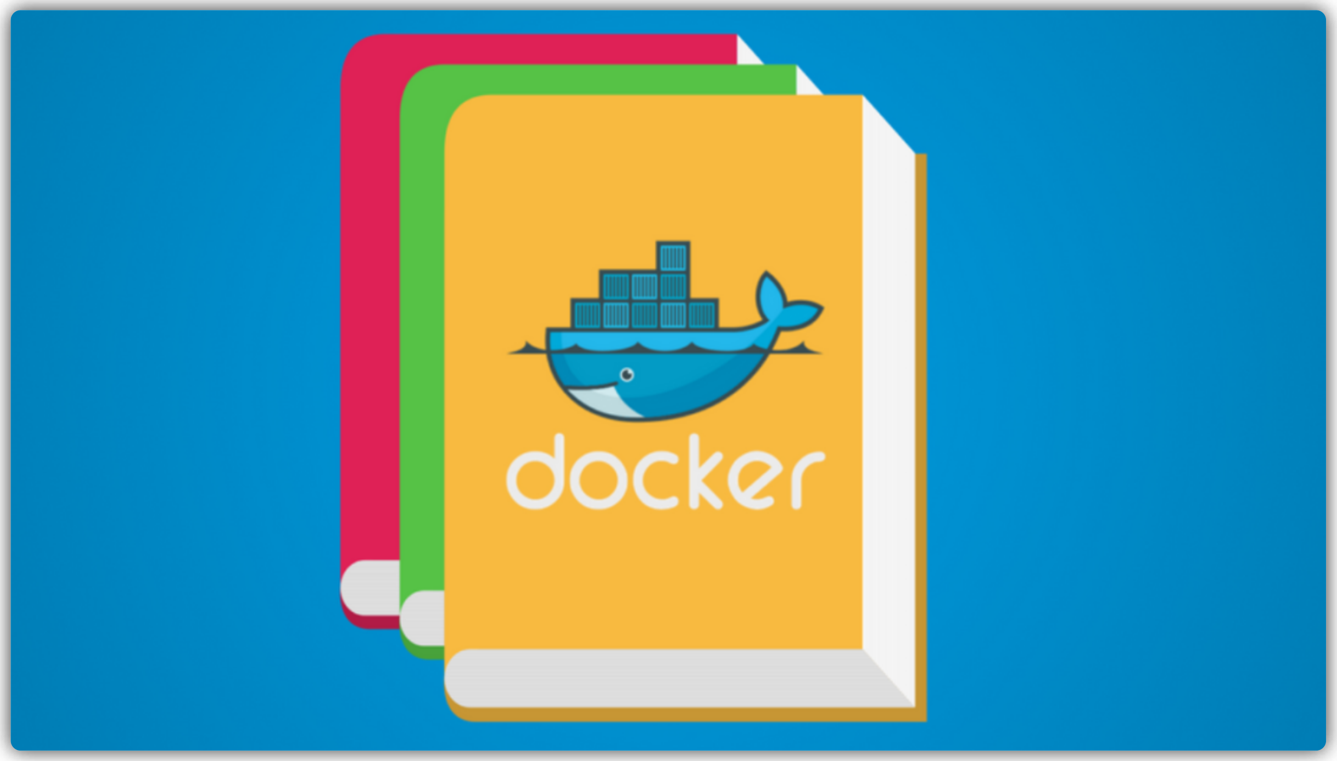
详解dockerfile之五个实战项目

Original 奋斗的小白 马哥Linux运维 Today

收录于话题

#Docker 1个



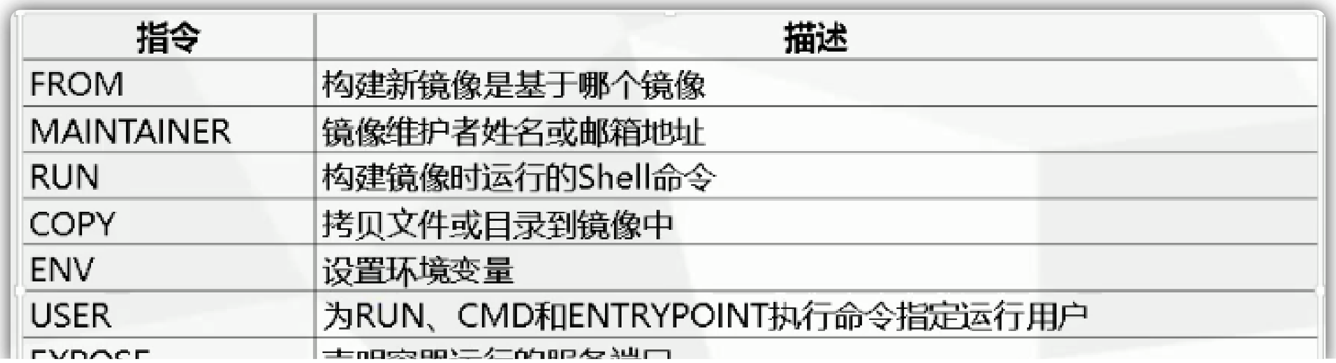
docker 的使用过程：它分为**镜像构建**与**容器启动**

dockerfile:镜像的构建过程。即创建一个镜像，它包含安装运行所需的环境、程序代码等。这个创建过程就是使用 dockerfile 来完成的。然后执行docker build . 就能制作镜像。从上往下依次执行dockerfile里面的命令dockerfile的作用是从无到有的构建镜像。它包含安装运行所需的环境、程序代码等。这个创建过程就是使用 dockerfile 来完成的。Dockerfile - 为 docker build 命令准备的，用于建立一个独立的 image

