**Common-web 部署文档**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 日期 | 作者 | 修订要点 |
| V1.00 | 2023年5月23日 |  | 新作成 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[部署文档 2](#_Toc134531977)

[1、 打包 2](#_Toc134531978)

[2、 将打包好的.jar文件上传至服务器 3](#_Toc134531979)

[3、 创建Dockerfile文件 3](#_Toc134531980)

[3.1 首先跳转目标路径 3](#_Toc134531981)

[3.2 创建Dockerfile文件 3](#_Toc134531982)

[3.3 输入以下内容 4](#_Toc134531983)

[4、 打成镜像文件上传至镜像库 5](#_Toc134531984)

[.4.1、找出上个镜像的版本信息 5](#_Toc134531985)

[4.2、构建镜像 5](#_Toc134531986)

[4.3、上传至镜像仓库 5](#_Toc134531987)

[5、 启动镜像 6](#_Toc134531988)

[1、找到以前部署过的程序 8](#_Toc134531989)

[2、编辑设置 8](#_Toc134531990)

[3、添加环境变量 10](#_Toc134531991)

[4、运行 10](#_Toc134531992)

[5、swagger文档地址 11](#_Toc134531993)

# 部署文档

部署地址：192.168.10.191 root/YTIOCity2022

使用xshell等工具连接

# 打包

将Web项目打包

命令：

npm run build

例

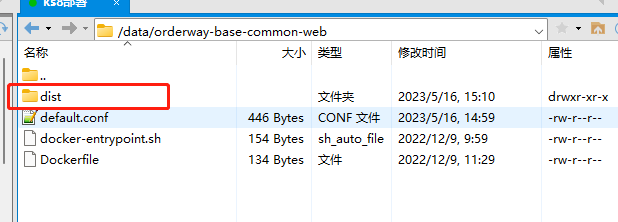


# 将打包好的dist文件上传至服务器

使用xftp等工具

上传地址为：/data/ orderway-base-common-web

例：



# 创建Dockerfile文件

### 3.1 首先跳转目标路径

cd /data/ orderway-base-common-web

### 3.2 创建Dockerfile文件

vi Dockerfile



### 3.3 输入以下内容

FROM nginx:latest

ENV TZ=Asia/Shanghai

COPY ./default.conf /etc/nginx/conf.d/

COPY ./dist /usr/share/nginx/html/

EXPOSE 8020

例如：



# 创建docker-entrypoint.sh 文件

### 1 首先跳转至目标路径

cd /data/orderway-base-common-web

### 2 创建docker-entrypoint.sh 文件

vi docker-entrypoint.sh

#!/usr/bin/env bash

set -e

cd /usr/share/nginx/html/

sed -i "/BAER\_API\_URL/$BAER\_API\_URL/" /etc/nginx/conf.d/default.conf

exec nginx -g 'daemon off;'

例如：



# 创建default.conf 文件

### 1 首先跳转至目标路径

cd /data/orderway-base-common-web

### 2 创建default.conf 文件

touch default.conf

-------------------------------------------------------------

server {

listen 8444;

server\_name localhost;

client\_max\_body\_size 100M;

underscores\_in\_headers on;

access\_log /dev/stdout main;

error\_log /dev/stderr;

location / {

root /usr/share/nginx/html/;

index index.html index.htm;

try\_files $uri $uri/ /index.html;

}

#location /common {

# alias /usr/share/nginx/html/;

# try\_files $uri $uri/ /common/index.html;

# }

location /api/ {

proxy\_pass http://192.168.10.221:31725/;

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Real-Port $remote\_port;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_redirect http:// $scheme://; #做https跳转

client\_max\_body\_size 1000m;

}

location /api\_common/ {

proxy\_pass http://192.168.10.221:31725/;

proxy\_set\_header Host $http\_host;

proxy\_set\_header X-Real-IP $remote\_addr;

proxy\_set\_header X-Real-Port $remote\_port;

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_redirect http:// $scheme://; #做https跳转

client\_max\_body\_size 1000m;

}

location /big\_data\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

proxy\_send\_timeout 180s; # 设置发送超时时间

proxy\_read\_timeout 180s; # 设置读取超时时间

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:31391/;

}

location /big\_data\_product\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:30994/;

}

location /big\_data\_tupu\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:30469/;

}

location /big\_data\_log\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:31082/;

}

location /big\_data\_celue\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:31024/;

}

location /orderway\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.221:30736/;

}

location /big\_data\_command\_web/ {

add\_header Access-Control-Allow-Origin \* always;

add\_header Access-Control-Allow-Headers \*;

add\_header Access-Control-Allow-Methods "GET, POST, PUT, OPTIONS";

if ($request\_method = 'OPTIONS') {

return 200;

}

proxy\_pass http://192.168.10.240:8060/;

}

error\_page 500 502 503 504 /50x.html;

location = /50x.html {

root /usr/share/nginx/html;

}

}----------------------------------------------------------------------

# 打成镜像文件上传至镜像库

#### .4.1、找出上个镜像的版本信息

docker images | grep common

例：

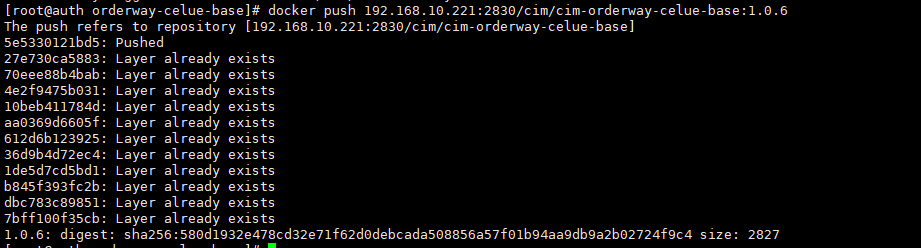


#### 4.2、构建镜像

镜像名：版本号 + 1

192.168.10.221:2830/cim/orderway-common-web 1.0.12.

