**Jenkins配置自动化构建**

**持续集成概述**

随着软件开发复杂度的不断提高，团队开发成员间如何更好地协同工作以确保软件开发的质量已经慢慢成为开发过程中不可回避的问题。尤其是近些年来，敏捷（Agile） 在软件工程领域越来越红火，如何能再不断变化的需求中快速适应和保证软件的质量也显得尤其的重要。

持续集成正是针对这一类问题的一种软件开发实践。它倡导团队开发成员必须经常集成他们的工作，甚至每天都可能发生多次集成。而每次的集成都是通过自动化的构建来验证，包括自动编译、发布和测试，从而尽快地发现集成错误，让团队能够更快的开发内聚的软件。

持续集成的核心价值在于：

1. 持续集成中的任何一个环节都是自动完成的，无需太多的人工干预，有利于减少重复过程以节省时间、费用和工作量；
2. 持续集成保障了每个时间点上团队成员提交的代码是能成功集成的。换言之，任何时间点都能第一时间发现软件的集成问题，使任意时间发布可部署的软件成为了可能；
3. 持续集成还能利于软件本身的发展趋势，这点在需求不明确或是频繁性变更的情景中尤其重要，持续集成的质量能帮助团队进行有效决策，同时建立团队对开发产品的信心。

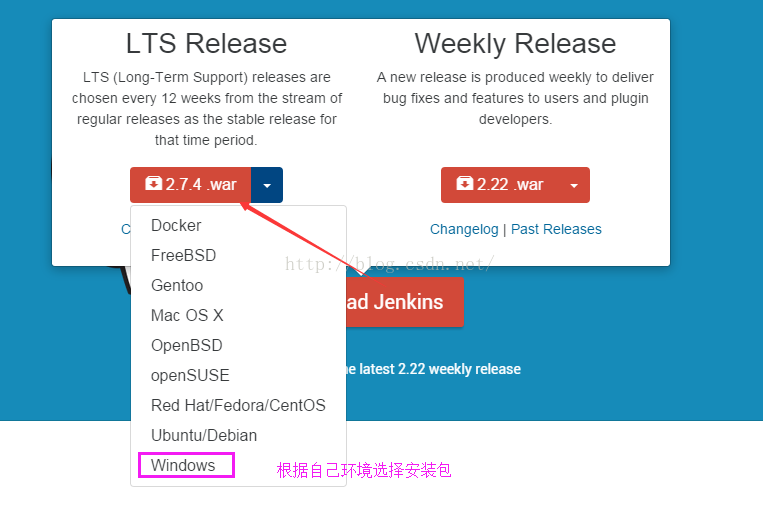
官网下载地址：https://jenkins.io/index.html

**1.下载安装Jenkins**

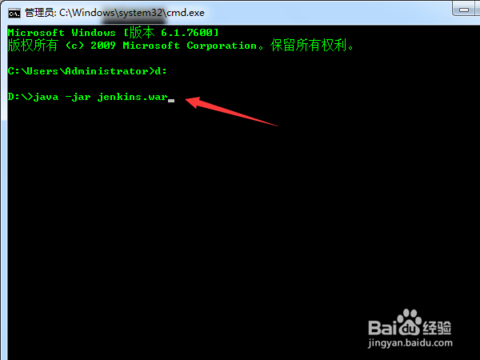
（1）点击Download Jenkins进入下载页



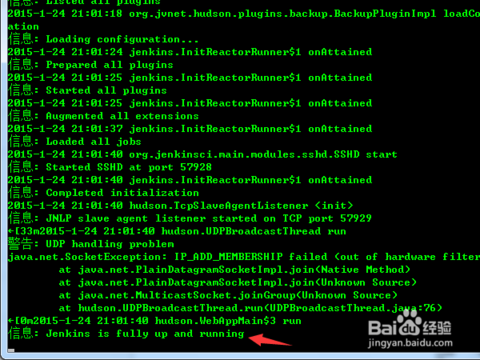
