# 提交代码到远程Git仓库

1. 准备 Git 仓库

码云：https://gitee.com

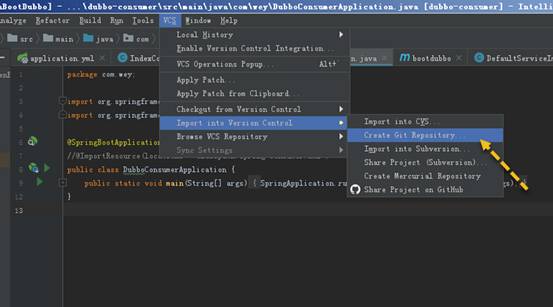
#### 1.1. 通过网站右上角的「+」号，选择「新建仓库」，进入新建仓库页面



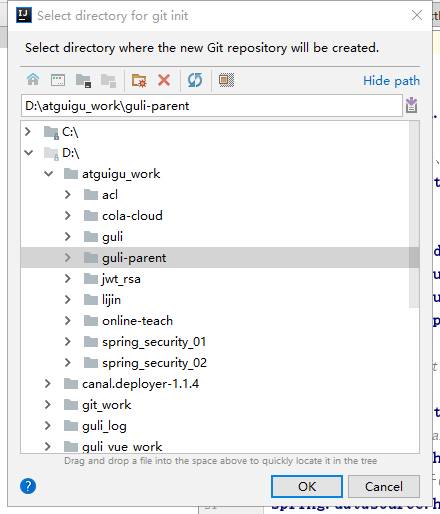
#### 1.2. 新建仓库



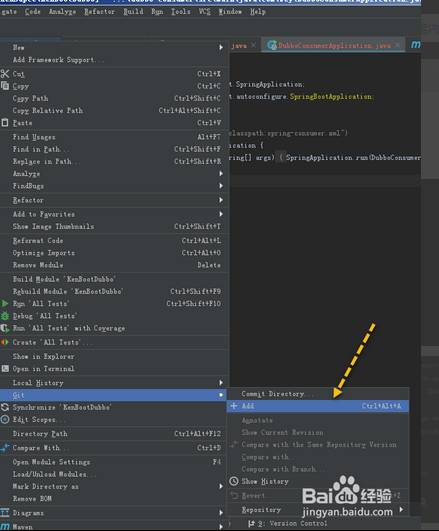
#### 1.3. 打开项目并点击菜单栏上的【CVS】--》【Import into version control】--》【Create Git Repository】创建本地仓库



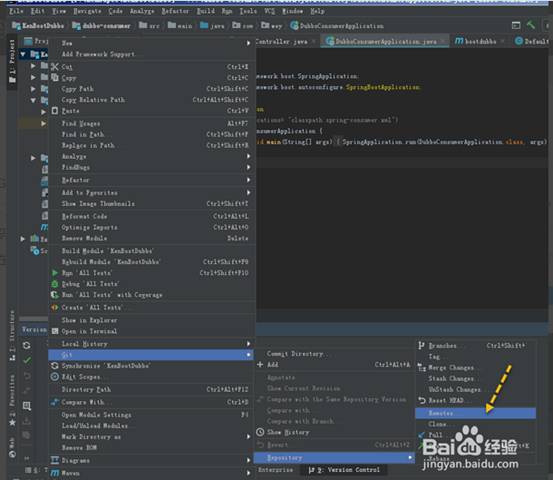
#### 1.4. 在打开的【Create Git Repository】对话框内选择本地仓库的位置，这里我选择项目的根目录。



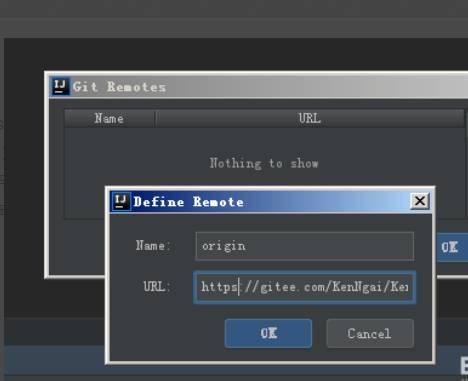
#### 1.5. 右击项目点击【Git】--》【Add】，接着点击【Git】--》【Commit Directory】在打开的窗口中选择要上传到本地仓库的代码并添加注释后提交到本地仓库内。



