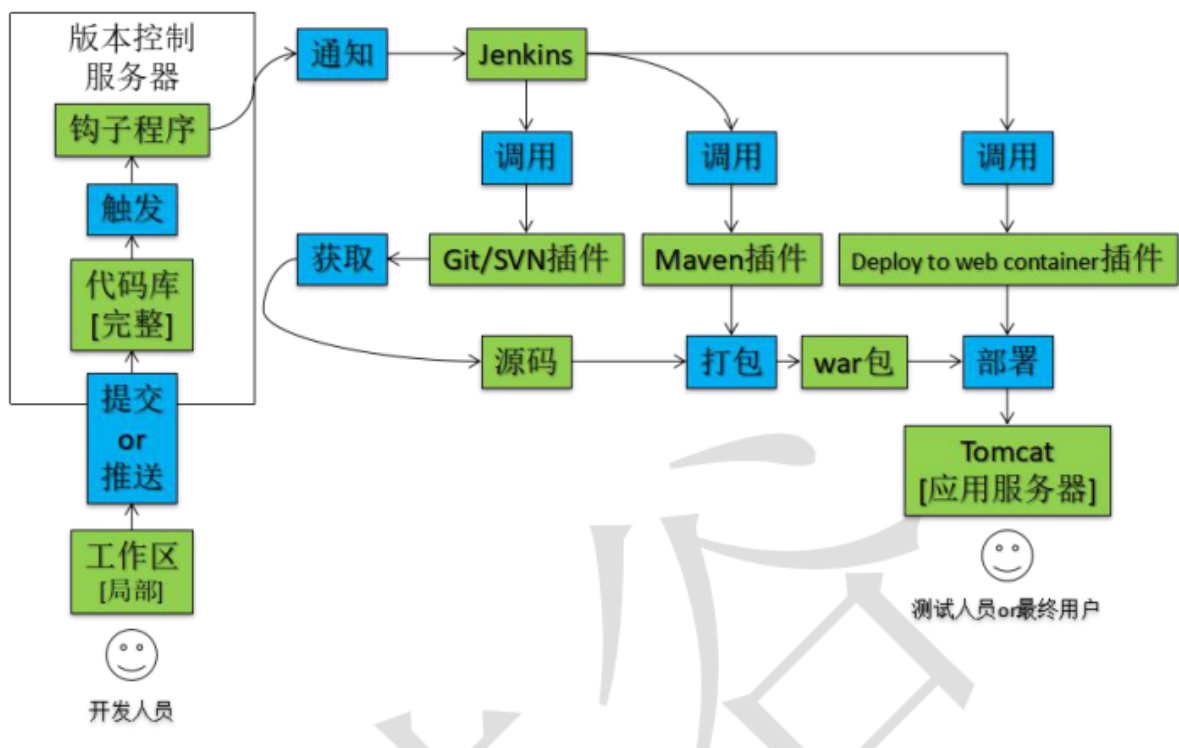


Jenkins部署手册

自动化部署流程图：



环境准备说明

- 虚拟机Linux系统
- 代码版本控制工具
 - Subversion服务器
 - 版本库钩子程序
- 持续集成系统（Jenkins）
 - JDK
 - Tomcat
 - Maven
 - jenkins
 - 主体程序
 - SVN插件
 - Maven插件
 - Deploy to Web Container 插件
- 应用发布子系统
 - JDK
 - Tomcat

环境准备工作

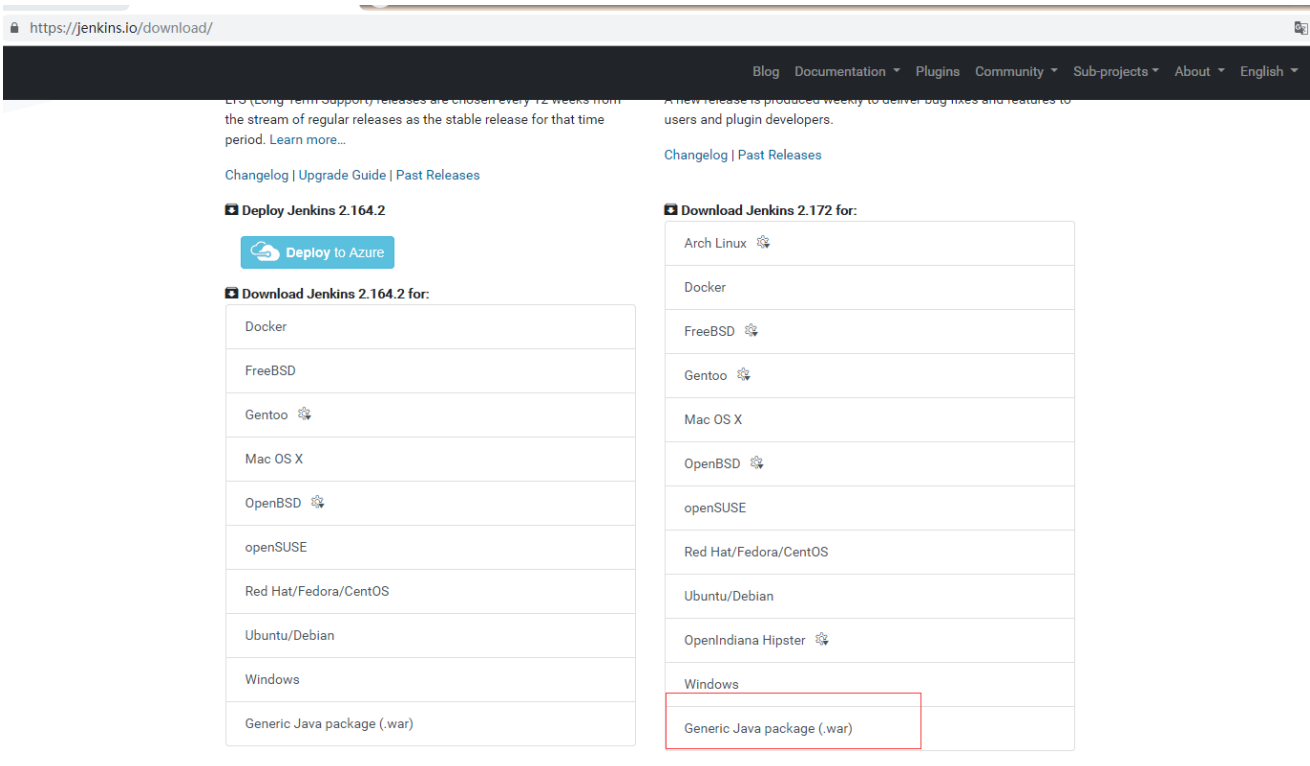
一、JDK、Maven准备

具体配置不详细说明。以下是安装完成的检测。

```
[root@zxxh opt]# java -version
java version "1.8.0_151"
Java(TM) SE Runtime Environment (build 1.8.0_151-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.151-b12, mixed mode)
[root@zxxh opt]# mvn -v
Apache Maven 3.5.2 (138ed61fd100ec658bfa2d307c43b76940a5d7d; 2017-10-18T15:58:13+08:00)
Maven home: /usr/local/maven/maven3.5
Java version: 1.8.0_151, vendor: Oracle Corporation
Java home: /usr/local/java8/jdk1.8/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "2.6.32-696.el6.x86_64", arch: "amd64", family: "unix"
[root@zxxh opt]#
```