（2）根据自己运行环境选择安装包



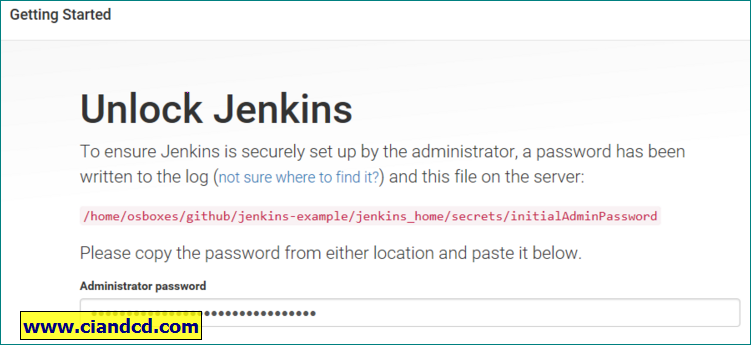
解压jenkins包，把解压之后放到一个目录下后为jenkins.war，是把jenkins.war拷贝到D盘下，打开cmd命令窗口，输入java -jar jenkins.war,后进行回车

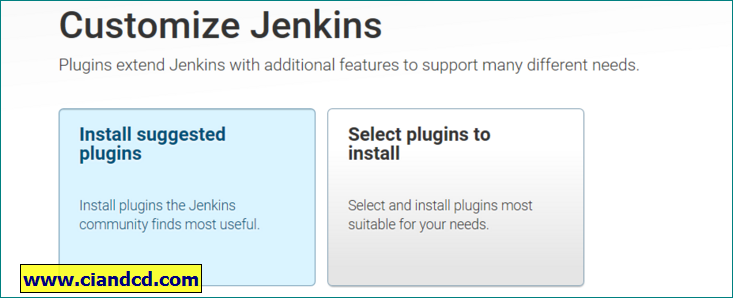


jenkins运行成功后，看到jenkins is fully up and running 运行完成

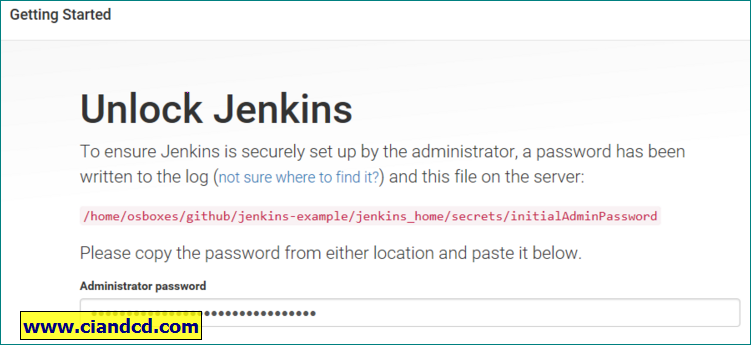


在网页输入localhost，在加默认的端口为8080,等待网页打开，如图所示，jenkins搭建完成，接下来是jenkins服务

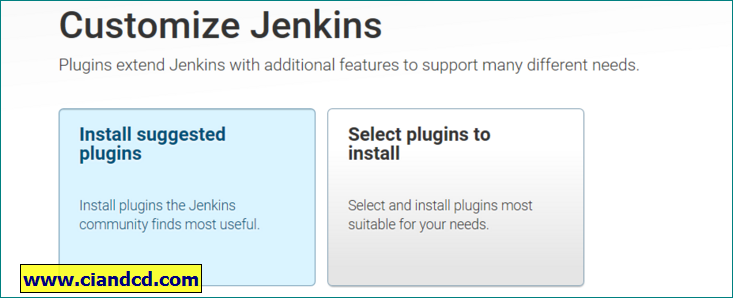




[第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins\_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

[](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

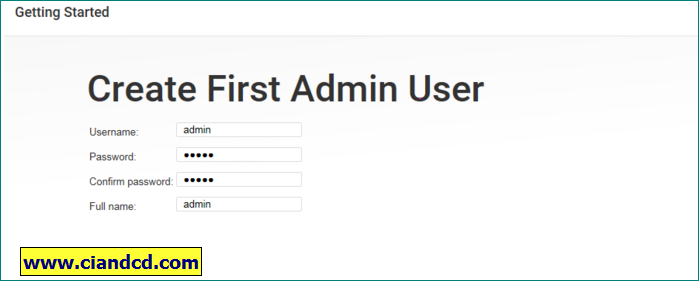
[然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