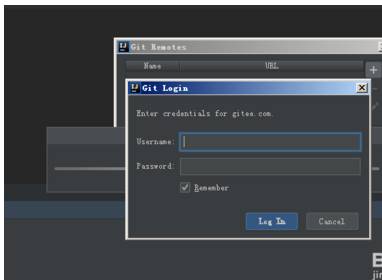
#### 1.6. 右击项目点击【Git】--》【Repository】--》【Remotes...】。在打开的【Git Remotes】窗口中添加码云的远程仓库。码云的远程仓库地址可以在码云仓库内找到。







#### 1.7. 点击【OK】后接着输入码云的账号密码。



#### 1.8. 上传代码到码云，右击项目点击【Git】--》【Repository】--》【Push...】在打开的【Push commits】内可以看到已提交到本地仓库的提交信息。点击【Push】按钮将本地仓库的代码上传到码云上，上传成功后就可以在码云上看到啦。

## 课堂笔记

###### Git操作（码云）

|  |
| --- |
| **1、提交过程**    **2、使用Git远程仓库使用码云**  **第一步 进入码云官方网站，注册用户**  [**https://gitee.com/**](https://gitee.com/)  **第二步 创建远程仓库**  **点击右上角的 ＋ 号进行创建**    **填写仓库信息**      **看到创建成功的仓库** |

###### Git仓库（提交代码）

|  |
| --- |
| **1、准备工作：安装git**  （1）在idea配置git环境    **2、创建本地git仓库**  **（1）选择创建本地git仓库**    **（2）选择当前项目目录作为本地git仓库目录**    **（3）创建成功，代码颜色变化**    **3、右键点击项目，选择Git，添加项目代码到本地库**    **（1）选择Git里面Add，添加代码到本地库**    **4、设置远程Git仓库地址**    **（1）设置码云地址**    **（2）点击OK，连接码云远程Git仓库，输入码云用户名和密码**  **\* 第一次需要输入，后面可以保存，之后不需要再输入了**    **5、把本地Git库内容，添加码云远程Git库**                **点击Push就可以了** |

###### Git操作远程库（细节）

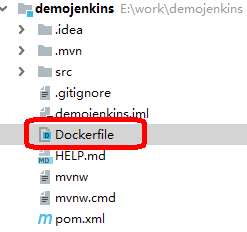
|  |
| --- |
| 1、在码云创建仓库时候，不要选择下面内容    2、如果保存输入码云用户名和密码，如何删除？        **3、让本地代码和远程库断开连接**    找到项目文件夹，删除文件夹.git |

# 项目部署

## 1. 准备代码，提交到码云Git库[包含dockfile]

**代码中需要包含以下几部分内容：**

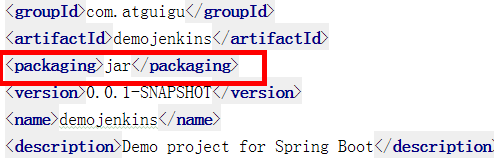
**（1）代码中需要包含Dockerfile文件**

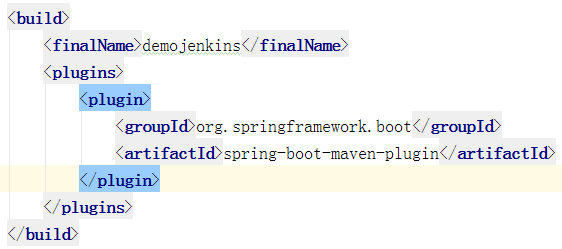


**文件内容：**

**FROM openjdk:8-jdk-alpine  
VOLUME /tmp  
COPY ./target/demojenkins.jar demojenkins.jar  
ENTRYPOINT ["java","-jar","/demojenkins.jar", "&"]**

