

第1章 准备Maven环境

注意：下列操作都是在jenkins主机部署

1.安装maven

```
1 tar zxf apache-maven-3.3.9-bin.tar.gz -C /opt/
2 cd /opt/
3 ln -s apache-maven-3.3.9 maven
```

2.设置maven国内源

在mirror标签下添加mirror区块

```
1 vim /opt/maven/conf/settings.xml
2     <mirror>
3         <id>nexus-aliyun</id>
4         <mirrorOf>*</mirrorOf>
5         <name>Nexus aliyun</name>
6         <url>http://maven.aliyun.com/nexus/content/groups/public</url>
7     </mirror>
```

3.配置环境变量

```
1 echo 'export PATH=$PATH:/opt/maven/bin' >> /etc/profile
2 source /etc/profile
3 mvn -v
```

4.使用mvn打包测试项目

```
1 tar zxf hello-world-war.tar.gz
2 cd hello-world-war/
3 mvn package
```

5.检查是否生成war包

```
1 ll target
```

第2章 上传代码到gitlab

以下操作在gitlab服务器进行：

1.在gitlab中创建项目

Blank project	Create from template	Import project
Project name <input type="text" value="Java Helloworld"/>		
Project URL <input type="text" value="http://10.0.0.200/"/> <input type="text" value="dev"/>		Project slug <input type="text" value="java-helloworld"/>
Want to house several dependent projects under the same namespace? Create a group.		
Project description (optional) <div>Description format</div>		
Visibility Level ⓘ <input checked="" type="radio"/> <input type="lock"/> Private Project access must be granted explicitly to each user.		
<input type="radio"/> <input type="shield"/> Internal This project cannot be internal because the visibility of dev is private. To make this project internal, you must first change the visibility of the parent group.		
<input type="radio"/> <input type="globe"/> Public This project cannot be public because the visibility of dev is private. To make this project public, you must first change the visibility of the parent group.		
<input type="checkbox"/> Initialize repository with a README Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.		
<input type="button" value="Create project"/>		<input type="button" value="Cancel"/>

2.在gitlab主机提交测试项目

```
1 tar xzf hello-world-war.tar.gz
2 cd hello-world-war/
3 git remote remove origin
4 git init
5 git remote add origin git@10.0.0.200:dev/java-helloworld.git
6 git add .
7 git commit -m "Install commit"
8 git push -u origin master
```

3.gitlab上查看项目

J

Java Helloworld

Project ID: 6

🔔

☆ Star

0

🍴 Fork

0

Clone

🔗 Add license

🔗 1 Commit

🔗 1 Branch

🔗 0 Tags

📁 82 KB Files

master


java-helloworld /

+

History

Find file

Web IDE



Merge branch 'dev' into 'master'
cto authored just now

5f538c35

📖 README

📄 Add CHANGELOG

📄 Add CONTRIBUTING

🔧 Enable Auto DevOps

🔧 Add Kubernetes cluster

🔧 Set up CI/CD

Name	Last commit	Last update
dist	test	1 year ago
src/main/webapp	Install commit	9 minutes ago
.gitignore	test	1 year ago
README.md	test	1 year ago
pom.xml	test	1 year ago

第3章 Jenkins配置Maven项目

1.配置Maven目录

进入Jenkins的系统管理-->全局工具配置页面-->Maven安装

Maven

Maven 安装

新增 Maven

Maven

Name

Maven

MAVEN_HOME

/opt/maven/

☐ Install automatically

删除 Maven

2.新建Maven项目

输入一个任务名称

java-helloworld

» 必填项



Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build :



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.

3.gitlab项目里激活部署公钥

Java Helloworld

- Project
- Repository
- Issues
- Merge Requests
- CI / CD
- Operations
- Wiki
- Snippets
- Settings
- Repository

Key

Paste a machine public key here. Read more about how to generate it [here](#)

☐ Write access allowed

Allow this key to push to repository as well? (Default only allows pull access.)

Add key

Enabled deploy keys 1 Privately accessible deploy keys 0 Publicly accessible deploy keys 0

Deploy key	Project usage	Created
jenkis 0b:66:7a:c2:ea:2f:cf:7c:bd:6f:cf:d0:7f:a3:4c:aa	Current project dev / h5game	22 hours ago

3.配置项目的gitlab地址

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 `git@10.0.0.200:dev/java-helloworld.git`

Credentials `- 无 -` 添加

高级...

