[shell] docker deploy service script

```
#! /bin/bash

# service name
SERVICE_NAME=service-name
# service port (--net=host invalid)
OPEN_PORT=7000
#
INSTANCES=1

# log path
LOG_PATH=/logs

# author: wzhz
# email: itqmdx@gmail.com
# version: v0.4
version=v0.4

# local ip (--net=host invalid)
IP=$(ip a | grep inet | grep -v 127.0.0.1 | grep -v inet6 | grep -v docker | awk '{print $2}'
| tr -d 'addr:' | awk -F '/' '{print $1}' | head -1)
# get ip
# ip addr | grep 'state UP' -A2 | tail -n1 | awk '{print $2}' | awk -F"/" '{print $1}'
DATEVERSION=$(date +'%Y.%m.%d.%H')

# script_dir
script_dir=$(readlink -f $0)
bootpath=$(dirname $script_dir)

# logspath=$bootpath/logs
# configpath=$bootpath/config
# jarpath=$bootpath/jar

RED='\e[1;31m'
GREEN='\e[1;32m'
```

```
YELLOW='\033[1;33m'
BLUE='\E[1;34m'
PINK='\E[1;35m'
RES='\033[0m'

# get all filename in specified path
getFileName() {
  path=$1
  files=$(ls $bootpath/jar)
  for filename in $files
  do
    echo $filename # >> filename.txt
  done

  for file in `find  $1 -name "*.jar"`
  do
    echo $file
  done
}

# touch Dockerfile
createDockerfile() {
#  --Dspring.config.location=/config/*
cat > ./Dockerfile << EOF
FROM openjdk:8
VOLUME /logs
EXPOSE $OPEN_PORT
ENV TZ=Asia/Shanghai JAVA_OPTS="-server -Xms512m -Xmx512m -XX:PermSize=64M -
XX:MaxNewSize=256m -XX:MaxPermSize=128m -Djava.awt.headless=true "
RUN ln -snf /usr/share/zoneinfo/\$TZ /etc/localtime && echo \$TZ > /etc/timezone
ADD *.jar app.jar
ENTRYPOINT exec java \$JAVA_OPTS -Djava.security.egd=file:/dev/./urandom -jar /app.jar
EOF
}

# delete old and images
deleteOldImage() {
  docker image rm -f $SERVICE_NAME:$DATEVERSION >> /dev/null 2>&1;
  docker image ls
}
# delete old containers
```

```
deleteOldContainer() {
  OLD_INSTANCES=$(docker container ps -a | grep -i $SERVICE_NAME | wc -l);
  for((i=0;i<$OLD_INSTANCES;i++));
  do
    docker container stop $SERVICE_NAME-$i >> /dev/null 2>&1;
    docker container rm -f $SERVICE_NAME-$i >> /dev/null 2>&1;
  done
  # rm -rf $bootpath/logs;
  if docker container ps -a | grep -i $SERVICE_NAME;then
    echo -e $RED hase $OLD_INSTANCES instances. $RES
  fi
  docker container ps
}

# build docker image
buildImage() {
  docker build -t $SERVICE_NAME:$DATEVERSION . ;
  docker image ls
}

# run docker container
runImage() {
  for((i=0;i < $INSTANCES;i++));
  do
    name=$SERVICE_NAME-$i
    port=$(($OPEN_PORT+$i))
    docker container rm -f $name >> /dev/null 2>&1
    echo create container is $name:$IP:$port;

    # docker run \
    # --net=host \
    # -v $bootpath/logs:$LOG_PATH \
    # -v $bootpath/config:/config \
    # --name $name \
    # --restart=on-failure:10 \
    # -d $SERVICE_NAME >> /dev/null 2>&1
    docker run \
    --expose=$port \
    -p $port:$port \
    -v $bootpath/logs:$LOG_PATH \
    -e server.port=$port \
```

```
    -e spring.application.name=$SERVICE_NAME \
    -e spring.cloud.client.ip-address=$IP \
    -e EUREKA_INSTANCE_INSTANCE-ID=$IP:$SERVICE_NAME:$port \
    -e EUREKA_INSTANCE_IP-ADDRESS=$IP \
    -e SERVER_PORT=$port \
    -e JAVA_OPTS=-Xmx512m \
    --name $name \
    --restart=on-failure:10 \
    -d $SERVICE_NAME:$DATEVERSION # >> /dev/null 2>&1
    CONTAINERID_NEW=`docker container ps -a | grep ${name}| awk '{print $NF}'`
    echo new container created successfully is $CONTAINERID_NEW
    if [ $i -lt $INSTANCES ];then
      sleep 1;
    fi
  done
  docker container ps
}
startContainer() {
  for((i=0;i < $INSTANCES;i++));
  do
    name=$SERVICE_NAME-$i
    port=$(($OPEN_PORT+$i))

    docker container start $name >> /dev/null 2>&1
    echo start container is $name:$IP:$port
    if [ $i -lt $INSTANCES ];then
      sleep 1;
    fi
  done
  docker container ps
}
restartContainer() {
  for((i=0;i < $INSTANCES;i++));
  do
    name=$SERVICE_NAME-$i
    port=$(($OPEN_PORT+$i))

    docker container restart $name >> /dev/null 2>&1
    echo restart container is $name:$IP:$port
    if [ $i -lt $INSTANCES ];then
      sleep 1;
```

```
    fi
  done
  docker container ps
}
stopContainer() {
  for((i=0;i < $INSTANCES;i++));
  do