**（2）在项目pom文件中指定打包类型，包含build部分内容**





## 2. 安装JAVA 运行环境

第一步：上传或下载安装包

cd/usr/local

jdk-8u121-linux-x64.tar.gz

第二步：解压安装包

tar -zxvf jdk-8u121-linux-x64.tar.gz

第三步：建立软连接

ln -s /usr/local/jdk1.8.0\_51 /usr/local/jdk

第四步：修改环境变量

vim /etc/profile

export JAVA\_HOME=/usr/local/jdk

export JRE\_HOME=$JAVA\_HOME/jre

export CLASSPATH=.:$CLASSPATH:$JAVA\_HOME/lib:$JRE\_HOME/lib

export PATH=$PATH:$JAVA\_HOME/bin:$JRE\_HOME/bin

通过命令source /etc/profile让profile文件立即生效

source /etc/profile

第五步、测试是否安装成功

②、使用java -version，出现版本

## 3. 安装maven

第一步：上传或下载安装包

cd/usr/local

apache-maven-3.6.1-bin.tar.gz

第二步：解压安装包

tar -zxvf apache-maven-3.6.1-bin.tar.gz

第三步：建立软连接

ln -s /usr/local/apache-maven-3.6.1 /usr/local/maven

第四步：修改环境变量

vim /etc/profile

export MAVEN\_HOME=/usr/local/maven

export PATH=$PATH:$MAVEN\_HOME/bin

通过命令source /etc/profile让profile文件立即生效

source /etc/profile

第五步、测试是否安装成功

mvn –v

## 4. 安装git

# 安装git

yum -y install git

# 查看git安装位置

which git

## 5. 安装docker

参考文档：

<https://help.aliyun.com/document_detail/60742.html?spm=a2c4g.11174283.6.548.24c14541ssYFIZ>

第一步：安装必要的一些系统工具

yum install -y yum-utils device-mapper-persistent-data lvm2

第二步：添加软件源信息

yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo

第三步：更新并安装Docker-CE

yum makecache fast

yum -y install docker-ce

第四步：开启Docker服务

service docker start

第五步、测试是否安装成功

docker -v

## 6. 安装Jenkins

第一步：上传或下载安装包http://updates.jenkins-ci.org/download/war/

cd/usr/local/jenkins

jenkins.war

第二步：启动

nohup java -jar  /usr/local/jenkins/jenkins.war >/usr/local/jenkins/jenkins.out &

nohup java -jar -Dhudson.model.DownloadService.noSignatureCheck=true /usr/local/jenkins/jenkins.war >/usr/local/jenkins/jenkins.out &

命令解释：

nohup和&的区别

&是指在后台运行

运行 nohup --help Run COMMAND, ignoring hangup signals. 可以看到是“运行命令，忽略挂起信号”就是不挂断的运行，注意没有后台运行功能。

就是指，用nohup运行命令可以使命令永久的执行下去，和用户终端没有关系，例如我们断开SSH连接都不会影响他的运行，注意了nohup没有后台运行的意思；&才是后台运行

**&是指在后台运行，但当用户推出(挂起)的时候，命令自动也跟着退出**

* **启动出现下面界面**

****

* **再次点击回车**

第二步：访问

<http://ip:8080>

## 7. 初始化 Jenkins 插件和管理员用户

### 7.1访问jenkins

<http://ip:8080>

### 7.2解锁jenkins

获取管理员密码

cat /root/.jenkins/secrets/initialAdminPassword

C:\Users\gouwenyong0000\Documents\My Knowledge\temp\71f0af91-e22f-4cbd-a7f3-89b17b7bb133\128\index_files\clip_image0229d3bd13f-f015-47a5-9cd1-e480f73bbed4.jpg





### 注意：配置国内的镜像

ps -ef | grep Jenkins

kill -9 pid

杀掉进程

官方下载插件慢 更新下载地址

cd {你的Jenkins工作目录}/updates  #进入更新配置位置 参考密码文件的地址

cd /root/.jenkins/updates/

sed -i 's/http:\/\/updates.jenkins-ci.org\/download/https:\/\/mirrors.tuna.tsinghua.edu.cn\/jenkins/g' default.json && sed -i 's/http:\/\/www.google.com/https:\/\/www.baidu.com/g' default.json

