Docker+Jenkins+Maven

1. 安装Jenkins

1.1 查找镜像

```
docker search jenkins
```

1.2 拉取镜像

```
docker pull jenkinsci/blueocean:latest
```

1.3 创建容器

```
docker run -d --name jenkins \
 --restart=always \
 -p 8080:8080 \
 -p 50000:50000 jenkinsci/blueocean:latest
```

1.4 访问Jenkins

1.4.1 访问

打开浏览器输入 http://ip:8080/

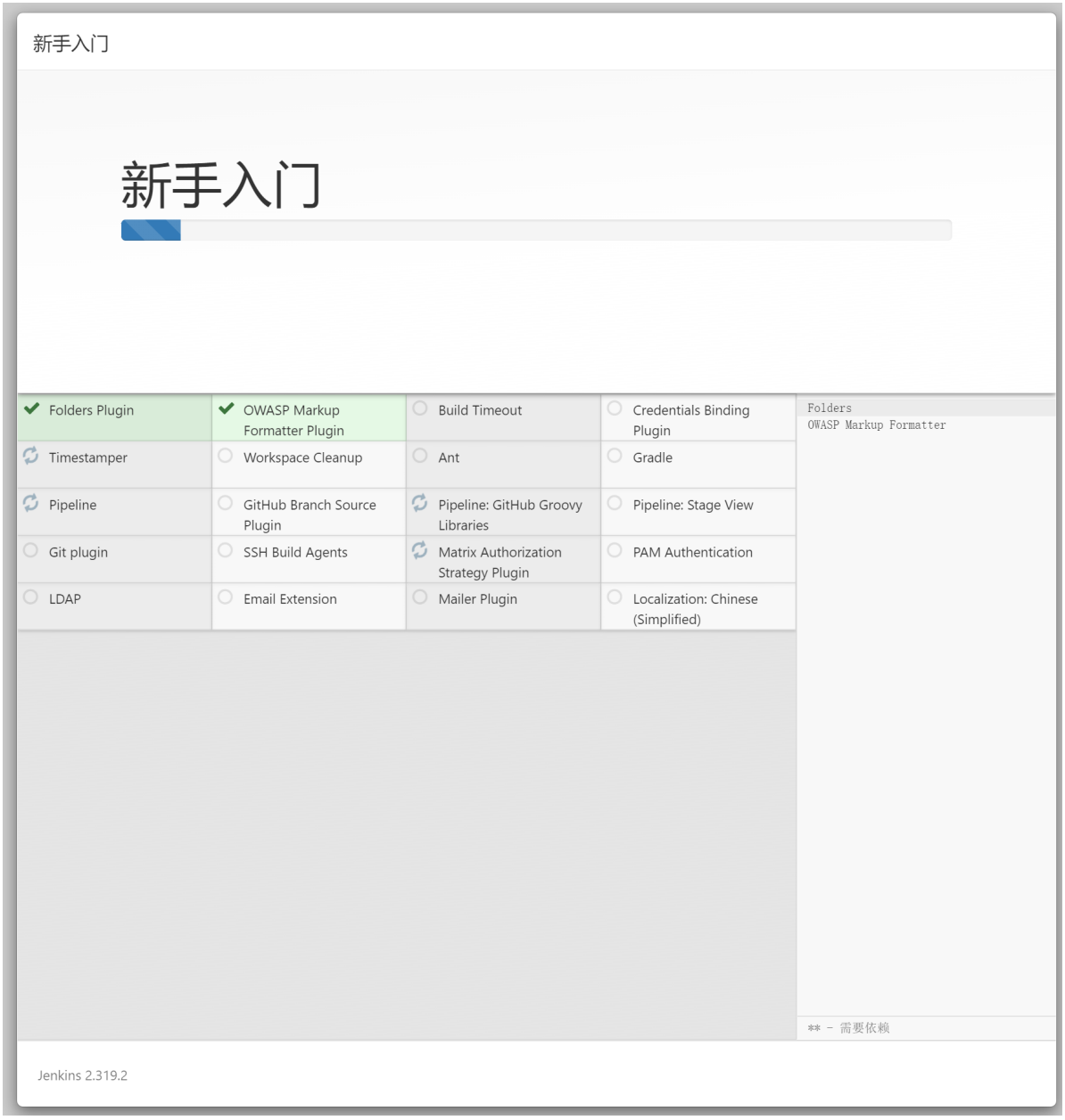
提示输入密码，密码路径 `/var/jenkins_home/secrets/initialAdminPassword`

```
# 查看密码
docker exec jenkins cat /var/jenkins_home/secrets/initialAdminPassword
```

1.4.2 进入插件安装页面


新手入门
自定义Jenkins
插件通过附加特性来扩展Jenkins以满足不同的需求。
安装推荐的插件
安装Jenkins社区推荐的插件。
选择插件来安装
选择并安装最适合的插件。
Jenkins 2.319.2

选择 安装推荐的插件


新手入门
新手入门
Folders Plugin
OWASP Markup Formatter Plugin
Build Timeout
Credentials Binding Plugin
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source Plugin
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git plugin
SSH Build Agents
Matrix Authorization Strategy Plugin
PAM Authentication
LDAP
Email Extension
Mailer Plugin
Localization: Chinese (Simplified)
Folders
OWASP Markup Formatter
** - 需要依赖
Jenkins 2.319.2