Add Repository

Branches to build

Branch Specifier (blank for 'any') `*/master`

Add Branch

源码库浏览器 (自动)

Additional Behaviours 新增

☐ Mercurial
☐ Subversion

4.设置打包命令

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

Build

Root POMpom.xml

Goals and optionsclean package

注意：clean和package是两个参数，先执行clean再执行package

高级...

5.测试构建是否成功

点击立即构建，然后查看jenkins的工作目录下是否成功生成了war包

```
1 [root@jenkins-201 ~]# ll /var/lib/jenkins/workspace/java-helloworld/target/
2 总用量 4
3 drwxr-xr-x 4 root root    54 8月  7 08:13 hello-world-war-1.0.0
4 -rw-r--r-- 1 root root 2403 8月  7 08:13 hello-world-war-1.0.0.war
5 drwxr-xr-x 2 root root    28 8月  7 08:13 maven-archiver
6 [root@jenkins-201 ~]#
```

6.将war包发送给web服务器

```
1 cd /var/lib/jenkins/workspace/java-helloworld/target/
2 scp hello-world-war-1.0.0.war 10.0.0.7:/tmp/
```

第4章 安装部署Tomcat

注意：以下操作在web服务器进行

1.安装java环境

```
1 yum install java -y
```

2.安装Tomcat

```
1 tar xf apache-tomcat-8.0.27.tar.gzls -C /opt/
2 cd /opt
3 ln -s apache-tomcat-8.0.27 tomcat
```

2.拷贝测试项目并用tomcat启动

```
1 rm -rf /opt/tomcat/webapps/*
2 cp /tmp/hello-world-war-1.0.0.war /opt/tomcat/webapps/ROOT.war
3 /opt/tomcat/bin/startup.sh
```

3.测试访问

← → ↻ ⓘ 不安全 | 10.0.0.7:8080

Hello World! v1.0

It is now Fri May 15 16:20:26 CST 2020

You are coming from 10.0.0.1

4.清理环境

测试完成后我们将war包删掉，后续我们使用jenkins自动部署

```
1 rm -rf /opt/tomcat/webapps/ROOT.war
```

第5章 Jenkins配置参数化构建

1.编写发布脚本

```
1 cat > /scripts/jenkins/java_deploy.sh << 'EOF'
2 #!/bin/bash
3
4 PATH_CODE=/var/lib/jenkins/workspace
5 PATH_WEB=/opt/tomcat/webapps
6 IP=10.0.0.7
7
8 #拷贝war包发送到web服务器代码目录
9 code_scp(){
10     ssh ${IP} "mkdir ${PATH_WEB}/java-${git_version} -p"
11     scp ${PATH_CODE}/java-helloworld/target/*.war ${IP}:${PATH_WEB}/java-
12     ${git_version}
13 }
```

```
13
14 #web服务器解压代码
15 code_unzip(){
16     ssh ${IP} "cd ${PATH_WEB}/java-${git_version} && unzip *.war && rm -rf
17     *.war"
18 }
19 #创建代码软链接
20 code_ln(){
21     ssh ${IP} "cd ${PATH_WEB} && rm -rf ROOT && ln -s java-${git_version} ROOT"
22 }
23
24 #重启tomcat
25 restart_tomcat(){
26     ssh ${IP} "cd /opt/tomcat/bin && ./shutdown.sh && ./startup.sh"
27 }
28
29 main(){
30     code_scp
31     code_unzip
32     code_ln
33 }
34
35 #选择发布还是回滚
36 if [ "${deploy_env}" == "deploy" ]
37 then
38     ssh ${IP} "ls ${PATH_WEB}/java-${git_version}" >/dev/null 2>&1
39     if [ $? == 0 -a ${GIT_COMMIT} == ${GIT_PREVIOUS_SUCCESSFUL_COMMIT} ]
40     then
41         echo "java-${git_version} 已部署,不允许重复构建"
42         exit
43     else
44         main
45         restart_tomcat
46     fi
47 elif [ "${deploy_env}" == "rollback" ]
48 then
49     code_ln
50     restart_tomcat
51 fi
52 EOF
```