这是直接修改的配置文件，如果前边Jenkins用sudo启动的话，那么这里的两个sed前均需要加上sudo

重启Jenkins，安装插件

https://blog.csdn.net/qq\_35024768/article/details/121326912

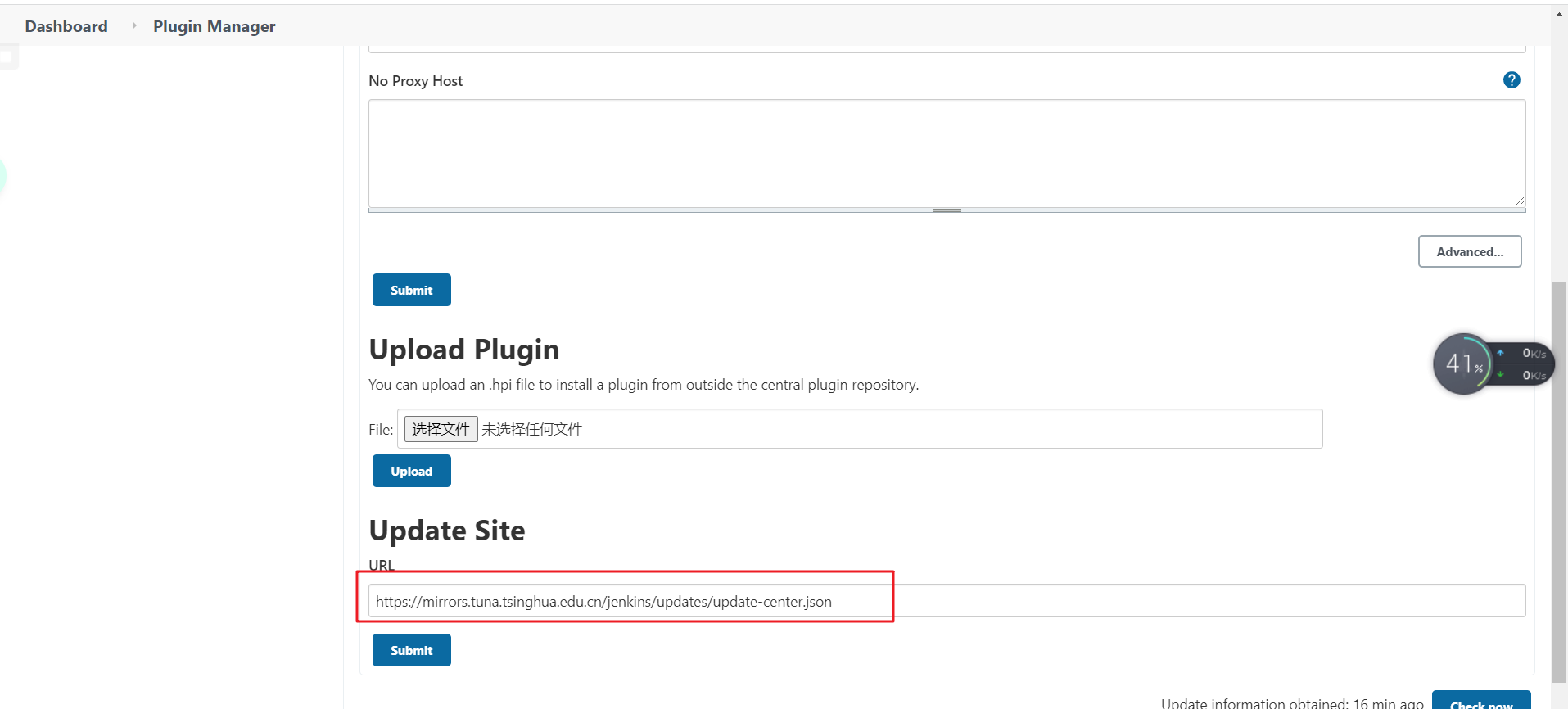
**未到插件安装界面 解决：如果没有找到updates   参考如下解决：**

**https://caoju.blog.csdn.net/article/details/120498306?spm=1001.2101.3001.6650.1&utm\_medium=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.no\_search\_link&depth\_1-utm\_source=distribute.pc\_relevant.none-task-blog-2%7Edefault%7ECTRLIST%7Edefault-1.no\_search\_link**

网上找了各种解决方法，修改jenkins插件的下载源地址：找到菜单Manage Jenkins → Manage Plugins → Advanced → Update Site，

