  Ubuntu Utopic 14.1015.04DockerAPT。

- Docker64Linux。
- DockerLinux3.10 `Ubuntu Precise 12.04`3.13。 3.10Docker。 `uname -r`。 `Ubuntu Precise (12.04 LTS)`。 WikiHowUbuntu。

APT

Docker。

1. `sudoroot`。
2. 。
3. APThttpsCA。

```
$ sudo apt-get update
$ sudo apt-get install apt-transport-https ca-certificates
```

4. `GPG`。 `hkp://ha.pool.sks-keyservers.net:80`ID`58118E89F3A912897C070ADBF76221572C52609Dadv keychainadv keychain` 。 `man apt-key`。

   ```
   $ sudo apt-key adv \
          --keyserver hkp://ha.pool.sks-keyservers.net:80 \
          --recv-keys 58118E89F3A912897C070ADBF76221572C52609D
   ```

5. Ubuntu。 APTDocker。 UbuntuLTS。

Ubuntu	
12.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-precise main`
Trusty 14.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-trusty main`
15.10	`deb https://apt.dockerproject.org/repo ubuntu-wily main`

Ubuntu	
Xenial 16.04LTS	`deb https://apt.dockerproject.org/repo ubuntu-xenial main`

Docker。 `https://master.dockerproject.org`。 docker`[arch=...]`。 Debian Multiarch wiki
。

6. `<REPO>`。

   $ echo“”| sudo tee /etc/apt/sources.list.d/docker.list

7. `sudo apt-get update`APT。

8. `APT`。

Docker。 URL `https://apt.dockerproject.org/repo/` 。 `***`。

```
$ apt-cache policy docker-engine

  docker-engine:
    Installed: 1.12.2-0~trusty
    Candidate: 1.12.2-0~trusty
    Version table:
   *** 1.12.2-0~trusty 0
          500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
          100 /var/lib/dpkg/status
       1.12.1-0~trusty 0
          500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
       1.12.0-0~trusty 0
          500 https://apt.dockerproject.org/repo/ ubuntu-trusty/main amd64 Packages
```

`apt-get upgrade` APT。

Ubuntu

Ubuntu Trusty14.04Wily15.10Xenial16.04`linux-image-extra-*`aufs。

`linux-image-extra-*`

1. Ubuntu。

2. `sudo apt-get update`。

3. 。

   ```
   $ sudo apt-get install linux-image-extra-$(uname -r) linux-image-extra-virtual
   ```

4. Docker

Ubuntu Precise12.04 LTSDocker3.13。 3.13。

`linux-image-generic-lts-trusty`	

	Linux。 `AUFS` 。 Docker。
`linux-headers-generic-lts-trusty`	`ZFSVirtualBox guest additions`。 `trusty`。 。
`xserver-xorg-lts-trusty`	Unity / Xorg。 Docker 。
`ligbl1-mesa-glx-lts-trusty`	backportedLTS Enablement Stack 。 5。

1. Ubuntu。
2. `sudo apt-get update`。
3. 。

   ```
   $ sudo apt-get install linux-image-generic-lts-trusty
   ```

4. 。
5. `sudo reboot`。
6. Docker。

。

Docker。 。

1. `sudo`Ubuntu。 `sudo -su` 。
2. `sudo apt-get update`APT。
3. `sudo apt-get install docker-engine`Docker。
4. `sudo service docker start` `docker`。
5. `docker`Hello。

   ```
   $ sudo docker run hello-world
   ```

。 。

rootDocker

`sudo``sudo docker`Unix。 `docker`dockerUnix/。

`docker`

1. `sudo`Ubuntu。
2. `sudo groupadd docker``docker`。

3. docker。

   ```
   $ sudo usermod -aG docker $USER
   ```

4. 。

5. sudodocker。

   ```
   $ docker run hello-world
   ```

```
Cannot connect to the Docker daemon. Is 'docker daemon' running on this host?
```

shellDOCKER_HOST。

```
$ env | grep DOCKER_HOST
```

。 。

```
$ unset DOCKER_HOST
```

~/.bashrc~/.profileDOCKER_HOST。

CentOSDocker-ceDocker-ee

Docker

-Docker-eeDocker-ceDocker

CentOSDocker-eeDocker-ce

Docker-ce

docker-ce

1. yum-utilsyum-config-manager

   ```
   $ sudo yum install -y yum-utils
   ```

2.
   ```
   $ sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
   ```

3. 。 docker.repo。 。

   ```
   $ sudo yum-config-manager --enable docker-ce-edge
   ```

- --disableyum-config-manager

◦ `--enable`◦ ◦

```
$ sudo yum-config-manager --disable docker-ce-edge
```

4. yum◦

```
$ sudo yum makecache fast
```

5. docker-ce

```
$ sudo yum install docker-ce-17.03.0.ce
```

6. Docker-ce

   `060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35`

   docker-ce

```
$ sudo yum install docker-ce-VERSION
```

   `VERSION`

7. docker-ce

```
$ sudo systemctl start docker
```

8. docker

```
$ sudo docker run hello-world
```

```
Hello from Docker!
This message shows that your installation appears to be working correctly.
```

-Docker-ee

Enterprise EditionEE<DOCKER-EE-URL>◦

1. https://cloud.docker.com/ ◦ ID<DOCKER-EE-URL>◦
2. `/etc/yum.repos.d/`Docker
3. Docker EEURL`/etc/yum/vars/`yum◦ <DOCKER-EE-URL>URL◦

```
$ sudo sh -c 'echo "<DOCKER-EE-URL>" > /etc/yum/vars/dockerurl'
```

4. yum-utilsyum-config-manager

```
$ sudo yum install -y yum-utils
```

5.
```
$ sudo yum-config-manager \
--add-repo \
<DOCKER-EE-URL>/docker-ee.repo
```

6. yum。

```
$ sudo yum makecache fast
```

7. docker-ee

```
sudo yum install docker-ee
```

8. docker-ee

```
$ sudo systemctl start docker
```

Docker https://riptutorial.com/zh-CN/docker/topic/658/docker

2: Docker Engine API

APIDockerDockerDocker。

Examples

LinuxDocker API

`/etc/init/docker.confDOCKER_OPTS`

```
DOCKER_OPTS='-H tcp://0.0.0.0:4243 -H unix:///var/run/docker.sock'
```

Docker deamon

```
service docker restart
```

Remote API

```
curl -X GET http://localhost:4243/images/json
```

systemdLinuxDocker API

LinuxsystemdUbuntu 16.04`-H tcp://0.0.0.0:2375/etc/default/docker -H tcp://0.0.0.0:2375`。

`/etc/systemd/system/docker-tcp.socket` 4243TCPdocker

```
[Unit]
Description=Docker Socket for the API
[Socket]
ListenStream=4243
Service=docker.service
[Install]
WantedBy=sockets.target
```

```
systemctl enable docker-tcp.socket
systemctl enable docker.socket
systemctl stop docker
systemctl start docker-tcp.socket
systemctl start docker
```

Remote API

```
curl -X GET http://localhost:4243/images/json
```

SystemdTLS

/ etc

```
cp /lib/systemd/system/docker.service /etc/systemd/system/docker.service
```

ExecStart/etc/systemd/system/docker.service

```
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:2376 \
  --tlsverify --tlscacert=/etc/docker/certs/ca.pem \
  --tlskey=/etc/docker/certs/key.pem \
  --tlscert=/etc/docker/certs/cert.pem
```

`dockerd`1.12`docker daemon` 。 2376dockersTLS2375。 TLSCA 。

systemdsystemd

```
systemctl daemon-reload
```

docker

```
systemctl restart docker
```

DockerTLSroot。

Go

`GoDocker Engine APICLIdocker pull your_image_name`。 ANSI 。

```
package yourpackage

import (
    "context"
    "encoding/json"
    "fmt"
    "io"
    "strings"

    "github.com/docker/docker/api/types"
    "github.com/docker/docker/client"
)

// Struct representing events returned from image pulling
type pullEvent struct {
    ID             string `json:"id"`
    Status         string `json:"status"`
    Error          string `json:"error,omitempty"`
    Progress       string `json:"progress,omitempty"`
    ProgressDetail struct {
        Current int `json:"current"`
        Total   int `json:"total"`
    } `json:"progressDetail"`
}

// Actual image pulling function
func PullImage(dockerImageName string) bool {
    client, err := client.NewEnvClient()
    if err != nil {
        panic(err)
    }

    resp, err := client.ImagePull(context.Background(), dockerImageName,
types.ImagePullOptions{})

    if err != nil {
        panic(err)
    }

    cursor := Cursor{}
    layers := make([]string, 0)
    oldIndex := len(layers)

    var event *pullEvent
    decoder := json.NewDecoder(resp)

    fmt.Printf("\n")
    cursor.hide()

    for {
        if err := decoder.Decode(&event); err != nil {
            if err == io.EOF {
                break
            }

            panic(err)
        }

        imageID := event.ID

        // Check if the line is one of the final two ones
        if strings.HasPrefix(event.Status, "Digest:") || strings.HasPrefix(event.Status,
"Status:") {
            fmt.Printf("%s\n", event.Status)
            continue
        }

        // Check if ID has already passed once
        index := 0
        for i, v := range layers {
            if v == imageID {
                index = i + 1
                break
            }
        }

        // Move the cursor
        if index > 0 {
            diff := index - oldIndex

            if diff > 1 {
                down := diff - 1
                cursor.moveDown(down)
            } else if diff < 1 {
                up := diff*(-1) + 1
                cursor.moveUp(up)
            }

            oldIndex = index
        } else {
            layers = append(layers, event.ID)
            diff := len(layers) - oldIndex

            if diff > 1 {
                cursor.moveDown(diff) // Return to the last row
            }

            oldIndex = len(layers)
        }

        cursor.clearLine()

        if event.Status == "Pull complete" {
            fmt.Printf("%s: %s\n", event.ID, event.Status)
        } else {
            fmt.Printf("%s: %s %s\n", event.ID, event.Status, event.Progress)
        }

    }

    cursor.show()

    if strings.Contains(event.Status, fmt.Sprintf("Downloaded newer image for %s",
dockerImageName)) {
        return true
    }

    return false
}
```

ANSI

```
package yourpackage

import "fmt"

// Cursor structure that implements some methods
// for manipulating command line's cursor
type Cursor struct{}

func (cursor *Cursor) hide() {
    fmt.Printf("\033[?25l")
}

func (cursor *Cursor) show() {
    fmt.Printf("\033[?25h")
}

func (cursor *Cursor) moveUp(rows int) {
    fmt.Printf("\033[%dF", rows)
}

func (cursor *Cursor) moveDown(rows int) {
    fmt.Printf("\033[%dE", rows)
}

func (cursor *Cursor) clearLine() {
    fmt.Printf("\033[2K")
}
```

PullImage。 Docker。

cURL

cURLDocker API。 map[string][]stringJSONGo。

```
curl --unix-socket /var/run/docker.sock \
    -XGET "http:/v1.29/images/json" \
    -G \
    --data-urlencode 'filters={"reference":{"yourpreciousregistry.com/path/to/image": true},
"dangling":{"true": true}}'
```

-G--data-urlencodeHTTP GETPOST。 URL?。

Docker Engine API https://riptutorial.com/zh-CN/docker/topic/3935/docker-engine-api

3: Docker --net。

docker0。 。

NIC。

。 “”。

docker

Examples

```
$ docker run -d --name my_app -p 10000:80 image_name
```

--net = bridgedocker。 。 BRIDGE。

```
$ docker run -d --name my_app -net=host image_name
```

。

。 IP。 “”。 web_container_1web_container_2web_container_2。 web_container_1

```
$ docker run -d --name web1 -p 80:80 USERNAME/web_container_1
```

。 。 。 。

```
$ docker run -d --name web2 --net=container:web1 USERNAME/web_container_2
```

。 。 。 。

Docker --net。 https://riptutorial.com/zh-CN/docker/topic/9643/docker---net-----

4: Dockerfiles

DockerfilesDocker。 Docker。 DockerfilesDocker。 `<INSTRUCTION><argument(s)>`。 Dockerfiles docker `docker build`Docker。

Dockerfiles

```
# This is a comment
INSTRUCTION arguments
```

- `#`
-
- Dockerfile`FROM`

DockerfileDockerDocker“”。 Dockerfile。 `COPY``ADD`。

Docker

```
# escape=`
```

Docker`` ` ``\ 。 Windows Docker。

Examples

HelloWorld Dockerfile

Dockerfile

```
FROM alpine
CMD ["echo", "Hello StackOverflow!"]
```

DockerAlpine `FROM` `CMD` 。

```
docker build -t hello .
docker run --rm hello
```

```
Hello StackOverflow!
```