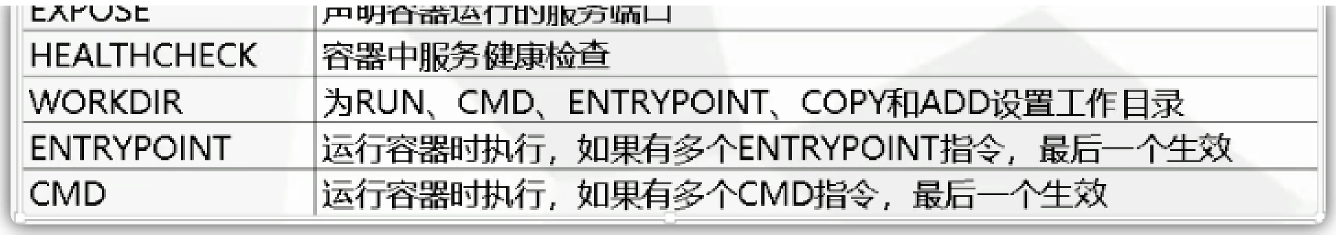
docker-compse.yml 记录一个项目(project, 一般是多个镜像)的构建过程。docker-compose是编排容器的。可以同时管理多个 container ，包括他们之间的关系、用官方 image 还是自己 build 、各种网络端口定义、储存空间定义等 他们之间的关系可以分为

```
1.dockerfile: 构建镜像；

2.docker run: 启动容器；

3.docker-compose: 启动服务；
```

Dockerfile常用指令



指令	描述
FROM	构建新镜像是基于哪个镜像
MAINTAINER	镜像维护者姓名或邮箱地址
RUN	构建镜像时运行的Shell命令
COPY	拷贝文件或目录到镜像中
ENV	设置环境变量
USER	为RUN、CMD和ENTRYPOINT执行命令指定运行用户
EXPOSE	声明容器运行的服务端口



EXPOSE	声明容器运行的服务端口
HEALTHCHECK	容器中服务健康检查
WORKDIR	为RUN、CMD、ENTRYPOINT、COPY和ADD设置工作目录
ENTRYPOINT	运行容器时执行，如果有多个ENTRYPOINT指令，最后一个生效
CMD	运行容器时执行，如果有多个CMD指令，最后一个生效

```
docker  build   .       #当前目录执行`
`docker  build -t  shykes/myapp  .   # -t 指定dockerfe的名字`
`docker  build -t  shykes/myapp  -f /path/Dockerfile  /path   #  -f 指定路径
```

构建容器前先开启内核路由转发,否则创建的容器无法连接网络。

```
echo -e "net.ipv4.ip_forward = 1\nnet.ipv4.conf.default.rp_filter = 0 \nnet.ipv4.
sysctl -p
```

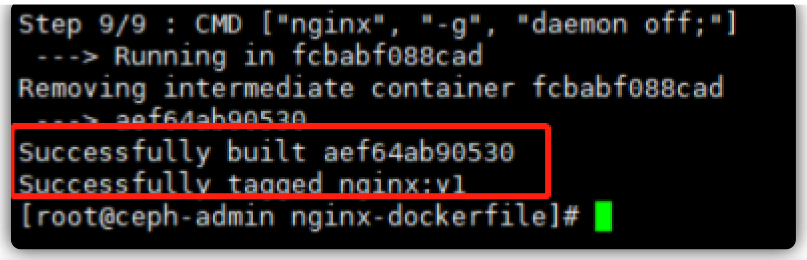
实例一 构建一个nginx镜像

需求:以centos为基础镜像,构建一个nginx源码编译安装的镜像，同时安装相应的管理工具

```
mkdir   /root/nginx-dockerfile   && cd   /root/nginx-dockerfile
cat Dockerfile
FROM centos:7
MAINTAINER zhangfan
```

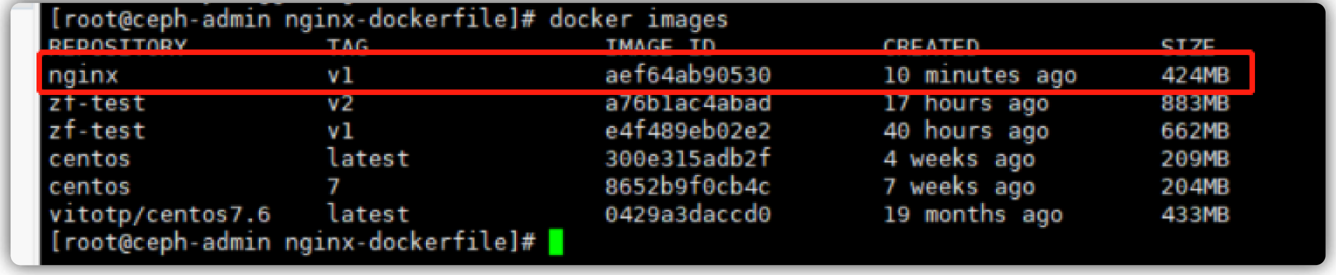
```
COPY CentOS-Base.repo  /etc/yum.repos.d/  #拷贝本地源到镜像中
#安装基础管理命令
RUN yum install -y gcc gcc-c++ make \
    openssl-devel pcre-devel gd-devel \
    iproute net-tools telnet wget curl && \
    yum clean all && \
    rm -rf /var/cache/yum/*
#下载源码nginx包并编译安装
RUN wget http://nginx.org/download/nginx-1.15.5.tar.gz && \
    tar zxf nginx-1.15.5.tar.gz && \
    cd nginx-1.15.5 &&\
    ./configure --prefix=/usr/local/nginx \
    --with-http_ssl_module \
    --with-http_stub_status_module && \
    make -j 4 && make install && \
    rm -rf /usr/local/nginx/html/* && \
    echo "ok" >> /usr/local/nginx/html/status.html && \
    cd / && rm -rf nginx-1.12.2* && \
    ln -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime

ENV PATH $PATH:/usr/local/nginx/sbin   #声明环境变量
#COPY nginx.conf /usr/local/nginx/conf/nginx.conf    #拷贝项目nginx配置
WORKDIR /usr/local/nginx   设置工作目录
EXPOSE 80      指定端口
CMD ["nginx", "-g", "daemon off;"]
```

```
docker  build  -t  nginx:v1        #当前目录执行，执行完无报错，说明构建成功
```



```
Step 9/9 : CMD ["nginx", "-g", "daemon off;"]
 ---> Running in fcbabf088cad
Removing intermediate container fcbabf088cad
 ---> aef64ab90530
Successfully built aef64ab90530
Successfully tagged nginx:v1
[root@ceph-admin nginx-dockerfile]#
```

```
docker images
```