进入界面：http://ip:8080/pluginManager/advanced

把URL改为https://mirrors.tuna.tsinghua.edu.cn/jenkins/updates/update-center.json



**1、javac InstallCert.java**

**2、java InstallCert <hostname>**

**此处hostname 为 mirrors.tuna.tsinghua.edu.cn，国内jenkins下载插件的地址会重定向到清华大学提供的镜像java InstallCert mirrors.tuna.tsinghua.edu.cn**

**echo $JAVA\_HOME # 查看安装位置**

**3、然后将生成的 jssecacerts 文件，拷贝到jdk中，目录位置：%JAVA\_HOME%\jre\lib\security**

**（例如 cp jssecacerts /usr/local/jdk1.8.0\_51/jre/lib/security/）**

**4.最后重新启动jenkins，证书才能生效。http://ip:8080/restart**

**注意：在linux上去编译执行InstallCert.java可能会报错，可以在windows的环境下执行完得到jssecacerts 文件再把jssecacerts 文件放到linux里的/usr/local/jdk1.8.0\_60/jre/lib/security下**



手动安装插件：

Timestamper

Credentials Binding

Email Extension

Localization: Chinese (Simplified)

Build Timeout

Pipeline

Pipeline: Stage View

Ant

GitHub Branch Source

SSH Build Agents

Pipeline: GitHub Groovy Libraries

Matrix Authorization Strategy

Mailer

OWASP Markup Formatter

Gradle

PAM Authentication

Git

Workspace Cleanup

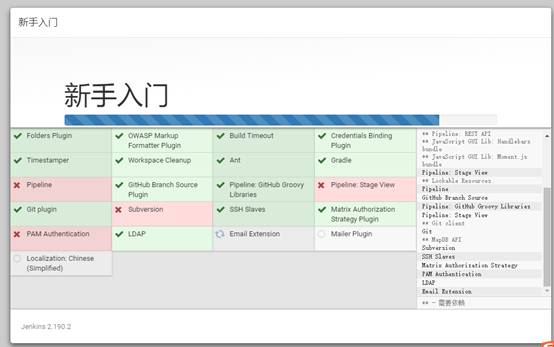
LDAP

Folders

### 7.3选择“继续”



### 7.4选择“安装推荐插件”



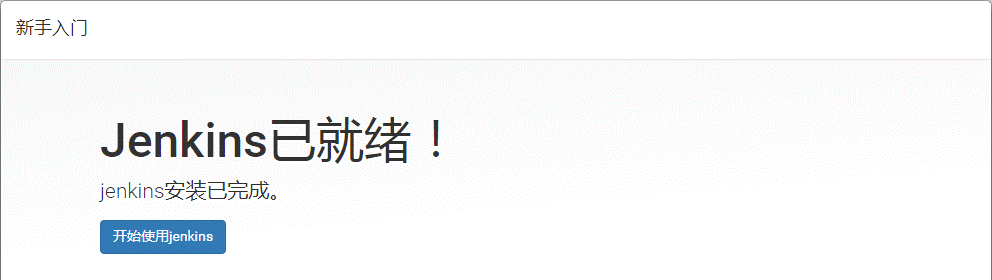
### 7.5插件安装完成，创建管理员用户



### 7.6保存并完成



### 7.7进入完成页面



## 8. 配置 Jenkins 构建工具



### 8.1全局工具配置



#### 8.1.1配置jdk

JAVA\_HOME：/usr/local/jdk



#### 8.1.2配置maven

MAVEN\_HOME：/usr/local/maven



#### 8.1.2配置git

查看git安装路径：which git



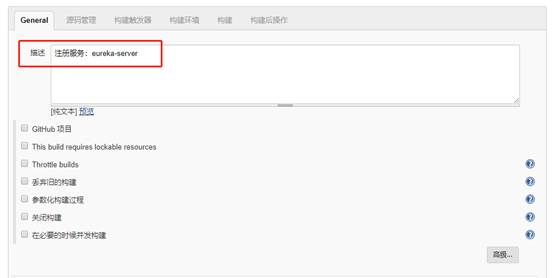
## 9. 构建作业

### 9.1点击创建一个新任务，进入创建项目类型选择页面



填好信息点击“确认”