二、Jenkins下载

下载地址：<https://jenkins.io/download/>



这里下载的是war

三、Tomcat准备并配置Jenkins

tomcat_8088 : Jenkins主体程序是war项目，放在tomcat服务器上运行。

tomcat_8089 : 应用发布的tomcat。

应用发布子系统tomcat配置

这里tomcat是tomcat_8089

Tomcat 服务器的账号密码

修改文件： /opt/tomcat_8089/conf/tomcat-users.xml

```
[root@zxh opt]# cd /opt/tomcat_8089/conf
[root@zxh conf]# ls -l
total 228
drwxr-x---. 3 root root 4096 Apr 15 01:05 Catalina
-rw-----. 1 root root 13548 Jun 21 2018 catalina.policy
-rw-----. 1 root root 7576 Jun 21 2018 catalina.properties
-rw-----. 1 root root 1338 Jun 21 2018 context.xml
-rw-----. 1 root root 1149 Jun 21 2018 jaspic-providers.xml
-rw-----. 1 root root 2313 Jun 21 2018 jaspic-providers.xsd
-rw-----. 1 root root 3622 Jun 21 2018 logging.properties
-rw-----. 1 root root 7511 Apr 15 01:03 server.xml
-rw-----. 1 root root 2164 Jun 21 2018 tomcat-users.xml
-rw-----. 1 root root 2633 Jun 21 2018 tomcat-users.xsd
-rw-----. 1 root root 169322 Jun 21 2018 web.xml
[root@zxh conf]#
```

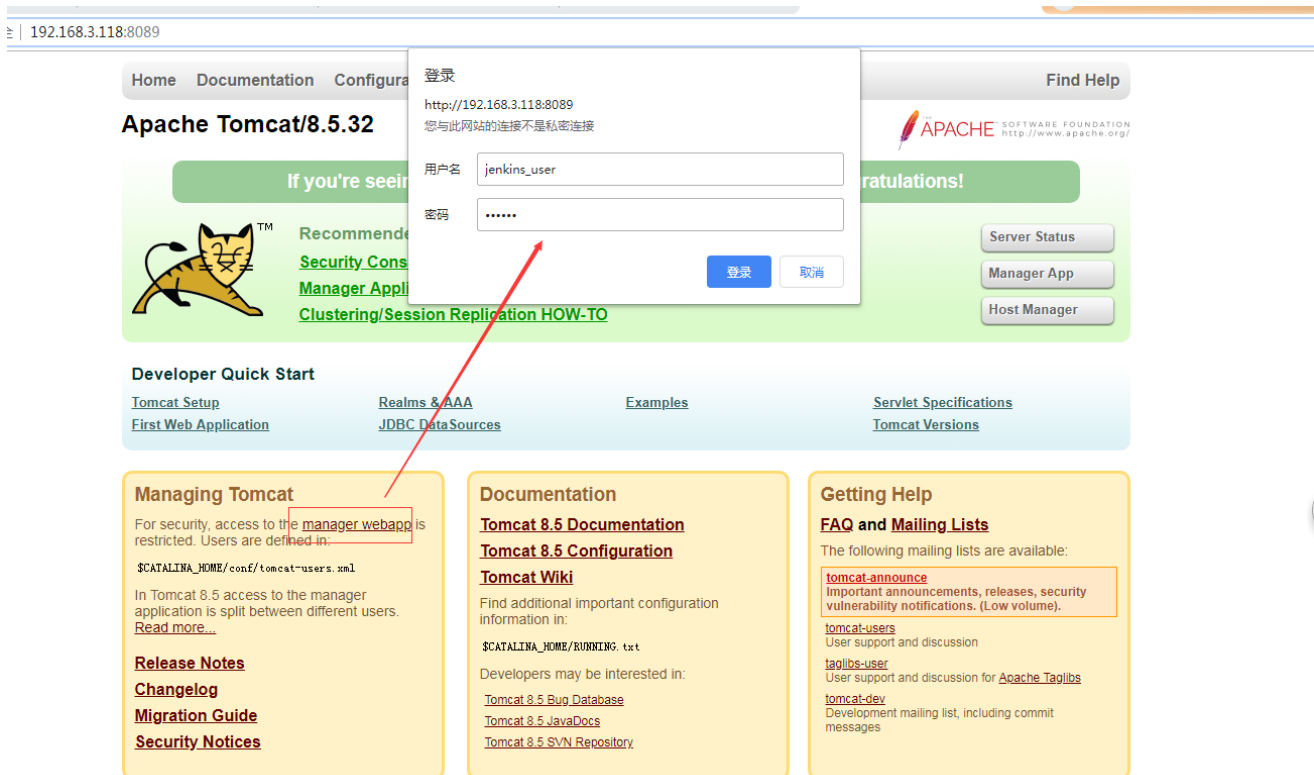
修改这文件

增加如下角色

```
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<user username="jenkins_user" password="123456" roles="manager-gui,manager-script,manager-
jmx,manager-status"/>
```

```
them. You will also need to set the passwords to something appropriate.
-->
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
<user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<user username="jenkins_user" password="123456" roles="manager-gui,manager-script,manager-jmx,manager-status"/>
</tomcat-users>
```

启动tomcat，访问manager_webapp 可以登录即可



如果403错误，参考如下解决方案：

<https://blog.csdn.net/w770583069/article/details/76084863/>

1、Jenkins 主体程序安装配置

这里tomcat为tomcat_8088

1.1、把jenkins.war 放在 Tomcat 的/webapps 目录下

1.2、修改编解码字符集

在Tomcat的/server.xml 修改URL 地址的编解码字符集：URIEncoding="UTF-8"

```
<Connector port="8088" protocol="HTTP/1.1"
    connectionTimeout="20000"
    redirectPort="8443" URIEncoding="UTF-8"/>
```

1.3、启动Tomcat，访问jenkins程序

访问地址：<http://192.168.3.118:8088/jenkins>

入门

解锁 Jenkins

为了确保管理员安全地安装 Jenkins，密码已写入到日志中（[不知道在哪里？](#)）该文件在服务器上：

```
/root/.jenkins/secrets/initialAdminPassword
```

请从本地复制密码并粘贴到下面。

管理员密码

1.4、解锁Jenkins

按照提示可知密码存放在 `/root/.jenkins/secrets/initialAdminPassword` 文件中。

```
# cat /root/.jenkins/secrets/initialAdminPassword
```

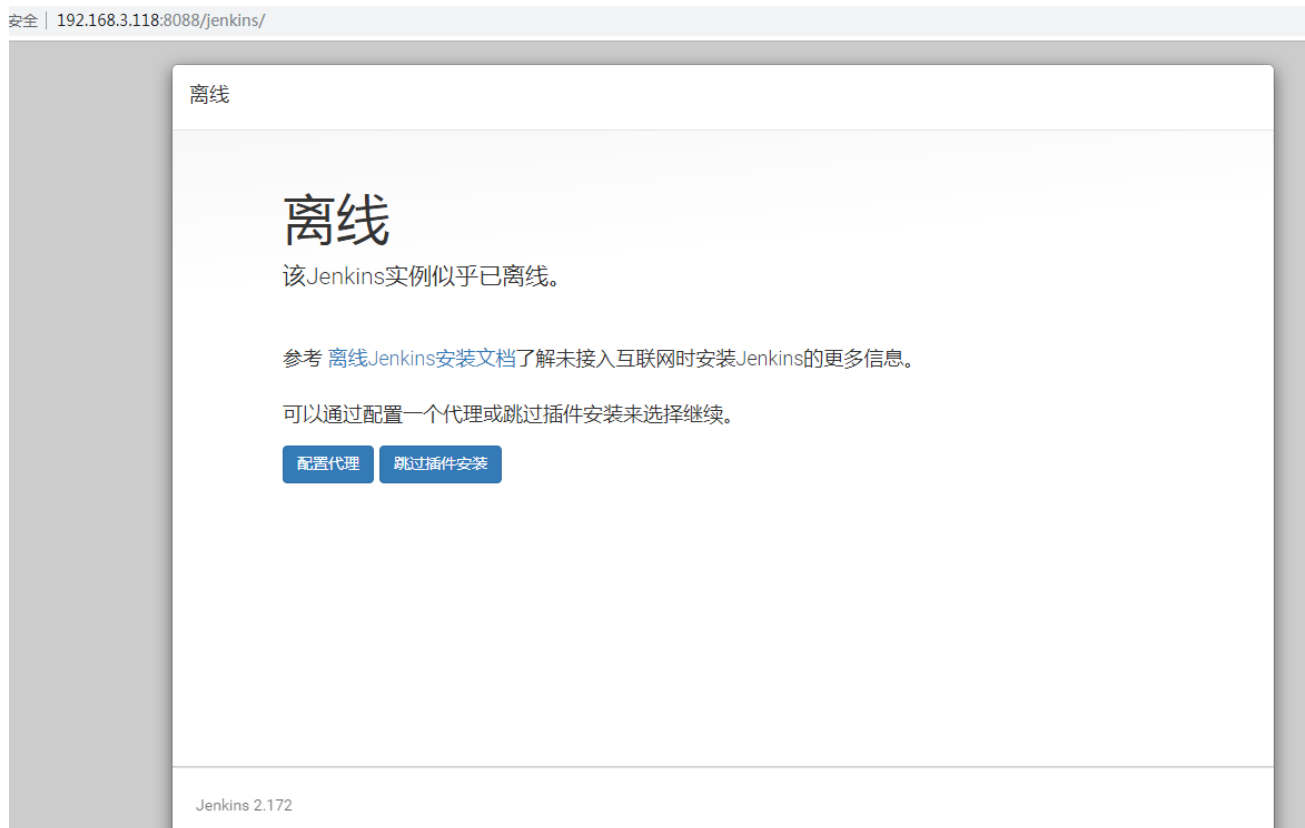
```
[root@zxh bin]# cat /root/.jenkins/secrets/initialAdminPassword
eb9f1af54e774fde9d963d65c5a3fdab
[root@zxh bin]#
```