等待安装


新手入门
新手入门
Folders Plugin
OWASP Markup Formatter Plugin
Build Timeout
Credentials Binding Plugin
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source Plugin
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git plugin
SSH Build Agents
Matrix Authorization Strategy Plugin
PAM Authentication
LDAP
Email Extension
Mailer Plugin
Localization: Chinese (Simplified)
Folders
OWASP Markup Formatter
Build Timeout
Credentials Binding
Timestamper
** Resource Disposer
Workspace Cleanup
Ant
Gradle
** Pipeline: REST API
** JavaScript GUI Lib: Handlebars bundle
** JavaScript GUI Lib: Moment.js bundle
Pipeline: Stage View
** Lockable Resources
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
** jnr-posix API
PAM Authentication
LDAP
Email Extension
Mailer
** - 需要依赖
Jenkins 2.319.2

1.4.3创建管理员用户

新手入门

创建第一个管理员用户

用户名: admin

密码: •••••

确认密码: •••••

全名: admin

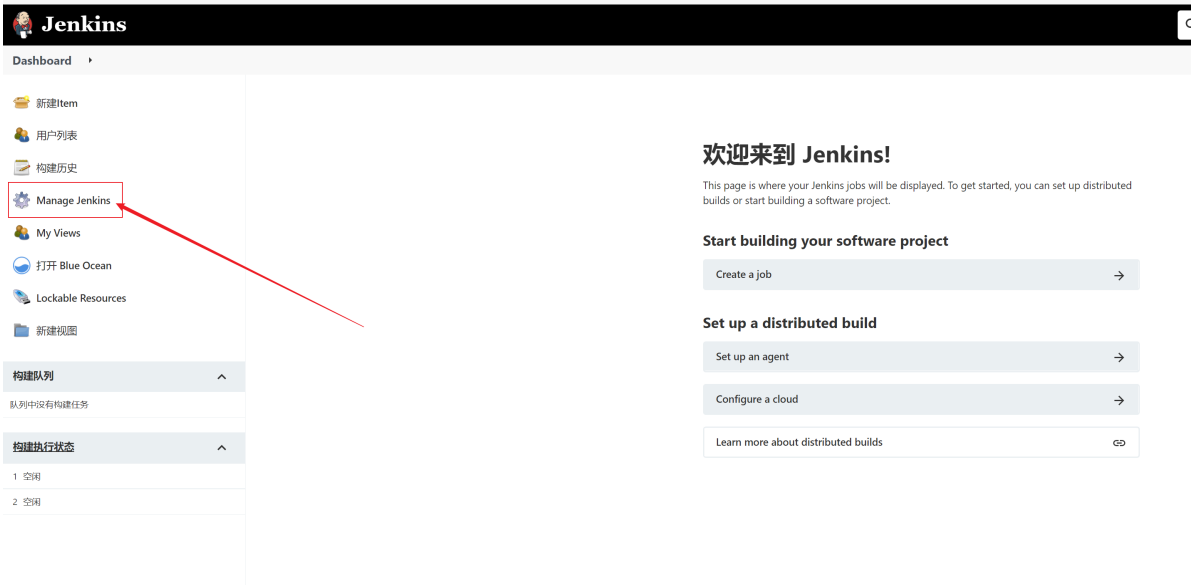
电子邮件地址: 10086@china.com

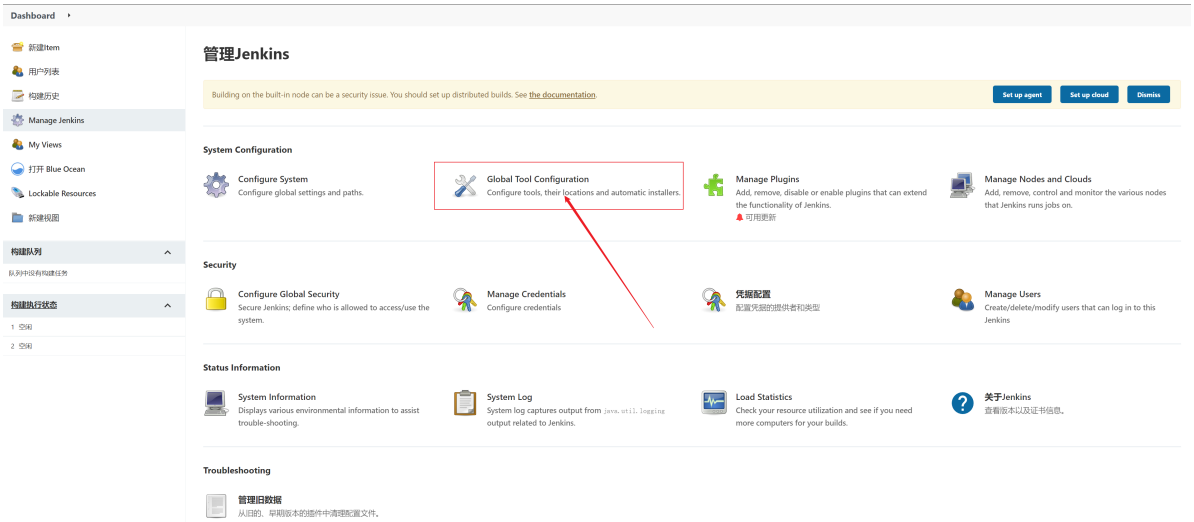
Jenkins 2.319.2

使用admin账户继续 保存并完成


新手入门
Jenkins已就绪！
Jenkins安装已完成。
开始使用Jenkins
Jenkins 2.319.2

2. 全局工具配置


Jenkins
Dashboard
新建Item
用户列表
构建历史
Manage Jenkins
My Views
打开 Blue Ocean
Lockable Resources
新建视图
构建队列
队列中没有构建任务
构建执行状态
1 空闲
2 空闲
欢迎来到 Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds

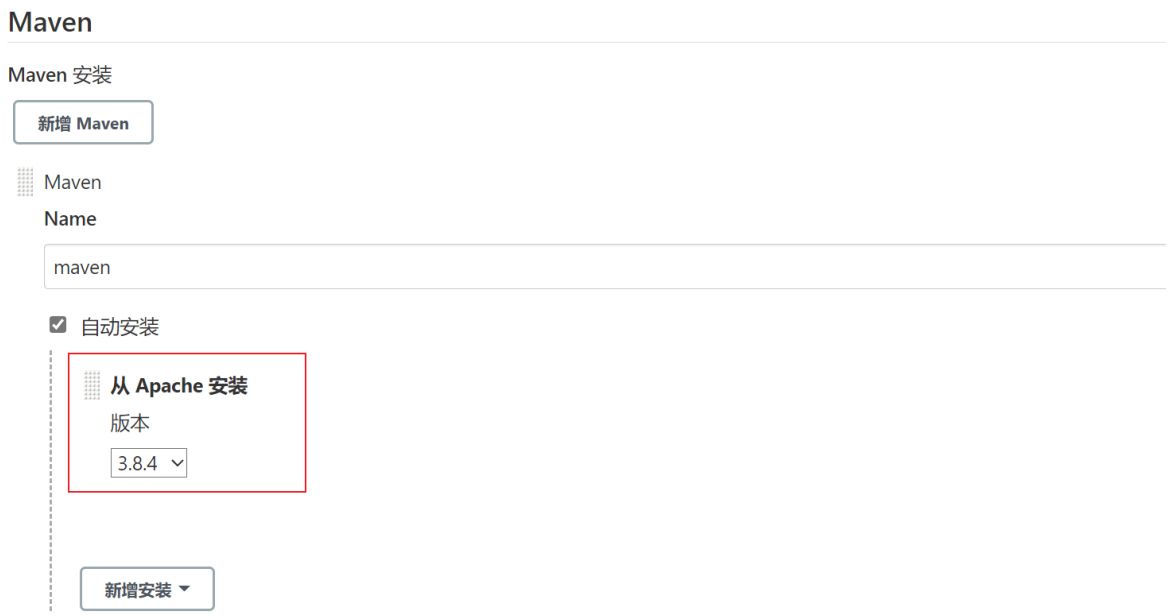


2.1 配置JDK

容器自带JDK,进入容器查看JDK路径

```
# 进入容器
docker exec -it -u root jenkins /bin/sh
# 查看jdk路径
echo $JAVA_HOME
# 查看JDK版本
java --version
```

2.2 配置Maven



配置完成之后 点击 保存

2.3 安装插件

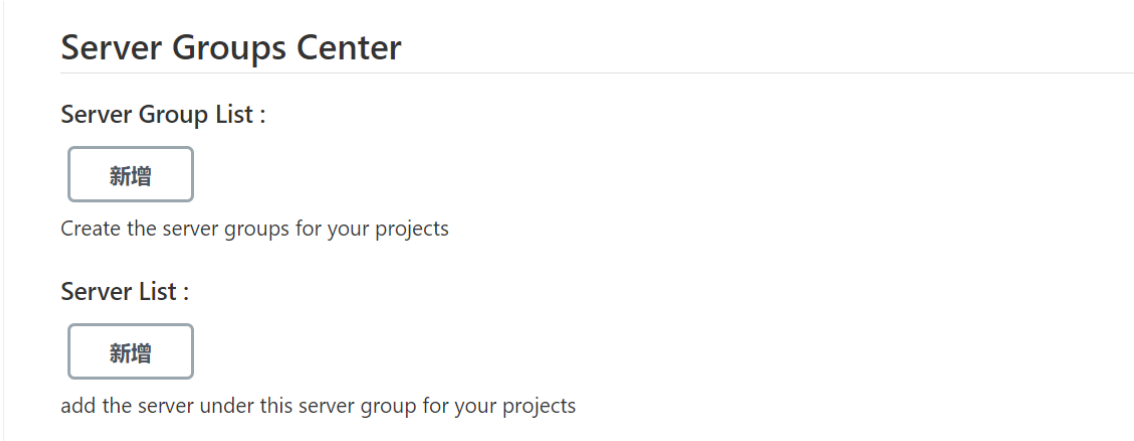
主要安装3个插件,搜索插件名勾选,然后点击下载

- Maven Integration (maven插件)
- Git Parameter(git参数定义插件)
- SSH2 Easy)(远程服务器上传文件及执行脚本插件)

> **SSH2 Easy在Jenkins上的存在bug**
>
> **com.jcraft.jsch:0.1.48 => com.jcraft.jsch:0.1.53**
>
> 可以手动修改 `com.jcraft.jsch:0.1.48 => com.jcraft.jsch:0.1.53` 然后打包成hpi上传插件的方式安装