### 9.2配置“General”

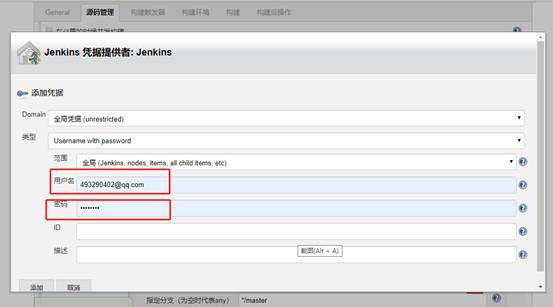


### 9.3配置“源码管理”

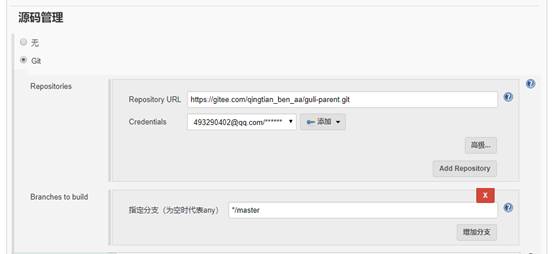
填写源码的git地址



添加git用户，git的用户名与密码



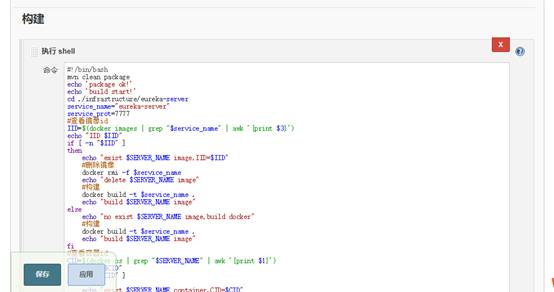
选择添加的用户，上面的红色提示信息消失，说明连接成功，如下图



### 9.4构建作业

**到源码中找到docker脚本**

选择“执行shell”



#!/bin/bash

#maven打包

mvn clean package

echo 'package ok!'

echo 'build start!'

cd ./infrastructure/eureka\_server

service\_name="eureka-server"

service\_prot=8761

#查看镜像id

IID=$(docker images | grep "$service\_name" | awk '{print $3}')

echo "IID $IID"

if [ -n "$IID" ]

then

    echo "exist $SERVER\_NAME image,IID=$IID"

    #删除镜像

    docker rmi -f $service\_name

    echo "delete $SERVER\_NAME image"

    #构建

    docker build -t $service\_name .

    echo "build $SERVER\_NAME image"

else

    echo "no exist $SERVER\_NAME image,build docker"

    #构建

    docker build -t $service\_name .

    echo "build $SERVER\_NAME image"

fi

#查看容器id

CID=$(docker ps | grep "$SERVER\_NAME" | awk '{print $1}')

echo "CID $CID"

if [ -n "$CID" ]

then

    echo "exist $SERVER\_NAME container,CID=$CID"

    #停止

    docker stop $service\_name

    #删除容器

    docker rm $service\_name

else

    echo "no exist $SERVER\_NAME container"

fi

#启动

docker run -d --name $service\_name --net=host -p $service\_prot:$service\_prot $service\_name

