**基于Docker+Jenkins实现自动化部署**

# 传统部署与自动化部署

# 使用Jenkins与Git自动化部署原理

# 使用码云搭建Git代码存储仓库

<https://gitee.com/login>

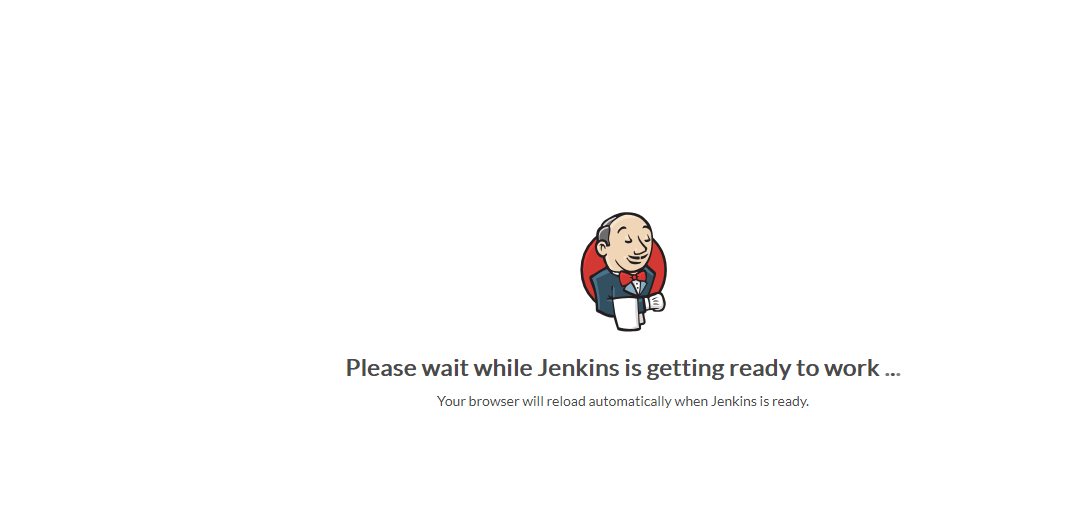
1. 使用码云创建私有私有git仓库
2. 将本地springboot项目上传到git仓库中

# 基于Docker安装Jenkins环境

1.使用docker 安装jenkins

|  |
| --- |
| docker run -p 8080:8080 -p 50000:50000 -v jenkins\_data:/var/jenkins\_home jenkinsci/blueocean |