安装完成之后重启Jenkins

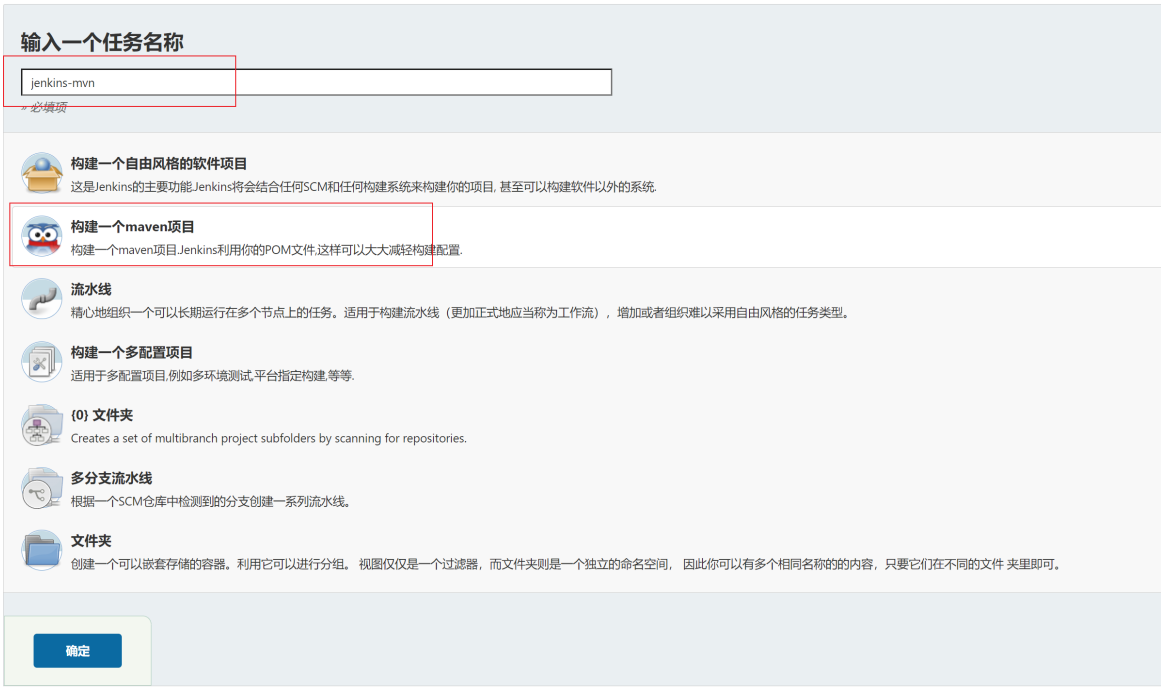
3.配置远程服务器ssh连接



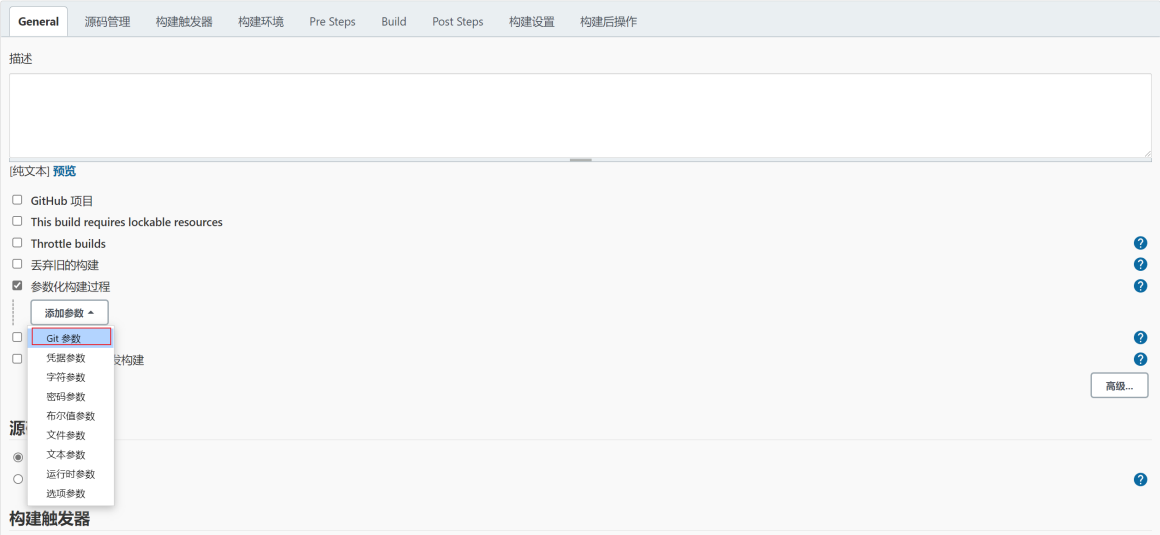
新增组和服务器

4.创建Maven项目

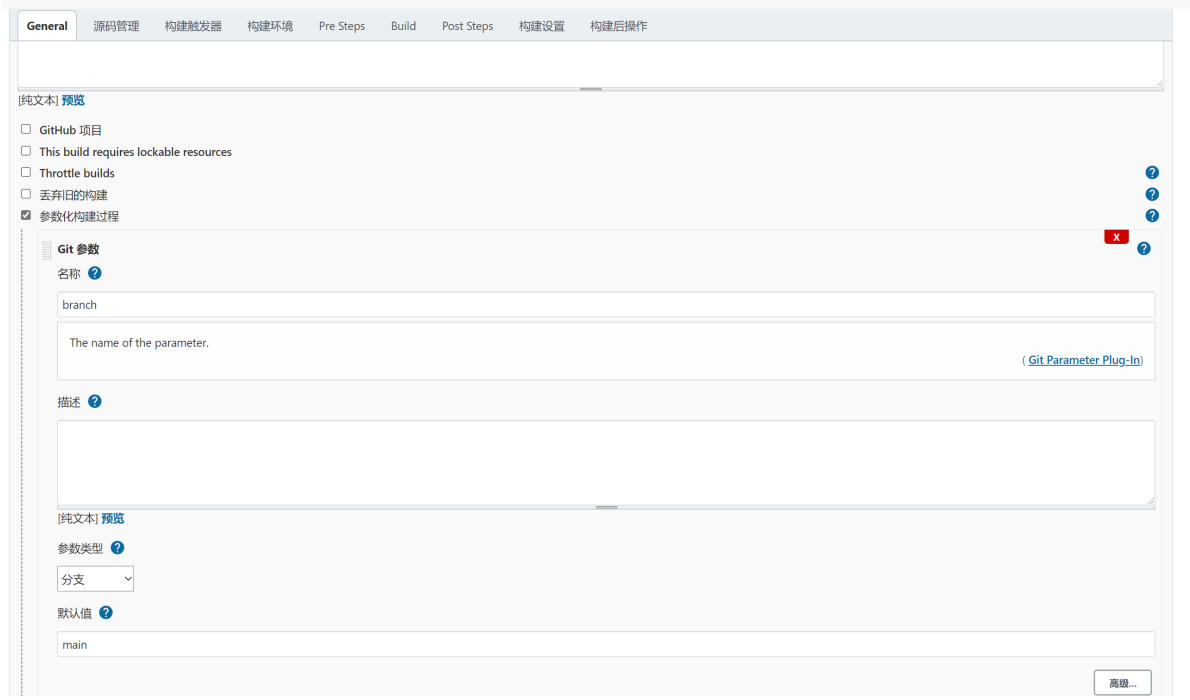
4.1创建Maven项目

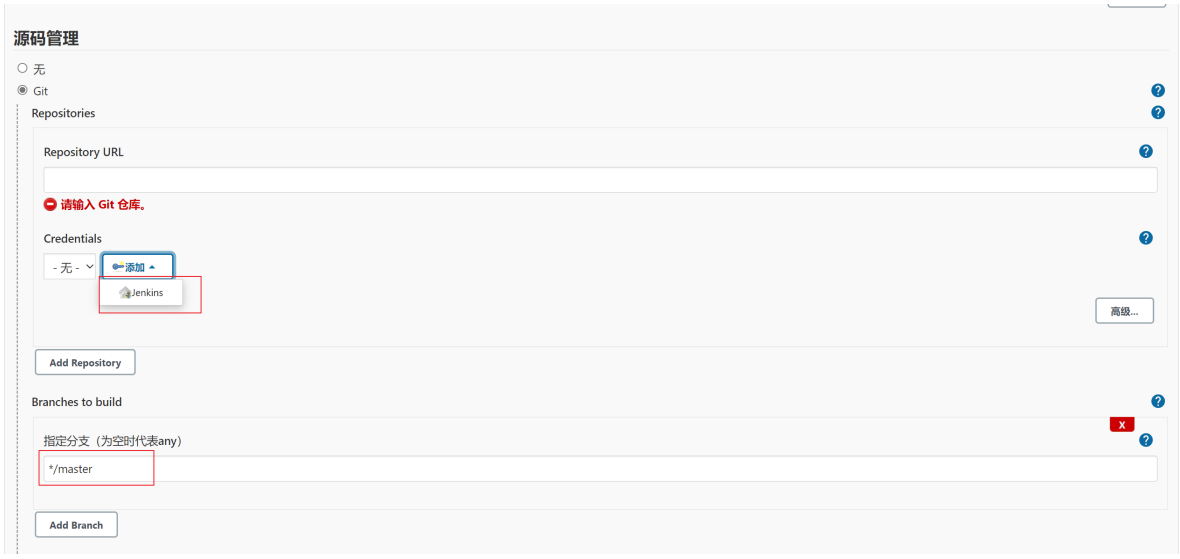


4.2配置git参数

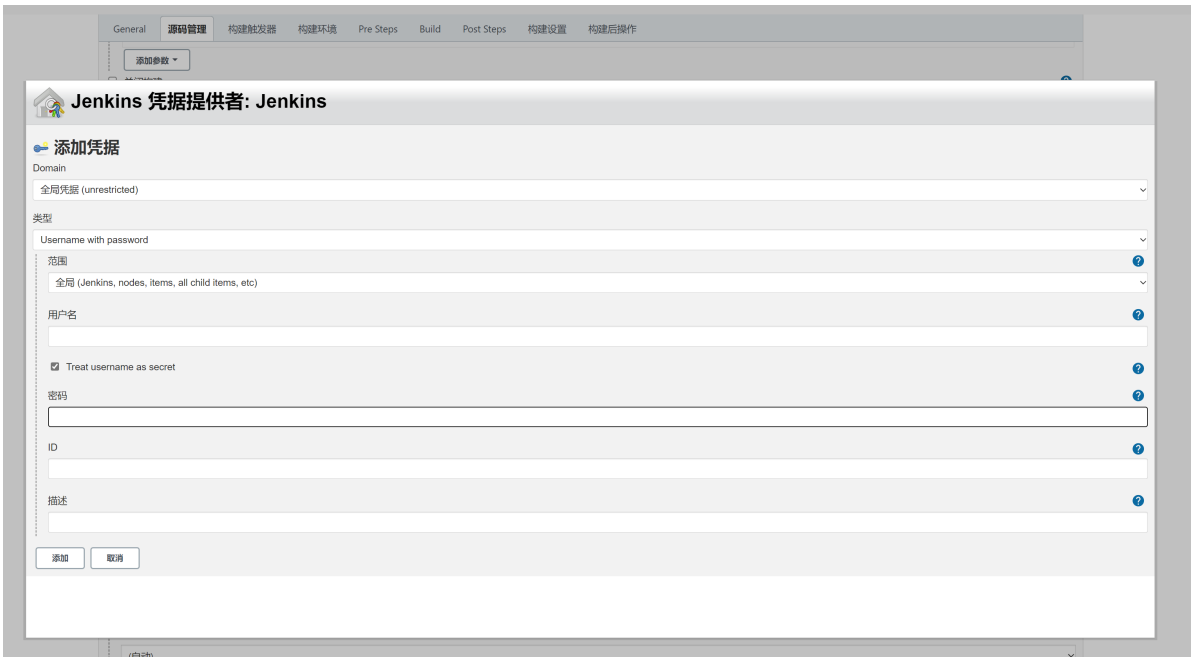


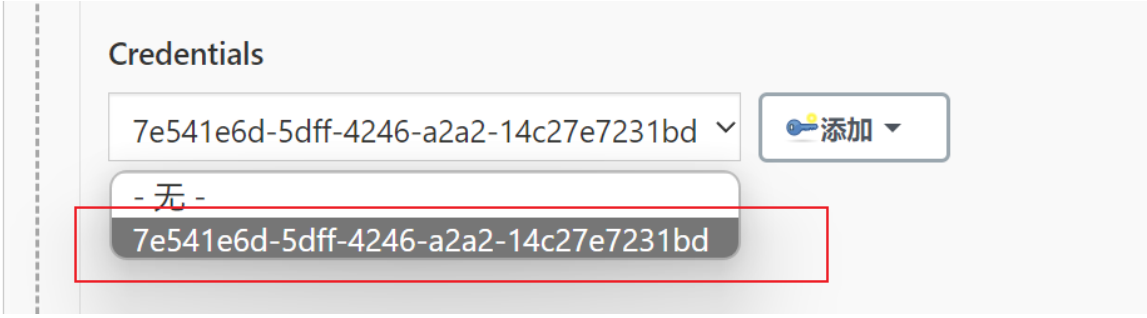
配置git

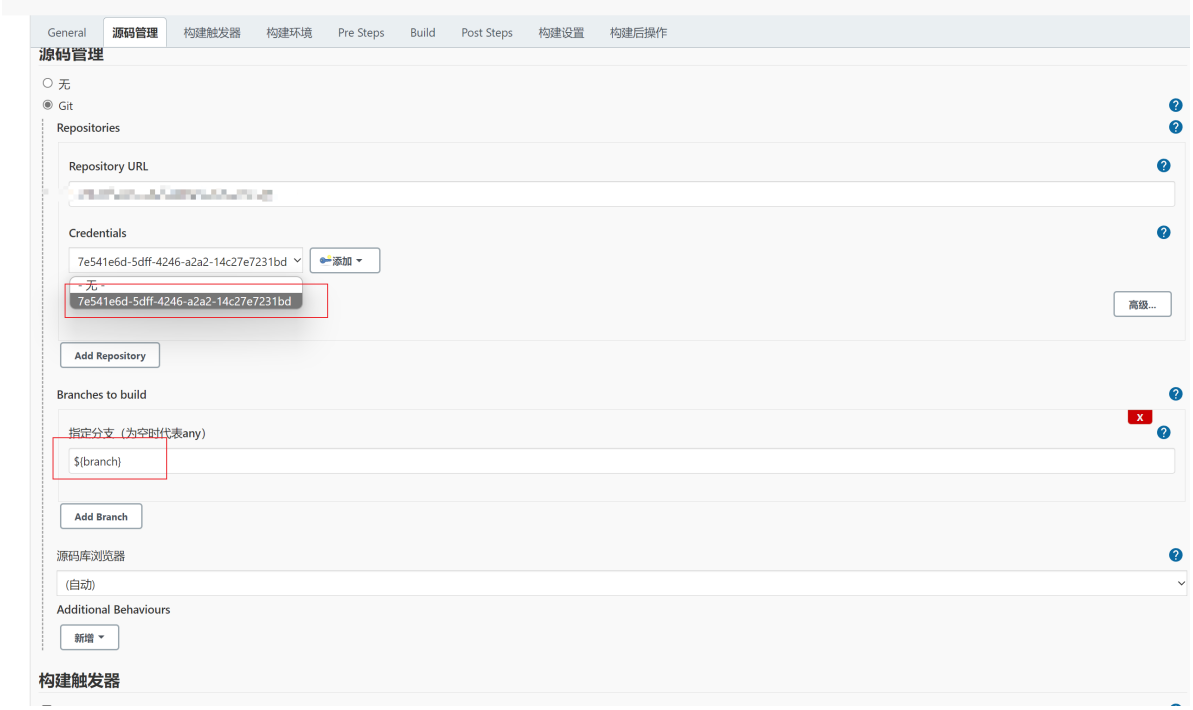