#查看启动日志

#docker logs -f  $service\_name

保存上面的构建作业



### 9.5构建

构建作业之后，就可以执行构建过程了。

#### 9.5.1执行构建过程



#### 9.5.2构建结构

第一列是 "上次构建状态显示"，是一个圆形图标，一般分为四种：

IMG_256

蓝色：构建成功；

IMG_257

黄色：不确定，可能构建成功，但包含错误；

IMG_258

红色：构建失败；

IMG_259

灰色：项目从未构建过，或者被禁用；

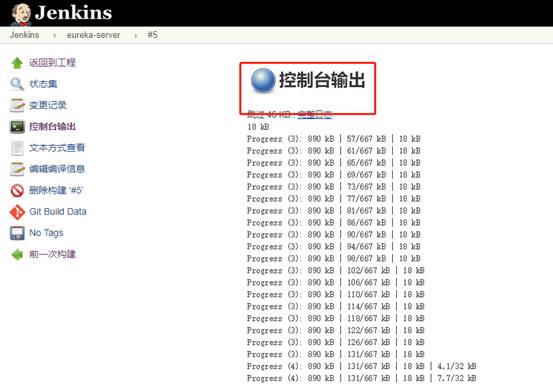
如上显示蓝色，表示构建成功。

注意：手动触发构建的时间与自动定时构建的时间互不影响。

#### 9.5.3查看控制台输出



日志内容：





## \*\*\*课堂笔记

###### Day19内容介绍

|  |
| --- |
| 1、持续化部署工具 jenkins  （1）手动打包运行过程  （2）使用jenkins实现完整过程 |

###### Jenkins（手动打包）

|  |
| --- |
| **1、创建普通SpringBoot工程**      **2、把工程进行打包，运行**  （1）把项目打包，因为SpringBoot工程，通过main方法执行，打包变成jar包形式，使用maven进行打包操作  \* 前提条件：安装好maven，配置环境变量    \* 进入到项目目录中，使用maven命令： mvn clean package      （2）打包完成之后，在项目target目录中找到jar包    **（3）把打成jar包运行**  \* 进入jar包所在目录，使用命令运行： java –jar jar包名称 |

###### Jenkins（安装）

|  |
| --- |
| **1、在liunx系统中安装相关软件**  **（1）安装java环境（jdk环境）**  第一步 上传jdk安装介质  第二步 解压  第三步 配置环境变量  第四步 使用命令查看jdk是否安装成功    **（2）安装maven环境**  第一步 上传maven安装介质    第二步 解压  第三步 配置环境变量  第四步 使用命令查看maven是否安装成功    **（3）安装Git环境**  **yum -y install git**  **（4）安装Docker**  第一步：安装必要的一些系统工具  yum install -y yum-utils device-mapper-persistent-data lvm2    第二步：添加软件源信息  yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linux/centos/docker-ce.repo    第三步：更新并安装Docker-CE  yum makecache fast  yum -y install docker-ce    第四步：开启Docker服务  service docker start    第五步、测试是否安装成功  docker -v    **2、安装jenkins**  **（1）把jenkins的war包上传到linux系统中**      **（2）启动war包，使用命令 java –jar**  nohup java -jar  /usr/local/jenkins/jenkins.war >/usr/local/jenkins/jenkins.out &   * **启动出现下面界面**      * **再次点击回车**   **（3）访问jenkins管理页面**  \* 使用当前linux系统ip地址:8080端口号  \* **首先进入初始化界面，需要多等一会，之后进入到下面界面**    **（4）解锁操作**  **\* 找到管理员密码进行输入**     * **查看提示文件内容，复制默认密码到输入框里面**        * **点击继续**      * **进入安装插件页面，不要往下进行了，下载插件会很慢，配置国内镜像，为了提高下载速度，关掉浏览器，去配置镜像**      * **配置国内镜像**   **cd /root/.jenkins/updates  #进入更新配置位置**  sed -i 's/http:\/\/updates.jenkins-ci.org\/download/https:\/\/mirrors.tuna.tsinghua.edu.cn\/jenkins/g' default.json && sed -i 's/http:\/\/www.google.com/https:\/\/www.baidu.com/g' default.json       * **重启jenkins，运行管理界面，安装插件**        * **安装插件之后，插件管理员用户名和密码**      * **实例配置**      * **安装完成** |

###### Jenkins（配置）

|  |
| --- |
| 1、配置自动化部署需要环境      2、配置jdk环境    3、配置maven环境    4、配置git环境 |

###### Jenkins（自动化过程）

|  |
| --- |
| 1、准备工作  （1）需要项目工程，需要做修改  第一 项目需要Dockerfile文件      第二 在项目pom文件添加打包类型 和 maven插件      **2、在jenkins管理界面创建自动化任务**  **（1）**    **（2）**    **（3）把代码提交码云Git仓库里面**  （4）  **在作业创建页面指定代码位置**    **（5）**      **（6）修改脚本路径，进行复制**      （7）执行作业（任务）  **\* 需要docker启动起来**  **service docker start**    或者    **（8）点击Build Now执行任务**     * 看到控制台日志输出 |

###### Jenkins（idea打包）

|  |
| --- |
| 第一步    第二步    第三步    第四步    第五步    第六步    第七步      最后看到打包的结果 |