#### 4.3、上传至镜像仓库

docker push 192.168.10.221:2830/cim/orderway-common-web 1.0.12 

# 启动镜像

K8s网址：<http://192.168.10.221:30880/login?referer=/clusters> admin/YTIOCity2022

平台管理 -> 应用负载 -> 工作负载



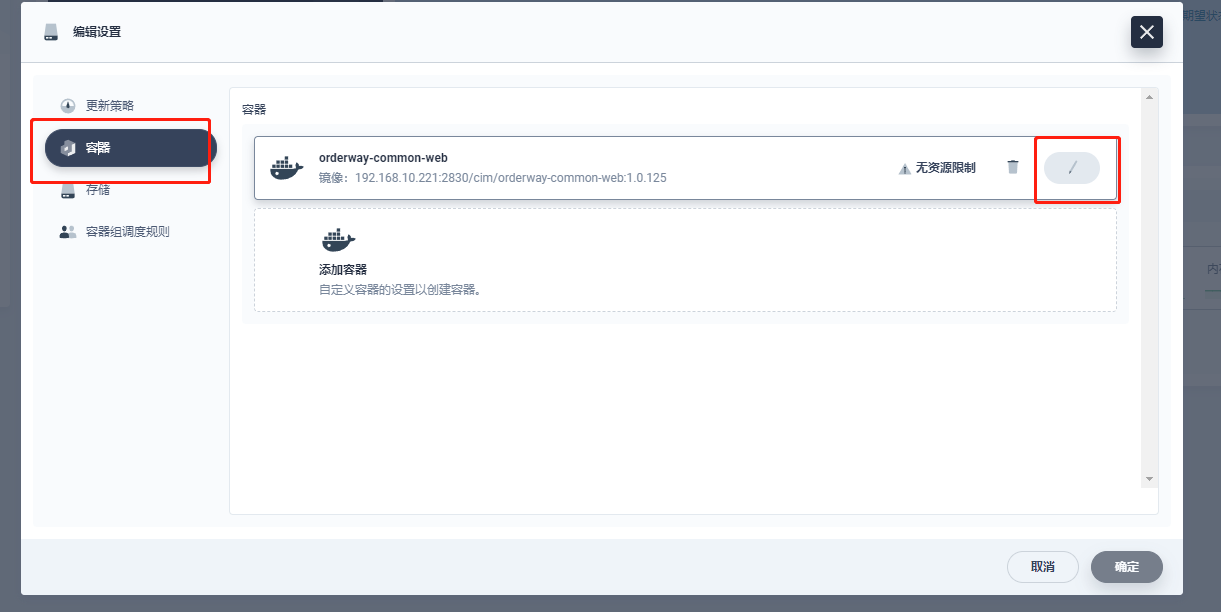
#### 1、找到以前部署过的程序



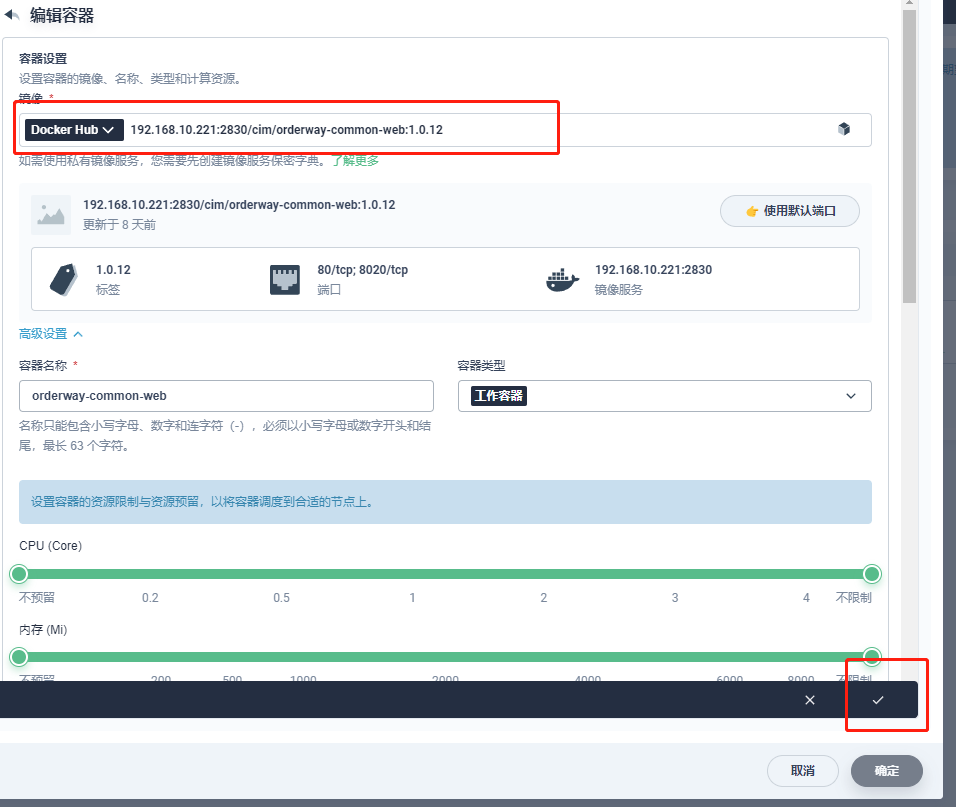
#### 2、编辑设置

例：





修改镜像名为刚上传的镜像名



#### 4、运行