密码

将密码复制并粘贴到管理员密码输入框，点击“继续”

2、Jenkins配置

解锁之后，由于网络原因导致Jenkins离线



解决方案参考：<https://blog.51cto.com/13568014/2350363>

修改 `hudson.model.UpdateCenter.xml` 文件：

首先查询 `hudson.model.UpdateCenter.xml` 文件所在位置，然后修改

```
# find / -name hudson.model.UpdateCenter.xml
/root/.jenkins/hudson.model.UpdateCenter.xml

# vi /root/.jenkins/hudson.model.UpdateCenter.xml
```

```
[root@zxh logs]# find / -name hudson.model.UpdateCenter.xml
/root/.jenkins/hudson.model.UpdateCenter.xml
[root@zxh logs]# vi /root/.jenkins/hudson.model.UpdateCenter.xml
```

将https改为http

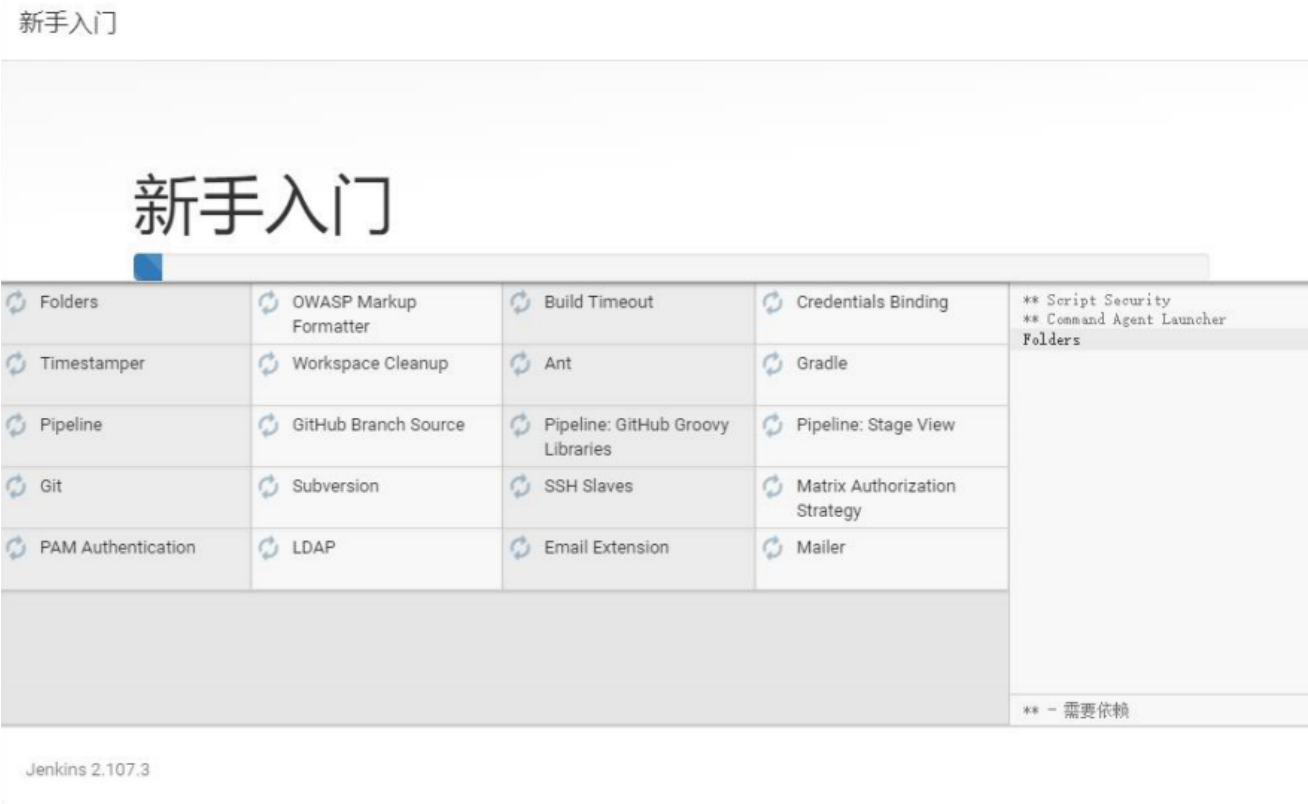
```
<?xml version='1.1' encoding='UTF-8'?>
<sites>
  <site>
    <id>default</id>
    <url>https://updates.jenkins.io/update-center.json</url>
  </site>
</sites>
~
~
```

改为http

重启tomcat。再次访问<http://192.168.3.118:8088/jenkins/>



这里选择“安装推荐的插件”，选择后会自动安装插件



自动安装完成后进入创建第一个管理员用户，界面如下：

新手入门

创建第一个管理员用户

用户名:

zjh

密码:

.....

确认密码:

.....

全名:

zhixinhua

电子邮件地址:

793597462@qq.com

Jenkins 2.172

使用admin账户继续

保存并完成

保存并完成。后面跳过设置RUL

新手入门

Jenkins已就绪！

您已经跳过了 Jenkins URL的配置。

要配置 Jenkins URL的话，到“Jenkins管理”页面。

Jenkins安装已完成。

开始使用Jenkins

点击开始会用Jenkins。jenkins登录后界面如下：



3、Jenkins卸载

因为这里Jenkins是war方式，所以卸载方式很简单。将tomcat webapp的项目删除 并清空 .jenkins 目录

```
# find / -name .jenkins //查找.jenkins目录的位置
```

```
[root@zxx bin]# find / -name .jenkins
/root/.jenkins
[root@zxx bin]# rm -rf /root/.jenkins/
[root@zxx bin]# find / -name .jenkins
[root@zxx bin]#
```

4、系统初始化配置



4.1、全局安全配置



4.2、全局工具配置

在配置之前，要清楚JDK、Maven的安装位置

```
[root@zxh bin]# echo $MAVEN_HOME
/usr/local/maven/maven3.5
[root@zxh bin]# echo $JAVA_HOME
/usr/local/java8/jdk1.8
[root@zxh bin]#
```

Maven Configuration 和 JDK

全局工具配置

Maven 配置

默认 settings 提供

文件系统中的 settings 文件

选择这个选项

文件路径

/usr/local/maven/maven3.5/conf/settings.xml

maven settings.xml的位置

默认全局 settings 提供

文件系统中的全局 settings 文件

设置maven settings.xml

文件路径

/usr/local/maven/maven3.5/conf/settings.xml

JDK

JDK 安装

新增 JDK

JDK

别名

jdk8

随便起一个名称

JAVA_HOME

/usr/local/java8/jdk1.8

java_home的目录

☐ 自动安装

去掉勾选

新增 JDK

系统下JDK 安装列表

删除 JDK

Maven 安装

Gradle

Gradle 安装

新增 Gradle

系统下Gradle 安装列表

Ant

Ant 安装

新增 Ant

系统下Ant 安装列表

Maven

Maven 安装

新增 Maven

Maven

Name

MyMaven

随便起一个maven名字

MAVEN_HOME

/usr/local/maven/maven3.5

配置MAVEN_HOME目录

☐ 自动安装

去掉勾选

新增 Maven

系统下Maven 安装列表

删除 Maven

Docker

Docker 安装

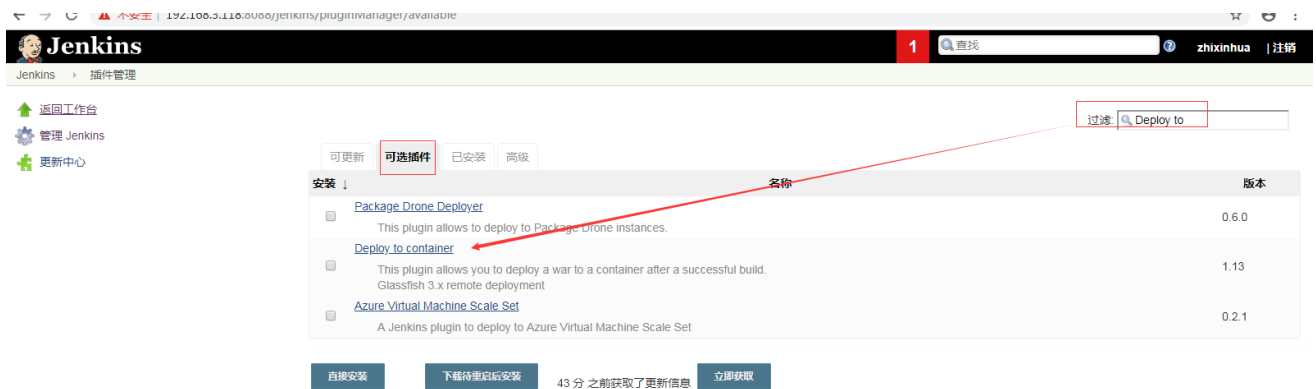
新增 Docker

系统下Docker 安装列表

5、插件安装

Jenkins要自动部署应用，需要用到Deploy to Web Container 插件。 插件安装：

【系统管理】-->【插件管理】-->【可选插件】->搜索 Deploy to container



选择插件，点击直接安装：

安装插件时受到网络状况的影响有可能会失败，不要紧，多试几次，直到成功。

Pipeline: Declarative	完成
Lockable Resources	完成
Pipeline	完成
GitHub API	完成
Git	完成
GitHub	完成
GitHub Branch Source	完成
Pipeline: GitHub Groovy Libraries	完成
Pipeline: Stage View	完成
Git	完成
MapDB API	完成
Subversion	完成
SSH Slaves	完成
Matrix Authorization Strategy	完成
PAM Authentication	完成
LDAP	完成
Email Extension	完成
Mailer	完成
Localization: Chinese (Simplified)	完成
Deploy to container	完成

[返回首页](#)
(返回首页使用已经安装好的插件)

[安装完成后重启Jenkins\(空闲时\)](#)

四、window 下SVN安装

参考：<https://blog.csdn.net/jinmie0193/article/details/81583264>

在安装时候我选择的是http

Initial Server Configuration

Please adjust the default configuration settings if necessary.



Location:	<input type="text" value="E:\learn\Visual SVN\"/>	<input type="button" value="Browse..."/>
Repositories:	<input type="text" value="E:\learn\Visual SVN\0_Repositories\"/>	<input type="button" value="Browse..."/>
Server Port:	<input type="text" value="443"/> <input type="checkbox"/> Use secure connection (https://)	
Backups:	<input type="text" value="E:\learn\Visual SVN backup\"/>	<input type="button" value="Browse..."/>

<https://blog.csdn.net/jinmiao0193>

Back

Next

Cancel

五、创建工程（实战）

[Jenkins日常运维笔记-重启数据覆盖问题、迁移、基于java代码发版\(maven构建\)](#)

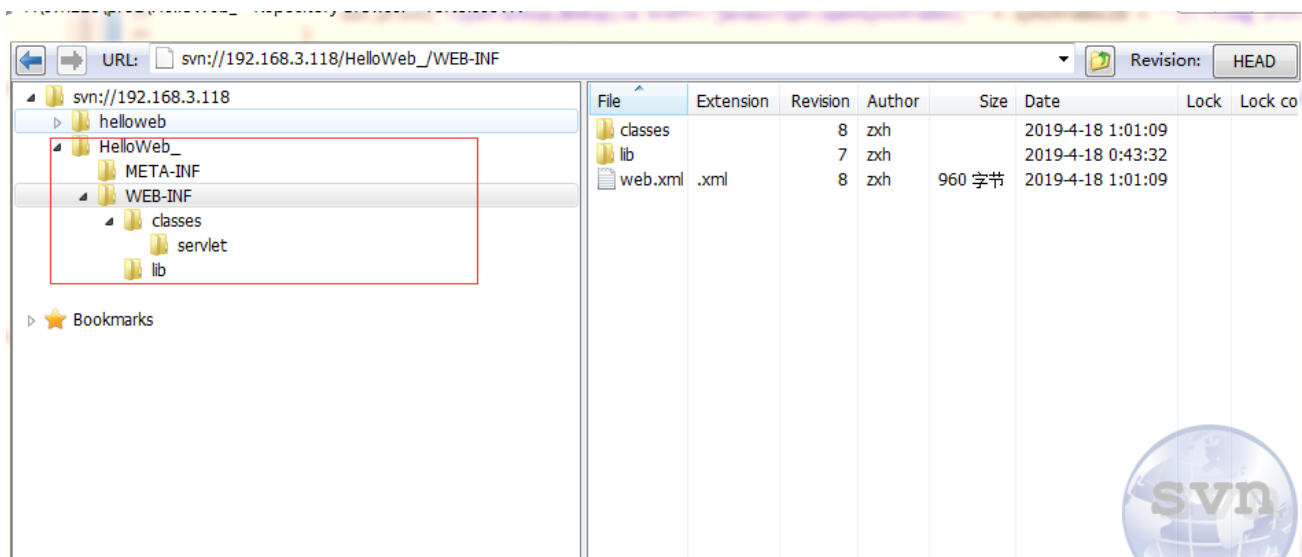
[Tomcat 部署项目的三种方法](#)

实战一：奇葩项目构建

这个项目svn存放的都是tomcat已编译的代码，对于这种奇葩的需求，jenkins的工作是：jenkins拉取svn代码，然后复制到tomcat再重新启动。

以下模拟这种需求：

1、svn的代码如下（里面都是已经编译好的代码）：



2、新建一个任务，构建一个自由风格的项目



3、项目构建配置

3.1、general设置

General

源码管理

构建触发器

构建环境

构建

构建后操作

描述

奇葩项目构建

写上描述，方便维护

[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

☐ 关闭构建

☐ 在必要的时候并发构建

高级...

3.2、源码管理设置

General

源码管理

构建触发器

构建环境

构建

构建后操作

源码管理

无

Git

Subversion

Modules

Repository URL

Repository URL

svn://192.168.3.118/HelloWeb_

Credentials

zxh/***** (118 svn)

添加

Local module directory

Local module directory

Repository depth

infinity

Ignore externals

Ignore externals

Cancel process on externals fail

Cancel process on externals fail

Add module...

Additional Credentials

Add additional credentials...