Docker`COPY`

```
COPY localfile.txt containerfile.txt
```

```
COPY ["local file", "container file"]
```

COPY◦ images/

```
COPY *.jpg images/
```

images/◦ Docker◦

DockerfileEXPOSE

```
EXPOSE 8080 8082
```

DockerDockerfile◦

Dockerfiles

Docker◦ ◦ ◦

```
RUN apt-get -qq update
RUN apt-get -qq install some-package
```

- ◦
- RUNapt-get updateapt-get install ◦ apt-get install docker◦ apt-get update◦

```
RUN apt-get -qq update && \
    apt-get -qq install some-package
```

◦

Dockerfile◦ ◦

```
LABEL maintainer John Doe <john.doe@example.com>
```

◦ ◦

Dockerfile◦ ◦

```
USER daemon
```

USERUIDRUN CMDENTRYPOINTDockerfile ◦

WORKDIR

```
WORKDIR /path/to/workdir
```

WORKDIRENTRYPOINTRUN CMD ENTRYPOINT COPYADD◦ WORKDIRDockerfile◦

Dockerfile◦ WORKDIR◦

```
WORKDIR /a
WORKDIR b
WORKDIR c
RUN pwd
```

Dockerfilepwd/a/b/c 。

WORKDIRENV。 Dockerfile。

```
ENV DIRPATH /path
WORKDIR $DIRPATH/$DIRNAME
RUN pwd
```

Dockerfilepwd/path/$DIRNAME

```
VOLUME ["/data"]
```

VOLUME。 JSON VOLUME ["/var/log/"]VOLUME /var/logVOLUME /var/log /var/db 。 Docker/。

docker run。 Dockerfile

```
FROM ubuntu
RUN mkdir /myvol
RUN echo "hello world" > /myvol/greeting
VOLUME /myvol
```

Dockerfiledocker/ myvol。

。

JSON"“。

COPY

COPY

```
COPY <src>... <dest>
COPY ["<src>",... "<dest>"] (this form is required for paths containing whitespace)
```

COPY<src><dest>。

<src>。

<src>Gofilepath.Match。

```
COPY hom* /mydir/        # adds all files starting with "hom"
COPY hom?.txt /mydir/    # ? is replaced with any single character, e.g., "home.txt"
```

<dest>WORKDIR。

```
COPY test relativeDir/   # adds "test" to `WORKDIR`/relativeDir/
COPY test /absoluteDir/  # adds "test" to /absoluteDir/
```

UIDGID0。

stdin `docker build - < somefile` COPY 。

`COPY`

- `<src>`;`COPY` ../something / somethingdocker builddocker。
- `<src>`。 。
- `<src>`。 `<dest>`/ `<src><dest>/base(<src>)` 。
- `<src><dest>/`。
- `<dest>` `<src><dest>` 。
- `<dest>`。

ENVARG

ENV

```
ENV <key> <value>
ENV <key>=<value> ...
```

`ENV<key>`。 “”Dockerfile。

`ENV`。 `ENV <key> <value>` 。 `<value>` - 。

`ENV <key>=<value> ...`。 =。 。

```
ENV myName="John Doe" myDog=Rex\ The\ Dog \
    myCat=fluffy
```

```
ENV myName John Doe
ENV myDog Rex The Dog
ENV myCat fluffy
```

。

`ENV`。 `docker run --env <key>=<value>` inspect`docker run --env <key>=<value>`。

ARG

`ARG` 。 `ARG`。

```
ENV DEBIAN_FRONTEND noninteractive
```

`apt-get``docker exec -it the-container bash``apt-get`Debian。

```
ARG DEBIAN_FRONTEND noninteractive
```

```
RUN <key>=<value> <command>
```

EXPOSE

```
EXPOSE <port> [<port>...]
```

`EXPOSE`Docker。 `EXPOSE`。 `-p``-P`。 `docker run [OPTIONS] IMAGE [COMMAND][ARG...]`。 。

```
docker run -p 2500:80 <image name>
```

<image>802500。

`-P`。 Docker。

```
LABEL <key>=<value> <key>=<value> <key>=<value> ...
```

`LABEL`。 `LABEL`。 `LABEL`。

```
LABEL "com.example.vendor"="ACME Incorporated"
LABEL com.example.label-with-value="foo"
LABEL version="1.0"
LABEL description="This text illustrates \
that label-values can span multiple lines."
```

。 Docker`LABEL`。 `LABEL`。 。

```
LABEL multi.label1="value1" multi.label2="value2" other="value3"
```

```
LABEL multi.label1="value1" \
      multi.label2="value2" \
      other="value3"
```

`FROM``LABEL` 。 Docker/。

docker inspect。

```
"Labels": {
    "com.example.vendor": "ACME Incorporated"
    "com.example.label-with-value": "foo",
    "version": "1.0",
    "description": "This text illustrates that label-values can span multiple lines.",
    "multi.label1": "value1",
    "multi.label2": "value2",
    "other": "value3"
},
```

CMD

CMD

```
CMD ["executable","param1","param2"] (exec form, this is the preferred form)
CMD ["param1","param2"] (as default parameters to ENTRYPOINT)
CMD command param1 param2 (shell form)
```

DockerfileCMD。 CMDCMD。

CMD。 ENTRYPOINT。

CMDENTRYPOINTJSONCMDENTRYPOINT。

execJSON“'。

shellexecshell。 shell。 CMD ["echo", "$HOME"]$HOME。 shellshellshell CMD ["sh", "-c", "echo $HOME"] 。

shellexec CMD。

CMDshell/bin/sh -c

```
FROM ubuntu
CMD echo "This is a test." | wc -
```

shellJSON。 CMD。

```
FROM ubuntu
CMD ["/usr/bin/wc","--help"]
```

ENTRYPOINTCMD。 ENTRYPOINT 。

docker runCMD。

RUNCMD。 RUN; CMD。

MAINTAINER

```
MAINTAINER <name>
```

MAINTAINER“”。

Docker MAINTAINER。 LABEL。 LABELdocker inspect。

```
LABEL maintainer="someone@something.com"
```

```
FROM <image>
```

```
FROM <image>:<tag>
```

```
FROM <image>@<digest>
```

FROM。 DockerfileFROM。 - 。

FROMDockerfile。

FROMDockerfile。 FROMID。

。 。 。

RUN

```
RUN <command> (shell form, the command is run in a shell, which by default is /bin/sh -c on
Linux or cmd /S /C on Windows)
RUN ["executable", "param1", "param2"] (exec form)
```

RUN。 DockerfileDockerfile 。

RUNDocker。

execshellshellRUN。

SHELLshellshell。

shell\ RUN。

```
RUN /bin/bash -c 'source $HOME/.bashrc ;\
echo $HOME'
```

```
RUN /bin/bash -c 'source $HOME/.bashrc ; echo $HOME'
```

“/ bin / sh”shellshellexec。 RUN ["/bin/bash", "-c", "echo hello"]

execJSON " ' 。

shellexecshell。 shell。 RUN ["echo", "$HOME"]$HOME。 shellshellshell RUN ["sh", "-c", "echo $HOME"] 。

JSON。 Windows。 JSONshell RUN ["c:\windows\system32\tasklist.exe"]

RUN ["c:\\windows\\system32\\tasklist.exe"]

RUN

◦ RUN apt-get dist-upgrade -y◦ --no-cacheRUNdocker build --no-cache◦

Dockerfile◦

ADDRUN◦ ◦

ONBUILD

```
ONBUILD [INSTRUCTION]
```

ONBUILD◦ DockerfileFROM◦

◦

◦

Python◦ ADDRUN ◦ Dockerfile◦

ONBUILD◦

ONBUILD◦ ◦

OnBuild◦ docker inspect◦ FROM◦ FROMONBUILD◦ FROM◦ FROM◦

◦ “”◦

```
[...]
ONBUILD ADD . /app/src
ONBUILD RUN /usr/local/bin/python-build --dir /app/src
[...]
```

ONBUILDONBUILD ONBUILDONBUILD◦

ONBUILDFROMMAINTAINER◦

STOPSIGNAL

```
STOPSIGNAL signal
```

STOPSIGNAL◦ 9SIGNAMESIGKILL◦

HEALTHCHECK

```
HEALTHCHECK [OPTIONS] CMD command (check container health by running a command inside the
container)
HEALTHCHECK NONE (disable any healthcheck inherited from the base image)
```

HEALTHCHECKDocker◦ Web◦

。 。 。 。

CMD

```
--interval=DURATION (default: 30s)
--timeout=DURATION (default: 30s)
--retries=N (default: 3)
```

。

。

。

HEALTHCHECKDockerfile 。 HEALTHCHECK。

CMDshellHEALTHCHECK CMD /bin/check-running execDockerfile;ENTRYPOINT 。

。

- 0: success -
- 1: unhealthy -
- 2: starting -

“”2“”“”。

```
HEALTHCHECK --interval=5m --timeout=3s \
  CMD curl -f http://localhost/ || exit 1
```

stdoutstderrUTF-8docker inspect。 4096。

health_status。

Docker 1.12HEALTHCHECK。

```
SHELL ["executable", "parameters"]
```

SHELLshellshell。 Linuxshell["/bin/sh", "-c"] Windows["cmd", "/S", "/C"] 。 SHELLJSONDockerfile 。

SHELLWindowsshellcmdpowershellshshell。

SHELL。 SHELLSHELL。

```
FROM windowsservercore

# Executed as cmd /S /C echo default
RUN echo default

# Executed as cmd /S /C powershell -command Write-Host default
```

```
RUN powershell -command Write-Host default

# Executed as powershell -command Write-Host hello
SHELL ["powershell", "-command"]
RUN Write-Host hello

# Executed as cmd /S /C echo hello
SHELL ["cmd", "/S"", "/C"]
RUN echo hello
```

Dockerfileshell `SHELL` `RUN` `CMD` `ENTRYPOINT` 。

Windows`SHELL`

```
...
RUN powershell -command Execute-MyCmdlet -param1 "c:\foo.txt"
...
```

docker

```
cmd /S /C powershell -command Execute-MyCmdlet -param1 "c:\foo.txt"
```

◦ cmd.exeshell。 shell`RUN`powershell -command。

◦ `RUN`JSON

```
...
RUN ["powershell", "-command", "Execute-MyCmdlet", "-param1 \"c:\\foo.txt\""]
...
```

JSONcmd.exe。 `SHELL`shellWindowsescape parser

```
# escape=`

FROM windowsservercore
SHELL ["powershell","-command"]
RUN New-Item -ItemType Directory C:\Example
ADD Execute-MyCmdlet.ps1 c:\example\
RUN c:\example\Execute-MyCmdlet -sample 'hello world'
```

```
PS E:\docker\build\shell> docker build -t shell .
Sending build context to Docker daemon 3.584 kB
Step 1 : FROM windowsservercore
 ---> 5bc36a335344
Step 2 : SHELL powershell -command
 ---> Running in 87d7a64c9751
 ---> 4327358436c1
Removing intermediate container 87d7a64c9751
Step 3 : RUN New-Item -ItemType Directory C:\Example
 ---> Running in 3e6ba16b8df9


Directory: C:\
Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----         6/2/2016   2:59 PM                Example


 ---> 1f1dfdcec085
Removing intermediate container 3e6ba16b8df9
Step 4 : ADD Execute-MyCmdlet.ps1 c:\example\
 ---> 6770b4c17f29
Removing intermediate container b139e34291dc
Step 5 : RUN c:\example\Execute-MyCmdlet -sample 'hello world'
 ---> Running in abdcf50dfd1f
Hello from Execute-MyCmdlet.ps1 - passed hello world
 ---> ba0e25255fda
Removing intermediate container abdcf50dfd1f
Successfully built ba0e25255fda
PS E:\docker\build\shell>
```

`SHELL`shell。 Windows`SHELL cmd /S /C /V:ON|OFF` 。

shellzshcshtcshLinux`SHELL`。

Docker 1.12`SHELL`。

Debian / Ubuntu

。 。 。 -y。 。

```
FROM debian

RUN apt-get update \
 && DEBIAN_FRONTEND=noninteractive apt-get install -y \
    git \
    openssh-client \
    sudo \
    vim \
    wget \
 && apt-get clean \
 && rm -rf /var/lib/apt/lists/*
```

Dockerfiles https://riptutorial.com/zh-CN/docker/topic/3161/dockerfiles

5: Dockerfile

1. `FROM`
2. `MAINTAINER LABEL`
3. `apt-get install apk add`
4. `bower.json package.json build.gradle requirements.txt`
5. `npm install pip install`
6.
7. `CMD ENTRYPOINT ENV EXPOSE`