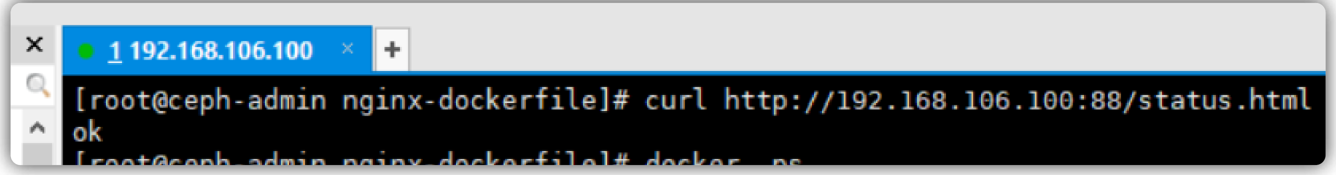


```
[root@ceph-admin nginx-dockerfile]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
nginx               v1                  aef64ab90530        10 minutes ago      424MB
zf-test             v2                  a76b1ac4abad        17 hours ago        883MB
zf-test             v1                  e4f489eb02e2        40 hours ago        662MB
centos              latest              300e315adb2f        4 weeks ago         209MB
centos              7                   8652b9f0cb4c        7 weeks ago         204MB
vitotp/centos7.6    latest              0429a3daccd0        19 months ago       433MB
[root@ceph-admin nginx-dockerfile]#
```

说明已经构建成功

验证

```
docker run -d   --name  nginx01  -p  88:80 nginx:v1  #创建容器
curl http://192.168.106.100:88/status.html
```

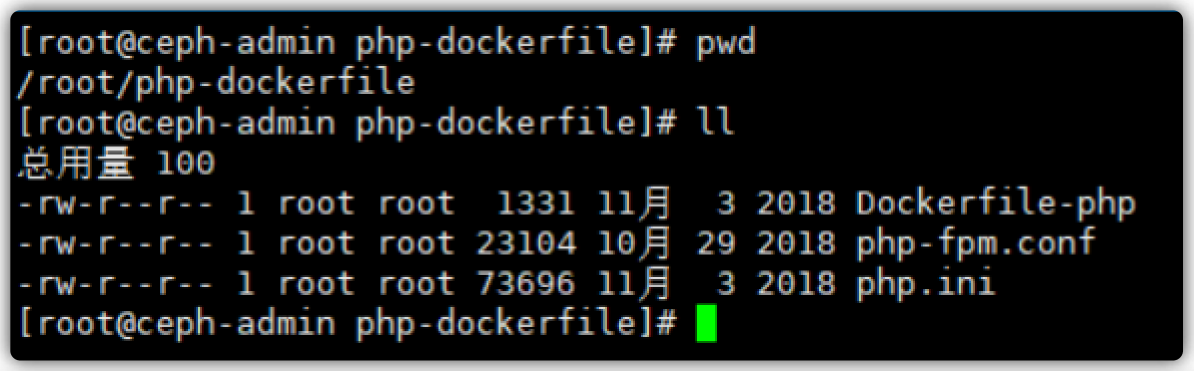


```
[root@ceph-admin nginx-dockerfile]# curl http://192.168.106.100:88/status.html
ok
[root@ceph-admin nginx-dockerfile]# docker  ps
```

说明已经构建成功

实例二 构建php基础镜像

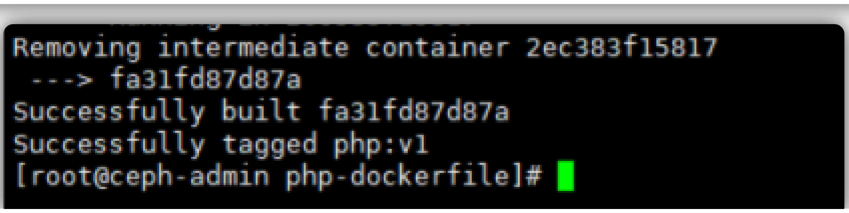
创建php-dockerfile的目录



```
[root@ceph-admin php-dockerfile]# pwd
/root/php-dockerfile
[root@ceph-admin php-dockerfile]# ll
总用量 100
-rw-r--r-- 1 root root  1331 11月  3 2018 Dockerfile-php
-rw-r--r-- 1 root root 23104 10月 29 2018 php-fpm.conf
-rw-r--r-- 1 root root 73696 11月  3 2018 php.ini
[root@ceph-admin php-dockerfile]# 
```