Check-out Strategy

Use 'svn update' as much as possible

Use 'svn update' whenever possible, making the build faster. But this causes the artifacts from the previous build to remain when a new build starts.

svn项目地址

添加svn访问的用户和密码，记得添加完之后要选择已添加的

3.3、构建触发器

构建触发器有很多构建方式：远程构建、其他工程构建后触发、定时构建等等。这里我们使用远程构建。

远程构建：

1、自定义一个身份令牌

2、构建访问路径：`JENKINS_URL /job/helloweb1/build?token= TOKEN_NAME` 或者 `/buildWithParameters?token= TOKEN_NAME`

这里的 `JENKINS_URL` 就是我们访问jenkins的路径，目前是：<http://192.168.3.118:8088/jenkins>

所以，根据规则可知，本任务的远程构建访问路径是：

`http://192.168.3.118:8088/jenkins/job/helloweb1/build?token=HELLOWEB_TOKEN1` 或者

`http://192.168.3.118:8088/jenkins/job/helloweb1/buildWithParameters?token=HELLOWEB_TOKEN1`

浏览器直接访问以上任意一个路径，或者使用curl命令请求该地址。都会触发构建。



3.4、构建

构建选择执行shell构建，shell主要实现的逻辑是：

1、复制代码到tomcat webapps下

2、重新启动tomcat




```
rm -rf /opt/tomcat_8089/webapps/helloweb1
cp -rf /root/.jenkins/workspace/helloweb1 /opt/tomcat_8089/webapps/

sh /opt/tomcat.sh
```

由于构建的时候，jenkins会将代码复制到jenkins的目录，这里是 `/root/.jenkins/workspace`。

控制台输出

```
Started by user unknown or anonymous
Running as SYSTEM
Building in workspace /root/.jenkins/workspace/helloweb1
Updating svn://192.168.3.118/HelloWeb_ at revision '2019-04-18T02:10:20.386 +0800' --quiet
Using sole credentials zzh/***** (118 svn) in realm '<svn://192.168.3.118:3690> 6adfe923-c1e3-4c86-b271-1f179da9c9c2'
At revision 8
```

/opt/tomcat.sh脚本：

```
# 防止jenkins会在构建完成后使用processTreeKiller杀掉了所有子进程
export BUILD_ID=dontkillme
# 设置tomcat启动和关闭的命令地址
tomcat_home=/opt/tomcat_8089
SHUTDOWN=$tomcat_home/bin/shutdown.sh
STARTTOMCAT=$tomcat_home/bin/startup.sh
echo "shudwon $tomcat_home"
# 关闭tomcat
$SHUTDOWN
# 用kill命令再次关闭
ps -ef | grep tomcat | grep $tomcat_home | grep -v 'grep' | awk '{print $2}' | xargs kill -9

#删除日志文件，如果你不先删除可以不要下面一行
#rm $tomcat_home/logs/* -rf
#删除tomcat的临时目录
#rm $tomcat_home/work/* -rf

sleep 3
echo "startup $tomcat_home"
# 启动tomcat
$STARTTOMCAT

# 日志输出后n行就行，千万不用一直输出，不然一直构建
#tail -f $tomcat_home/logs/catalina.out
tail -n 100 $tomcat_home/logs/catalina.out
```

如果遇到如下问题：

jenkins自动部署中执行shell脚本启动tomcat，但是tomcat不启动的问题。

解决方案参考：https://blog.csdn.net/weixin_39483907/article/details/80840948

4、完成以上配置就可以立即构建

实战二：maven项目构建

1、创建一个任务

这里也是构建一个自由风格的项目（参考上面）

2、项目构建配置

2.1、general设置

参考上面

2.2、源码管理

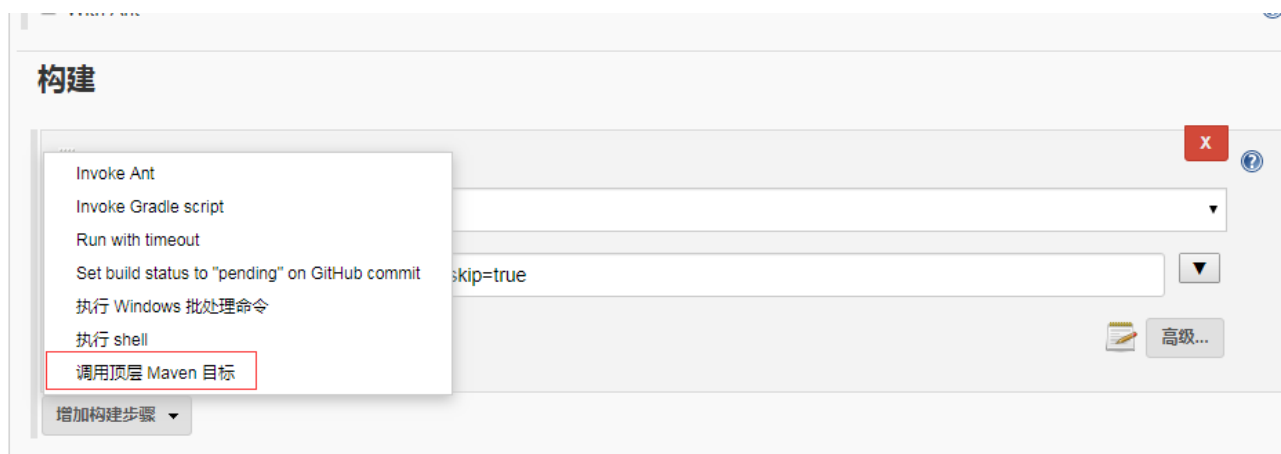
参考上面。这里要主要的是，Repository URL 要到项目目录（即pom.xml上一级）

2.3、构建触发器

同上

2.4、构建

构建maven



构建

调用顶层 Maven 目标

Maven 版本

MyMaven

目标

clean install -Dmaven.test.skip=true

这里选择之前全局配置的maven

构建是执行的maven命令，跳过单元测试

增加构建步骤

高级...

2.5、构建后操作

选择 Deploy war/ear to a container

构建后操作

Deploy war/ear to a container

WAR/EAR files

target/helloweb-1.0-SNAPSHOT.war

Aggregate downstream test results

Publish JUnit test result report

归档成品

构建其他工程

记录文件的指纹用于追踪

Git Publisher

Deploy war/ear to a container

E-mail Notification

Editable Email Notification

Set GitHub commit status (universal)

Set build status on GitHub commit [deprecated]

Delete workspace when build is done

增加构建后操作步骤

构建后操作

Deploy war/ear to a container

WAR/EAR files

target/helloweb-1.0-SNAPSHOT.war

war包已项目根目录为基准的相对目录

Context path

helloweb

浏览器访问时的项目名

Containers

Tomcat 8.x

Credentials

jenkins_user/*****

tomcat用户名和密码

Tomcat URL

http://192.168.3.118:8089

访问路径

应用服务tomcat配置

Deploy on failure

增加构建后操作步骤

3、完成以上配置就可以立即构建

4、使用钩子自动构建（看情况定是否需要）

设置钩子之后，只要svn更新了代码，jenkins就会触发构建。

修改svn的post-commit文件

```
[root@zxh repositories]# pwd
/opt/svn/repositories
[root@zxh repositories]# ls
conf  db  format  hooks  locks  README.txt
[root@zxh repositories]# cd hooks
[root@zxh hooks]# ls
post-commit      post-lock.tmpl      post-unlock.tmpl    pre-lock.tmpl      pre-unlock.tmpl
post-commit.tmpl post-revprop-change.tmpl pre-commit.tmpl      pre-revprop-change.tmpl start-commit.tmpl
[root@zxh hooks]#
```



(1) 修改之前最后先备份post-commit：`cp post-commit post-commit_back`（这里注意不要使用任何扩展名。如果按照我们习惯的使用.sh 扩展名则钩子程序无法正常工作）