Docker。

> Dockerfile。 。

Examples

Dockerfile

```
# Base image
FROM python:2.7-alpine

# Metadata
MAINTAINER John Doe <johndoe@example.com>

# System-level dependencies
RUN apk add --update \
    ca-certificates \
    && update-ca-certificates \
    && rm -rf /var/cache/apk/*

# App dependencies
COPY requirements.txt /requirements.txt
RUN pip install -r /requirements.txt

# App codebase
WORKDIR /app
COPY . ./

# Configs
ENV DEBUG true
EXPOSE 5000
CMD ["python", "app.py"]
```

MAINTAINERDocker 1.13LABEL。

LABEL Maintainer =“John Doe johndoe@example.com”

Dockerfile https://riptutorial.com/zh-CN/docker/topic/6448/dockerfile

6: DockerDocker

Examples

DockerJenkins CI

JenkinsDockerDockerHostDockerDocker。 DockerDocker。 Jenkins Docker ImageDocker。 Dockerfile

```
FROM jenkins

USER root

RUN cd /usr/local/bin && \
curl https://master.dockerproject.org/linux/amd64/docker > docker  && \
chmod +x docker  && \
groupadd -g 999 docker && \
usermod -a -G docker jenkins

USER Jenkins
```

DockerfileImageDockerDocker。 Docker。 `RUN`UID 999UNIXJenkins。 。 ImageDockerJenkins ImageDockerHostDocker。 UNIX Socket `/var/run/docker.sock`。 UnixJenkins。 `docker run -v /var/run/docker.sock:/var/run/docker.sock --name jenkins MY_CUSTOM_IMAGE_NAME` 。 docker `docker:root`DockerfileUIDJenkins。 Jenkins ContainerDocker。 run`-v jenkins_home:/var/jenkins_home`Jenkins_home。

DockerDocker https://riptutorial.com/zh-CN/docker/topic/8012/dockerdocker

7: Docker

Examples

```
docker events
```

```
docker run... & docker events --filter 'container=$(docker ps -lq)'
```

```
docker ps -lq llast qquiet。 id。
```

Docker https://riptutorial.com/zh-CN/docker/topic/6200/docker

8: Docker

DockerDocker。 Docker。 。 - 。

docker。

Examples

A

```
[root@localhost ~]# docker run -it -v  /data  --name=vol3   8251da35e7a7 /bin/bash
root@d87bf9607836:/# cd /data/
root@d87bf9607836:/data# touch abc{1..10}
root@d87bf9607836:/data# ls
```

abc1 abc10 abc2 abc3 abc4 abc5 abc6 abc7 abc8 abc9

B[cont + P + Q]

```
[root@localhost ~]# docker ps
```

IDd87bf9607836 8251da35e7a7"/ bin / bash"31vol3 [root @ localhost~]

C'docker inspect'

```
[root@localhost ~]# docker inspect  d87bf9607836
```

"Mounts"[{"Name""cdf78fbf79a7c9363948e133abe4c572734cd788c95d36edea0448094ec9121c" "Source""/ var / lib / docker / volumes / cdf78fbf79a7c9363948e133abe4c572734cd788c95d36edea0448094ec9121c / _data" "Destination""/ data""Driver""local" """"RW"

D

```
[root@localhost ~]# docker run -it  --volumes-from  vol3  8251da35e7a7  /bin/bash

root@ef2f5cc545be:/# ls
```

bin boot data dev etc home lib libst media mnt opt proc root run sbin srv sys tmp usr var

`root@ef2f5cc545be:/# ls` / data abc1 abc10 abc2 abc3 abc4 abc5 abc6 abc7 abc8 abc9

E

```
[root@localhost ~]# docker run -it  -v  /etc:/etc1 8251da35e7a7 /bin/bash
```

/ etc/ etc1

Docker https://riptutorial.com/zh-CN/docker/topic/5908/docker

9: Docker

Docker。

- [] []

Examples

-v--volume。 /etc/mnt/etc

```
(on linux) docker run -v "/etc:/mnt/etc" alpine cat /mnt/etc/passwd
(on windows)  docker run -v "/c/etc:/mnt/etc" alpine cat /mnt/etc/passwd
```

。 :ro

```
docker run -v "/etc:/mnt/etc:ro" alpine touch /mnt/etc/passwd
```

```
docker volume create --name="myAwesomeApp"
```

。 -v--volume--volume docker run

```
docker run -d --name="myApp-1" -v="myAwesomeApp:/data/app" myApp:1.5.3
```

/。

```
docker run -d --name="myApp-2" --volumes-from "myApp-1" myApp:1.5.3
```

myApp-2myApp:1.5.3myApp-1myAwesomeApp。 myAwesomeAppmyApp-2/data/appmyApp-1/data/app 。

Docker https://riptutorial.com/zh-CN/docker/topic/1318/docker

10: Docker

docker。

`docker-machine`Docker。

`docker-machine`。 “”。 Docker。

Examples

Docker Machine

shell。

`docker-machine env`docker-machine

`eval $(docker-machine env)`docker-machineshelldocker-machine。

shell--no-proxydocker-machine `eval $(docker-machine env --no-proxy)`

docker-machines `eval $(docker-machine env --no-proxy machinename)`

SSHdocker

shell

- docker-machine

`docker-machine ssh` to ssh into default docker-machine

`docker-machine ssh machinename`sshdocker-machine

- ◦ docker-machine`uptime`uptime`docker-machine ssh default uptime`

Docker

`docker-machine``docker-machine`Docker。 SSHSSL。

Virtualbox

```
docker-machine create --driver virtualbox docker-host-1
```

Docker`generic`

```
docker-machine -D create -d generic --generic-ip-address 1.2.3.4 docker-host-2
```

`--driver``--driver`。

-
-

docker-machinesdockerDocker。

```
docker-machine ls
```

```
NAME              ACTIVE   DRIVER    STATE     URL                            SWARM   DOCKER
ERRORS
docker-machine-1 -         ovh       Running   tcp://1.2.3.4:2376                     v1.11.2
docker-machine-2 -         generic   Running   tcp://1.2.3.5:2376                     v1.11.2
```

```
docker-machine ls --filter state=running
```

```
docker-machine ls --filter state=
```

'side-project-'Golang

```
docker-machine ls --filter name="^side-project-"
```

```
docker-machine ls --format '{{ .URL }}'
```

https://docs.docker.com/machine/reference/ls/ 。

Docker

◦ docker

```
docker-machine upgrade docker-machine-name
```

dockerIP

dockerIP

```
docker-machine ip machine-name
```

Docker https://riptutorial.com/zh-CN/docker/topic/1349/docker

11: docker

Examples

`docker inspect`。

Docker`docker inspect -f ... container_id`

```
docker inspect -f ... $(docker ps -q)
```

```
docker command | grep or awk | tr or cut
```

`docker inspect`“”htophttps://hub.docker.com/r/jess/htop/pid ae1

```
docker inspect -f '{{.Created}}' ae1
```

```
2016-07-14T17:44:14.159094456Z
```

```
docker inspect -f '{{.Path}}' ae1
```

```
htop
```

`docker inspect``docker inspect`

```
"State": { "Status": "running", "Running": true, "Paused": false, "Restarting": false,
"OOMKilled": false, "Dead": false, "Pid": 4525, "ExitCode": 0, "Error": "", "StartedAt": "2016-
07-14T17:44:14.406286293Z", "FinishedAt": "0001-01-01T00:00:00Z"{ ...}
```

```
docker inspect -f '{{.State}}' ae1
```

```
{running true false false false false 4525 0 2016-07-14T17:44:14.406286293Z 0001-01-
01T00:00:00Z}
```

State.Pid

```
docker inspect -f '{{ .State.Pid }}' ae1
```

```
4525
```

docker inspect`[]`

```
docker inspect -f '{{ .Config.Env }}' 7a7
```

```
[DISPLAY=:0 PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin LANG=fr_FR.UTF-8
LANGUAGE=fr_FR:en LC_ALL=fr_FR.UTF-8 DEBIAN_FRONTEND=noninteractive HOME=/home/gg WINEARCH=win32
WINEPREFIX=/home/gg/.wine_captvty]
```

0

```
docker inspect -f '{{ index ( .Config.Env) 0 }}' 7a7
```

```
DISPLAY=:0
```

10

```
docker inspect -f '{{ index ( .Config.Env) 1 }}' 7a7
```

```
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
```

```
docker inspect -f '{{ len .Config.Env }}' 7a7

9

docker inspect -f "{{ index .Config.Cmd $[$(docker inspect -format '{{ len .Config.Cmd }}'
$CID)-1]}}" 7a7
```

docker https://riptutorial.com/zh-CN/docker/topic/6470/docker-

12: Docker

Examples

`registry:latest` v1。 Python。 v2Go。 “”v2 v1

```
docker run -d -p 5000:5000 --name="registry" registry:2
```

Docker Distribution 。

。

AWS S3

AWS S3。 。 `config.yml`

```
storage:
    s3:
        accesskey: AKAAAAAACCCCCCCBBBDA
        secretkey: rn9rjnNuX44iK+26qpM4cDEoOnonbBW98FYaiDtS
        region: us-east-1
        bucket: registry.example.com
        encrypt: false
        secure: true
        v4auth: true
        chunksize: 5242880
        rootdirectory: /registry
```

`accesskey``secretkey`S3IAM 。 `AmazonS3FullAccess`。 `region`S3。 `bucket`。 `encrypt`。 `secure`HTTPS。 `v4auth`truefalse。 `chunksize`S3 API5。 `rootdirectory`S3。

。

Docker https://riptutorial.com/zh-CN/docker/topic/4173/docker

13: Docker

Examples

Docker

```
sudo docker stats $(sudo docker inspect -f "{{ .Name }}" $(sudo docker ps -q))
```

CPU

Docker https://riptutorial.com/zh-CN/docker/topic/5863/docker

14: Docker

Examples

Containerip

IPWeb。

docker-machineMacOSXWindows。

```
$ docker-machine ls

NAME      ACTIVE    DRIVER       STATE     URL                         SWARM
default   *         virtualbox   Running   tcp://192.168.99.100:2376
```

```
$ docker-machine ip default

192.168.99.100
```

Docker

```
docker network create app-backend
```

appBackend。 。

```
docker network ls
```

Docker。 bridge hostnull。 bridge。

```
docker network connect app-backend myAwesomeApp-1
```

myAwesomeApp-1app-backend。 DNSDNS。 DNSbridge。

```
docker network disconnect app-backend myAwesomeApp-1
```

myAwesomeApp-1app-backend。 DNS。

Docker

```
docker network rm app-backend
```

Dockerapp-backend。 。 bridge hostnull。

Docker

```
docker network inspect app-backend
```

app-backend。

```
[
    {
        "Name": "foo",
        "Id": "a0349d78c8fd7c16f5940bdbaf1adec8d8399b8309b2e8a969bd4e3226a6fc58",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.18.0.0/16",
                    "Gateway": "172.18.0.1/16"
                }
            ]
        },
        "Internal": false,
        "Containers": {},
        "Options": {},
        "Labels": {}
    }
]
```

Docker https://riptutorial.com/zh-CN/docker/topic/3221/docker

15: Docker

Docker 。 Swarm。

- docker swarm init [OPTIONS]
- swarm/ docker swarm join [OPTIONS] HOSTPORT
- docker service create [OPTIONS] IMAGE [COMMAND] [ARG ...]
- docker service inspect [OPTIONS] SERVICE [SERVICE ...]
- docker service ls [OPTIONS]
- docker service rm SERVICE [SERVICE ...]
- docker service scale SERVICE = REPLICAS [SERVICE = REPLICAS ...]
- docker service ps [OPTIONS] SERVICE [SERVICE ...]
- docker service update [OPTIONS] SERVICE

Swarm

- Docker Engine
-
-
-
-
-
-
-
-
-

SwarmDocker Swarm

SwarmCLI

```
docker swarm init [OPTIONS]
```

swarm/

```
docker swarm join [OPTIONS] HOST:PORT
```

```
docker service create [OPTIONS] IMAGE [COMMAND] [ARG...]
```

```
docker service inspect [OPTIONS] SERVICE [SERVICE...]
```

```
docker service ls [OPTIONS]
```

```
docker service rm SERVICE [SERVICE...]
```

```
docker service scale SERVICE=REPLICAS [SERVICE=REPLICAS...]
```

```
docker service ps [OPTIONS] SERVICE [SERVICE...]
```

```
docker service update [OPTIONS] SERVICE
```