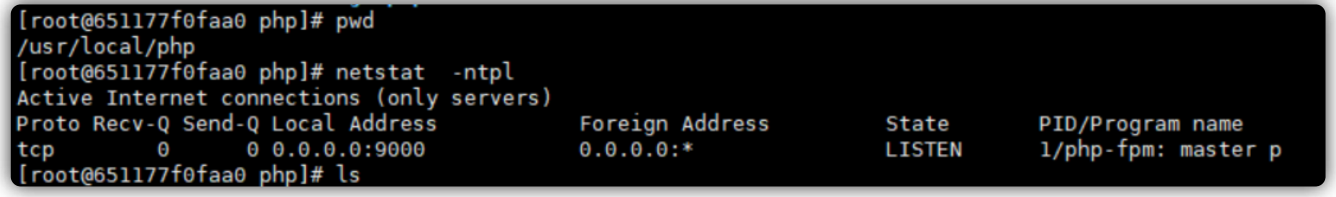
```
cat Dockerfile

FROM centos:7
MAINTAINER zhangfan
#安装php基础依赖包和基本工具
RUN yum install epel-release -y && \
    yum install -y gcc gcc-c++ make gd-devel libxml2-devel \
    libcurl-devel libjpeg-devel libpng-devel openssl-devel \
    libmcrypt-devel libxslt-devel libtidy-devel autoconf \
    iproute net-tools telnet wget curl && \
    yum clean all && \
    rm -rf /var/cache/yum/*
#编译安装php模块
RUN wget http://docs.php.net/distributions/php-5.6.36.tar.gz && \
    tar zxf php-5.6.36.tar.gz && \
    cd php-5.6.36 && \
    ./configure --prefix=/usr/local/php \
    --with-config-file-path=/usr/local/php/etc \
    --enable-fpm --enable-opcache \
    --with-mysql --with-mysqli --with-pdo-mysql \
    --with-openssl --with-zlib --with-curl --with-gd \
    --with-jpeg-dir --with-png-dir --with-freetype-dir \
    --enable-mbstring --with-mcrypt --enable-hash && \
    make -j 4 && make install && \
    cp php.ini-production /usr/local/php/etc/php.ini && \
    cp sapi/fpm/php-fpm.conf /usr/local/php/etc/php-fpm.conf && \
    sed -i "90a \daemonize = no" /usr/local/php/etc/php-fpm.conf && \
    mkdir /usr/local/php/log && \
    cd / && rm -rf php* && \
    ln -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime

ENV PATH $PATH:/usr/local/php/sbin:/usr/local/php/bin  #声明环境变量
COPY php.ini /usr/local/php/etc/    #替换修改过的配置文件
COPY php-fpm.conf /usr/local/php/etc/  #替换修改过的配置文件
WORKDIR /usr/local/php    #声明工作路径
EXPOSE 9000      #指定端口
CMD ["php-fpm"]  #指定启动程序
```

```
docker  build -t php:v1 . #当前目录构建
```



```
Removing intermediate container 2ec383f15817
 ---> fa31fd87d87a
Successfully built fa31fd87d87a
Successfully tagged php:v1
[root@ceph-admin php-dockerfile]#
```

```
docker run -d  --name  php01  php:v1
docker exec  -it  php01 bash
 bin/php  -v     #查看版本
```

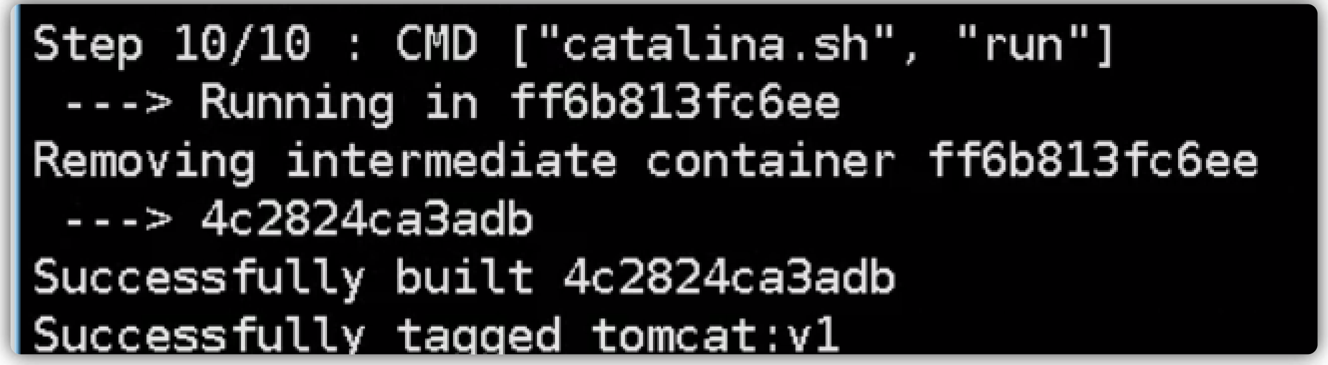


```
[root@651177f0faa0 php]# pwd
/usr/local/php
[root@651177f0faa0 php]# netstat  -ntpl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:9000            0.0.0.0:*               LISTEN      1/php-fpm: master p
[root@651177f0faa0 php]# ls
```