2.访问jenkins地址 8080



注意：第一次启动的时候正在加载jenkins大概会等待3-10分钟。

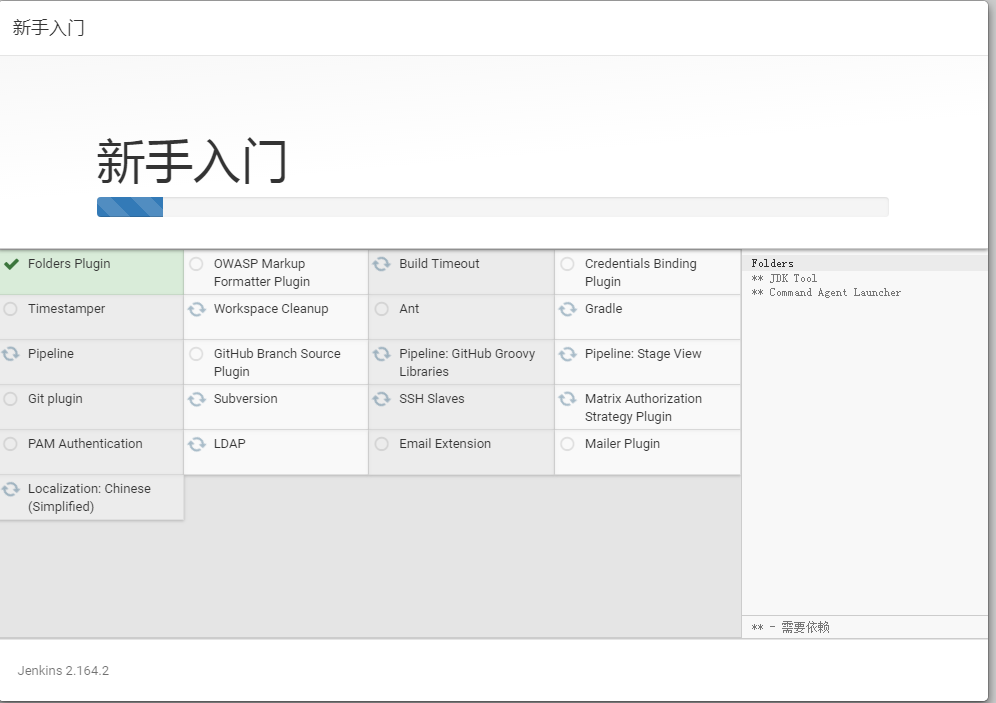
3.解锁jenkins



docker exec -it 7f485bd95c3b /bin/bash 进入jenkins容器

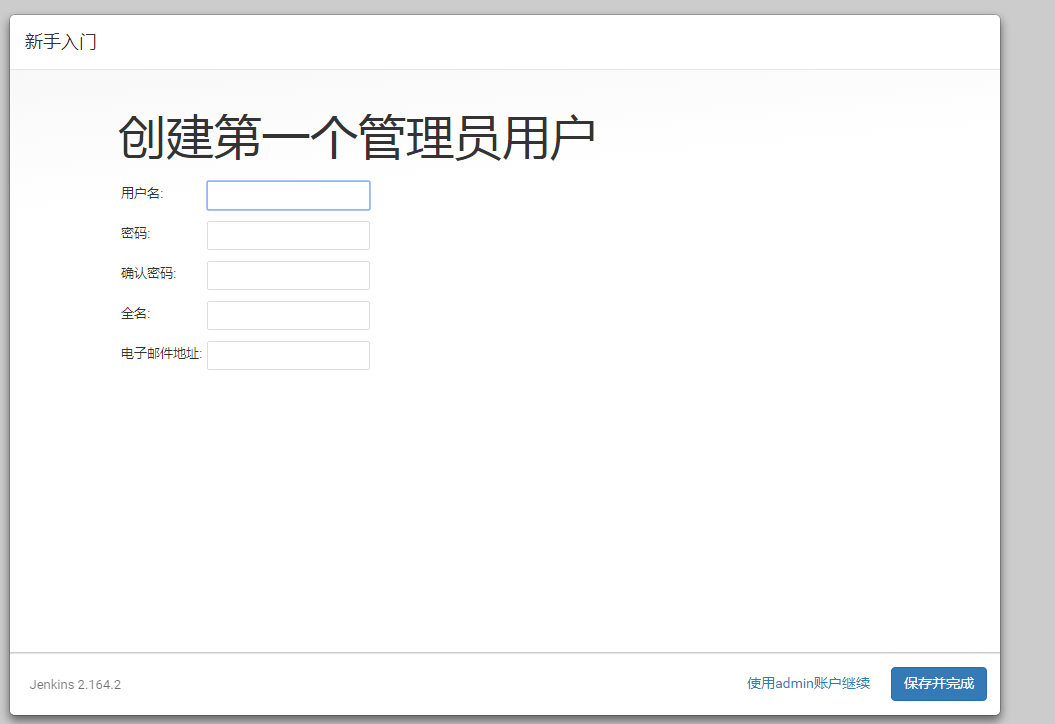
cat /var/jenkins\_home/secrets/initialAdminPassword