Examples

docker-machineVirtualBoxLinuxswarm

```
# Create the nodes
# In a real world scenario we would use at least 3 managers to cover the fail of one manager.
docker-machine create -d virtualbox manager
docker-machine create -d virtualbox worker1

# Create the swarm
# It is possible to define a port for the *advertise-addr* and *listen-addr*, if none is
defined the default port 2377 will be used.
docker-machine ssh manager \
    docker swarm init \
    --advertise-addr $(docker-machine ip manager)
    --listen-addr $(docker-machine ip manager)

# Extract the Tokens for joining the Swarm
# There are 2 different Tokens for joining the swarm.
MANAGER_TOKEN=$(docker-machine ssh manager docker swarm join-token manager --quiet)
WORKER_TOKEN=$(docker-machine ssh manager docker swarm join-token worker --quiet)


# Join a worker node with the worker token
docker-machine ssh worker1 \
    docker swarm join \
    --token $WORKER_TOKEN \
    --listen-addr $(docker-machine ip worker1) \
    $(docker-machine ip manager):2377
```

。

```
# grab the ipaddress:port of the manager (second last line minus the whitespace)
export MANAGER_ADDRESS=$(docker swarm join-token worker | tail -n 2 |  tr -d '[[:space:]]')

# grab the manager and worker token
export MANAGER_TOKEN=$(docker swarm join-token manager -q)
export WORKER_TOKEN=$(docker swarm join-token worker -q)
```

-q。 swarm。

swarm。

```
docker swarm join --token $WORKER_TOKEN $MANAGER_ADDRESS
```

。

Webapi。

swarm。 NASNFSGFS2。 。 Docker。 `/nfs/` shared。

。

IP`10.0.9.0/24` `hello-network`

```
docker network create \
  --driver overlay \
  --subnet 10.0.9.0/24 \
  --opt encrypted \
  hello-network
```

。 postgresql。 `nfs/postgres`

```
docker service create --replicas 1 --name hello-db \
       --network hello-network -e PGDATA=/var/lib/postgresql/data \
       --mount type=bind,src=/nfs/postgres,dst=/var/lib/postgresql/data \
       kiasaki/alpine-postgres:9.5
```

`--network hello-network`、`--mount`。

API

API`username/hello-api`API。

```
docker service create --replicas 1 --name hello-api \
       --network hello-network \
       -e NODE_ENV=production -e PORT=80 -e POSTGRESQL_HOST=hello-db \
       username/hello-api
```

。 Docker swarmDNSAPIDNS。

nginxAPI。 nginx

```
docker service create --replicas 1 --name hello-load-balancer \
       --network hello-network \
       --mount type=bind,src=/nfs/nginx/nginx.conf,dst=/etc/nginx/nginx.conf \
       -p 80:80 \
       nginx:1.10-alpine
```

`-p`。 。

- 。
- 。
- Drain。 。

```
#Following commands can be used on swarm manager(s)
docker node update --availability drain node-1
#to verify:
docker node ls
```

Swarm

`docker node promote`

```
docker node promote node-3 node-2

Node node-3 promoted to a manager in the swarm.
Node node-2 promoted to a manager in the swarm.
```

`docker node demote`

```
docker node demote node-3 node-2

Manager node-3 demoted in the swarm.
Manager node-2 demoted in the swarm.
```

```
#Run the following on the worker node to leave the swarm.

docker swarm leave
Node left the swarm.
```

*Manager*Manager。 --force

```
#Manager Node

docker swarm leave --force
Node left the swarm.
```

Swarm`docker node ls`。

```
docker node rm node-2

node-2
```

Docker https://riptutorial.com/zh-CN/docker/topic/749/docker

16:

◦ SSHshell◦ `SIGNALSIGINT`◦ ◦ `supervisordSIGNAL`◦

“”◦ docker`docker host docker exec -it container_name /bin/bahs` ◦ sshshell◦

Examples

Dockerfile + supervisord.conf

Apache WebSSH`supervisord` ◦

`supervisord.conf`

```
[supervisord]
nodaemon=true

[program:sshd]
command=/usr/sbin/sshd -D

[program:apache2]
command=/bin/bash -c "source /etc/apache2/envvars && exec /usr/sbin/apache2 -DFOREGROUND"
```

`Dockerfile`

```
FROM ubuntu:16.04
RUN apt-get install -y openssh-server apache2 supervisor
RUN mkdir -p /var/lock/apache2 /var/run/apache2 /var/run/sshd /var/log/supervisor
COPY supervisord.conf /etc/supervisor/conf.d/supervisord.conf
CMD ["/usr/bin/supervisord"]
```

```
docker build -t supervisord-test .
```

```
$ docker run -p 22 -p 80 -t -i supervisord-test
2016-07-26 13:15:21,101 CRIT Supervisor running as root (no user in config file)
2016-07-26 13:15:21,101 WARN Included extra file     "/etc/supervisor/conf.d/supervisord.conf"
during parsing
2016-07-26 13:15:21,112 INFO supervisord started with pid 1
2016-07-26 13:15:21,113 INFO spawned: 'sshd' with pid 6
2016-07-26 13:15:21,115 INFO spawned: 'apache2' with pid 7
...
```

https://riptutorial.com/zh-CN/docker/topic/4053/

17: DockerIptables

iptablesdocker。

。 。

- iptables -I DOCKER [RULE ...] [ACCEPT | DROP] //a DOCKER
- iptables -D DOCKER [RULE ...] [ACCEPT | DROP] //DOCKER
- ipset restore </etc/ipfriends.conf //ipset *ipfriends*

ext_if	Docker。
XXX.XXX.XXX.XXX	DockerIP。
YYY.YYY.YYY.YYY	DockerIP。
ipfriends	DockerIPipset。

Dockeriptables。 “”。

nginx-proxy+HTTPSWeb。 IP XXX.XXX.XXX.XXXWeb。

```
$ iptables -A INPUT -i eth0 -p tcp -s XXX.XXX.XXX.XXX -j ACCEPT
$ iptables -P INPUT DROP
```

。

Docker。 。 INPUTFORWARD。

。 Dockeriptables。 DOCKERFORWARD。

```
$ iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination
DOCKER-ISOLATION  all  --  anywhere             anywhere
DOCKER     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
DOCKER     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere             ctstate RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain DOCKER (2 references)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             172.18.0.4           tcp dpt:https
ACCEPT     tcp  --  anywhere             172.18.0.4           tcp dpt:http

Chain DOCKER-ISOLATION (1 references)
target     prot opt source               destination
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere
RETURN     all  --  anywhere             anywhere
```

---

https://docs.docker.com/v1.5/articles/networking/ DockerIP。

```
$ iptables -I DOCKER -i ext_if ! -s 8.8.8.8 -j DROP
```

DOCKER。 Docker。

- IPIPsrc IP。
- 8.8.8.8。
- HTTPWebIP。

ipset 。 IPipsetIP。 ipsetiptable。

```
$ iptables -I DOCKER -i ext_if -m set ! --match-set my-ipset src -j DROP
```

。 。 docker

```
$ iptables -I DOCKER -i ext_if -m state --state ESTABLISHED,RELATED -j ACCEPT
```

iptables。 DROPESTABLISHEDDOCKERipset。

iptable-Iiptables

```
// Drop rule for non matching IPs
$ iptables -I DOCKER -i ext_if -m set ! --match-set my-ipset src -j DROP
// Then Accept rules for established connections
$ iptables -I DOCKER -i ext_if -m state --state ESTABLISHED,RELATED -j ACCEPT
$ iptables -I DOCKER -i ext_if ... ACCEPT // Then 3rd custom accept rule
$ iptables -I DOCKER -i ext_if ... ACCEPT // Then 2nd custom accept rule
$ iptables -I DOCKER -i ext_if ... ACCEPT // Then 1st custom accept rule
```

。

Examples

DockerIP

ipset 。 。 Debian。

```
$ apt-get update
$ apt-get install ipset
```

ipsetDockerIP。

```
$ vi /etc/ipfriends.conf
# Recreate the ipset if needed, and flush all entries
create -exist ipfriends hash:ip family inet hashsize 1024 maxelem 65536
flush
# Give access to specific ips
add ipfriends XXX.XXX.XXX.XXX
add ipfriends YYY.YYY.YYY.YYY
```

ipset。

```
$ ipset restore < /etc/ipfriends.conf
```

Docker。

```
$ docker ps
```

iptables。 。

```
// All requests of src ips not matching the ones from ipset ipfriends will be dropped.
$ iptables -I DOCKER -i ext_if -m set ! --match-set ipfriends src -j DROP
// Except for requests coming from a connection already established.
$ iptables -I DOCKER -i ext_if -m state --state ESTABLISHED,RELATED -j ACCEPT
```

。

```
$ iptables -D DOCKER -i ext_if -m set ! --match-set ipfriends src -j DROP
$ iptables -D DOCKER -i ext_if -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Docker

iptables

DockerIptables https://riptutorial.com/zh-CN/docker/topic/9201/dockeriptables

18: API v2Docker/

dockerDocker Hub。 。

sudo docker run -p 50005000	docker50005000。
--name	“docker ps”。
-v'pwd'/ certs/ certs	/ certsCURRENT_DIR / certs“” 。
-e REGISTRY_HTTP_TLS_CERTIFICATE = / certs / server.crt	/certs/server.crt。 env
-e REGISTRY_HTTP_TLS_KEY = / certs / server.key	RSAserver.key。
-v / root / images/ var / lib / registry /	。 / root / images。 。
2	docker hub«2»2。

docker-engine

SSL

Examples

RSA `openssl genrsa -des3 -out server.key 4096`

Openssl。 。 123456。

```
openssl req -new -key server.key -out server.csr
```

。 “”docker。 mydomain.com

RSA `cp server.key server.key.org && openssl rsa -in server.key.org -out server.key`

。 .key.csr.crt。

```
openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

server.keyserver.crt 。

。

server.keyserver.crt/ root / certs

docker`cd` *certs*。

> level = fatal msg =“open /certs/server.crt”

`cd /root` `sudo docker run -p 5000:5000 --restart=always --name registry -v `pwd`/certs:/certs -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/server.crt -e REGISTRY_HTTP_TLS_KEY=/certs/server.key -v /root/Documents:/var/lib/registry/ registry:2`/ `sudo docker run -p 5000:5000 --restart=always --name registry -v `pwd`/certs:/certs -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/server.crt -e REGISTRY_HTTP_TLS_KEY=/certs/server.key -v /root/Documents:/var/lib/registry/ registry:2`

。

docker

。 *server.crt*docker。 。

server.crt/etc/docker/certs.d/mydomain.com:5000/。 *ca-certificates.crt* `mv /etc/docker/certs.d/mydomain.com:5000/server.crt /etc/docker/certs.d/mydomain.com:5000/ca-certificates.crt`

PULL `docker pull mydomain.com:5000/nginx`

1. hub.docker.comdocker `docker pull nginx`
2. `docker tag IMAGE_ID mydomain.com:5000/nginx` `docker tag IMAGE_ID mydomain.com:5000/nginx` `docker images`IMAGE_ID
3. `docker push mydomain.com:5000/nginx`

API v2Docker/ https://riptutorial.com/zh-CN/docker/topic/8707/api---v2docker-

19:

- docker volume create --name <volume_name><volume_name>
- docker run -v <volume_name><mount_point> -d crramirez / limesurveylatest<mount_point><volume_name>

--name <volume_name>	
-v <volume_name><mount_point>	

docker。 Docker。

- -v。
- docker volume rm
- --volumes-from。
- 。
- docker volume。

Examples

docker。 Limesurvey

```
docker volume create --name mysql
docker volume create --name upload

docker run -d --name limesurvey -v mysql:/var/lib/mysql -v upload:/app/upload -p 80:80
crramirez/limesurvey:latest
```

。 。

```
docker volume create --name=data
echo "Hello World" |  docker run -i --rm=true -v data:/data ubuntu:trusty tee /data/hello.txt
```

```
docker run -d --name backup -v data:/data ubuntu:trusty tar -czvf /tmp/data.tgz /data
docker cp backup:/tmp/data.tgz data.tgz
docker rm -fv backup
```

```
tar -xzvf data.tgz
cat data/hello.txt
```

https://riptutorial.com/zh-CN/docker/topic/7429/

20: 1.12

Examples

1.12

consul dockerDocker 1.12swarmdocker swarmconsul。 http://qnib.org/2016/08/11/consul-service/ 。 docker swarm。 ipsdocker swarmdns。

docker 1.12 swarm。

docker。 syslogdockerd`--log-driver=syslog`。

```
docker network create consul-net -d overlay
```