2.配置参数化构建

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

☒ This project is parameterized

Git Parameter

Name:

Description:

[Plain text] [预览](#)

Parameter Type:

Default Value:

[高级...](#)

Choice Parameter

名称:

选项:

描述:

[Plain text] [预览](#)

[添加参数](#)

[保存](#) [应用](#)

最后还需要配置一下git仓库地址,注意需要修改拉取的版本的变量为 `$git_version`

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: [添加](#)

[高级...](#)
[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any'):

[Add Branch](#)

jenkins构建查看会发现已经有了选项,但是因为还没提交代码,所以版本号还没有:

返回面板

状态

修改记录

工作空间

Build with Parameters

删除 Maven project

配置

模块

收藏夹

打开 Blue Ocean

重命名

Maven project java-helloworld

需要如下参数用于构建项目:

git_version

发布新版本

deploy_env deploy

deploy: 发布版本
rollback: 回滚版本

开始构建

Build History 构建历史

find

#1 2020-8-7 上午8:13

RSS 全部 RSS 失败

3.配置构建后操作

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

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.

Execute shell

命令 `bash -x /scripts/jenkins/java_deploy.sh`

查看 可用的环境变量列表

高级...

Add post-build step

4.项目打标签测试

删除以前的标签:

```
1 git tag
2 git tag -d v1.0 v2.0 v3.0 v4.0
```

发布v2.0版本

```
1 cd hello-world-war/src/main/webapp/
2 git checkout master
3 git pull
4 git checkout -b v2.0
5 vim src/main/webapp/index.jsp
6 git status
7 git add .
8 git commit -m "v2.0 稳定版"-
9 git push -u origin v2.0
```

发布v3.0版本

```
1 cd hello-world-war/src/main/webapp/
2 git checkout master
3 git pull
4 git checkout -b v3.0
5 vim src/main/webapp/index.jsp
6 git status
7 git add .
8 git commit -m "v3.0 稳定版"
9 git push -u origin v3.0
```

5.测试发版

发布v2.0版本：



The image shows the Jenkins web interface for a project named 'java-helloworld'. The left sidebar contains a list of actions: '返回面板' (Return to dashboard), '状态' (Status), '修改记录' (Edit record), '工作空间' (Workspace), 'Build with Parameters' (highlighted with a red box), '删除 Maven project' (Delete Maven project), '配置' (Configure), '模块' (Module), '收藏夹' (Favorites), '打开 Blue Ocean' (Open Blue Ocean), and '重命名' (Rename). The main content area is titled 'Maven project java-helloworld' and displays the '需要如下参数用于构建项目:' (Need the following parameters for building the project:). The 'git_version' parameter is set to 'v2.0' (highlighted with a red box). The 'deploy_env' parameter is set to 'deploy' (highlighted with a red box). Below these parameters, there is a '开始构建' (Start build) button. The interface also includes a '发布新版本' (Release new version) button and a 'rollback' button.

web服务器查看:

←

→

↻

ⓘ

不安全

10.0.0.7:8080

Hello World! v2.0

It is now Fri Aug 07 09:27:45 CST 2020

You are coming from 10.0.0.1

目录查看:

```
1 [root@web-7 ~]# ll /opt/tomcat/webapps/
2 总用量 0
3 drwxr-xr-x 4 root root 54 8月  7 09:26 java-v2.0
4 lrwxrwxrwx 1 root root  9 8月  7 09:27 ROOT -> java-v2.0
```

测试发布v3.0版本:

Jenkins

Jenkins

java-helloworld

返回面板

状态

修改记录

工作空间

Build with Parameters

删除 Maven project

配置

模块

收藏夹

打开 Blue Ocean

重命名

Maven project java-helloworld

需要如下参数用于构建项目:

git_version

v2.0

v3.0

发布新版本

deploy_env

deploy

deploy: 发布版本

rollback: 回滚版本

开始构建

浏览器查看:

←

→

↺

ⓘ

不安全

10.0.0.7:8080

Hello World! v3.0

It is now Fri Aug 07 09:39:44 CST 2020

You are coming from 10.0.0.1

代码目录查看

```
1 [root@web-7 ~]# ll /opt/tomcat/webapps/
2 总用量 0
3 drwxr-xr-x 4 root root 54 8月 7 09:26 java-v2.0
4 drwxr-xr-x 4 root root 54 8月 7 09:39 java-v3.0
5 lrwxrwxrwx 1 root root 9 8月 7 09:39 ROOT -> java-v3.0
```