[](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

[有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

[如下，选择跳过plugins以后再安装。](第一次启动Jenkins需要配置初始管理员密码，手动将/home/osboxes/github/jenkins-example/jenkins_home/secrets/initialAdminPassword中的内容复制，然后粘贴到管理员密码框中。然后需要等到几十分钟，等Jenkins初始化结束进入下一步， 选择安装建议的plugins。有的时候由于网络问题，会提示跳过plugin的安装。如果需要离线安装plugins，可以使用脚本https://gist.github.com/micw/e80d739c6099078ce0f3来确保plugin所依赖的plugins也被自动的安装。如下，选择跳过plugins以后再安装。)

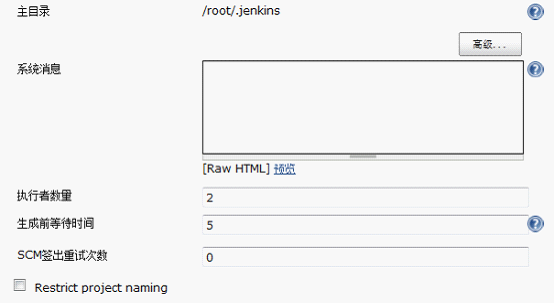
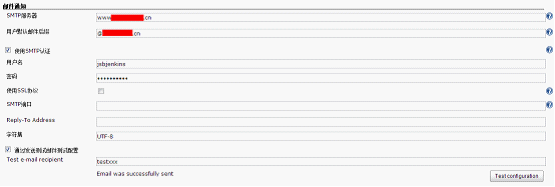
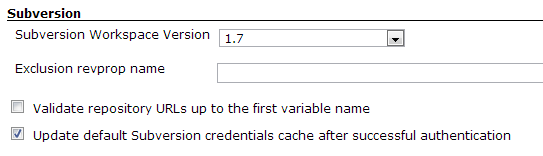
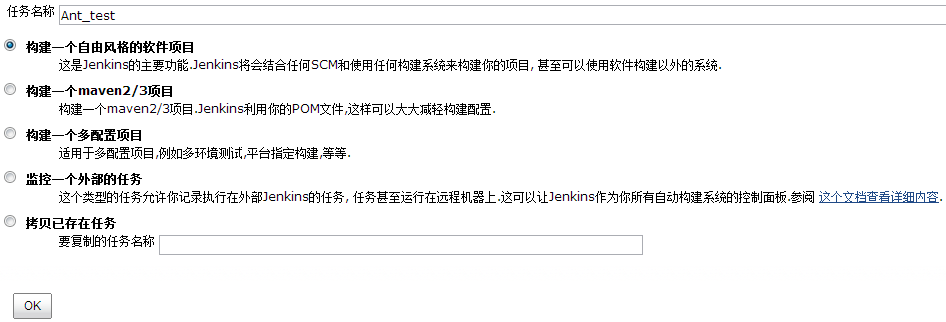
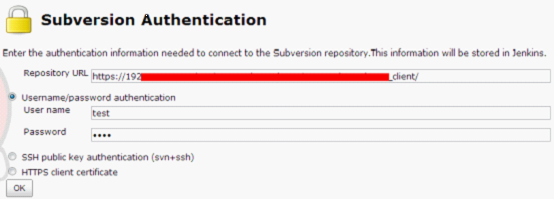
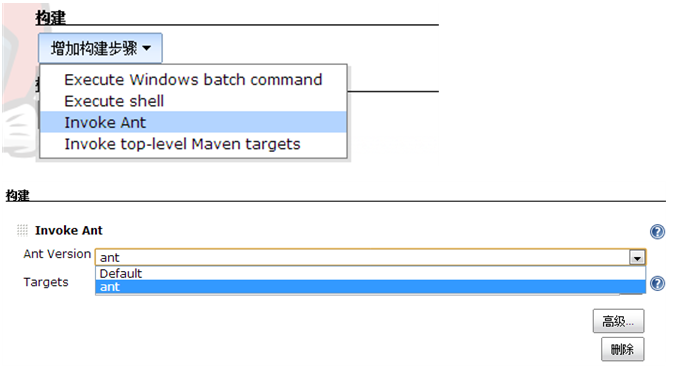
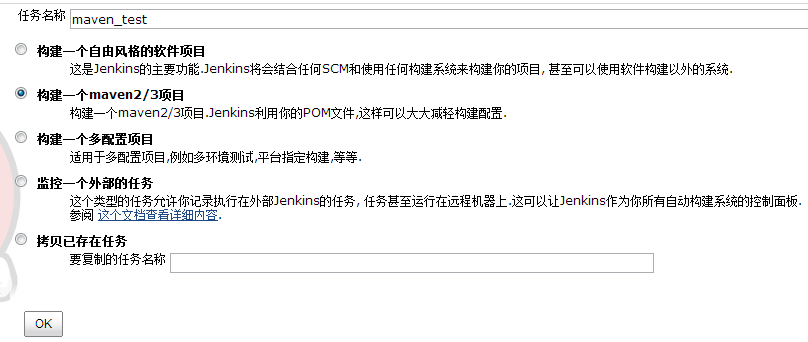
然后会提示设置管理员用户和密码，如下：



