

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

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

<input checked="" type="checkbox"/> Folders Plugin	<input checked="" type="checkbox"/> OWASP Markup Formatter Plugin	<input type="checkbox"/> Build Timeout	<input type="checkbox"/> Credentials Binding Plugin	Folders OWASP Markup Formatter
<input checked="" type="checkbox"/> Timestampers	<input type="checkbox"/> Workspace Cleanup	<input type="checkbox"/> Ant	<input type="checkbox"/> Gradle	
<input checked="" type="checkbox"/> Pipeline	<input type="checkbox"/> GitHub Branch Source Plugin	<input checked="" type="checkbox"/> Pipeline: GitHub Groovy Libraries	<input type="checkbox"/> Pipeline: Stage View	
<input type="checkbox"/> Git plugin	<input type="checkbox"/> SSH Build Agents	<input checked="" type="checkbox"/> Matrix Authorization Strategy Plugin	<input type="checkbox"/> PAM Authentication	
<input type="checkbox"/> LDAP	<input type="checkbox"/> Email Extension	<input type="checkbox"/> Mailer Plugin	<input type="checkbox"/> Localization: Chinese (Simplified)	
				** - 需要依赖

等待安装

新手入门

✓ Folders Plugin	✓ OWASP Markup Formatter Plugin	✓ Build Timeout	✓ Credentials Binding Plugin	Folders
✓ Timestampers	✓ Workspace Cleanup	✓ Ant	✓ Gradle	OWASP Markup Formatter
✓ Pipeline	✓ GitHub Branch Source Plugin	✓ Pipeline: GitHub Groovy Libraries	✓ Pipeline: Stage View	Build Timeout
✓ Git plugin	✓ SSH Build Agents	✓ Matrix Authorization Strategy Plugin	✓ PAM Authentication	Credentials Binding
✓ LDAP	✓ Email Extension	✓ Mailer Plugin	<input type="radio"/> Localization: Chinese (Simplified)	Timestampers
				** 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

1.4.3创建管理员用户

创建第一个管理员用户

用户名:	<input type="text" value="admin"/>
密码:	<input type="password" value="....."/>
确认密码:	<input type="password" value="....."/>
全名:	<input type="text" value="admin"/>
电子邮件地址:	<input type="text" value="10086@china.com"/>