说明构建成功没问题

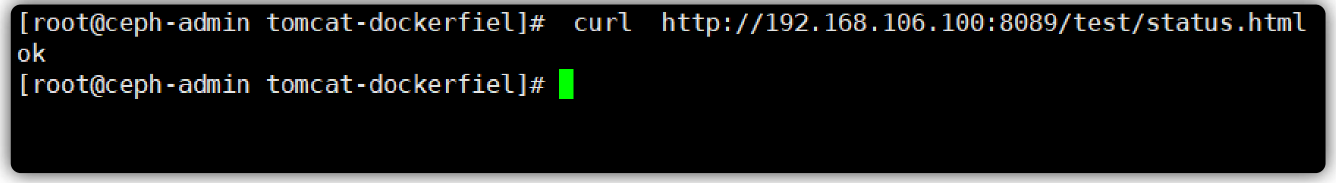
实例三 构建tomcat镜像

```
FROM centos:7                         #指定基础镜像
MAINTAINER zhangfan                   #指定作者
ENV VERSION=8.5.61                    #定义版本
RUN yum install java-1.8.0-openjdk wget curl unzip iproute net-tools -y && \
    yum clean all && \
    rm -rf /var/cache/yum/*
#https://mirrors.tuna.tsinghua.edu.cn/apache/tomcat/tomcat-8/v8.5.61/bin/apache-t
#下载安装二进制包
RUN wget https://mirrors.tuna.tsinghua.edu.cn/apache/tomcat/tomcat-8/v${VERSION}/
    tar zxf apache-tomcat-${VERSION}.tar.gz && \
    mv apache-tomcat-${VERSION} /usr/local/tomcat && \
    rm -rf apache-tomcat-${VERSION}.tar.gz /usr/local/tomcat/webapps/* && \
    mkdir /usr/local/tomcat/webapps/test && \
    echo "ok" > /usr/local/tomcat/webapps/test/status.html && \
    sed -i '1a JAVA_OPTS="-Djava.security.egd=file:/dev/./urandom"' /usr/local/to
    ln -sf /usr/share/zoneinfo/Asia/Shanghai /etc/localtime
#设定环境变量
ENV PATH $PATH:/usr/local/tomcat/bin
#设置工作目录
WORKDIR /usr/local/tomcat
#指定端口
EXPOSE 8080
CMD ["catalina.sh", "run"]  #配置前台启动
```



```
Step 10/10 : CMD ["catalina.sh", "run"]
 ---> Running in ff6b813fc6ee
Removing intermediate container ff6b813fc6ee
 ---> 4c2824ca3adb
Successfully built 4c2824ca3adb
Successfully tagged tomcat:v1
```

```
docker run -d   --name  tomcat01  -p 8089:8080  tomcat:v1
curl  http://192.168.106.100:8089/test/status.html  #出现如图所示的测试页面说明构建成功
```

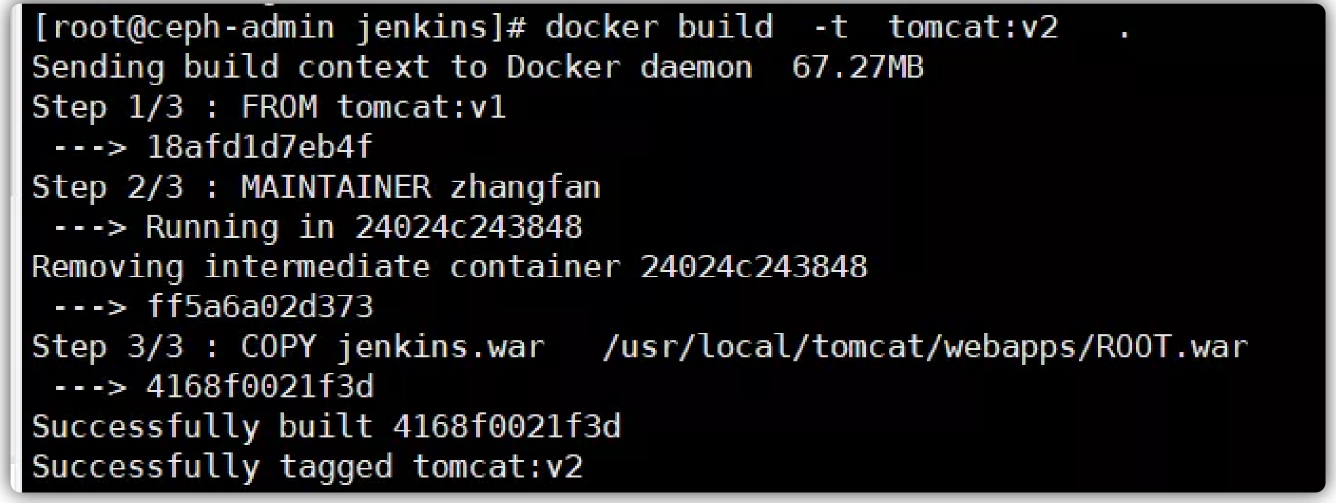


```
[root@ceph-admin tomcat-dockerfiel]#  curl  http://192.168.106.100:8089/test/status.html
ok
[root@ceph-admin tomcat-dockerfiel]# 
```

实例四 构建jenkins项目

```
mkdir  /root/jenkins-dockerfile  &&cd  /root/jenkins-dockerfile
wget https://get.jenkins.io/war-stable/2.263.1/jenkins.war
cat  Dockerfile
FROM tomcat:v1                          #指定刚才构建的tomcat为基础镜像
MAINTAINER zhangfan              #指定作者
COPY jenkins.war   /usr/local/tomcat/webapps/ROOT.war   #将下载的jenkins.war  拷贝到
```