1--replicas1

```
docker service create --name consul-seed \
  -p 8301:8300 \
  --network consul-net \
  -e 'CONSUL_BIND_INTERFACE=eth0' \
  consul agent -server -bootstrap-expect=3  -retry-join=consul-seed:8301 -retry-join=consul-
cluster:8300
```

。

```
docker service create --name consul-cluster \
  -p 8300:8300 \
  --network consul-net \
  --replicas 3 \
  -e 'CONSUL_BIND_INTERFACE=eth0' \
  consul agent -server -retry-join=consul-seed:8301 -retry-join=consul-cluster:8300
```

consul。 docker

```
docker exec <containerid> consul members
```

1.12 https://riptutorial.com/zh-CN/docker/topic/6437/1-12

21: Docker ImageMongoChef

Docker ImageMongoChef。

Examples

1. MongoBase 64。 chef data_bags
2. suppermarketdocker cookbook。 custom_mongo“docker”“> 2.0”metadata.rb
3.
4. MongoRep Set

1

mongo-keyfiledata_bagkeyfile。 chefdata_bags。

```
openssl rand -base64 756 > <path-to-keyfile>
```

```
{
  "id": "keyfile",
  "comment": "Mongo Repset keyfile",
  "key-file": "generated base 64 key above"
}
```

2docker cookbookcustom_mongo cookbook

```
knife cookbook site download docker
knife cookbook create custom_mongo
```

custom_mongometadat.rb

```
depends          'docker', '~> 2.0'
```

3

```
default['custom_mongo']['mongo_keyfile'] = '/data/keyfile'
default['custom_mongo']['mongo_datadir'] = '/data/db'
default['custom_mongo']['mongo_datapath'] = '/data'
default['custom_mongo']['keyfilename'] = 'mongodb-keyfile'
```

```
#
# Cookbook Name:: custom_mongo
# Recipe:: default
#
# Copyright 2017, Innocent Anigbo
#
# All rights reserved - Do Not Redistribute
```

```
#

data_path = "#{node['custom_mongo']['mongo_datapath']}"
data_dir = "#{node['custom_mongo']['mongo_datadir']}"
key_dir = "#{node['custom_mongo']['mongo_keyfile']}"
keyfile_content = data_bag_item('mongo-keyfile', 'keyfile')
keyfile_name = "#{node['custom_mongo']['keyfilename']}"

#chown of keyfile to docker user
execute 'assign-user' do
 command "chown 999 #{key_dir}/#{keyfile_name}"
 action :nothing
end

#Declaration to create Mongo data DIR and Keyfile DIR
%W[ #{data_path} #{data_dir} #{key_dir} ].each do |path|
directory path do
  mode '0755'
  end
end

#declaration to copy keyfile from data_bag to keyfile DIR on your mongo server
file "#{key_dir}/#{keyfile_name}" do
  content keyfile_content['key-file']
  group 'root'
  mode '0400'
  notifies :run, 'execute[assign-user]', :immediately
end

#Install docker
docker_service 'default' do
  action [:create, :start]
end

#Install mongo 3.4.2
docker_image 'mongo' do
  tag '3.4.2'
  action :pull
end
```

mongo-role

```
{
  "name": "mongo-role",
  "description": "mongo DB Role",
  "run_list": [
    "recipe[custom_mongo]"
  ]
}
```

mongo

```
knife node run_list add FQDN_of_node_01 'role[mongo-role]'
knife node run_list add FQDN_of_node_02 'role[mongo-role]'
knife node run_list add FQDN_of_node_03 'role[mongo-role]'
```

4Mongorepset

Mongo。 01--authMongo

```
docker run --name mongo -v /data/db:/data/db -v /data/keyfile:/opt/keyfile --hostname="mongo-
01.example.com" -p 27017:27017 -d mongo:3.4.2 --keyFile /opt/keyfile/mongodb-keyfile --auth
```

01docker containershelladmin

```
docker exec -it mongo /bin/sh
    mongo
    use admin
    db.createUser( {
         user: "admin-user",
         pwd: "password",
         roles: [ { role: "userAdminAnyDatabase", db: "admin" } ]
       });
```

root

```
db.createUser( {
         user: "RootAdmin",
         pwd: "password",
         roles: [ { role: "root", db: "admin" } ]
       });
```

01Docker。 DIR。 01Mongorepset

```
docker rm -fv mongo
docker run --name mongo-uat -v /data/db:/data/db -v /data/keyfile:/opt/keyfile --
hostname="mongo-01.example.com" -p 27017:27017 -d mongo:3.4.2 --keyFile /opt/keyfile/mongodb-
keyfile --replSet "rs0"
```

rep set0203mongo

```
docker run --name mongo -v /data/db:/data/db -v /data/keyfile:/opt/keyfile --hostname="mongo-
02.example.com" -p 27017:27017 -d mongo:3.4.2 --keyFile /opt/keyfile/mongodb-keyfile --replSet
"rs0"
docker run --name mongo -v /data/db:/data/db -v /data/keyfile:/opt/keyfile --hostname="mongo-
03.example.com" -p 27017:27017 -d mongo:3.4.2 --keyFile /opt/keyfile/mongodb-keyfile --replSet
"rs0"
```

01root

```
use admin
db.auth("RootAdmin", "password");
rs.initiate()
```

0123repset0

```
rs.add("mongo-02.example.com")
rs.add("mongo-03.example.com")
```

db.printSlaveReplicationInfoSyncedToBehind。 0

```
rs0:PRIMARY> db.printSlaveReplicationInfo()
   source: mongo-02.example.com:27017
           syncedTo: Mon Mar 27 2017 15:01:04 GMT+0000 (UTC)
           0 secs (0 hrs) behind the primary
   source: mongo-03.example.com:27017
           syncedTo: Mon Mar 27 2017 15:01:04 GMT+0000 (UTC)
           0 secs (0 hrs) behind the primary
```

Docker ImageMongoChef https://riptutorial.com/zh-CN/docker/topic/10014/docker-imagemongochef

22: docker build

`docker build -t mytag .---> Running in d9a42e53eb5a The command '/bin/sh -c returned a non-zero code: 127` 127“121276

Examples

```
docker build -t mytag .
```

```
---> Running in d9a42e53eb5a
```

shell

```
docker run -it d9a42e53eb5a /bin/bash
```

/ bin / bash/ bin / sh

docker build https://riptutorial.com/zh-CN/docker/topic/8078/docker-build

23:

Examples

Wordpress

DockerfileFROM php5.6-apache

Dockerfile https://github.com/docker-library/php/blob/master/5.6/apache/Dockerfile

FROM debianjessieDebian jessie。

https://riptutorial.com/zh-CN/docker/topic/8077/

24:

Examples

`docker inspect`

```
docker run
```

`docker run -e password=abc`

`docker run --env-file myfile`

myfile

`password1=abc password2=def`

`docker run -v $(pwd)/my-secret-file:/secret-file`

keywhiz https://square.github.io/keywhiz/

https://www.hashicorp.com/blog/vault.html

etcd https://xordataexchange.github.io/crypt/

https://riptutorial.com/zh-CN/docker/topic/6481/

25:

-	Dockerfile `FROM` 。

Examples

Dockerfile

Dockerfile`docker build`。

```
docker build -t image-name path
```

Dockerfile`Dockerfile` `-f`Dockerfile。

```
docker build -t image-name -f Dockerfile2 .
```

`dockerbuild-example:1.0.0``dockerbuild-example:1.0.0``Dockerfile`

```
$ ls
Dockerfile Dockerfile2

$ docker build -t dockerbuild-example:1.0.0 .

$ docker build -t dockerbuild-example-2:1.0.0 -f Dockerfile2 .
```

`docker build` usage 。

~ Dockerfile。 `docker build -t mytag .``docker build -t mytag .`

docker。 Dockerfile。

`.dockerignore` 。 `.gitignore`。

Dockerfile

```
FROM node:5
```

`FROM`。 。

```
WORKDIR /usr/src/app
```

`WORKDIR``cd` 。 `RUN cd` 。

```
RUN npm install cowsay knock-knock-jokes
```

RUN。

```
COPY cowsay-knockknock.js ./
```

COPY*path*docker build *path*。

```
CMD node cowsay-knockknock.js
```

CMD。 docker run。

;Dockerfile 。

ENTRYPOINTCMD

Dockerfile。 CMDENTRYPOINTENTRYPOINT/bin/sh -c 。 /。 ENTRYPOINT CMD。

Dockerfile

```
FROM ubuntu:16.04
CMD ["/bin/date"]
```

/bin/sh -cENTRYPOINT/bin/date 。 /bin/sh -c /bin/date 。

```
$ docker build -t test .
$ docker run test
Tue Jul 19 10:37:43 UTC 2016
```

CMD 。

```
$ docker run test /bin/hostname
bf0274ec8820
```

ENTRYPOINTDocker CMD。 Dockerfile

```
FROM ubuntu:16.04
ENTRYPOINT ["/bin/echo"]
CMD ["Hello"]
```

```
$ docker build -t test .
$ docker run test
Hello
```

/bin/echo

```
$ docker run test Hi
Hi
```

Dockerfileecho--entrypoint

```
$ docker run --entrypoint=/bin/hostname test
b2c70e74df18
```

`ENTRYPOINTCMD。`

Dockerfile

```
EXPOSE <port> [<port>...]
```

Docker

> `EXPOSE`Docker。 `EXPOSE`。 `-p`–`P`。 。

Dockerfile

```
EXPOSE 8765
```

`docker run`

```
-p 8765:8765
```

ENTRYPOINTCMD

。 Dockerfile

`ENTRYPOINT [ "nethogs"] CMD ["wlan0"]`

a

`docker built -t inspector .`

Dockerfile

`docker run -it --net=host --rm inspector`

nethogswlan0

eth0wlan1ra1 ...

`docker run -it --net=host --rm inspector eth0`

`docker run -it --net=host --rm inspector wlan1`

Docker Hub

Docker Hubdocker repo host。 `docker login`docker hub。

```
docker login
```

```
Login with your Docker ID to push and pull images from Docker Hub.
If you don't have a Docker ID, head over to https://hub.docker.com to create one.

Username: cjsimon
Password:
Login Succeeded
```

docker。 。 。

```
docker login quay.io
```

。 `server/username/reponame:tag` 。 Docker Hub。 。

```
docker tag mynginx quay.io/cjsimon/mynginx:latest
```

。 。

`docker images`。 push。

```
docker push quay.io/cjsimon/mynginx:latest
```

`-a`

```
docker pull quay.io/cjsimon/mynginx:latest
```

Dockerfile`wget`GitHub。

Docker

```
$ docker build --build-arg http_proxy=http://myproxy.example.com:3128 \
               --build-arg https_proxy=http://myproxy.example.com:3128 \
               --build-arg no_proxy=internal.example.com \
               -t test .
```

`build-arg`。

https://riptutorial.com/zh-CN/docker/topic/713/

26:

Examples

DockervolumeDocker。 Dockerdocker volume。 。

Web“”Docker。

“”Docker。 “”VOLUME Dockerfile-vdocker run-v /path/on/container 。 “”--volumes-from docker run--volumes-from。

```
docker run -d --name "mysql-data" -v "/var/lib/mysql" alpine /bin/true
```

“”。 。

```
docker run -d --name="mysql" --volumes-from="mysql-data" mysql
```

mysqlmysql-data 。

Dockervolume。

volume

```
docker run -d --name "mysql-1" -v "/var/lib/mysql" mysql
```

mysql。 /var/lib/mysql。 。 。 。 。

--volumes-from

```
docker run -d --name="mysql-2" --volumes-from="mysql-1" mysql
```

mysql-2mysql-1/var/lib/mysql 。

https://riptutorial.com/zh-CN/docker/topic/3224/

27:

- docker inspect [OPTIONS] CONTAINER | IMAGE [CONTAINER | IMAGE ...]