6.测试回滚

当前版本为v3.0,我们选择回滚到v2.0

Jenkins

Jenkins > java-helloworld >

返回面板

状态

修改记录

工作空间

Build with Parameters

删除 Maven project

配置

模块

收藏夹

打开 Blue Ocean

重命名

Maven project java-helloworld

需要如下参数用于构建项目:

git_version

v2.0

v3.0

deploy_env

rollback

发布新版本

deploy: 发布版本

rollback: 回滚版本

开始构建

web浏览器查看发现已经回滚成功了:



不安全

10.0.0.7:8080

Hello World! v2.0

It is now Fri Aug 07 09:41:08 CST 2020

You are coming from 10.0.0.1

查看代码目录：

```
1 [root@web-7 ~]# ll /opt/tomcat/webapps/  
2 总用量 0  
3 drwxr-xr-x 4 root root 54 8月 7 09:26 java-v2.0  
4 drwxr-xr-x 4 root root 54 8月 7 09:39 java-v3.0  
5 lrwxrwxrwx 1 root root 9 8月 7 09:41 ROOT -> java-v2.0
```

第7章 sonar添加java项目

1.SonarQube添加新项目



2.创建新令牌

欢迎使用SonarQube!
让我们分析一个新项目。

1 创建一个令牌

java: a82e776ea9fe8efa91c53d041d969ee23d2a27d1 ✖

此令牌用于执行分析时认证时使用，如果这个令牌存在问题，可以随时在你的用户账号下取消这个令牌。

继续

3.选择java项目

欢迎使用SonarQube!
让我们分析一个新项目。

跳过教程
随时在帮助里找到它

1 创建一个令牌

✔ java: a82e776ea9fe8efa91c53d041d969ee23d2a27d1

2 分析你的项目

项目的主要语言是什么?

Java

C# 或 VB.NET

其他 (JS, Python, PHP, ...)

主要使用Java开发: 使用了什么构建技术?

Maven

Gradle

在你的电脑上使用Maven执行SonarQube扫描

使用Maven执行SonarQube分析是非常简单的。只需要在你的项目目录下执行如下命令。

```
mvn sonar:sonar \
-Dsonar.host.url=http://10.0.0.203:9000 \
-Dsonar.login=a82e776ea9fe8efa91c53d041d969ee23d2a27d1
```

复制

请访问 [SonarQube Maven扫描器官方文档](#) 了解更多信息。

分析完成后，可以通过日志最后的URL浏览你的项目。

完成教程

4.将命令写入sonar-scanner配置文件

```
1 cat > /opt/sonar-scanner/conf/sonar-scanner.properties <<'EOF'
2 sonar.host.url=http://10.0.0.203:9000
3 sonar.login=a82e776ea9fe8efa91c53d041d969ee23d2a27d1
4 sonar.sourceEncoding=UTF-8
5 EOF
```

第8章 jenkins配置sonar

1.配置构建后步骤

参数命令:


```
1 sonar.projectName=${JOB_NAME}
2 sonar.projectKey=java
3 sonar.sources=.
```


2.提交新版本发版测试


```
1 cd hello-world-war/src/main/webapp/
2 git checkout master
3 git pull
4 git checkout -b v4.0
5 vim src/main/webapp/index.jsp
6 git status
7 git add .
8 git commit -m "v4.0 稳定版"-
9 git push -u origin v4.0
```


3.jenkins发版测试


Jenkins ▶ java-helloworld ▶


 返回面板


 状态


 修改记录


 工作空间


 Build with Parameters


 删除 Maven project


 配置

 模块

 收藏夹

 SonarQube

 打开 Blue Ocean

 重命名

Maven project java-helloworld

需要如下参数用于构建项目:

git_version

发布新版本

deploy_env

deploy: 发布版本
rollback: 回滚版本

开始构建

4.检查发版状态

web服务状态:



不安全 | 10.0.0.7:8080

Hello World! v4.0

It is now Fri Aug 07 09:58:38 CST 2020

You are coming from 10.0.0.1

sonar状态:

