

CentOS7 下 svn+tomcat9.0+maven3.3+jenkins 实现 web 项目自动构建与远程发布

by:授客 QQ: 1033553122

目录

一、	实践环境	1
二、	安装 SVN(如果没的话)	1
三、	安装 java.....	1
四、	安装 Apache Tomcat.....	2
五、	安装 maven.....	5
六、	安装 jenkins.....	6
七、	Jenkins 基本设置	6
1)	系统管理-系统设置	6
2)	系统管理-安全设置	7
3)	系统管理-插件管理	9
八、	自动构建任务与自动部署.....	10

一、实践环境

CentOS 7 操作系统(CentOS-7-x86_64-DVD-1503-01.iso)

下载地址: http://ftp.riken.jp/Linux/centos/7/isos/x86_64/

Java(jdk-8u65-linux-x64.tar.gz)

下载地址:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Apache Tomcat(apache-tomcat-9.0.0.M1.tar.gz)

下载地址: <http://tomcat.apache.org/download-90.cgi>

maven(apache-maven-3.3.9-bin.tar.gz)

下载地址: <http://maven.apache.org/download.cgi>

jenkins(jenkins.war)

下载地址: <https://wiki.jenkins-ci.org/display/JENKINS/Meet+Jenkins>

Deploy to container Plugin(deploy-1.0.ph)

下载地址: <https://updates.jenkins-ci.org/download/plugins/deploy/>

以上软件包集合下载地址: <http://pan.baidu.com/s/1c1xnUfu>

二、安装 SVN(如果没的话)

参考文章: [CentOS7 下配置 svn 的安装及基础配置介绍](#)

三、安装 java

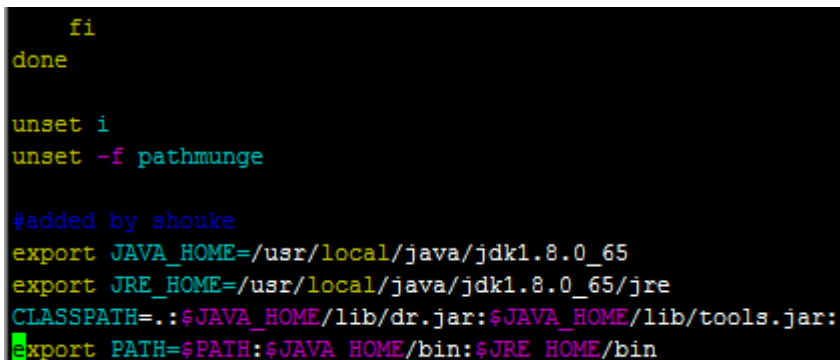
```
[root@localhost tmp]# mkdir -p /usr/local/java
[root@localhost tmp]# mv jdk-8u65-linux-x64.tar.gz /usr/local/java
[root@localhost tmp]# cd /usr/local/java/
[root@localhost java]# tar -xvzf jdk-8u65-linux-x64.tar.gz
.....
[root@localhost java]# rm -rf jdk-8u65-linux-x64.tar.gz
```

环境变量配置

```
[root@localhost java]# vim /etc/profile
```

添加如下内容:

```
#added by shouke
export JAVA_HOME=/usr/local/java/jdk1.8.0_65
export JRE_HOME=/usr/local/java/jdk1.8.0_65/jre
CLASSPATH=.:$JAVA_HOME/lib/dr.jar:$JAVA_HOME/lib/tools.jar:
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
```



```
fi
done

unset i
unset -f pathmunge

#added by shouke
export JAVA_HOME=/usr/local/java/jdk1.8.0_65
export JRE_HOME=/usr/local/java/jdk1.8.0_65/jre
CLASSPATH=.:$JAVA_HOME/lib/dr.jar:$JAVA_HOME/lib/tools.jar:
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin
```

```
[root@localhost bin]# source /etc/profile
```

查看是否安装成功

```
[root@localhost java]# java -version
java version "1.8.0_65"
Java(TM) SE Runtime Environment (build 1.8.0_65-b17)
Java HotSpot(TM) 64-Bit Server VM (build 25.65-b01, mixed mode)
[root@localhost java]# javac -version
javac 1.8.0_65
```

参考连接:

http://docs.oracle.com/javase/8/docs/technotes/guides/install/linux_jdk.html#BJFJJJFG

四、安装 Apache Tomcat

```
[root@localhost tmp]# tar -xvzf apache-tomcat-9.0.0.M1.tar.gz
.....
[root@localhost tmp]# mkdir -p /usr/local/apache-tomcat
```

```
[root@localhost tmp]# mv apache-tomcat-9.0.0.M1 /usr/local/apache-tomcat/
```

环境变量配置:

```
[root@localhost java]# vim /etc/profile
```

.....

```
#added by shouke
```

```
export JAVA_HOME=/usr/local/java/jdk1.8.0_65
```

```
export JRE_HOME=/usr/local/java/jdk1.8.0_65/jre
```

```
export CATALINA_BASE=/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
```

```
export CATALINA_HOME=/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
```

```
CLASSPATH=.:$JAVA_HOME/lib/dr.jar:$JAVA_HOME/lib/tools.jar:
```

```
export PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin:$CATALINA_BASE:$CATALINA_HOME
```

```
[root@localhost bin]# source /etc/profile
```

设置管理员帐号密码

```
[root@localhost apache-tomcat-9.0.0.M1]# ls
```

```
bin  conf  lib  LICENSE  logs  NOTICE  RELEASE-NOTES  RUNNING.txt  temp
webapps  work
```

```
[root@localhost apache-tomcat-9.0.0.M1]# cd conf/
```

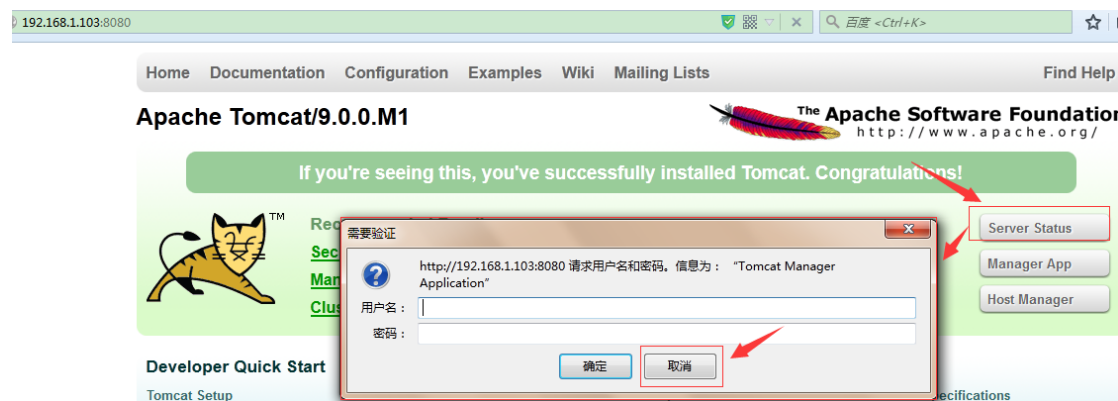
```
[root@localhost conf]# vim tomcat-users.xml
```

找到如下内容

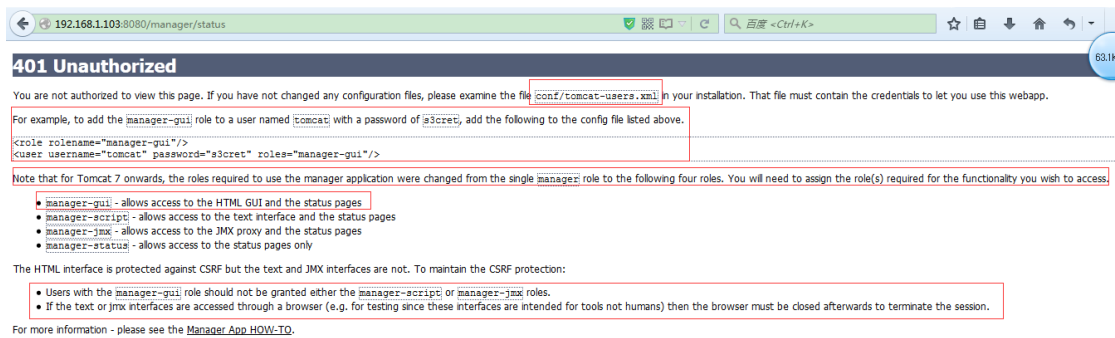
```
<!--
NOTE: The sample user and role entries below are wrapped in a comment
and thus are ignored when reading this file. Do not forget to remove
<!-- ... --> that surrounds them.
-->
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="tomcat" roles="tomcat"/>
<user username="both" password="tomcat" roles="tomcat,role1"/>
<user username="role1" password="tomcat" roles="role1"/>
-->
tomcat-users>
```

在其下新增用户和角色

问题: 怎么知道角色是啥角色呢? 如下, 点击对应按钮, 点击取消,



如下，它会告诉你怎么做的



可根据实际情况设置，修改后的文件配置如下

```
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="tomcat" roles="tomcat"/>
<user username="both" password="tomcat" roles="tomcat,role1"/>
<user username="role1" password="tomcat" roles="role1"/>
-->
<role rolename="manager-gui"/>
<role rolename="admin-gui"/>
<user username="hostadmin" password="123456" roles="admin-gui"/>
<user username="admin" password="huozhe" roles="manager-gui,admin-gui"/>
</tomcat-users>
```

说明: admin 用户可以访问 Server Status, Manager App, Host Manager
, hostadmin 只可访问 Host Manager

启动 Apache Tomcat

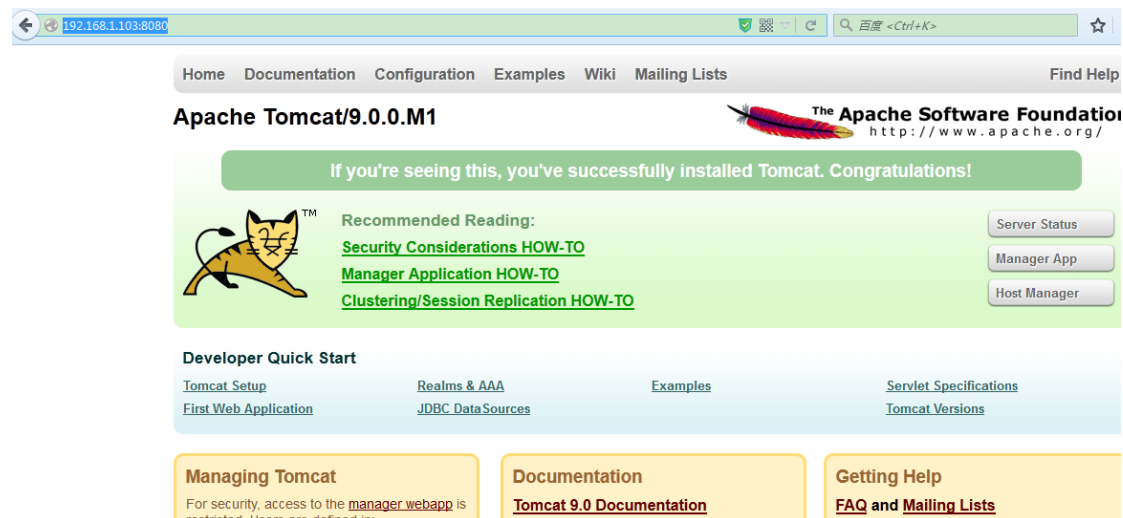
```
[root@localhost tmp]# cd /usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/
[root@localhost apache-tomcat-9.0.0.M1]# cd bin
[root@localhost bin]# sh startup.sh
Using CATALINA_BASE:   /usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
Using CATALINA_HOME:   /usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
Using CATALINA_TMPDIR: /usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/temp
Using JRE_HOME:        /usr/local/java/jdk1.8.0_65/jre
Using CLASSPATH:
/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/bin/bootstrap.jar:/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/bin/tomcat-juli.jar
Tomcat started.
[root@localhost bin]#
```

防火墙开放默认的 8080 端口

```
[root@localhost bin]# firewall-cmd --permanent --zone=public
--add-port=8080/tcp
success
```

```
[root@localhost bin]# firewall-cmd --reload
```

访问测试



五、安装 maven

```
[root@localhost tmp]# mkdir -p /usr/local/maven
[root@localhost tmp]# tar -xvzf apache-maven-3.3.9-bin.tar.gz
.....
[root@localhost tmp]# mv apache-maven-3.3.9 /usr/local/maven/
```

环境变量设置

```
[root@localhost tmp]# vim /etc/profile
export JAVA_HOME=/usr/local/java/jdk1.8.0_65
export JRE_HOME=/usr/local/java/jdk1.8.0_65/jre
export CATALINA_BASE=/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
export CATALINA_HOME=/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1
export MAVEN_HOME=/usr/local/maven/apache-maven-3.3.9
CLASSPATH=.:$JAVA_HOME/lib/dr.jar:$JAVA_HOME/lib/tools.jar:
export
PATH=$PATH:$JAVA_HOME/bin:$JRE_HOME/bin:$CATALINA_BASE:$CATALINA_HOME:$MAVEN_HOME/bin
[root@localhost tmp]# source /etc/profile
```

查看是否安装成功

```
[root@localhost tmp]# mvn -v
Apache Maven 3.3.9 (bb52d8502b132ec0a5a3f4c09453c07478323dc5;
2015-11-11T00:41:47+08:00)
Maven home: /usr/local/maven/apache-maven-3.3.9
Java version: 1.8.0_65, vendor: Oracle Corporation
Java home: /usr/local/java/jdk1.8.0_65/jre
```

Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "3.10.0-229.el7.x86_64", arch: "amd64", family:
"unix"

参考连接: <http://maven.apache.org/install.html>

六、安装 jenkins

```
[root@localhost tmp]# ls
jenkins.war
[root@localhost tmp]# cp jenkins.war
/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/webapps/
```

重启 apache tomcat 服务器

```
[root@localhost bin]# sh
/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/bin/shutdown.sh
[root@localhost bin]# sh
/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/bin/startup.sh
```

访问 jenkins



七、Jenkins 基本设置

1) 系统管理-系统设置

JDK

JDK 安装

JDK

别名

JDK

JAVA_HOME

/usr/local/java/jdk1.8.0_65/

☐ 自动安装

删除 JDK

新增 JDK

系统下JDK 安装列表

Ant

Ant 安装

新增 Ant

系统下Ant 安装列表

Maven

Maven 安装

Maven

Name

Maven

MAVEN_HOME

/usr/local/maven/apache-maven-3.3.9/

☐ 自动安装

Subversion

Subversion Workspace Version

1.7

Exclusion revprop name

☐ Validate repository URLs up to the first variable name

☒ Update default Subversion credentials cache after successful authentication

2) 系统管理-安全设置

Configure Global Security

☒ 启用安全

JNLP节点代理的TCP端口 ☐ 指定端口: ☒ 随机选取 ☐ 禁用

Disable remember me ☐

访问控制

安全域

☒ Jenkins专用户户数据库

☒ 允许用户注册

☐ LDAP

☐ Servlet容器代理

☐ Unix用户/组数据库

授权策略

☐ 任何用户可以做任何事(没有任何限制)

☐ 安全矩阵

☒ 登录用户可以做任何事

☐ 遗留模式

☐ 项目矩阵授权策略

保存

应用

提交后如下



Jenkins

Jenkins >

 用户

 任务历史

 Credentials

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

用户名:


密码:

☐ 在这台计算机上保持登录状态

登录




[创建一个用户账号](#) 如果你没有注册用户.

注册并登陆



Jenkins

Jenkins ▸

 用户
 任务历史
 Credentials

构建队列
队列中没有构建任务

构建执行状态
1 空闲

Sign up

用户名:

密码:

确认密码:

全名:

电子邮件地址:

Sign up

3) 系统管理-插件管理

1) Deploy to container Plugin

安装方法 1、在线安装

系统管理->管理插件->可选插件->过滤, 输入 Deploy to container Plugin,

Jenkins ▸ 插件管理

 返回

 系统管理

可更新 可选插件 已安装 高级

安装 ↓

☐ [Deploy to container Plugin](#)
This plugin takes a war/ear file and depl

直接安装 **下载待重启后安装**

 返回

 系统管理

 管理插件

安装/更新 插件中

准备

- Checking internet

Deploy to container Plugin  等待

重启 Jenkins  等待

 [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)