大概需要等待3-10分钟

创建新的用户





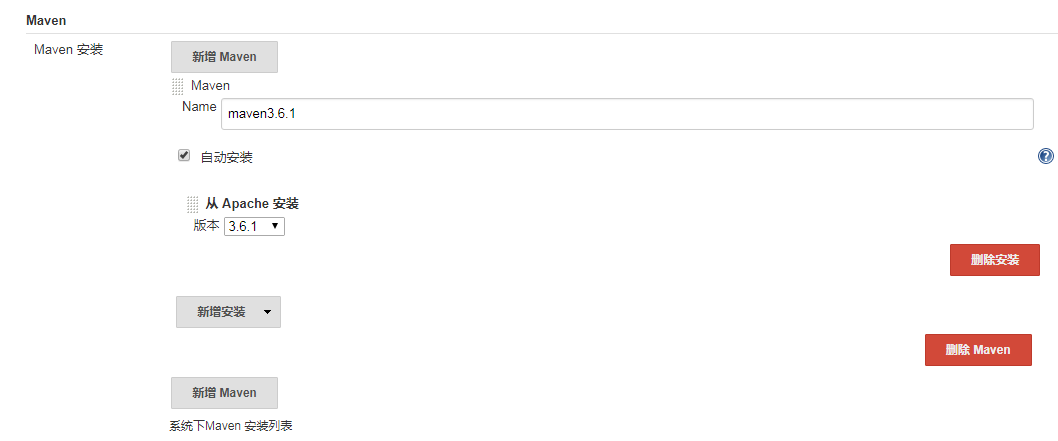
## Jenkins全局工具配置

进入到jenkins容器中 echo $JAVA\_HOME 获取java环境安装地址

### JDK环境安装

|  |
| --- |
|  |

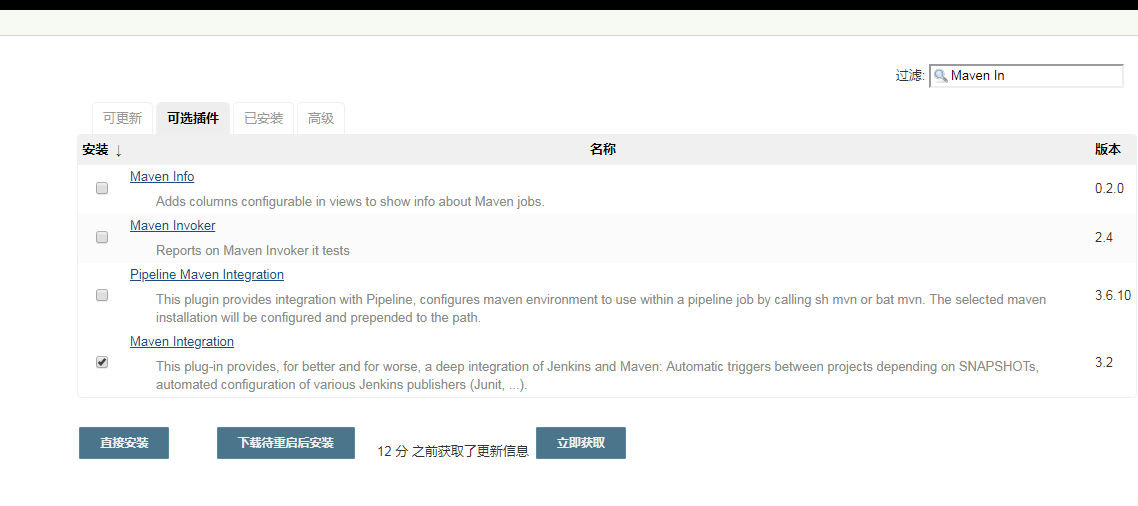
### Maven环境安装

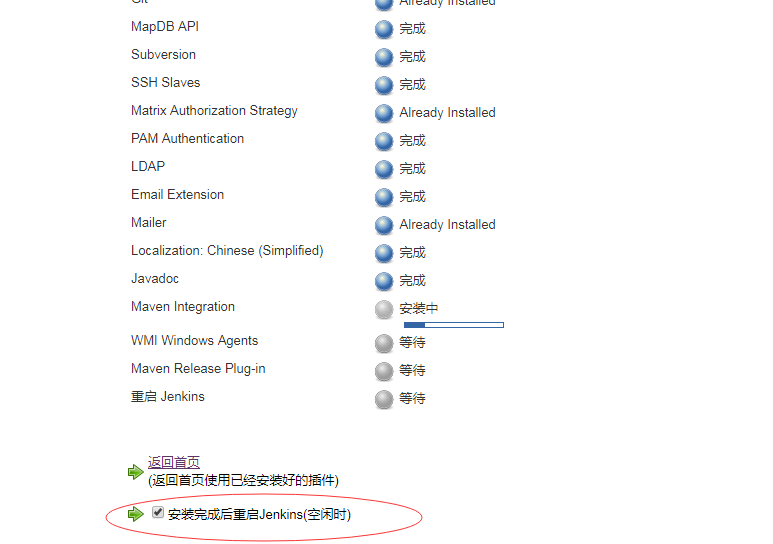


### 安装Jenkins对应Maven插件

找到 “系统管理“ - “安装插件” ，点击 “可选插件”，找到如下maven插件的版本

插件名称 [Maven Integration](https://plugins.jenkins.io/maven-plugin)