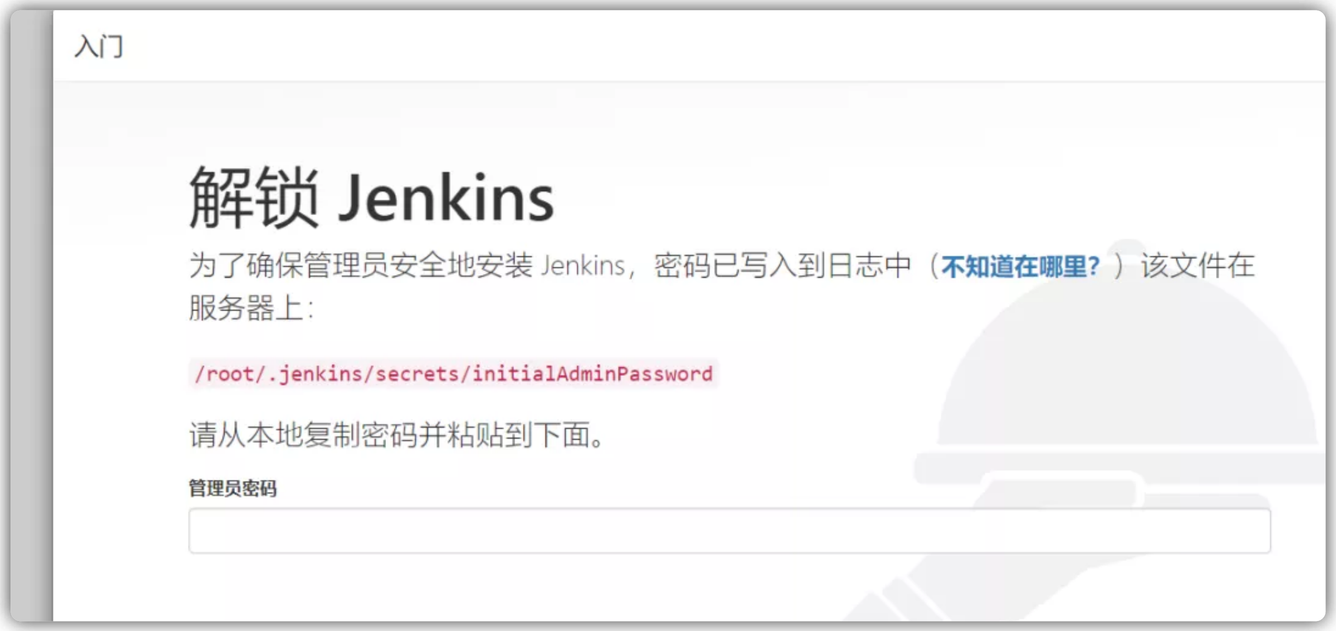
```
docker build  -t  tomcat:v2   .
```



```
[root@ceph-admin jenkins]# docker build  -t  tomcat:v2   .
Sending build context to Docker daemon  67.27MB
Step 1/3 : FROM tomcat:v1
 ---> 18afd1d7eb4f
Step 2/3 : MAINTAINER zhangfan
 ---> Running in 24024c243848
Removing intermediate container 24024c243848
 ---> ff5a6a02d373
Step 3/3 : COPY jenkins.war   /usr/local/tomcat/webapps/ROOT.war
 ---> 4168f0021f3d
Successfully built 4168f0021f3d
Successfully tagged tomcat:v2
```

```
docker  run  -d  --name tomcat02  -p 8888:8080  tomcat:v2
```

http://192.168.106.100:8888/login?from=%2F　访问该地址。可以看到jenkins的初始化页面，说明项目构建成功



解锁 Jenkins

为了确保管理员安全地安装 Jenkins，密码已写入到日志中（不知道在哪里?）该文件在服务器上:

/root/.jenkins/secrets/initialAdminPassword

请从本地复制密码并粘贴到下面。

管理员密码

实例五　快速构建LNMP网站平台

1. 自定义网络

```
docker  network  create lnmp
```

2. 创建Mysql容器

```
docker  run  -d  \
--name  lnmp_mysql  \
--net lnmp  \
--mount   src=myql-vol,dst=/var/lib/mysql  \      #指定数据卷
-e MYSQL_ROOT_PASSWORD=123456 \   #指定数据库密码
-e MYSQL_DATABASE=wordpress \      #创建数据库
mysql:5.7 --character-set-server=utf8     #设置字符集
```

3.创建PHP容器

```
docker run -d --name lnmp_php  --net  lnmp  --mount  src=wwwroot,dst=/wwwroot  ph
```

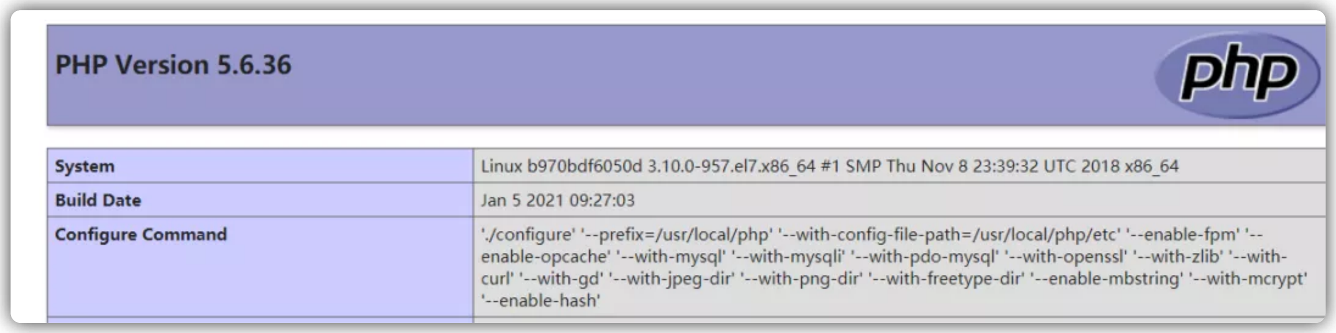
4创建nginx容器

```
cat   nginx.conf

  user                 nobody;
worker_processes     4;
worker_rlimit_nofile 65535;

error_log  logs/error.log  notice;

pid        /var/run/nginx.pid;

events {
    use epoll;
    worker_connections  4096;
```

```
}

http {

    include       mime.types;
    default_type  application/octet-stream;

    log_format  main '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    access_log off;
    keepalive_timeout  65;

    client_max_body_size         64m;
    server {
        listen 80;
        server_name www.ctnrs.com;
        index index.php index.html;

        access_log logs/www.ctnrs.com_access.log;
        error_log logs/www.ctnrs.com_error.log;

        # location ~ .*\.(js|css|html|png|gif|jpg|jpeg)$ {
        location / {
            root /wwwroot;
        }

        location ~* \.php$ {
            root /wwwroot;
            fastcgi_pass lnmp_php:9000;
            fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
            include fastcgi_params;
        }
    }
}
```

```
docker run -d --name  lnmp_nginx  --net lnmp -p 8000:80   --mount  type=bind,src=
cd   /var/lib/docker/volumes/wwwroot/_data
cat  test.php
<?php phpinfo();?>
```