Examples

```
docker inspect <container>
```

```
docker inspect -f '<format>' <container>
```

```
docker inspect -f '{{ .NetworkSettings }}' <container>
```

IP

```
docker inspect -f '{{ .NetworkSettings.IPAddress }}' <container>
```

-fGo

```
docker inspect -f '{{ json .NetworkSettings }}' {{containerIdOrName}}
```

jsonJSON。

pythonJSON

```
docker inspect -f '{{ json .NetworkSettings }}' <container> | python -mjson.tool
```

docker。

“ jq ”`docker inspect`。

```
docker inspect -f '{{ json .NetworkSettings }}' aa1 | jq [.Gateway]
```

```
[
  "172.17.0.1"
]
```

。 `docker inspect`。 `Config.Env` `index`

```
docker inspect --format '{{ index (index .Config.Env) 0 }}' <container>
```

`1`

```
docker inspect --format '{{ index (index .Config.Env) 1 }}' <container>
```

`len`

```
docker inspect --format '{{ len .Config.Env }}' <container>
```

```
docker inspect -format "{{ index .Config.Cmd $[$(docker inspect -format '{{ len .Config.Cmd
}}' <container>)-1]}}" <container>
```

`docker inspect`keyvalue`docker inspect` jess / spotify

```
"Config": { "Hostname": "8255f4804dde", "Domainname": "", "User": "spotify", "AttachStdin":
false, "AttachStdout": false, "AttachStderr": false, "Tty": false, "OpenStdin": false,
"StdinOnce": false, "Env": [ "DISPLAY=unix:0",
"PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin", "HOME=/home/spotify" ],
"Cmd": [ "-stylesheet=/home/spotify/spotify-override.css" ], "Image": "jess/spotify", "Volumes":
null, "WorkingDir": "/home/spotify", "Entrypoint": [ "spotify" ], "OnBuild": null, "Labels": {}
},
```

Config

```
docker inspect -f '{{.Config}}' 825
```

```
{8255f4804dde spotify false false false map[] false false false [DISPLAY=unix:0
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin HOME=/home/spotify] [-
stylesheet=/home/spotify/spotify-override.css] false jess/spotify map[] /home/spotify [spotify]
false [] map[] }
```

Config.Image

```
docker inspect -f '{{index (.Config) "Image" }}' 825
```

```
jess/spotify
```

Config.Cmd

```
docker inspect -f '{{.Config.Cmd}}' 825
```

```
[-stylesheet=/home/spotify/spotify-override.css]
```

ID。 CentOS 6

```
➜  ~ docker images
REPOSITORY        TAG               IMAGE ID          CREATED           SIZE
centos            centos6           cf2c3ece5e41      2 weeks ago       194.6 MB
```

- ➜ ~ docker inspect cf2c3ece5e41
- ➜ ~ docker inspect centos:centos6

JSON

```
[
    {
        "Id": "sha256:cf2c3ece5e418fd063bfad5e7e8d083182195152f90aac3a5ca4dbfbf6a1fc2a",
        "RepoTags": [
            "centos:centos6"
        ],
        "RepoDigests": [],
        "Parent": "",
        "Comment": "",
        "Created": "2016-07-01T22:34:39.970264448Z",
        "Container": "b355fe9a01a8f95072e4406763138c5ad9ca0a50dbb0ce07387ba905817d6702",
        "ContainerConfig": {
            "Hostname": "68a1f3cfce80",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) CMD [\"/bin/bash\"]"
            ],
            "Image":
"sha256:cdbcc7980b002dc19b4d5b6ac450993c478927f673339b4e6893647fe2158fa7",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {
                "build-date": "20160701",
                "license": "GPLv2",
                "name": "CentOS Base Image",
                "vendor": "CentOS"
            }
        },
        "DockerVersion": "1.10.3",
        "Author": "https://github.com/CentOS/sig-cloud-instance-images",
        "Config": {
            "Hostname": "68a1f3cfce80",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/bash"
            ],
            "Image":
"sha256:cdbcc7980b002dc19b4d5b6ac450993c478927f673339b4e6893647fe2158fa7",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": {
                "build-date": "20160701",
                "license": "GPLv2",
                "name": "CentOS Base Image",
                "vendor": "CentOS"
            }
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 194606575,
        "VirtualSize": 194606575,
        "GraphDriver": {
            "Name": "aufs",
            "Data": null
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
                "sha256:2714f4a6cdee9d4c987fef019608a4f61f1cda7ccf423aeb8d7d89f745c58b18"
            ]
        }
    }
]
```

docker `inspect--format`Go 。 pipe / sed / grep。

IP

```
docker inspect --format '{{ .NetworkSettings.IPAddress }}' 7786807d8084
```

。

init PID

```
docker inspect --format '{{ .State.Pid }}' 7786807d8084
```

`/procstrace`。

```
docker inspect --format 'Container {{ .Name }} listens on {{ .NetworkSettings.IPAddress }}:{{
range $index, $elem := .Config.ExposedPorts }}{{ $index }}{{ end }}' 5765847de886 7786807d8084
```

```
Container /redis listens on 172.17.0.3:6379/tcp
Container /api listens on 172.17.0.2:4000/tcp
```

docker inspect

`docker inspect`。

stdoutstderr`docker inspect`。

`docker inspect <container-id> | grep Source`

stdoutstderr。

stdout / stderr

```
docker logs --follow <containerid>
```

◦ docker◦

https://riptutorial.com/zh-CN/docker/topic/1336/

28:

Examples

dockerubuntu

docker**2 GB RAM** 。 **4GB** 。

1. gitmake

   ```
   sudo apt-get install make git-core -y
   ```

2. 4.2

   ```
   sudo apt-get install linux-generic-lts-xenial
   ```

3.
   ```
   sudo reboot
   ```

4. `criudocker checkpoint`

   ```
   sudo apt-get install libprotobuf-dev libprotobuf-c0-dev protobuf-c-compiler protobuf-
   compiler python-protobuf libnl-3-dev libcap-dev -y
   wget http://download.openvz.org/criu/criu-2.4.tar.bz2 -O - | tar -xj
   cd criu-2.4
   make
   make install-lib
   make install-criu
   ```

5. criu

   ```
   sudo criu check
   ```

6. dockerdockerdocker

   ```
   cd ~
   wget -qO- https://get.docker.com/ | sh
   sudo usermod -aG docker $(whoami)
   ```

- **docker**。

   ```
   git clone https://github.com/boucher/docker
   cd docker
   git checkout docker-checkpoint-restore
   make #that will take some time - drink a coffee
   DOCKER_EXPERIMENTAL=1 make binary
   ```

7. docker。 。 `<version>`

```
sudo service docker stop
sudo cp $(which docker) $(which docker)_ ; sudo cp ./bundles/latest/binary-client/docker-
<version>-dev $(which docker)
sudo cp $(which docker-containerd) $(which docker-containerd)_ ; sudo cp
./bundles/latest/binary-daemon/docker-containerd $(which docker-containerd)
sudo cp $(which docker-containerd-ctr) $(which docker-containerd-ctr)_ ; sudo cp
./bundles/latest/binary-daemon/docker-containerd-ctr $(which docker-containerd-ctr)
sudo cp $(which docker-containerd-shim) $(which docker-containerd-shim)_ ; sudo cp
./bundles/latest/binary-daemon/docker-containerd-shim $(which docker-containerd-shim)
sudo cp $(which dockerd) $(which dockerd)_ ; sudo cp ./bundles/latest/binary-
daemon/dockerd $(which dockerd)
sudo cp $(which docker-runc) $(which docker-runc)_ ; sudo cp ./bundles/latest/binary-
daemon/docker-runc $(which docker-runc)
sudo service docker start
```

- 。 docker_ 。

docker。

```
# create docker container
export cid=$(docker run -d --security-opt seccomp:unconfined busybox /bin/sh -c 'i=0; while
true; do echo $i; i=$(expr $i + 1); sleep 1; done')

# container is started and prints a number every second
# display the output with
docker logs $cid

# checkpoint the container
docker checkpoint create $cid checkpointname

# container is not running anymore
docker np

# lets pass some time to make sure

# resume container
docker start $cid --checkpoint=checkpointname

# print logs again
docker logs $cid
```

https://riptutorial.com/zh-CN/docker/topic/5291/

29:

- [] [[]]
- docker inspect [OPTIONS] CONTAINER | IMAGE [CONTAINER | IMAGE ...]
- docker pull [OPTIONS] NAME [TAG | @DIGEST]
- docker rmi [OPTIONS] IMAGE [IMAGE ...]
- docker tag [OPTIONS] IMAGE [TAG] [REGISTRYHOST /] [USERNAME /] NAME [TAG]

Examples

Docker Hub

Docker Hub。 DockerDocker HubDocker。 Docker`ubuntu``docker run ubuntu`Docker`ubuntu`。 `docker pull``docker pull`Docker Hub。

```
docker pull ubuntu
docker pull ubuntu:14.04
```

。 。 `registry.example.com``ubuntu:14.04`

```
docker pull registry.example.com/username/ubuntu:14.04
```

```
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
hello-world         latest              693bce725149        6 days ago          967 B
postgres            9.5                 0f3af79d8673        10 weeks ago        265.7 MB
postgres            latest              0f3af79d8673        10 weeks ago        265.7 MB
```

Docker

ID	`693bce725149`
	`hello-world` *`:latest`*
+	`hello-world:latest`
	`hello-world@sha256:e52be8ffeeb1f374f440893189cd32f44cb166650e7ab185fa7735b7dc48d619`

。 `docker images --digests` 。

`docker rmi`

```
docker rmi <image name>
```

。 。

```
docker rmi registry.example.com/username/myAppImage:1.3.5
```

ID

```
docker rmi 693bce725149
```

IDID

```
docker rmi 693
```

; docker rmi“”。

。 。

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED                  STATUS
PORTS               NAMES
5483657ee07b        hello-world         "/hello"            Less than a second ago   Exited
(0) 2 seconds ago                       small_elion

$ docker rmi hello-world
Untagged: hello-world:latest

$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED                  STATUS
PORTS               NAMES
5483657ee07b        693bce725149        "/hello"            Less than a second ago   Exited
(0) 12 seconds ago                       small_elion
```

```
docker rmi $(docker images -qa)
```

-f

```
docker rmi -f $(docker images -qa)
```

“”

```
docker images -q --no-trunc -f dangling=true | xargs -r docker rmi
```

Docker Hub

Docker Hub

```
docker search <term>
```

```
$ docker search nginx
NAME                      DESCRIPTION                                      STARS     OFFICIAL
AUTOMATED
nginx                     Official build of Nginx.                         3565      [OK]
jwilder/nginx-proxy       Automated Nginx reverse proxy for docker c...    717
```

```
[OK]
richarvey/nginx-php-fpm   Container running Nginx + PHP-FPM capable ...   232
[OK]
...
```

```
docker inspect <image>
```

JSON。 jq。

```
docker inspect <image> | jq -r '.[0].Author'
```

。

```
docker tag ubuntu:latest registry.example.com/username/ubuntu:latest
```

```
docker tag myApp:1.4.2 myApp:latest
docker tag myApp:1.4.2 registry.example.com/company/myApp:1.4.2
```

Docker

```
docker save -o ubuntu.latest.tar ubuntu:latest
```

ubuntu:latesttarballubuntu.latest.tar 。 tarballrsync。

tarball

```
docker load -i /tmp/ubuntu.latest.tar
```

ubuntu:latest。

https://riptutorial.com/zh-CN/docker/topic/690/

30:

- docker rm [OPTIONS] CONTAINER [CONTAINER ...]
- docker attach [OPTIONS] CONTAINER
- docker exec [OPTIONS] CONTAINER COMMAND [ARG ...]
- docker ps []
- docker logs [OPTIONS] CONTAINER
- docker inspect [OPTIONS] CONTAINER | IMAGE [CONTAINER | IMAGE ...]

- containerdocker`<container>`container id`<CONTAINER_NAME>` 。 ID。

Examples

```
$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                     NAMES
2bc9b1988080        redis               "docker-entrypoint.sh"   2 weeks ago         Up 2
hours          0.0.0.0:6379->6379/tcp    elephant-redis
817879be2230        postgres            "/docker-entrypoint.s"   2 weeks ago         Up 2
hours          0.0.0.0:65432->5432/tcp   pt-postgres
```

`docker ps`。 `-a`

```
$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                     NAMES
9cc69f11a0f7        docker/whalesay     "ls /"                   26 hours ago        Exited
(0) 26 hours ago                             berserk_wozniak
2bc9b1988080        redis               "docker-entrypoint.sh"   2 weeks ago         Up 2
hours                0.0.0.0:6379->6379/tcp    elephant-redis
817879be2230        postgres            "/docker-entrypoint.s"   2 weeks ago         Up 2
hours                0.0.0.0:65432->5432/tcp   pt-postgres
```

`-f`。

```
$ docker ps -a -f status=exited
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
PORTS                     NAMES
9cc69f11a0f7        docker/whalesay     "ls /"                   26 hours ago        Exited
(0) 26 hours ago
```

`-q`Container ID。 Unix`grep``awk`