缺点:可能无法下载, GFW 太强大了

安装方法 2、本地安装

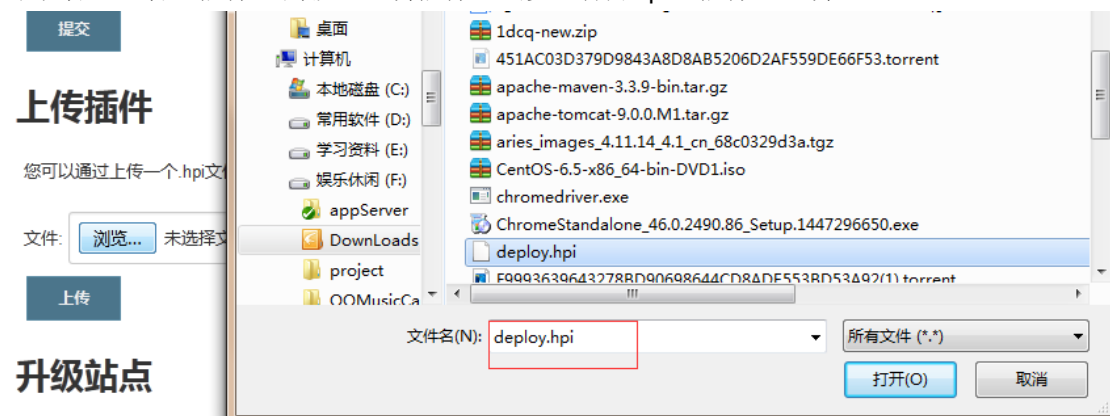
下载插件，插件集合下载地址：

<https://updates.jenkins-ci.org/download/plugins>

Deploy to container Plugin 下载地址：

<https://updates.jenkins-ci.org/download/plugins/deploy/>

系统管理->管理插件->高级->上传插件->浏览，打开 .phi 插件->上传



查看是否安装成功



八、自动构建任务与自动部署

点击 **【新建】**、**【创建一个新任务】** (从未创建过任务的情况)



The image shows the Jenkins 'New Item' dialog. On the left is a sidebar with navigation links: '新建' (New), '用户' (Users), '任务历史' (Task History), '系统管理' (System Management), and 'Credentials'. The main area is titled '构建队列' (Build Queue) and shows '队列中没有构建任务' (No build tasks in queue). Below that is '构建执行状态' (Build Execution Status) showing two '空闲' (Idle) states. On the right, the 'Item名称' (Item Name) field is set to 'test_project1'. There are four radio button options: '构建一个自由风格的软件项目' (Build a free-style software project), '构建一个maven项目' (Build a maven project), 'External Job', and '构建一个多配置项目' (Build a multi-configuration project). The '构建一个maven项目' option is selected. Below the options is an 'OK' button.

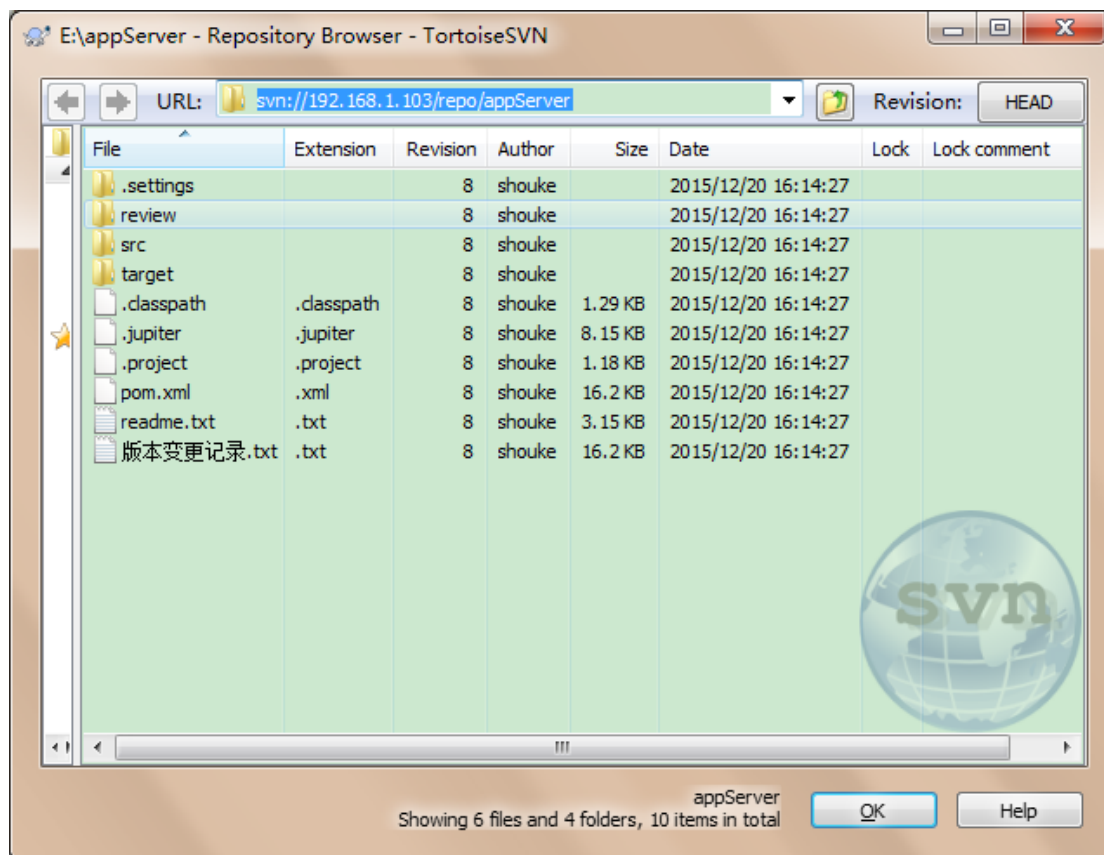
Item名称: test_project1

- ☐ 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
- ☒ 构建一个maven项目
构建一个maven项目.Jenkins利用你的POM文
- ☐ External Job
这个类型的任务允许你记录执行在外部Jenkin: [组内容](#).
- ☐ 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

OK

如上图,填写好 item 名称, 点击 **【OK】**

代码结构如下:



选择 Subversion, 填写版本库代码 url

源码管理

☐ None
☐ CVS
☐ CVS Projectset
☒ Subversion

Modules

Repository URL ?

Unable to access svn://192.168.1.103/repo/appServer : svn: E200015: No credential to try.
Authentication failed (show details)
(Maybe you need to enter credential?)

Local module directory (optional) ?

Repository depth ?

Ignore externals ☐ ?

Add more locations...


Check-out Strategy ?

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.

源码库浏览器

如上, 提示 No credential, 点击 enter credential, 打开如下界面

Subversion

 **Subversion Authentication**

Enter the authentication information needed to connect to the Subversion

Repository URL

☒ Username/password authentication

User name

Password

☐ SSH public key authentication (svn+ssh)

☐ HTTPS client certificate

OK

如上, 填写代码库所在 url 及用户名称和密码, 点击【OK】提交

返回到刚才的页面, 刷新, 重新填写, 结果如下

源码管理

☐ None
☐ CVS
☐ CVS Projectset
☒ Subversion

Modules

Repository URL

Local module directory (optional)

Repository depth

Ignore externals ☐

[Add more locations...](#)

Check-out Strategy

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.

源码库浏览器

构建触发器

构建触发器

☒ Build whenever a SNAPSHOT dependency is built
☐ 触发远程构建 (例如,使用脚本)
☐ Build after other projects are built
☐ Build periodically
☒ Poll SCM

日程表

⚠ Spread load evenly by using 'H/60 * * * *' rather than '* */60 * * * *'
Would last have run at Sunday, December 20, 2015 5:00:57 PM CST; would next run at Sunday, December 20, 2015 6:00:57 PM CST.

Ignore post-commit hooks ☐

说明:

1) Poll SCM: 定时检查源码变更 (根据 SCM 软件的版本号), 如果有更新就 checkout 最新 code 下来, 然后执行构建动作。

2) Build periodically: 按给定周期, 定时构建 (它不管源码是否发生变化)

示例:

* / 60 * * * * (意为每 60 分钟检查一次源码变化)

0 2 * * * (每天 2:00 执行一次构建)

这里和 linux crontab 文件配置是一致的。

参考连接:

<http://www.scmgalaxy.com/scm/setting-up-the-cron-jobs-in-jenkins-using-build-periodically-scheduling-the-jenkins-job.html>

Pre Steps

Add pre-build step ▾

Build

Root POM pom.xml

Goals and options

高级...

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.

Add post-build step ▾

```
[root@localhost workspace]# pwd
/root/.jenkins/jobs/test_project1/workspace
[root@localhost workspace]# ll
total 56
-rw-r-----. 1 root root 16629 Dec 20 17:21 pom.xml
-rw-r-----. 1 root root 3235 Dec 20 17:21 readme.txt
drwxr-x---. 2 root root 4096 Dec 20 17:21 review
drwxr-x---. 4 root root 4096 Dec 20 17:21 src
drwxr-x---. 8 root root 4096 Dec 20 17:21 target
-rw-r-----. 1 root root 16664 Dec 20 17:21 版本变更记录.txt
```

注意：这里的 pom 设置是有讲究的，参考连接：

http://my.oschina.net/u/260244/blog/318755#OSC_h3_16

如上，Post steps, 选择仅 build 成功时才运行 Post Steps

Post Steps

☒ Run only if build succeeds ☐ Run only if build succeeds or is unstable

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

- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Deploy artifacts to Maven repository
- Record fingerprints of files to track usage
- Deploy war/ear to a container

增加构建后操作步骤 ▾

保存 应用

如上，点击【增加构建后操作步骤】，选择 Deploy war/ear to a container, 设置远程发布项目

构建后操作

Deploy war/ear to a container

WAR/EAR files

target/appServer.war

Context path

/usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/webapps/test_dir

Containers

Tomcat 7.x

Manager user name

admin

Manager password

Tomcat URL

http://192.168.1.103:8080

删除

Add Container

Deploy on failure ☐

删除

增加构建后操作步骤

说明:

1) 构建时会在目录: `/root/.jenkins/jobs/item_name/workspace/target` 下生成 `filename.war` 文件, 本例中为:

`/root/.jenkins/jobs/test_project1/workspace/target/appServer.war`

2) WAR/EAR files: 填写 `.war`、`.ear` 文件所在的相对路径

3) Context Path: 填写要发布至远程服务器的位置, 通常是 tomcat 的 webapps

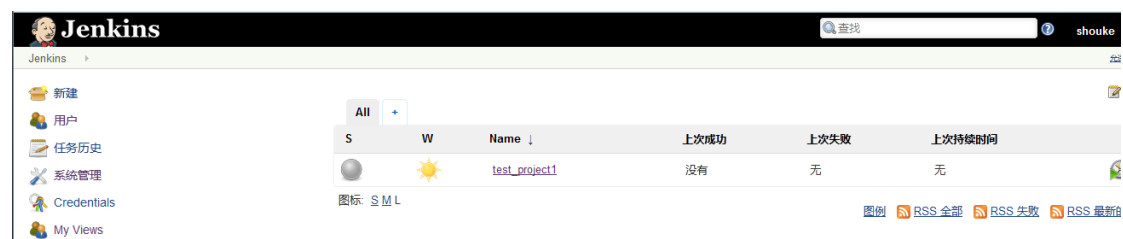
4) 通常不勾选【Deploy on failure】, 即构建失败则不发布

5) 这里也可以用 Publish over SSH 来实现远程发布, 参考连接:

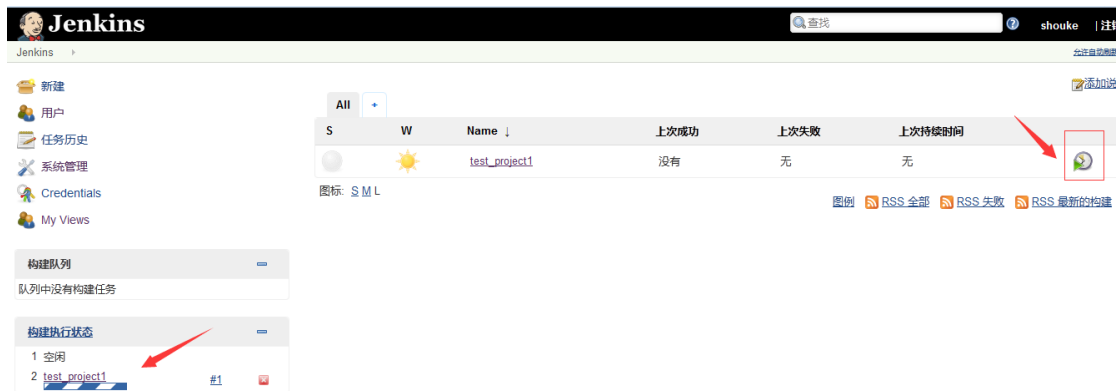
<http://jdkleo.iteye.com/blog/2159844>

如上, 点击 Add Container, 可以选择容器类型, 这里选择 Tomcat 7.x, 然后填写入 Tomcat 管理员(具有 manager-gui 角色的 tomcat 用户), 密码, Tomcat 连接