    name=$SERVICE_NAME-$i
    port=$(($OPEN_PORT+$i))

    docker container stop $name >> /dev/null 2>&1
    echo stop container is $name:$IP:$port
    if [ $i -lt $INSTANCES ];then
      sleep 1;
    fi
  done
  docker container ps
}
viewContainerLog() {
  if [ $INSTANCES -eq 1 ];then
    showLog $SERVICE_NAME-0
  else
    # more
    echo -e $GREEN show logs for containers: $RES
    docker ps -a | grep ${SERVICE_NAME}| awk '{print $1, $2, $(NF-1), $NF}'
    read -p 'please input a container id or name: ' input
    showLog $input
  fi
}
showLog() {
  docker container logs -f --tail=100 $1
}
readme() {
  echo -e $GREEN ---- deploy service script $RES
  echo -e $YELLOW ---- author: wzhz $RES
  echo -e $YELLOW ---- email: itqmdx@gmail.com $RES
  echo -e $YELLOW ---- version: $version $RES
}

current() {
  echo
```

```
  echo -e $PINK current time is $(date +'%Y-%m-%d %T') $RES
  echo
}

var() {
  echo
  echo IP $IP
  echo SERVICE_NAME $SERVICE_NAME
  echo OPEN_PORT $OPEN_PORT
  echo INSTANCES $INSTANCES
  echo DATEVERSION $DATEVERSION
  echo
}

# setting env var
setEnvironmentVariable() {
  ARRT=$1
  ARRT_NAME=`echo ${ARRT} | awk -F '=' '{print $1}'`
  ARRT_VALUE=`echo ${ARRT} | awk -F '=' '{print $2}'`
  # echo $ARRT_NAME is $ARRT_VALUE
  if [ $ARRT_NAME == 'name' ]; then
    SERVICE_NAME=$ARRT_VALUE
  elif [ $ARRT_NAME == 'port' ]; then
    OPEN_PORT=$ARRT_VALUE
  elif [ $ARRT_NAME == 'ip' ]; then
    IP=$ARRT_VALUE
  elif [ $ARRT_NAME == 'i' ]; then
    INSTANCES=$ARRT_VALUE
  else
    echo
    echo -e $RED $ARRT no matches found. $RES
    echo
  fi
}

functionItems() {
  echo
  echo -e $GREEN = 0. perform steps 7-8 and 1-3 automatically $RES
  echo -e $BLUE = 1. create current environment\'s Dockerfile $RES
  echo -e $BLUE = 2. build image $SERVICE_NAME $RES
  echo -e $BLUE = 3. run image $SERVICE_NAME $RES
```

```
  echo -e $BLUE = 4. start $SERVICE_NAME\'s containers $RES
  echo -e $BLUE = 5. restart $SERVICE_NAME\'s containers $RES
  echo -e $BLUE = 6. stop $SERVICE_NAME\'s containers $RES
  echo -e $BLUE = 7. delete $SERVICE_NAME\'s containers  $RES
  echo -e $BLUE = 8. delete image $SERVICE_NAME $RES
  echo -e $BLUE = 9. view $SERVICE_NAME\'s container log $RES
  echo -e $RED = 99. configure global information $RES
  echo
}

main() {
  functionItems
  read -p 'please input a function item no: ' input
  echo your input is $input
  case $input in
    0)
    deleteOldContainer
    echo -e $GREEN delete containers successfully. $RES
    deleteOldImage
    echo -e $GREEN delete image successfully. $RES
    createDockerfile
    echo -e $GREEN Dockerfile created successfully, default is based on openjdk:8. $RES
    buildImage
    echo -e $GREEN created successfully, image is $SERVICE_NAME. $RES
    runImage
    echo -e $GREEN runs successfully. $RES
    ;;
    1)
    createDockerfile
    echo -e $GREEN Dockerfile created successfully, default is based on openjdk:8. $RES
    cat Dockerfile
    ;;
    2)
    buildImage
    echo -e $GREEN created successfully, image is $SERVICE_NAME. $RES
    ;;
    3)
    runImage
    echo -e $GREEN runs successfully. $RES
    ;;
    4)
```

```
    startContainer
    echo -e $GREEN start container successfully. $RES
    ;;
    5)
    restartContainer
    echo -e $GREEN restart container successfully. $RES
    ;;
    6)
    stopContainer
    echo -e $GREEN stop container successfully. $RES
    ;;
    7)
    deleteOldContainer
    echo -e $GREEN delete containers successfully. $RES
    ;;
    8)
    deleteOldImage
    echo -e $GREEN delete image successfully. $RES
    ;;
    9)
    viewContainerLog
    echo -e $GREEN view container log complete. $RES
    ;;
    99)
    echo -e $YELLOW developing... $RES
    ;;
    *)
    echo -e $RED wrong input, exit 0. $RES
    exit 0
    ;;
  esac
}

echo -e $YELLOW working directory is $bootpath $RES
cd $bootpath;ls -all;

for arg in $@
do
  setEnvironmentVariable $arg
done
```

```
readme


current


var


while true
do
  main
  sleep 1
done
```

Revision #4
Created 16 March 2020 04:53:54 by
Updated 3 July 2020 05:59:25 by