http://192.168.106.100:8000/test.php



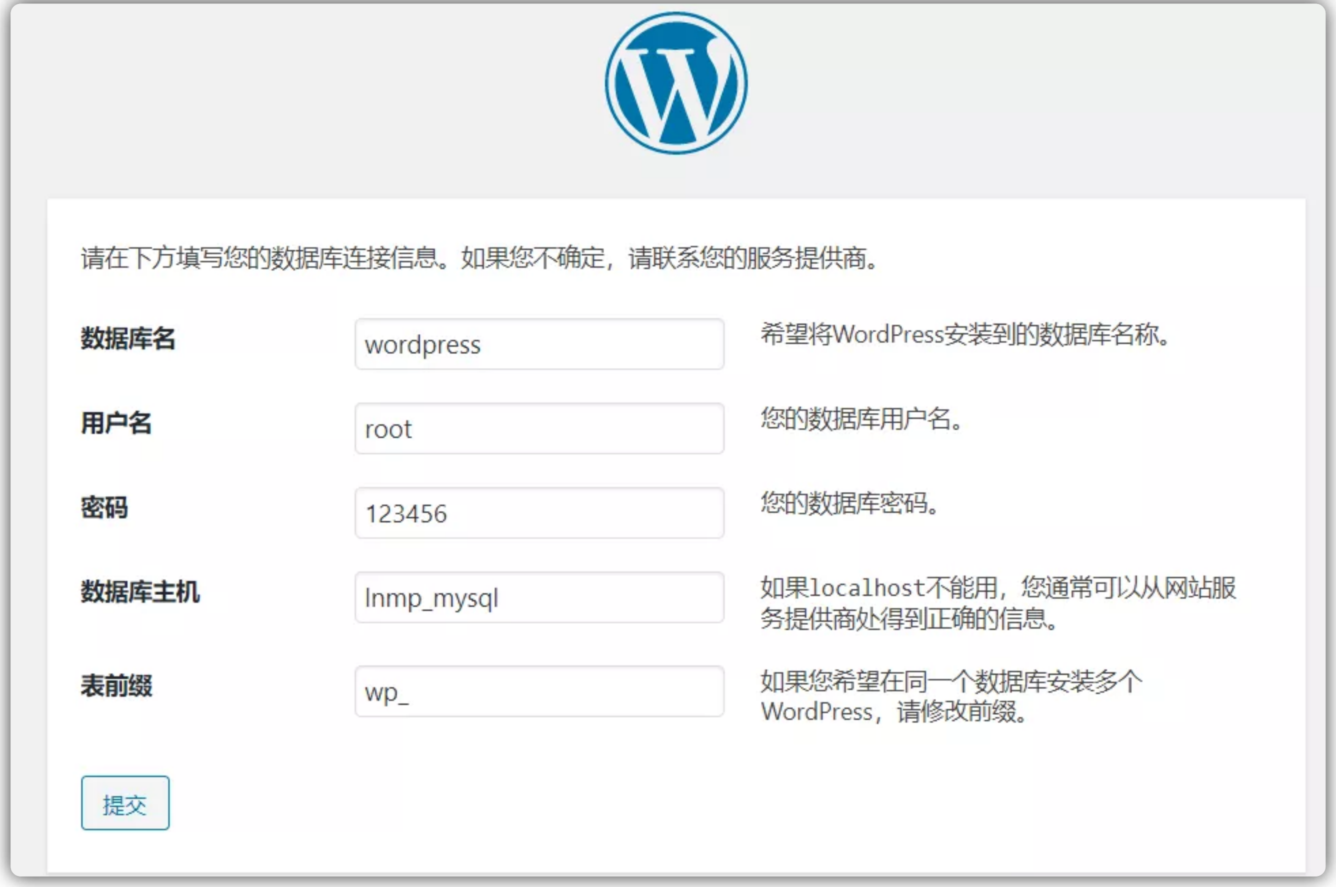
PHP Version 5.6.36

System	Linux b970bdf6050d 3.10.0-957.el7.x86_64 #1 SMP Thu Nov 8 23:39:32 UTC 2018 x86_64
Build Date	Jan 5 2021 09:27:03
Configure Command	'./configure' '--prefix=/usr/local/php' '--with-config-file-path=/usr/local/php/etc' '--enable-fpm' '--enable-opcache' '--with-mysql' '--with-mysqli' '--with-pdo-mysql' '--with-openssl' '--with-zlib' '--with-curl' '--with-gd' '--with-jpeg-dir' '--with-png-dir' '--with-freetype-dir' '--enable-mbstring' '--with-mcrypt' '--enable-hash'