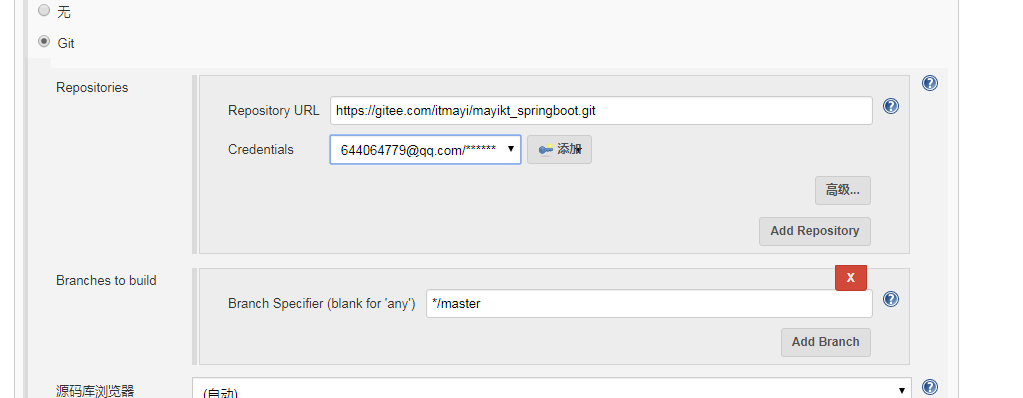


## Jenkins实现Springboot项目自动部署

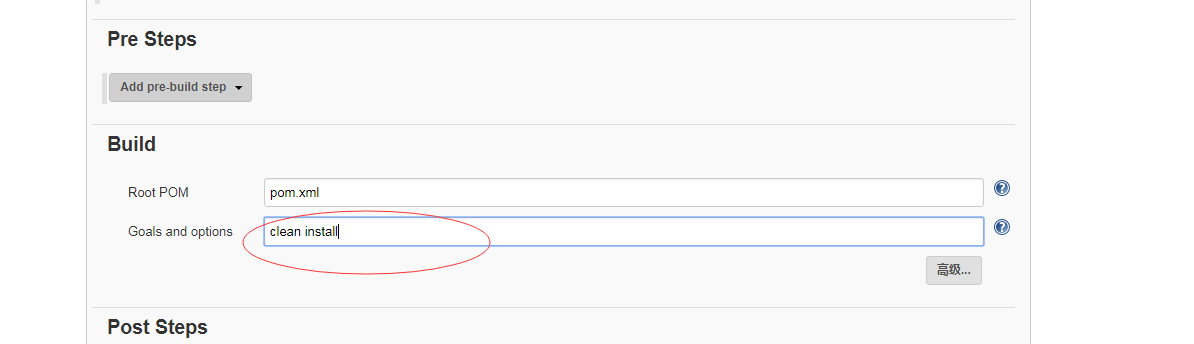
1. 新建一个发布任务



1. 配置任务git账号密码



3.项目打包



clean install

1. 点击立即构建



第一次构建可能耗时比较长，因为需要下载一些相关依赖jar包

## Jenkins启动成功之后执行shll脚本

|  |
| --- |
| #!/bin/bash  #服务名称  SERVER\_NAME=mayikt\_springboot  # 源jar路径,mvn打包完成之后，target目录下的jar包名称，也可选择成为war包，war包可移动到Tomcat的webapps目录下运行，这里使用jar包，用java -jar 命令执行  JAR\_NAME=mayikt\_springboot-0.0.1-SNAPSHOT  # 源jar路径  #/usr/local/jenkins\_home/workspace--->jenkins 工作目录  #demo 项目目录  #target 打包生成jar包的目录  JAR\_PATH=/var/jenkins\_home/workspace/mayikt\_springboot/target  # 打包完成之后，把jar包移动到运行jar包的目录--->work\_daemon，work\_daemon这个目录需要自己提前创建  JAR\_WORK\_PATH=/var/jenkins\_home/workspace/mayikt\_springboot/target  echo "查询进程id-->$SERVER\_NAME"  PID=`ps -ef | grep "$SERVER\_NAME" | awk '{print $2}'`  echo "得到进程ID：$PID"  echo "结束进程"  for id in $PID  do  kill -9 $id  echo "killed $id"  done  echo "结束进程完成"  #复制jar包到执行目录  echo "复制jar包到执行目录:cp $JAR\_PATH/$JAR\_NAME.jar $JAR\_WORK\_PATH"  cp $JAR\_PATH/$JAR\_NAME.jar $JAR\_WORK\_PATH  echo "复制jar包完成"  cd $JAR\_WORK\_PATH  #修改文件权限  chmod 755 $JAR\_NAME.jar  java -jar $JAR\_NAME.jar |

Jenkins启动SpringBoot后，停止转圈

将springboot启动方式设置为后台启动即可

|  |
| --- |