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

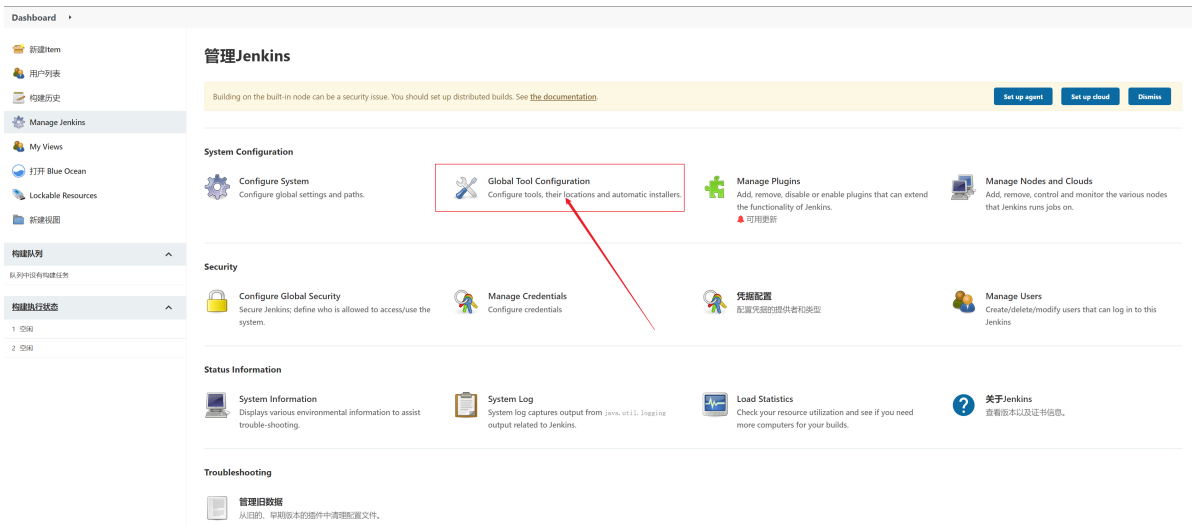
Create a job

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
```

JDK

JDK 安装

新增 JDK



JDK

别名

jdk

JAVA_HOME

/opt/java/openjdk

☐ 自动安装

2.2 配置Maven

Maven

Maven 安装

新增 Maven

Maven

Name

maven

☒ 自动安装

从 Apache 安装

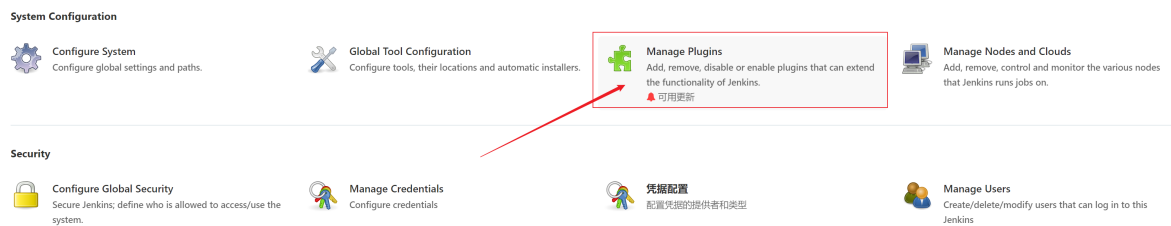
版本

3.8.4

新增安装

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

 新建任务


 用户列表


 构建历史

 系统管理

 我的视图

 打开 Blue Ocean

 Lockable Resources

 新建视图

管理Jenkins

Building on the built-in node can be a security issue. You should set up

系统配置



系统配置

配置全局设置和路径

Server Groups Center

Server Group List :

新增

Create the server groups for your projects

Server List :

新增

add the server under this server group for your projects

新增组和服务端

Server Groups Center

Server Group List :

Group Name	<input type="text" value="ssh-vm-group"/>	?
SSH Port	<input type="text" value="22"/>	?
User Name	<input type="text" value="root"/>	?
Password	<input type="password" value="....."/>	?

新增

Create the server groups for your projects

Server List :

Server Group:	<input type="text" value="ssh-vm-group"/>	?
Server Name	<input type="text" value="192.168.174.204"/>	?
Server IP	<input type="text" value="192.168.174.204"/>	?

新增

add the server under this server group for your projects

新增组之后需要先保存一下才能下拉

4.创建Maven项目

4.1创建Maven项目

输入一个任务名称

jenkins-mvn

必填项

构建一个自由风格的软件项目
这是Jenkins的主要功能.Jenkins将会结合任何SCM和任何构建系统来构建你的项目, 甚至可以构建软件以外的系统。

构建一个maven项目
构建一个maven项目.Jenkins利用你的POM文件.这样可以大大减轻构建配置。

流水线
精心地组织一个可以长期运行在多个节点上的任务。适用于构建流水线（更加正式地应当称为工作流），增加或者组织难以采用自由风格的任务类型。

构建一个多配置项目
适用于多配置项目,例如多环境测试,平台指定构建,等等。

(0) 文件夹
Creates a set of multibranch project subfolders by scanning for repositories.

多分支流水线
根据一个SCM仓库中检测到的分支创建一系列流水线。

文件夹
创建一个可以嵌套存储的容器。利用它可以进行分组。视图仅仅是一个过滤器，而文件夹则是一个独立的命名空间，因此你可以有多个相同名称的内容，只要它们在不同的文件 夹里即可。

确定

4.2配置git参数

General 源码管理 构建触发器 构建环境 Pre Steps Build Post Steps 构建设置 构建后操作

描述

[纯文本] 预览

☐ GitHub 项目

☐ This build requires lockable resources

☐ Throttle builds

☐ 丢弃旧的构建

☒ 参数化构建过程

添加参数

Git 参数

凭据参数

字符参数

密码参数

布尔值参数

文件参数

文本参数

运行时参数

选项参数

源

☒ 源码管理

☐ 构建后操作

构建触发器

高级...

配置git

General

源码管理

构建触发器

构建环境

Pre Steps

Build

Post Steps

构建设置

构建后操作

[纯文本] 预览

☐ GitHub 项目

☐ This build requires lockable resources

☐ Throttle builds

☐ 丢弃旧的构建

☒ 参数化构建过程

Git 参数

名称

?

branch

The name of the parameter.

(Git Parameter Plug-In)

描述

?

[纯文本] 预览

参数类型

?

分支

默认值

?

main

高级...

源码管理

☐ 无

☒ Git

Repositories

?

Repository URL

?

请输入 Git 仓库。

Credentials

?

- 无 -

添加

Jenkins

高级...

Add Repository

Branches to build

?

指定分支 (为空时代表any)

*/master

高级...

Add Branch

配置git远程仓库账号

General

源码管理

构建触发器

构建环境

Pre Steps

Build

Post Steps

构建设置

构建后操作

Jenkins 凭据提供者: Jenkins

添加凭据

Domain

全局凭据 (unrestricted)

类型

Username with password

范围

全局 (Jenkins, nodes, items, all child items, etc)

用户名

☐ Treat username as secret

密码

ID

描述

添加

取消

Credentials

7e541e6d-5dff-4246-a2a2-14c27e7231bd ▼

添加 ▼

- 无 -

7e541e6d-5dff-4246-a2a2-14c27e7231bd

General 源码管理 构建触发器 构建环境 Pre Steps Build Post Steps 构建设置 构建后操作

源码管理

☐ 无

☒ Git

Repositories

Repository URL

Credentials

7e541e6d-5dff-4246-a2a2-14c27e7231bd ▼ 添加 ▼

- 无 -

7e541e6d-5dff-4246-a2a2-14c27e7231bd

高级...

Add Repository

Branches to build

指定分支 (为空时代表any)

\$(branch)

Add Branch

源码库浏览器

(自动)

Additional Behaviours

新增 ▼

构建触发器

4.3 配置Maven打包命令

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

Pre Steps

Add pre-build step ▼

Build

Root POM

pom.xml

Goals and options

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

Maven打包命令

4.5 配置jar包上传指令



上传Jar包配置

Post Steps

☒ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☐ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

远传FTP上传

☐ Disable

Target Server ?

ssh-vm-group--192.168.174.204--192.168.174.204

localFilePath ?

/var/jenkins_home/workspace/\${JOB_NAME}/target/\${JOB_NAME}-0.0.1-SNAPSHOT.jar

remoteLocation ?

/usr/local/program/\${JOB_NAME}

fileName ?

\${JOB_NAME}.jar

Add post-build step v

\$(JOB_NAME) jenkins的预设参数,值为项目名

目标服务器, 提前在系统配置中配好

打包之后jar的路径

目标服务器路径, 需提前创建好文件夹

自定义上传后的文件名, 可不填

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

Post Steps

☒ Run only if build succeeds ☐ Run only if build succeeds or is unstable ☐ Run regardless of build result

Should the post-build steps run only for successful builds, etc.

远传FTP上传

☐ Disable

Target Server ?

ssh-vm-group--192.168.174.204--192.168.174.204

localFilePath ?

/var/jenkins_home/workspace/\${JOB_NAME}/target/\${JOB_NAME}-0.0.1-SNAPSHOT.jar

remoteLocation ?

/usr/local/program/\${JOB_NAME}

fileName ?

\${JOB_NAME}.jar

Add post-build step ^

- Invoke Ant
- Invoke Gradle script
- Run with timeout
- Set build status to "pending" on GitHub commit
- 执行 Windows 批处理命令
- 执行 shell
- 调用顶层 Maven 目标
- 远传FTP上传
- 远传FTP下载
- 远传执行命令**
- 远传执行脚本

4.6 执行远程命令

远程执行命令

☐ Disable

Target Server ?

vm~~204~~192.168.174.204

commands ?

cd /usr/local/program/jenkins-mvn/
sh app.sh restart

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} ."
    fi
}
```

```

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

Dashboardjenkins-mvn

返回面板

状态

修改记录

workspace

Build with Parameters

配置

删除 Maven project

模块

收藏夹

打开 Blue Ocean

重命名

Build History构建历史

Filter builds...

No builds

Atom feed 全部Atom feed 失败

Maven project jenkins-mvn

需要如下参数用于构建项目:

branch

origin/main

开始构建

控制台查看日志

Dashboardjenkins-mvn#1

返回到工程

状态集

变更记录

控制台输出

文本方式查看

控制台输出

Started by user admin

Running as SYSTEM

Building in workspace /var/jenkins_home/workspace/jenkins-mvn

Unpacking https://repo.maven.apache.org/maven2/org/apache/maven/apache-maven/3.8.4/apache-maven-3.8.4-bin.zip to /var/jenkins_home/tools/hudson.tasks.Maven_MavenInstallation

The recommended git tool is: NONE

using credential 7e541e6d-5d6f-4246-a2a2-14c27e7231bd

Cloning the remote Git repository

进度: 100%

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。