配置git远程仓库账号







4.3 配置Maven打包命令

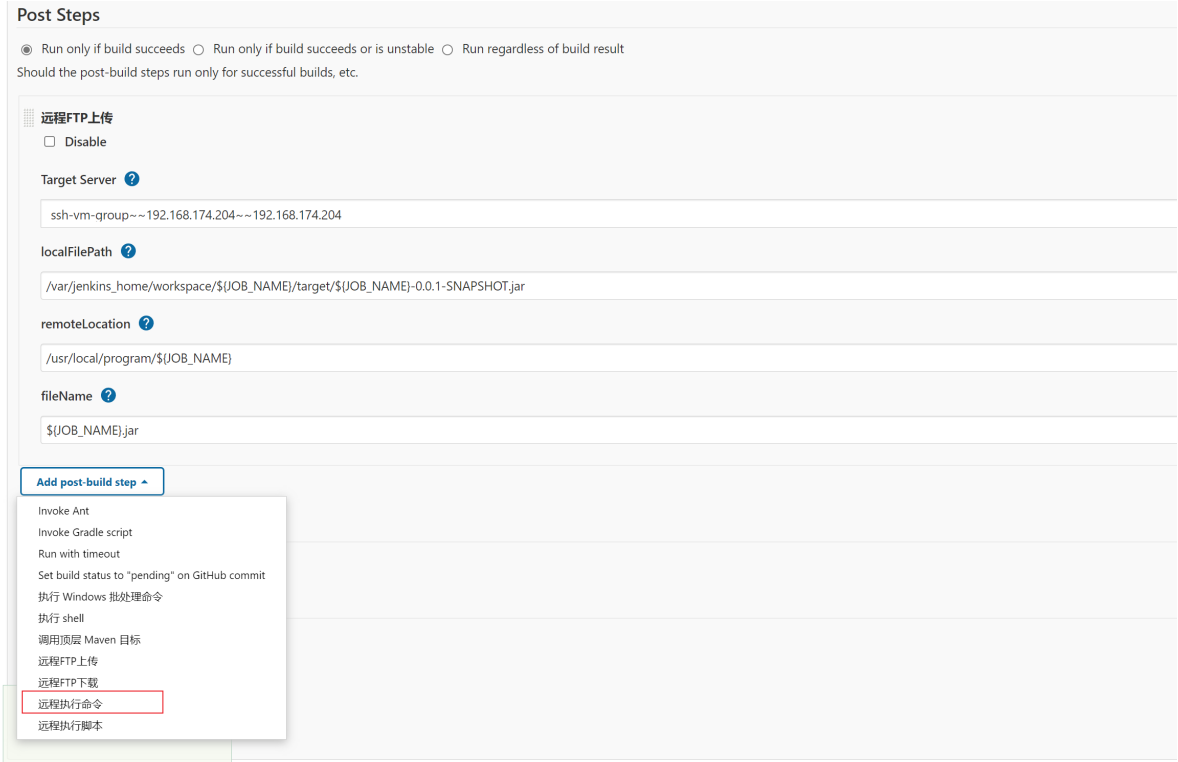
```
clean package -Dmaven.prod.skip=true -e
```

4.5 配置jar包上传指令

上传Jar包配置

注意,Jenkins打包好的文件在 `/var/jenkins_home/workspace/` 目录下,进入容器可以查看

`${JOB_NAME}` 是Jenkins的预设参数,值为项目名



4.6 执行远程命令

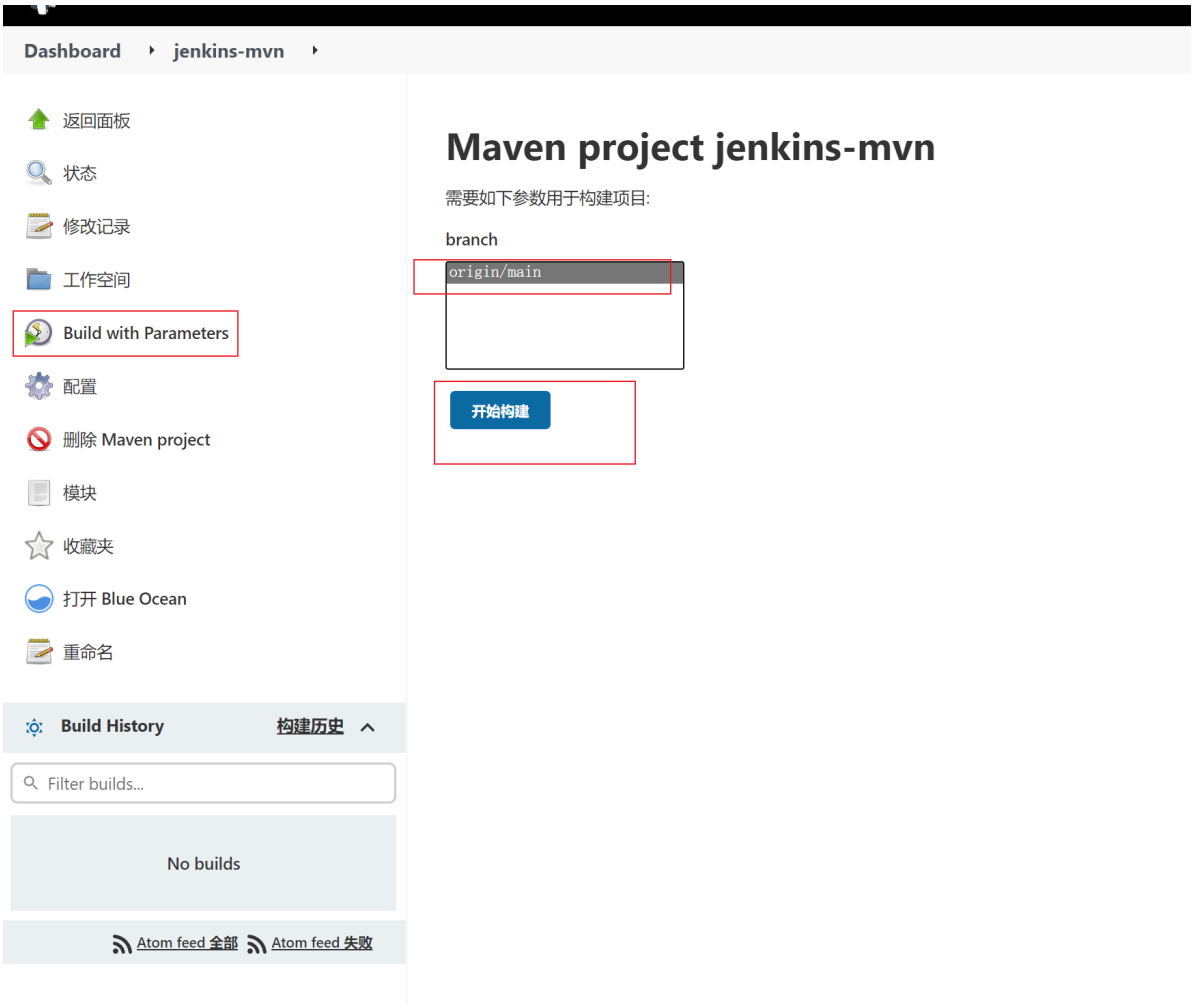
app.sh

```
#!/bin/bash


PROJECT_NAME=jenkins-mvn
# 日志前缀: jenkins-mvn-2022-02-02.log
LOG_PREFIX=/usr/local/program/${PROJECT_NAME}/logs/${PROJECT_NAME}/${PROJECT_NAME}-
# jar包路径
JAR_PATH=/usr/local/program/${PROJECT_NAME}/${PROJECT_NAME}.jar
# spring boot配置
ACTIVE=dev

usage() {
  echo "Usage: sh ShellName.sh [start|stop|restart|status]"
  exit 1
}

file_is_exist() {
  if [ -f "${JAR_PATH}" ]; then
    return 1
  else
    echo "${JAR_PATH} is not exist"
    exit 0
  fi
}
# shellcheck disable=SC2120
is_run() {
  file_is_exist
  #echo "ps -ef|grep ${JAR_PATH}|grep -v grep|awk '{print $2}' "
  pid=$(ps -ef | grep ${JAR_PATH} | grep -v grep | awk '{print $2}')
  if [ -z "${pid}" ]; then
    return 1
  else
    return 0
  fi
}

start() {
  is_run
  if [ $? -eq "0" ]; then
    echo "${JAR_PATH} is already running. pid = ${pid} ."
```