5. 以wordpress博客为例

```
cd   /var/lib/docker/volumes/wwwroot/_data
wget https://cn.wordpress.org/latest-zh_CN.tar.gz
```

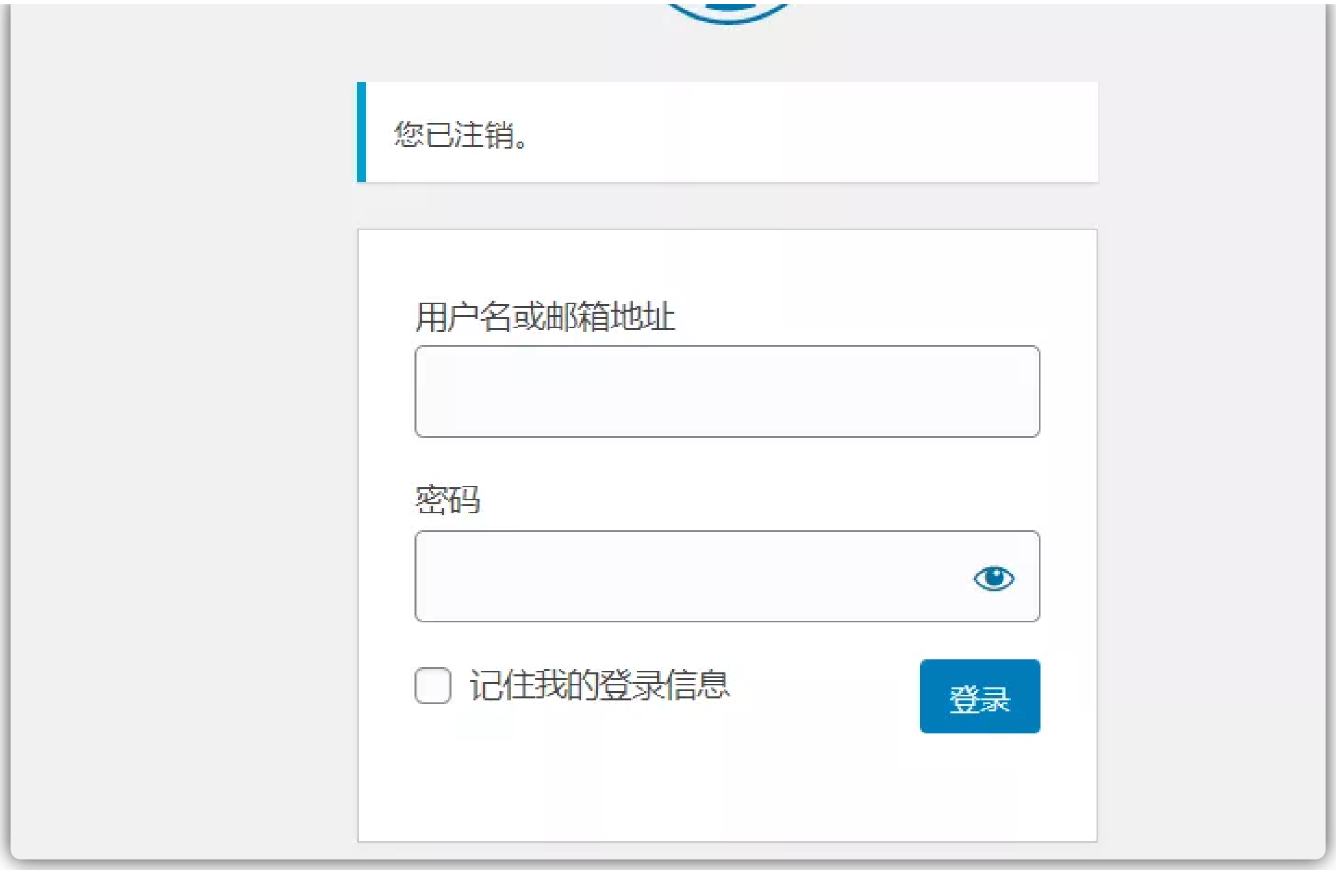
```
tar -xf  latest-zh_CN.tar.gz
http://192.168.106.100:8000/wordpress
```


请在下方填写您的数据库连接信息。如果您不确定，请联系您的服务提供商。

数据库名	wordpress	希望将WordPress安装到的数据库名称。
用户名	root	您的数据库用户名。
密码	123456	您的数据库密码。
数据库主机	lnmp_mysql	如果localhost不能用，您通常可以从网站服务提供商处得到正确的信息。
表前缀	wp_	如果您希望在同一个数据库安装多个WordPress，请修改前缀。

至此 wordpress 部署完成

http://192.168.106.100:8000/wordpress/wp-login.php?loggedout=true&wp_lang=zh_CN


您已注销。

用户名或邮箱地址

密码

通过dockerfile实现了两大项目的部署工作

基于tomcat的jenkins项目

基于PHP-LNMP的wordpress博客项目

| 公众号专属福利 1 |

| 2020全新专题课程限时免费中 |

Python：WEB SSH实战

Python:运维未来之路

Python:日志分析

| 公众号专属福利 2 |

| 长按识别免费领取实战手册|

Read more

喜欢此内容的人还喜欢

主流日志采集器，阴暗潮湿的地底世界

DevOps技术栈

机房布线的最高境界......

Linux学习

ElasticSearch 集群压力测试指南

奇妙的Linux世界