```
$ docker ps -aq
9cc69f11a0f7
2bc9b1988080
817879be2230
```

`docker run --name mycontainer1`mood_famousnostalgic_stallman `docker ps -f name=mycontainer1`

Docker

UUID	`9cc69f11a0f76073e87f25cb6eaf0e079fbfbd1bc47c063bcd25ed3722a8cc4a`
UUID	`9cc69f11a0f7`
	`berserk_wozniak`

`docker ps`。

UUIDDocker。 `docker run --name <given name> <image>` 。 Docker。

UUID“”UUID

```
docker stop <container> [<container>...]
```

SIGTERMSIGKILL。 。

```
docker start <container> [<container>...]
```

;。 `-a` -- `--attach` 。

```
docker ps --format 'table {{.ID}}\t{{.Names}}\t{{.Status}}'
```

```
docker ps --filter name=myapp_1
```

IP

IP

```
docker inspect <container id> | grep IPAddress
```

docker inspect

`docker inspect --format '{{ .NetworkSettings.IPAddress }}' ${CID}`

docker

```
docker restart <container> [<container>...]
```

--time 10

```
docker restart <container> --time 10
```

docker rm

```
docker rm <container name or id>
```

```
docker rm $(docker ps -qa)
```

docker。。。

xargs

```
docker ps -aq -f status=exited | xargs -r docker rm
```

docker ps -aq -f status=exited“”ID。

“”。

-f

```
docker rm -f <container name or id>
```

```
docker rm -f $(docker ps -qa)
```

dead

```
docker rm $(docker ps --all -q -f status=dead)
```

exited

```
docker rm $(docker ps --all -q -f status=exited)
```

。

1.3Unixdf

```
$ docker system df
```

```
$ docker system prune
```

docker

docker exec -it <container id> /bin/bash

。。。/ bin / bash/ bin / sh。

```
docker exec <container id> tar -czvf /tmp/backup.tgz /data
docker cp <container id>:/tmp/backup.tgz .
```

tar。docker cp。

```
Usage:  docker logs [OPTIONS] CONTAINER

Fetch the logs of a container

  -f, --follow=false        Follow log output
  --help=false              Print usage
  --since=                  Show logs since timestamp
  -t, --timestamps=false    Show timestamps
  --tail=all                Number of lines to show from the end of the logs
```

```
$ docker ps
CONTAINER ID    IMAGE    COMMAND                   CREATED     STATUS       PORTS
ff9716dda6cb    nginx    "nginx -g 'daemon off"  8 days ago  Up 22 hours  443/tcp,
0.0.0.0:8080->80/tcp

$ docker logs ff9716dda6cb
xx.xx.xx.xx - - [15/Jul/2016:14:03:44 +0000] "GET /index.html HTTP/1.1" 200 511
"https://google.com" "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/50.0.2661.75 Safari/537.36"
xx.xx.xx.xx - - [15/Jul/2016:14:03:44 +0000] "GET /index.html HTTP/1.1" 200 511
"https://google.com" "Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/50.0.2661.75 Safari/537.36"
```

```
docker attach --sig-proxy=false <container>
```

bashbash。

`Ctl-P Ctl-Q`

bash

```
docker exec -it <container> bash
```

/

```
docker cp CONTAINER_NAME:PATH_IN_CONTAINER PATH_IN_HOST
```

```
docker cp PATH_IN_HOST CONTAINER_NAME:PATH_IN_CONTAINER
```

jess /

https://hub.docker.com/r/jess/transmission/builds/bsn7eqxrkzrhxazcuytbmzp/

//

/ home / $ USER / abc

`docker cp transmission_id_or_name:/transmission/download .`

`/home/$USER/abc/transmission/download`

`docker cp`

docker

Docker。

```
docker rm -v <container id or name>
```

`-v`“”。

```
docker volume rm $(docker volume ls -qf dangling=true)
```

`docker volume ls -qf dangling=true` filterdocker。

`xargs`

```
docker volume ls -f dangling=true -q | xargs --no-run-if-empty docker volume rm
```

Docker

Dockertarball。 。 `docker-compose.yml` `docker run` `docker-compose.yml`。 Docker。 `docker export` `docker import` 。

```
docker export -o redis.tar redis
```

`redis`。 tarball

```
docker import ./redis.tar redis-imported:3.0.7
```

`redis-imported:3.0.7`。

```
docker import -c="ENV DEBUG true" -m="enable debug mode" ./redis.tar redis-changed
```

`-c`Dockerfile`CMD` `ENTRYPOINT` `ENV` `EXPOSE` `ONBUILD` `USER` `VOLUME` `WORKDIR` 。

https://riptutorial.com/zh-CN/docker/topic/689/

31:

Examples

systemd

```
[Service]

# empty exec prevents error "docker.service has more than one ExecStart= setting, which is
only allowed for Type=oneshot services. Refusing."
ExecStart=
ExecStart=/usr/bin/dockerd -H fd:// --log-driver=syslog
```

dockersyslog。 root`/etc/systemd/system/docker.service.d` Ubuntu 16.04。

Docker/。

```
ln -sf /dev/stdout /var/log/nginx/access.log
ln -sf /dev/stderr /var/log/nginx/error.log
```

。

https://riptutorial.com/zh-CN/docker/topic/7378/

32:

- docker stats [OPTIONS] [CONTAINER ...]
- docker logs [OPTIONS] CONTAINER
- docker top [OPTIONS] CONTAINER [ps OPTIONS]

Examples

`docker exec`。 “”。

```
docker exec -it container_id bash
```

```
docker exec -it container_id /bin/sh
```

shell。

```
docker exec container_id ls -la
```

`-u flag`uid=1013 gid=1023 。

```
docker exec -it -u 1013:1023 container_id ls -la
```

uidgid。

`docker run...; docker exec -it $(docker ps -lq) bash`

`docker ps -lq`l in `-lq` id。 bashshzsh

。 `top`

```
docker stats
```

```
docker stats 7786807d8084 7786807d8085
```

Docker

```
CONTAINER      CPU %    MEM USAGE / LIMIT    MEM %     NET I/O                BLOCK I/O
7786807d8084   0.65%    1.33 GB / 3.95 GB    33.67%    142.2 MB / 57.79 MB    46.32 MB / 0 B
```

`docker stats`id

`docker stats $(docker ps --format '{{.Names}}')`

。 `ps`。

```
docker top 7786807d8084
```

ps

```
docker top 7786807d8084 faux
```

root

```
docker top 7786807d8084 -u root
```

shellpsminimalistic docker top。

“”。 。

attach。

```
docker attach <container>
```

<container>ID。

```
docker attach c8a9cf1a1fa8
```

```
docker attach graceful_hopper
```

sudodocker。

> Attachshell。
>
> 。 Ctrl-c。

Ctrl-pCtrl-q

shellexec 。 ID

```
docker exec -i -t c8a9cf1a1fa8 /bin/bash
```

```
docker exec -i -t graceful_hopper /bin/bash
```

exec/bin/bash shell。 -i-tTTY。

> *attach* Ctrl-c*exec'd*。

。 7786807d8084tail -f some-application.log。

```
docker logs --follow --tail 10 7786807d8084
```

pid 1。

`--timestamps`。

- `docker run ... ; docker logs $(docker ps -lq)docker run ... ; docker logs $(docker ps -lq)`
- ID

`docker ps -a`

`docker logs container-id`

`docker logs containername`

Docker

Docker。 “”“”。 root`ps`“”

```
sudo ps aux
```

Docker。

◦ straceltracegdb。

https://riptutorial.com/zh-CN/docker/topic/1333/

33:

- docker run [OPTIONS] IMAGE [COMMAND] [ARG ...]

Examples

```
docker run hello-world
```

Docker Hubhello-world。 。

```
docker run docker/whalesay cowsay 'Hello, StackExchange!'
```

Docker`docker/whalesay``docker/whalesay cowsay 'Hello, StackExchange!'`。 `Hello, StackExchange!`。

```
docker run docker/whalesay ls /
```

```
docker run --entrypoint=/bin/bash docker/whalesay -c ls /
```

Docker。 。 。 。 Docker `--rm--rm`

```
docker run --rm ubuntu cat /etc/hosts
```

“ubuntu”**/ etc / hosts**。 。

> `--rm-d --detach` <1.13.0。

`--rm`Docker。 `docker rm -v my-container` 。 。

`docker run -it --rm -v /etc -v logs:/var/log centos /bin/produce_some_logs` `/etc/var/log`。 **--volumes-from -** 。

`small_roentgen` `docker run``docker run``small_roentgen``modest_dubinsky` 。 。 `--name`

```
docker run --name my-ubuntu ubuntu:14.04
```

;Docker。

Docker。 Docker。

。

```
docker run -p "8080:8080" myApp
docker run -p "192.168.1.12:80:80" nginx
docker run -P myApp
```

`EXPOSE` Dockerfile`--expose``docker run` `-p``-P`。 `-p`。 `-P`Docker。

```
docker run --restart=always -d <container>
```

DockerDocker。 Docker`--restart`。 Docker`--restart=always`。 `docker stop <container>` 。

`--restart=[policy]` `--restart`。 。

	。 。
[]	`non-zero exit status` 。 Docker。
	。 `always` Docker。 。
	。

`-d`

```
docker run -d busybox top
```

`-d`。 `-d=true` 。

-rm-d。

Docker。 UnionFS。

`-v`

```
docker run -d -v "/data" awesome/app bootstrap.sh
```

`/data` 。

- `--rm`。

```
docker run -d -v "/home/foo/data:/data" awesome/app bootstrap.sh
```

- 。

`/home/foo/data/data` 。 “”Linux `mount --bind` 。 。

UNIX

```
docker run -d -v $(pwd)/data:/data awesome/app bootstrap.sh
```

docker。

```
docker run -d -v "my-volume:/data" awesome/app bootstrap.sh
```

。

```
$ docker run -e "ENV_VAR=foo" ubuntu /bin/bash
```

-e--env。

```
$ docker run --env-file ./env.list ubuntu /bin/bash
```

```
# This is a comment
TEST_HOST=10.10.0.127
```

--env-fileVARIABLE=VALUE--env。 #。

--env-file -e / --env。 -e--env--env--env-var。

docker run。 --hostname

```
docker run --hostname redbox -d ubuntu:14.04
```

-it

```
$ docker run -it ubuntu:14.04 bash
root@8ef2356d919a:/# echo hi
hi
root@8ef2356d919a:/#
```

-iSTDIN-tTTY。

/

```
docker run -it -m 300M --memory-swap -1 ubuntu:14.04 /bin/bash
```

。 300M700M。

```
docker run -it -m 300M --memory-swap 1G ubuntu:14.04 /bin/bash
```

shell

---

execbash shell。

jovial_morseTTY bash shell

```
docker exec -it jovial_morse bash
```

-u--user。 。

    -u, --user UsernameUID <name|uid>[:<group|gid>]

dockeruserjovial_morse

```
docker exec -it -u dockeruser jovial_morse bash
```

root

root-u **root**。 Root。

```
docker exec -it -u root jovial_morse bash
```

run。

```
docker run -it dockerimage bash
```

Dockerfile。

docker build .docker build .

```
$ docker build .
Uploading context 10240 bytes
Step 1 : FROM busybox
Pulling repository busybox
 ---> e9aa60c60128MB/2.284 MB (100%) endpoint: https://cdn-registry-1.docker.io/v1/
Step 2 : RUN ls -lh /
 ---> Running in 9c9e81692ae9
total 24
drwxr-xr-x    2 root     root         4.0K Mar 12  2013 bin
drwxr-xr-x    5 root     root         4.0K Oct 19 00:19 dev
drwxr-xr-x    2 root     root         4.0K Oct 19 00:19 etc
drwxr-xr-x    2 root     root         4.0K Nov 15 23:34 lib
lrwxrwxrwx    1 root     root            3 Mar 12  2013 lib64 -> lib
dr-xr-xr-x  116 root     root            0 Nov 15 23:34 proc
lrwxrwxrwx    1 root     root            3 Mar 12  2013 sbin -> bin
dr-xr-xr-x   13 root     root            0 Nov 15 23:34 sys
drwxr-xr-x    2 root     root         4.0K Mar 12  2013 tmp
drwxr-xr-x    2 root     root         4.0K Nov 15 23:34 usr
 ---> b35f4035db3f
Step 3 : CMD echo Hello world
 ---> Running in 02071fceb21b
 ---> f52f38b7823e
```

---> Running in 02071fceb21b---> Running in 02071fceb21b

```
docker run -it 02071fceb21b bash
```

stdin

-idocker rundocker exec。

mariadbdump.sql

```
docker exec -i mariadb bash -c 'mariadb "-p$MARIADB_PASSWORD" ' < dump.sql
```

```
docker exec -i container command < file.stdin
```

```
docker exec -i container command <<EOF
inline-document-from-host-shell-HEREDOC-syntax
EOF
```

pty docker run -it ... Control P - Control Q。

```
docker run --name="test-app" --entrypoint="/bin/bash" example-app
```

test-appexample-appENTRYPOINT。 CMD

```
docker run --name="test-app" --entrypoint="/bin/bash" example-app /app/test.sh
```

ENTRYPOINTCMD。 /bin/bash /app/test.sh 。

```
docker run --add-host="app-backend:10.15.1.24" awesome-app
```

/etc/hosts--add-host <name>:<address> 。 app-backend10.15.1.24 。 。

。 -t-d。

```
docker run -t -d debian bash
```

```
docker stop mynginx
```

ID。

SIGTERMSIGKILL。

kill-sSIGKILL。

```
docker kill mynginx
```

```
docker kill -s SIGINT mynginx
```

。 docker ps -a。

`exec`Docker。 `docker ps`ID。

```
docker exec 294fbc4c24b3 echo "Hello World"
```

`-it`shell。

```
docker exec -it 294fbc4c24b3 bash
```

LinuxGUI

Docker GUI。

X11。 *DISPLAY*

```
docker run -v /tmp/.X11-unix:/tmp/.X11-unix -e DISPLAY=unix$DISPLAY <image-name>
```

X

```
cannot connect to X server unix:0
```

```
xhost +local:root
```

```
xhost -local:root
```

DockerfileX

```
FROM <iamge-name>
MAINTAINER <you>

# Arguments picked from the command line!
ARG user
ARG uid
ARG gid

#Add new user with our credentials
ENV USERNAME ${user}
RUN useradd -m $USERNAME && \
        echo "$USERNAME:$USERNAME" | chpasswd && \
        usermod --shell /bin/bash $USERNAME && \
        usermod  --uid ${uid} $USERNAME && \
        groupmod --gid ${gid} $USERNAME

USER ${user}

WORKDIR /home/${user}
```

`docker build` Dockerfile*ARG*

```
docker build --build-arg user=$USER --build-arg uid=$(id -u) --build-arg gid=$(id -g) -t <new-image-with-X11-enabled-name> -f <Dockerfile-for-X11> .
```

xauth

```
xauth nlist $DISPLAY | sed -e 's/^..../ffff/' | xauth -f /tmp/.docker.xauth nmerge -
```

/

```
docker run -e DISPLAY=unix$DISPLAY -v /tmp/.X11-unix:/tmp/.X11-unix -v
/tmp/.docker.xauth:/tmp/.docker.xauth:rw -e XAUTHORITY=/tmp/.docker.xauth
```

https://riptutorial.com/zh-CN/docker/topic/679/

34:

Examples

visualizer。 80809090.。

```
docker service create \
        --name=visualizer \
        --label com.my.custom.label=visualizer \
        --publish=9090:8080 \
        --mount type=bind,source=/var/run/docker.sock,target=/var/run/docker.sock \
        manomarks/visualizer:latest
```

hello world web80。

```
docker service create \
    --publish 80:80 \
    tutum/hello-world
```

“visualizer”

```
docker service rm visualizer
```

4

```
docker service scale visualizer=4
```

Docker Swarm。

```
docker service scale visualizer=0
```

https://riptutorial.com/zh-CN/docker/topic/8802/

35: Node.js

Examples

ContainerBasic Node.js

DockerNode.js。 DockerNode.jsDocker`latest` `node`。 `docker pull node`。 docker`pull` `s` `node`。

1. 。 `package.json`。 `package.json`

   ```
   {
     "name": "docker_web_app",
     "version": "1.0.0",
     "description": "Node.js on Docker",
     "author": "First Last <first.last@example.com>",
     "main": "server.js",
     "scripts": {
       "start": "node server.js"
     },
     "dependencies": {
       "express": "^4.13.3"
     }
   }
   ```

2. Node.jsWeb`server`。 `Express.js` `4.13.3`。 `server.js`

   ```
   var express = require('express');
   var PORT = 8080;
   var app = express();
   app.get('/', function (req, res) {
     res.send('Hello world\n');
   });

   app.listen(PORT);
   console.log('Running on http://localhost:' + PORT);
   ```

3. Docker`Dockerfile` 。 `Dockerfile`。

`Dockerfile`。 Windows。 Linux`touch Dockerfile` 。 Dockerfile

```
FROM node:latest
RUN mkdir -p /usr/src/my_first_app
WORKDIR /usr/src/my_first_app
COPY package.json /usr/src/my_first_app/
RUN npm install
COPY . /usr/src/my_first_app
EXPOSE 8080
```

- `FROM node:latest`Docker。 Docker Hub`latest`Docker`node` 。
- 。

```
RUN mkdir -p /usr/src/my_first_app
WORKDIR /usr/src/my_first_app
```

- package.jsonapp/usr/src/my_first_app。

```
COPY package.json /usr/src/my_first_app/
RUN npm install
```

- COPY . /usr/src/my_first_app。

- EXPOSE80808080 。

- npm startnode server.js 。 CMD。

```
CMD [ "npm", "start" ]
```

4. .dockerignoreDockerfile node_modulesNode.js。 .dockerignore

```
node_modules
npm-debug.log
```

5.

DockerfileDocker。 -tdocker images

```
 $ docker build -t <your username>/node-web-app .
```

Docker。

```
$ docker images

REPOSITORY                      TAG        ID              CREATED
node                            latest     539c0211cd76    10 minutes ago
<your username>/node-web-app    latest     d64d3505b0d2    1 minute ago
```

6.

nodeDockerfileDockerfile 。 <your username>/node-web-app。 -ddocker run。 -p。

```
$ docker run -p 49160:8080 -d <your username>/node-web-app
```

7. docker ps。 。

```
 CONTAINER ID         IMAGE                            COMMAND              CREATED
STATUS               PORTS                         NAMES
 7b701693b294      <your username>/node-web-app   "npm start"         20 minutes ago
Up 48 seconds       0.0.0.0:49160->8080/tcp   loving_goldstine
```

`docker logs <CONTAINER ID>`。 `docker logs 7b701693b294` 。

```
Running on http://localhost:8080
```

8. `docker ps0.0.0.0:49160->8080/tcp` 。 Docker`8080`49160。 `localhost:49160` 。

`curl`

```
$ curl -i localhost:49160

HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 12
Date: Sun, 08 Jan 2017 14:00:12 GMT
Connection: keep-alive

Hello world
```

Node.js https://riptutorial.com/zh-CN/docker/topic/8754/node-js

36:

tty:true	docker-compose.yml tty: truesh。

hostbridgedocker。 。 docker。

- Swarm docker network create --driver overlay
- docker / swarm

Examples

Docker

docker。

```
docker network create sample
docker run --net sample --name keys consul agent -server -client=0.0.0.0 -bootstrap
```

Dockerfile8500 8600 。

```
docker run --net sample -ti alpine sh
/ # wget -qO- keys:8500/v1/catalog/nodes
```

consulkeys keys。 Dockerdns --name。

v2。 。

example/docker-compose.yml

```
version: '2'
services:
  keys:
    image: consul
    command: agent -server -client=0.0.0.0 -bootstrap
  test:
    image: alpine
    tty: true
    command: sh
```

docker-compose up -dexample_default 。 docker network ls

```
 > docker network ls
NETWORK ID          NAME                    DRIVER              SCOPE
719eafa8690b        example_default         bridge              local
```

alpine

```
 > docker exec -ti example_test_1 sh
/ # nslookup keys
...
/ # wget -qO- keys:8500/v1/kv/?recurse
...
```

`networks:`docker network。

`--linklink:` sections `--link` -compose。

```
docker network create sample
docker run -d --net sample --name redis redis
```

redis。

```
> docker run --net sample --link redis:cache -ti python:alpine sh -c "pip install redis &&
python"
>>> import redis
>>> r = redis.StrictRedis(host='cache')
>>> r.set('key', 'value')
True
```

---

docker `1.10.0` - docker。 `legacy`。

https://riptutorial.com/zh-CN/docker/topic/6528/

37:

docker。 `--net--net docker network connect--net docker network connect` 。

Examples

LAN

```
docker network create -o "com.docker.network.bridge.enable_ip_masquerade"="false" lan-
restricted
```

-
    -
    -
-
    - docker`10.0.1.10:22`

```
docker network create -o "com.docker.network.bridge.enable_icc"="false" icc-restricted
```

-
    - `icc-restricted`。
-
    - docker
    -
    -

docker

```
iptables -I INPUT -i docker0 -m addrtype --dst-type LOCAL -j DROP
```

-
-
    - docker
    -
    -
    -
    - `docker0docker0`

docker

```
docker network create --subnet=192.168.0.0/24 --gateway=192.168.0.1 --ip-range=192.168.0.0/25
local-host-restricted
iptables -I INPUT -s 192.168.0.0/24 -m addrtype --dst-type LOCAL -j DROP
```

`local-host-restricted`

-
-
    - docker
    -
    -
    -
    - docker

`br-15bbe9bb5bf5` 。

https://riptutorial.com/zh-CN/docker/topic/6331/

S. No		Contributors
1	Docker	abaracedo, Aminadav, Braiam, Carlos Rafael Ramirez, Community, ganesshkumar, HankCa, Josha Inglis, L0j1k, mohan08p, Nathaniel Ford, schumacherj, Siddharth Srinivasan, SztupY, Vishrant
2	Docker Engine API	Ashish Bista, atv, BMitch, L0j1k, Radoslav Stoyanov, SztupY
3	Docker --net。	mohan08p
4	Dockerfiles	BMitch, foraidt, k0pernikus, kubanczyk, L0j1k, ob1, Ohmen, rosysnake, satsumas, Stephen Leppik, Thiago Almeida, Wassim Dhif, yadutaf
5	Dockerfile	akhyar, Philip
6	DockerDocker	Ohmen
7	Docker	Nathaniel Ford, user2915097
8	Docker	Amit Poonia, Rob Bednark, serieznyi
9	Docker	James Hewitt, L0j1k, NRKirby, Nuno Curado, Scott Coates, t3h2mas
10	Docker	Amine24h, kubanczyk, Nik Rahmel, user2915097, yadutaf
11	docker	user2915097
12	Docker	Ashish Bista, L0j1k
13	Docker	Kostiantyn Rybnikov
14	Docker	HankCa, L0j1k, Nathaniel Ford
15	Docker	abronan, Christian, Farhad Farahi, Jilles van Gurp, kstromeiraos , kubanczyk, ob1, Philip, Vanuan
16		h3nrik, Ohmen, Xavier Nicollet
17	DockerIptables	Adrien Ferrand
18	API v2Docker/	bastien enjalbert, kubanczyk
19		Carlos Rafael Ramirez, Vanuan
20	1.12	Jilles van Gurp

21	Docker ImageMongo Chef	Innocent Anigbo
22	docker build	user2915097
23		user2915097
24		user2915097
25		cjsimon, ETL, Ken Cochrane, L0j1k, Nathan Arthur, Nathaniel Ford, Nour Chawich, SztupY, user2915097, Wolfgang
26		GameScripting, L0j1k, melihovv
27		AlcaDotS, devopskata, Felipe Plets, h3nrik, Jilles van Gurp, L0j1k, Milind Chawre, Nik Rahmel, Stephen Leppik, user2915097, yadutaf
28		Bastian, Fuzzyma
29		akhyar, Björn Enochsson, dsw88, L0j1k, Nathan Arthur, Nathaniel Ford, Szymon Biliński, user2915097, Wolfgang, zygimantus
30		akhyar, atv, Binary Nerd, BrunoLM, Carlos Rafael Ramirez, Emil Burzo, Felipe Plets, ganesshkumar, L0j1k, Matt, Nathaniel Ford, Rafal Wiliński, Sachin Malhotra, serieznyi, sk8terboi87 ツ, tommyyards, user2915097, Victor Oliveira Antonino, Wolfgang, Xavier Nicollet, zygimantus
31		Jilles van Gurp, Vanuan
32		allprog, Binary Nerd, foraidt, L0j1k, Nathaniel Ford, user2915097, yadutaf
33		abaracedo, Adri C.S., AlcaDotS, atv, Binary Nerd, BMitch, Camilo Silva, Carlos Rafael Ramirez, cizixs, cjsimon, Claudiu, ElMesa, Emil Burzo, enderland, Felipe Plets, ganesshkumar, Gergely Fehérvári, ISanych, L0j1k, Nathan Arthur, Patrick Auld, RoyB, ssice, SztupY, Thomasleveil, tommyyards, VanagaS, Wolfgang, zinking
34		Mateusz Mrozewski, Philip
35	Node.js	Siddharth Srinivasan
36		Jett Jones
37		xeor