(2) 使用 `chmod` 命令设置为可执行权限：`chmod 777 post-commit`

(3) 修改post-commit文件：`vi post-commit`

```
# For more examples and pre-written hooks, see those in
# the Subversion repository at
# http://svn.apache.org/repos/asf/subversion/trunk/tools/hook-scripts/ and
# http://svn.apache.org/repos/asf/subversion/trunk/contrib/hook-scripts/

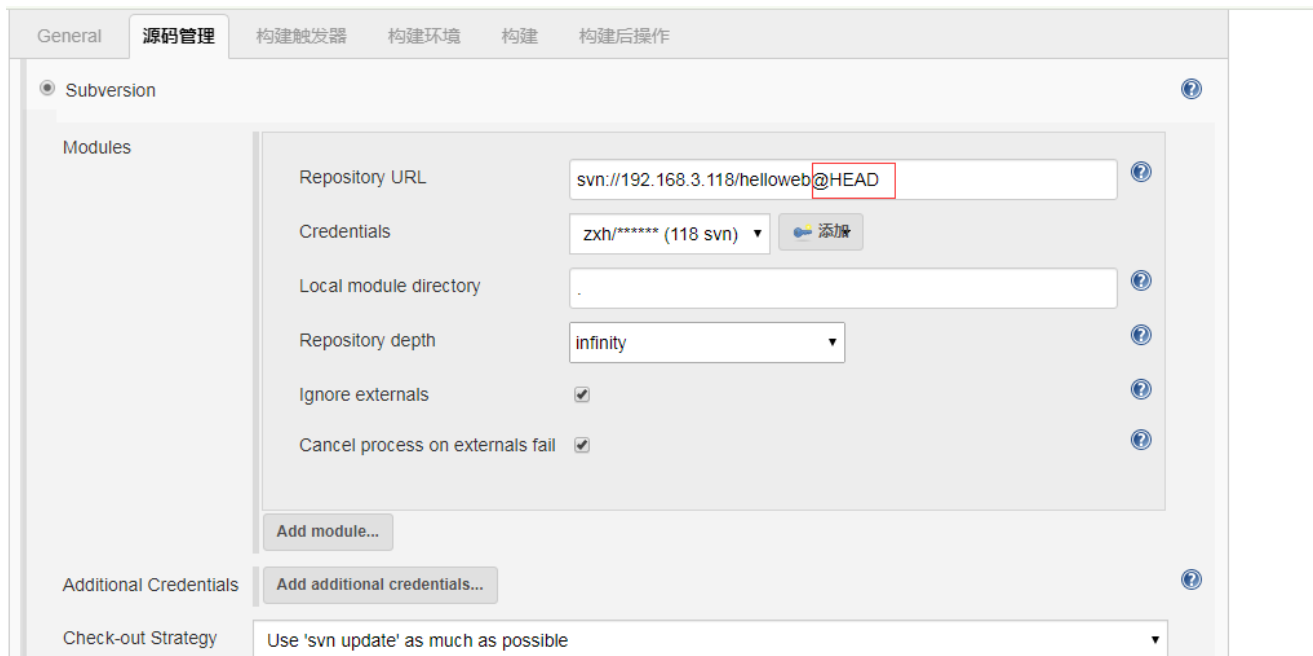
#REPOS="$1"
#REV="$2"

#mailer.py commit "$REPOS" "$REV" /path/to/mailer.conf
curl -X post -v -u zxh:123456 http://192.168.3.118:8088/jenkins/job/helloweb/build?token=HELLOWEB_TOKEN
```

其实就是利用svn访问触发构建的地址

(4) 补充

如果发生 Jenkins 服务器从 SVN 服务器下载代码不是最新版的情况，那么就在 SVN 服务器的 URL 地址后面加上 @HEAD 强制要求下载最新版。



实战三：ant 构建web项目

前提环境：在服务器安装ant。很简单，自行网上查找。

```
[root@zxh tomcat_8089]# echo $ANT_HOME
/usr/local/ant-1.10.5
[root@zxh tomcat_8089]# ant -v
Apache Ant(TM) version 1.10.5 compiled on July 10 2018
Trying the default build file: build.xml
Buildfile: build.xml does not exist!
Build failed
[root@zxh tomcat_8089]# ant
Buildfile: build.xml does not exist!
Build failed
[root@zxh tomcat_8089]#
```

安装目录

说明安装成功

ant编译项目的文件：build.xml



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<project name="SchedulePlatform" basedir="." default="war">
  <!--引入配置信息-->
```

```

<property file="build.properties" />
<property name="tomcat_home" location="/opt/tomcat_8089" />
<property name="java_home" location="/usr/local/java8/jdk1.8" />
<!-- java源代码目录 -->
<property name="src.dir" location="${basedir}/src" />
<!-- 构建目录 -->
<property name="build.dir" location="${basedir}/build" />
<!-- class文件目录 -->
<property name="build.classes" location="${build.dir}/classes" />
<!-- 打包目录 -->
<property name="build.war" location="${build.dir}/war" />

<!-- web 应用的名字，也是打包后war的名字 -->
<property name="web.name" value="SchedulePlatform" />
<!-- web 根目录 -->
<property name="web.root" value="WebRoot" />
<property name="web.WEB-INF" location="${web.root}/WEB-INF" />
<property name="web.lib" location="${web.WEB-INF}/lib" />

<!-- 定义编译时的classpath -->
<path id="compile.path">
  <!-- 拷贝第三方jar包 -->
  <fileset dir="${web.lib}" includes="*.lib">
    <include name="*.jar" />
  </fileset>
  <!-- 拷贝tomcat下的jar包 -->
  <fileset dir="${tomcat_home}/lib">
    <include name="**/*.jar" />
  </fileset>
</path>

<!-- 创建目录 -->
<target name="init" description="初始化" depends="clean">
  <!-- mkdir创建目录 -->
  <mkdir dir="${build.dir}" />
  <mkdir dir="${build.classes}" />
  <mkdir dir="${build.war}" />
  <echo>初始化工作结束！</echo>
</target>

<!-- web项目编译 -->
<target name="compile" depends="init" description="编译">
  <javac destdir="${build.classes}" srcdir="src" includeantruntime="false" fork="true"
bootclasspath="${java_home}/jre/lib/rt.jar" encoding="UTF-8">
    <compilerarg line="-encoding UTF-8" />
    <classpath refid="compile.path"/>
  </javac>
  <!-- 拷贝源码中的配置文件 -->
  <copy todir="${build.classes}">
    <fileset dir="${src.dir}">
      <exclude name="**/*.java" />
    </fileset>
  </copy>

```

```

    <echo message="编译完成！" />
</target>

<!--web项目打成war包-->
<target name="war" depends="compile" description="打包war文件">
    <war destfile="${build.war}/${web.name}.war">
        <fileset dir="${web.root}" includes="**/*.*" />
        <lib dir="${web.lib}" />
        <webinf dir="${web.WEB-INF}" />
        <classes dir="${build.classes}" />
    </war>
    <echo>打包完成！</echo>
</target>