| #!/bin/bash  #服务名称  SERVER\_NAME=mayikt\_springboot  # 源jar路径,mvn打包完成之后，target目录下的jar包名称，也可选择成为war包，war包可移动到Tomcat的webapps目录下运行，这里使用jar包，用java -jar 命令执行  JAR\_NAME=meitedu\_springboot-0.0.1-SNAPSHOT  # 源jar路径  #/usr/local/jenkins\_home/workspace--->jenkins 工作目录  #demo 项目目录  #target 打包生成jar包的目录  JAR\_PATH=/var/jenkins\_home/workspace/meitedu\_springboot/target/  # 打包完成之后，把jar包移动到运行jar包的目录--->work\_daemon，work\_daemon这个目录需要自己提前创建  JAR\_WORK\_PATH=/var/jenkins\_home/workspace/meitedu\_springboot/target/  echo "查询进程id-->$SERVER\_NAME"  PID=`ps -ef | grep "$SERVER\_NAME" | awk '{print $2}'`  echo "得到进程ID：$PID"  echo "结束进程"  for id in $PID  do  kill -9 $id  echo "killed $id"  done  echo "结束进程完成"  #复制jar包到执行目录  echo "复制jar包到执行目录:cp $JAR\_PATH/$JAR\_NAME.jar $JAR\_WORK\_PATH"  cp $JAR\_PATH/$JAR\_NAME.jar $JAR\_WORK\_PATH  echo "复制jar包完成"  cd $JAR\_WORK\_PATH  #修改文件权限  chmod 755 $JAR\_NAME.jar  BUILD\_ID=dontKillMe nohup java -jar $JAR\_NAME.jar & |

# 容器映射8081端口

1. 重启容器

systemctl restart  docker

1. 清空未运行的容器

docker rm $(sudo docker ps -a -q)

|  |
| --- |
| docker run -p 8080:8080 -p 8087:8087 -p 50000:50000 -v jenkins\_data:/var/jenkins\_home jenkinsci/blueocean |

最主要是思路 灰度发布

# Docker常用命令

<http://ms.mayikt.com/detail/279>

docker run -p 8080:8080 -p 8081:8081 -p 50000:50000 -v jenkins\_data:/var/jenkins\_home jenkinsci/blueocean -v /usr/local/mayikt-springboot: /usr/local/mayikt-springboot

第五期改进知识点 深度会加深点 3万月薪

**第五期重点讲解服务网格**

**Vue+服务网格构建2020年最新微服务电商项目**

**秒杀、会员、消息发布、在线教育视频网站架构、蚂蚁课堂架构方案**