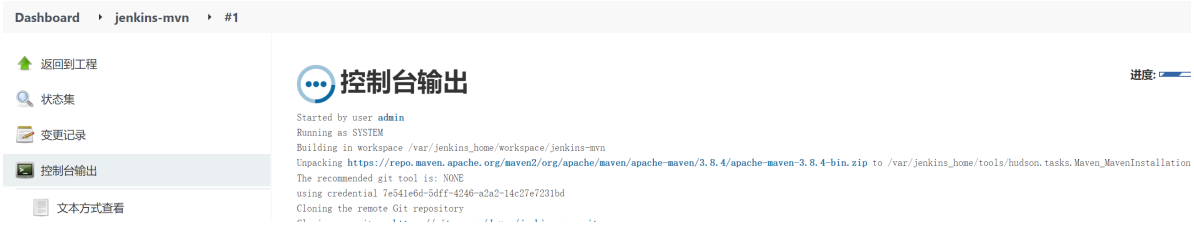
```
  else
    # No default log is generated. Use logback or log4j2
    echo " nohup java -Xms512m -Xmx1024m -XX:+HeapDumpOnOutOfMemoryError \ "
    echo "-XX:HeapDumpPath=./ -jar  \  "
    echo "-Dspring.profiles.active=${ACTIVE} \ "
    echo "${JAR_PATH} >/dev/null 2>&1 & "

    nohup java -Xms512m -Xmx1024m -XX:+HeapDumpOnOutOfMemoryError \
      -XX:HeapDumpPath=./  -jar \
      -Dspring.profiles.active=${ACTIVE} \
      ${JAR_PATH} >/dev/null 2>&1 &
    now_date=$(date +%Y-%m-%d)
    echo "If you need to view the log, please execute ' tail -20f
${LOG_PREFIX}${now_date}.log '"
#    sleep 3s
#    echo "tail -20f ${LOG_PREFIX}${now_date}.log"
#    tail -20f ${LOG_PREFIX}${now_date}.log
  fi
}

stop() {
  is_run
  if [ $? -eq "0" ]; then
    echo "kill -9 ${pid}"
    kill -9 $pid
    echo "${pid} Stopping."
  fi
  echo "${JAR_PATH} is NOT running."
}
parameter() {
  echo "jinfo -flags ${pid}"
  jinfo -flags ${pid}
}
status() {
  is_run
  if [ $? -eq "0" ]; then
    echo "${JAR_PATH} is running. Pid is ${pid}"
    parameter
  else
    echo "${JAR_PATH} is NOT running."
  fi
}
restart() {
  stop
  start
}

case "$1" in
"start")
  start
  ;;
"stop")
  stop
  ;;
"status")
  status
  ;;
"restart")
```

```
  restart
  ;;
*)
  usage
  ;;
esac
```

4.7构建测试



控制台查看日志



tips

执行脚本远程服务器命令行，拓展一下，可以获取当前项目版本号，然后打个tag到git仓库，以及发布成功之后钉钉机器人推送等。