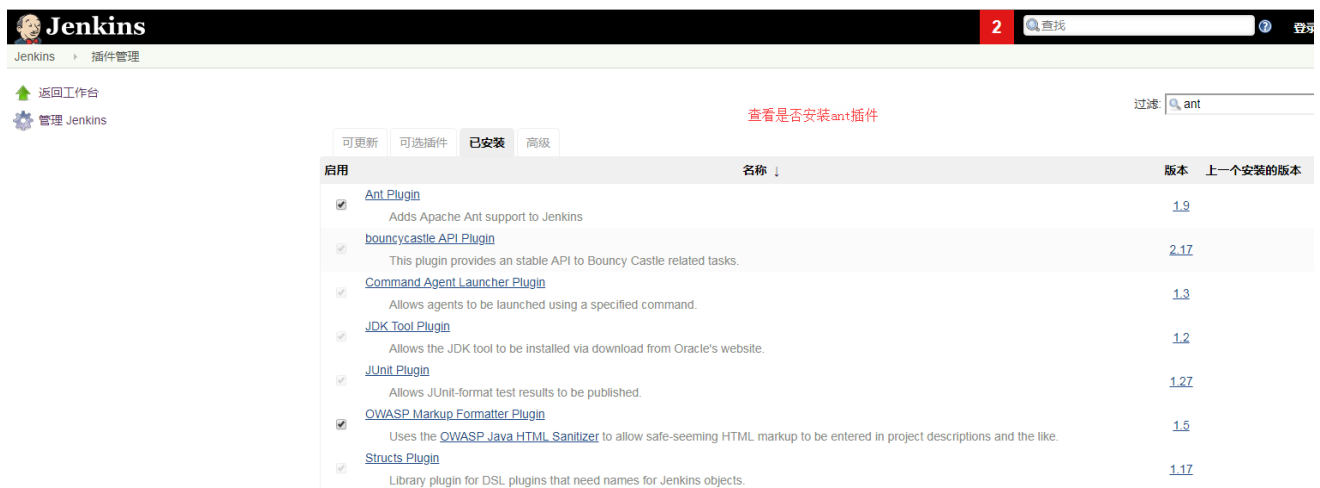
<!--创建目录-->
<target name="clean" description="清理">
    <delete dir="${build.classes}" />
    <delete dir="${build.war}" />
    <echo>清理完成！</echo>
</target>