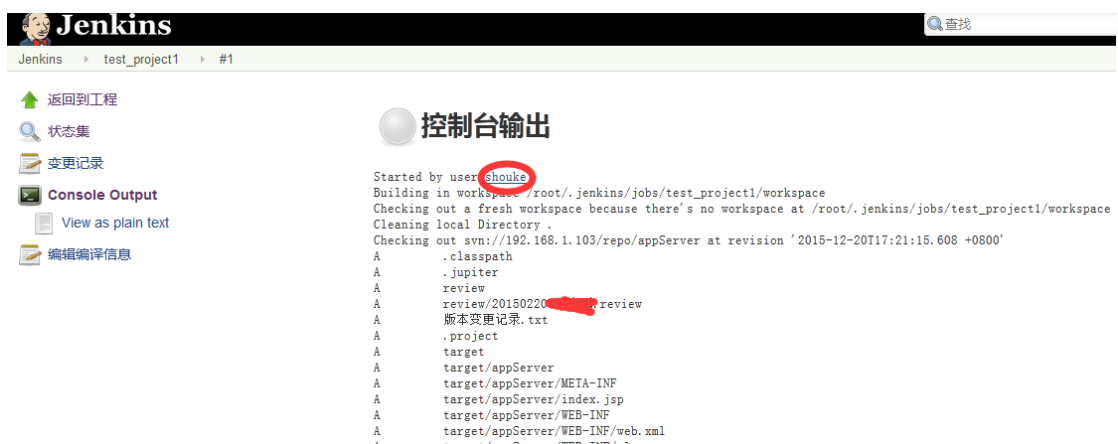
最后提交, 如下



如下, 点击右侧按钮, 开始第一次构建



点击连接查看 Console Output



如下，第一次会根据 pom.xml 下载相关文件

```
Parsing POMs
Discovered a new module com.idcq.appServer:appServer appserver Maven Webapp
Modules changed, recalculating dependency graph
[workspace] $ /usr/local/java/jdk1.8.0_65/bin/java -cp /root/.jenkins/plugins/maven-plugin/WEB-INF/lib/maven3-agent-1.5.jar:/usr/local/maven/apache-maven-3.3.9/boot/plexus-classworlds-2.5.2.jar:/usr/local/maven/apache-maven-3.3.9/conf/logging-jenkins-maven3-agent-Maven31Main /usr/local/maven/apache-maven-3.3.9/ /usr/local/apache-tomcat/apache-tomcat-9.0.0.M1/webapps/jenkins/WEB-INF/lib/remoting-2.53.2.jar /root/.jenkins/plugins/maven-plugin/WEB-INF/lib/maven31-interceptor-1.5.jar /root/.jenkins/plugins/maven-plugin/WEB-INF/lib/maven3-interceptor-commons-1.5.jar 37438
<===[JENKINS REMOTING CAPACITY]===>channel started
Executing Maven: -B -f /root/.jenkins/jobs/test_project1/workspace/pom.xml package
[INFO] Scanning for projects...
[WARNING]
[WARNING] Some problems were encountered while building the effective model for com.idcq.appServer:appServer:war:0.0.1-SNAPSHOT
[WARNING] 'dependencies.dependency.(groupId:artifactId:type:classifier)' must be unique: org.apache.lucene:lucene-analyzers-common:jar -> duplicate declaration of version 4.5.0 @ line 307, column 13
[WARNING] 'build.plugins.plugin.version' for org.apache.maven.plugins:maven-compiler-plugin is missing. @ line 16, column 12
[WARNING]
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING]
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
[INFO]
[INFO] -----
[INFO] Building appserver Maven Webapp 0.0.1-SNAPSHOT
[INFO] -----
[INFO] Downloading: https://nexus.codehaus.org/content/repositories/codehaus-snapshots/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2.6.pom
[INFO] Downloading: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/2.6/maven-resources-plugin-2.6.pom
```