</project>

```

1、检查jenkins是否安装ant插件

系统设置->插件管理，安装ant插件



The screenshot shows the Jenkins 'Plugin Management' page. The 'Ant Plugin' is listed as installed. The table below summarizes the visible plugins and their details.

启用	名称 ↓	版本	上一个安装的版本
<input checked="" type="checkbox"/>	Ant Plugin Adds Apache Ant support to Jenkins	1.9	
<input checked="" type="checkbox"/>	bouncycastle API Plugin This plugin provides an stable API to Bouncy Castle related tasks.	2.17	
<input checked="" type="checkbox"/>	Command Agent Launcher Plugin Allows agents to be launched using a specified command.	1.3	
<input checked="" type="checkbox"/>	JDK Tool Plugin Allows the JDK tool to be installed via download from Oracle's website.	1.2	
<input checked="" type="checkbox"/>	JUnit Plugin Allows JUnit-format test results to be published.	1.27	
<input checked="" type="checkbox"/>	OWASP Markup Formatter Plugin Uses the OWASP Java HTML Sanitizer to allow safe-seeming HTML markup to be entered in project descriptions and the like.	1.5	
<input checked="" type="checkbox"/>	Structs Plugin Library plugin for DSL plugins that need names for Jenkins objects.	1.17	

2、配置全局ant

Jenkins > 全局工具配置

☐ 自动安装

Delete Git

Add Git

Gradle

Gradle 安装

新增 Gradle

系统下Gradle 安装列表

Ant

Ant 安装

新增 Ant

Ant

Name

my_ant

ANT_HOME

/usr/local/ant-1.10.5

☐ 自动安装

删除 Ant

新增 Ant

系统下Ant 安装列表

Maven

Maven 安装...

Docker

Docker 安装

新增 Docker

3、构建自由风格项目。

general设置、源码管理、触发器和上面一样

4、构建

构建

Invoke Ant

选择刚刚配置的全局ant

Ant Version

my_ant

Targets

高级...

增加构建步骤

5、构建后操作

构建后操作

Deploy war/ear to a container

WAR/EAR files

build/war/SchedulePlatform.war

Context path

schedulePlatform

Containers

Tomcat 8.x

Credentials

jenkins_user/*****

Tomcat URL

http://192.168.3.118:8089

Add Container

Deploy on failure

增加构建后操作步骤

保存 应用

6、立即构建，查看日志如下

Buildfile: /root/.jenkins/workspace/SchedulePlatform/build.xml

clean:

[delete] Deleting directory /root/.jenkins/workspace/SchedulePlatform/build/classes
[delete] Deleting directory /root/.jenkins/workspace/SchedulePlatform/build/war
[echo] 清理完成!

init:

[mkdir] Created dir: /root/.jenkins/workspace/SchedulePlatform/build/classes
[mkdir] Created dir: /root/.jenkins/workspace/SchedulePlatform/build/war
[echo] 初始化工作结束!

compile:

[javac] Compiling 18 source files to /root/.jenkins/workspace/SchedulePlatform/build/classes
[javac] Note: Some input files use unchecked or unsafe operations.
[javac] Note: Recompile with -Xlint:unchecked for details.
[copy] Copying 1 file to /root/.jenkins/workspace/SchedulePlatform/build/classes
[echo] 编译完成!

war:

[war] Building war: /root/.jenkins/workspace/SchedulePlatform/build/war/SchedulePlatform.war
[echo] 打包完成!

BUILD SUCCESSFUL

Total time: 5 seconds

Deploying /root/.jenkins/workspace/SchedulePlatform/build/war/SchedulePlatform.war to container Tomcat 8.x Remote with context schedulePlatform
Redeploying [/root/.jenkins/workspace/SchedulePlatform/build/war/SchedulePlatform.war]
Undeploying [/root/.jenkins/workspace/SchedulePlatform/build/war/SchedulePlatform.war]
Deploying [/root/.jenkins/workspace/SchedulePlatform/build/war/SchedulePlatform.war]
Finished: SUCCESS

六、插件的安装方式

插件的安装位置在jenkins目录下的/plugins

可以 `find / -name plugins` 查找

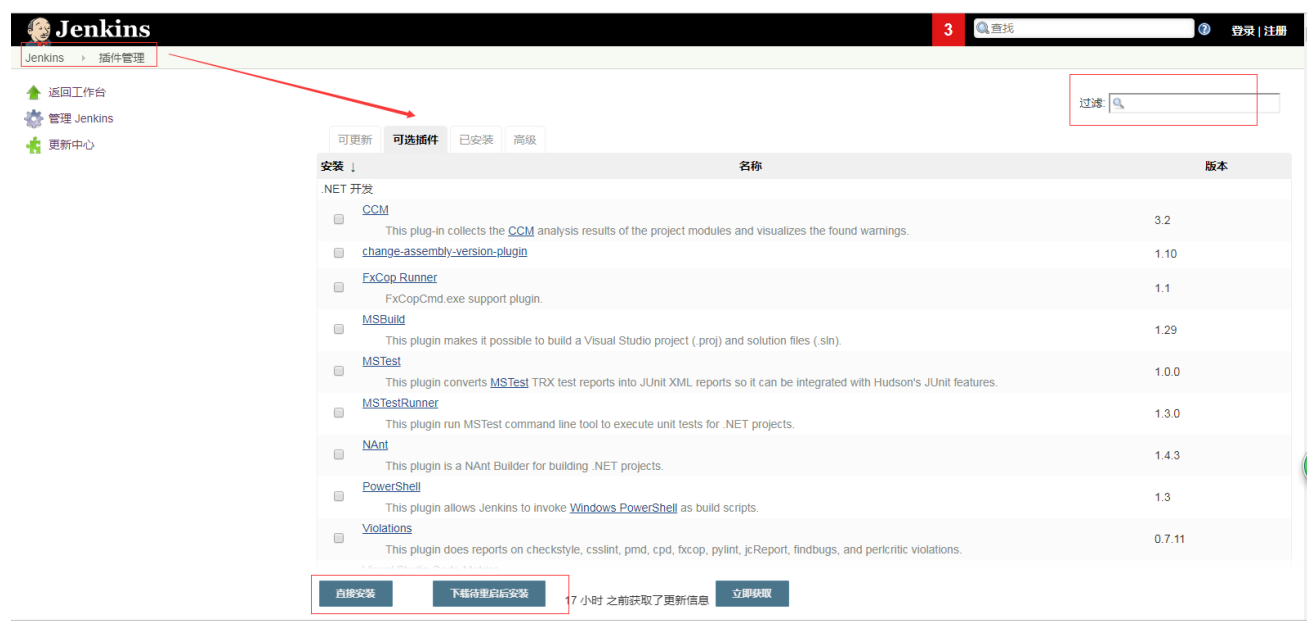
```
[root@localhost bin]# ^C
[root@localhost bin]# ls /root/.jenkins/plugins
ace-editor.jpi.tmp      git-client.jpi      momentjs             script-security.jpi
ant                    github              momentjs.jpi         ssh-credentials
antisamy-markup-formatter  github-api          pam-auth             ssh-credentials.jpi
antisamy-markup-formatter.jpi  github-api.jpi     pam-auth.jpi         ssh-slaves
ant.jpi                github-branch-source  pipeline-build-step  ssh-slaves.jpi
apache-httpcomponents-client-4-api  github-branch-source.jpi  pipeline-build-step.jpi  structs
apache-httpcomponents-client-4-api.jpi  github.jpi          pipeline-github-lib     structs.jpi
authentication-tokens          git.jpi             pipeline-github-lib.jpi  subversion
authentication-tokens.jpi       git-server          pipeline-graph-analysis  subversion.jpi
bouncycastle-api             git-server.jpi      pipeline-graph-analysis.jpi  timestamp
bouncycastle-api.jpi          gradle              pipeline-input-step      timestamp.jpi
branch-api                   gradle.jpi          pipeline-input-step.jpi   token-macro
branch-api.jpi               handlebars          pipeline-milestone-step   token-macro.jpi
build-timeout                handlebars.jpi      pipeline-milestone-step.jpi  workflow-aggregator
build-timeout.jpi            jackson2-api       pipeline-model-api        workflow-aggregator.jpi
cloudbees-folder             jackson2-api.jpi   pipeline-model-api.jpi    workflow-api
cloudbees-folder.jpi         jdk-tool           pipeline-model-declarative-agent  workflow-api.jpi
command-launcher             jdk-tool.jpi       pipeline-model-declarative-agent.jpi  workflow-basic-steps
command-launcher.jpi         jquery-detached     pipeline-model-definition  workflow-basic-steps.jpi
credentials                   jquery-detached.jpi  pipeline-model-definition.jpi  workflow-cps
credentials-binding           jsch                pipeline-model-extensions  workflow-cps-global-lib.jpi.tmp
credentials-binding.jpi      jsch.jpi            pipeline-model-extensions.jpi  workflow-cps.jpi
```

卸载插件，直接使用 `rm -rf 插件` 移除。（慎重操作）

也可以在已安装中找到插件，卸载。

方式一：在线安装

【插件管理】-->【可选插件】-->【搜索安装的插件】-->安装



方式二：离线安装（在线安装网络不好时候可选离线方式）

【插件管理】-->【高级】-->上传下载好的插件（.hpi 文件）-->安装

Jenkins 3 查找 登录 | 注册

Jenkins > 插件管理

返回工作台
管理 Jenkins
更新中心

可更新 可选插件 已安装 **高级**

代理设置

服务器

端口

用户名

密码

不通过代理的主机

提交 高级...

上传插件

您可以通过上传一个 **hpi** 文件来安装插件。

文件: 未选择任何文件

网络禁止

插件下载地址：<https://plugins.jenkins.io/>

https://plugins.jenkins.io

Jenkins Blog Documentation **Plugins** Community Sub-projects About English



Plugins Index

Discover the 1000+ community contributed Jenkins plugins to support building, deploying and automating any project.

Browse Find plugins... 搜索插件

Browse categories	New Plugins	Recently updated	Trending
Platforms	SCM Skip	SCM Skip	Script Security
User interface	Metrics DataDog	Service Now	JUnit
Administration	UIPath	Matrix Authorization Strategy	Credentials
Source code management	Deep Security Smart Check	Violation Comments to Bitbucket Server	Struts
Build management	ICQ Notification	Violation Comments to GitHub	Pipeline: Step API
	CCTray XML (cc.xml)	Violation Comments to GitLab	Mailer
	GitHub Status Wrapper	Slack Notification	SSH Credentials
	QRebel	Worksoft Execution Manager	Matrix Project
	MATLAB	MathWorks Polyspace	JDK Tool
	MISRA Compliance Report	NowSecure Auto	SCM API

在连接...

Jenkins Blog Documentation **Plugins** Community Sub-projects About English

Browse Deploy to container 搜索插件

Sort relevance
● Relevance
○ Most installed
○ Trending
○ Title
○ Release date

Categories
■ Platforms
■ iOS development
■ .NET
■ Android development
■ Ruby development
■ User interface
■ User interface
■ List view column plugins
■ Administration
■ Agent controllers
■ Page decorators
■ Users and security
■ Cluster management
■ CLI extensions

1 2 3 4 5

1 to 50 of 779

Deploy to container
Installs: 18369
Jenkins 1.642.4++
Artifact uploaders,
This plugin allows you to deploy a war to a container after a successful build.
Glassfish 3.x remote
Kohsuke Kawaguchi
Meikel Bode
(2 other contributors)

Allyun-Container-Service-Deploy
Installs: 112
Jenkins 1.651.2++
This plugin deploy applications to Aliyun Container Service.
qinyujia

Azure Container Service
Installs: 739
Jenkins 1.651.3++
Deploy Kubernetes, DC/OS, Docker Swarm application configurations to Azure Container Service cluster.
Azure DevOps Team

Kubernetes Continuous Deploy
Installs: 2938
Jenkins 1.651.3++
A Jenkins plugin to deploy configurations to Kubernetes cluster.
Azure DevOps Team

Azure Container Agents
Installs: 554
Jenkins 1.651.3++
A Jenkins plugin to provisions agents on Azure Container Service and Azure Container Instances
Azure DevOps Team

Amazon Elastic Service
Installs: 3718
Jenkins 2.107.3++
Cluster management controllers, Use Amazon EC2 Service to provide agents.
Philipp Garbe
Douglas Manley
(1 other contributors)

Deploy WebLogic
Installs: 1656
Jenkins 1.580.1++
Artifact uploaders, Clean-up actions,
This plugin allows you to deploy an artifact (JAR, EAR, WAR) to any WebLogic target (admin server, managed server,
D.W.

Amazon EC2 Container Service with autoscaling
Installs: 1167
Jenkins 1.609++
Cluster management, Agent controllers,
Use Amazon EC2 Container Service to provide elastic slaves.
Jan Roehrich

OSF Builder Suite For Salesforce Commerce
Installs: 37
Jenkins 2.107.1++
Deploy your build to a Salesforce Commerce Cloud instance
G.R.

Azure Container Registry Tasks
Installs: 214
Jenkins 1.651.3++
Queue an Azure Container Registry quick task request for building image. Source code can be hosted on a Github repo
A.C.

Xebialabs XL Deploy
Installs: 572
Jenkins 1.642.3++
Artifact uploaders, Clean-up actions, Deployment plugins,
Package and deploy your applications from Jenkins with Xebialabs XL Deploy.
XX

Anchore Conta Scanner
Installs: 470
Jenkins 1.625.3++
Build tools, SCM co
This plugin provid image scanning us Engine
Daniel Nurni

→ ↻ https://plugins.jenkins.io/deploy

Jenkins Blog Documentation **Plugins** Community Sub-projects About English **Download**

Find plugins

Deploy to container 1.13
Minimum Jenkins requirement: 1.642.4
ID: deploy

Installs: 18369
GitHub →
Last released: 2 years ago

Maintainers
Kohsuke Kawaguchi
Meikel Bode
Daniel Barth
Leandro Kersting de Freitas

Dependencies
Credentials v.2.1.13 (required)
bouncycastle API v.2.16.0 (implied) (what's this?)
Command Agent Launcher v.1.0 (implied) (what's this?)
JDK Tool v.1.0 (implied) (what's this?)
JAXB v.2.3.0 (implied) (what's this?)

Archives
Get past versions

20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0

Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar

Labels
Artifact uploaders

Are you maintaining this plugin?
Visit the [Jenkins Plugin Wiki](#) to edit this content.

参考文章

<https://mp.weixin.qq.com/s/g6RXis8lw7I2Fcvimj-slg>