设置完成之后就可以啦



安装后自动创建了一个windows服务：Jenkins，默认使用的端口是8080，如果需要修改，打开安装目录下的jenkins.xml文件，修改  <arguments>-Xrs -Xmx256m -Dhudson.lifecycle=hudson.lifecycle.WindowsServiceLifecycle -jar "%BASE%\jenkins.war" --httpPort=8081</arguments>后保存，启动jenkins服务。

1. **3.1 系统管理**
2. 在已运行的Jenkins主页中，点击左侧的系统管理进入如下界面：
3. 
4. 图3 Jenkins系统管理
5. **3.1.1 提示信息**
6. Ps：版本不同提示的消息有可能不同
7. **3.1.1.1 Utf-8编码**
8. Your [**Container**](http://lib.csdn.net/base/docker) doesn't use UTF-8 to decode URLs. If you use non-ASCII characters as a job name etc, this will cause problems. See [Containers](http://wiki.jenkins-ci.org/display/JENKINS/Containers) and [Tomcat i18n](http://wiki.jenkins-ci.org/display/JENKINS/Tomcat#Tomcat-i18n) for more details.
9. Jenkins建议在tomcat中使用utf-8编码，配置tomcat下conf目录的server.xml文件
10. http://images.cnitblog.com/blog/529896/201308/12163628-5e18c80f1c44409fb584a8a618299e12.png
11. 图4 URIEncoding编码设置
12. Ps：如果Job的控制台中文输出乱码，请将URIEncoding=”utf-8”更改为useBodyEncodingForURI="true"
13. **3.1.1.2 新的版本**
14. New version of Jenkins (1.518.JENKINS-14362-jzlib) is available for download (changelog).
15. 提示有新的版本可以下载了,喜欢更新的点击download去下载吧！
16. **3.1.1.3 安全设置**
17. http://images.cnitblog.com/blog/529896/201308/12163634-a71e6cae19ce4f059fb19d845b7f5db7.png
18. 图5 安全提示消息
19. 詹金斯允许网络上的任何人代表您启动进程。考虑至少启用身份验证来阻止滥用。点击Dismiss忽略该消息,点击Setup Security进入设置界面.详细设置请参考 Configure Global Security(安全设置) 章节
20. **3.1.2 系统设置**
21. 在已运行的Jenkins主页中，点击左侧的系统管理—>系统设置进入如下界面：
22. 
23. 图6 系统设置界面
24. **3.1.2.1 JDK、Maven、Ant配置**
25. 配置一个JDK、Ant、Maven实例，请在每一节下面单击Add(新增) 按钮，这里将添加实例的名称和绝对地址。下图描述了这两个部分。
26. 
27. 图7 JDK配置界面
28. JDK别名：给你看的，随便你自己，叫阿猫阿狗都可以
29. JAVA\_HOME：这个是本机JDK的安装路径（错误的路径会有红字提示你的）
30. 自动安装：不推荐这个选项
31. 后面Ant与Maven的配置是一样的，JDK去[**Oracle**](http://lib.csdn.net/base/oracle)官网下载，Ant与Maven去apache官网下载
32. Ps：每个文本框后面都有个问号，点击问号就会出现帮助信息
33. **3.1.2.2 邮件通知配置**
34. **3.1.2.2.1 配置发件人地址**
36. 
37. 图8 发件人地址配置界面
38. System Admin e-mail address：Jenkins邮件发送地址，如果你这个没有配置，等着发邮件的时候报错吧，当时我也是这儿没有配置，郁闷了我一周的时间。⊙﹏⊙b汗
39. **3.1.2.2.2 配置邮件通知**
40. 
41. 图9 邮件通知
42. 这个就非常的简单了，根据的的邮箱提供者的参数配置就行了。
43. Ps：小技巧：用户默认邮件后缀配置了后，以后你填写邮件地址只需要@之前的就行了
44. **3.1.2.3 Subversion配置**
45. 
46. 图10 Subversion配置
47. Subversion Workspace Version：Subversion 的版本号，选择你对应的版本号就行了
48. **3.1.3 Configure Global Security(安全设置)**
49. 在已运行的Jenkins主页中，点击左侧的系统管理—>Configure Global Security进入如下界面：
50. 
51. 图11 安全设置界面
52. 设置如上图，保存后系统管理中就出现管理用户的选项。页面右上角也会出现登录/注册的选项。
53. **3.1.4 管理用户设置**
54. 在右上角点击注册
55. 
56. 图12 注册用户界面
57. 点击sign up按钮，提示你现在已经登录.返回首页.
58. 登录后和匿名账号看到的首页有几点不同，如下图红框所示：
59. 
60. 图13 用户登录界面
61. **3.1.5 管理插件设置**
62. 建议先阅读[Jenkins插件](http://www.cnblogs.com/zz0412/p/jenkins02.html#_Jenkins插件)章节，在回来安装如下所示的插件。这个插件将生成的构件（war或者ear）部署到主流的服务器上。
63. 插件名称：[Deploy Plugin](https://wiki.jenkins-ci.org/display/JENKINS/Deploy+Plugin)
64. 插件介绍：This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build
65. **3.2 项目构建设置**
66. **3.2.1 构建自由风格的Job**
67. **3.2.1.1 新建自由风格构建任务**
68. 在已运行的Jenkins主页中，点击左侧的新建Job进入如下界面：
69. 
70. 图14 新建Job界面
71. 这时，需要为新的构建任务指定一个名称。（这里输入的任务名称为：Ant\_test）这里有几种的任务类型可供选择，鉴于初步介绍，先选择构建一个自由风格的软件项目。对于其他的类型,经常使用的是拷贝已存在任务;这主要为了能在现有的任务基础上新建任务。点击OK按钮，
72. **3.2.1.2 构建任务配置**
73. **3.2.1.2.1 源码管理配置**
74. 演示是使用Subversion的链接，在Repository URL中输入你的项目链接，如果没有权限则会提示如下图：
76. 图15 Subversion配置界面
77. 点击 enter credential 输入用户名和密码(我猜大家一般都是使用的用户名和密码登陆的)
78. 
79. 图16 Subversion权限认证界面
80. Ps：svn的用户名和密码设置了是没有办法在web界面修改的。如果要修改则先去Jenkins目录删除hudson.scm.SubversionSCM.xml文件（点到为止，毕竟这只是入门教程）
81. **3.2.1.2.2 构建触发器**
82. 在其他项目构建完成后才执行构建：指定的项目完成构建后，触发此项目的构建。
83. Poll SCM ：这是CI 系统中常见的选项。当您选择此选项，您可以指定一个定时作业表达式来定义Jenkins每隔多久检查一下您源代码仓库的变化。如果发现变化，就执行一次构建。例如，表达式中填写0,15,30,45 \* \* \* \*将使Jenkins每隔15分钟就检查一次您源码仓库的变化。
84. Build periodically ：此选项仅仅通知Jenkins按指定的频率对项目进行构建，而不管SCM是否有变化。如果想在这个Job中运行一些测试用例的话，它就很有帮助。
85. **3.2.1.2.3 Ant构建配置**
86. 因为我的项目是用ant脚本实现的编译和打包，所以我选择的是Invoke Ant，Ant Version选择你Ant配置的那个名字，注意不要选择default喔，那个选择了没有用。
87. 
88. 图17 Ant构建配置界面
89. 如果你的构建脚本build.xml不在workspace根目录、或者说你的构建脚本不叫build.xml。那么需要在高级里设置Build File选项的路径，指明你的脚本。注意：是相对路径
90. 部署请参考：[war文件部署](http://www.cnblogs.com/zz0412/p/jenkins02.html#_War文件部署)章节
91. **3.2.2 构建Maven风格的Job**
92. **3.2.2.1 新建Maven构建任务**
93. 
94. 图18 新建Job界面
95. 这时，需要为新的构建任务指定一个名称。（这里输入的任务名称为：maven\_test）这里有几种的任务类型可供选择，鉴于初步介绍，先选择构建一个maven2/3项目。对于其他的类型,经常使用的是拷贝已存在任务;这主要为了能在现有的任务基础上新建任务。点击OK按钮，
96. **3.2.2.2 构建任务配置**

 （2）暂时不用的相关设置



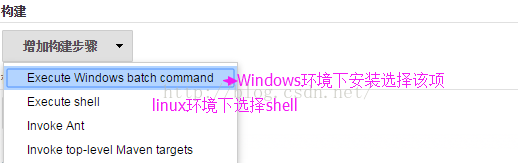
（3）源码管理



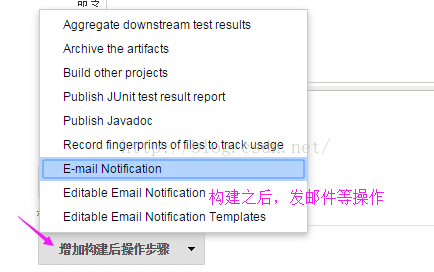
（4）构建触发器



（5）构建——增加构建步骤



（6）构建后操作——增加构建后操作步骤



（7）点击保存，跳转到下图，一个基本job项目建立；



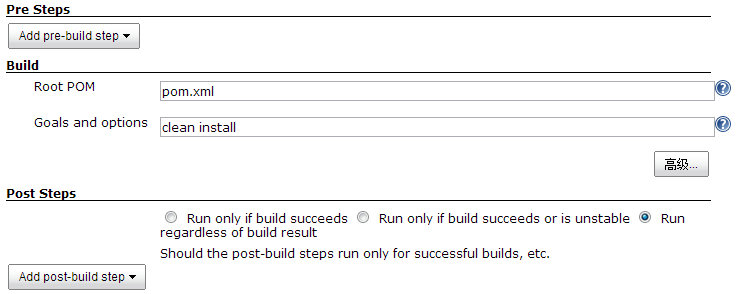
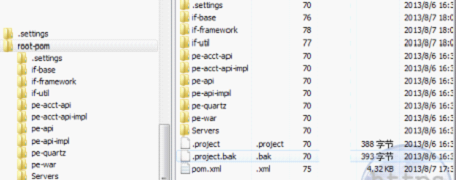
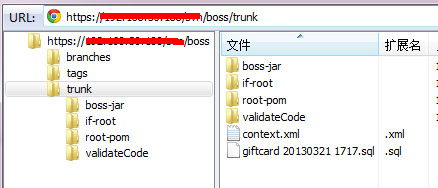
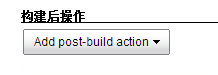
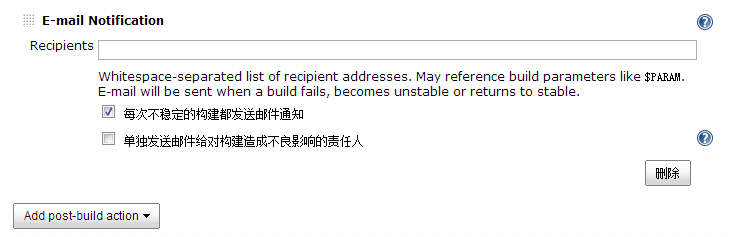
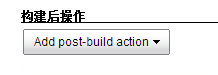
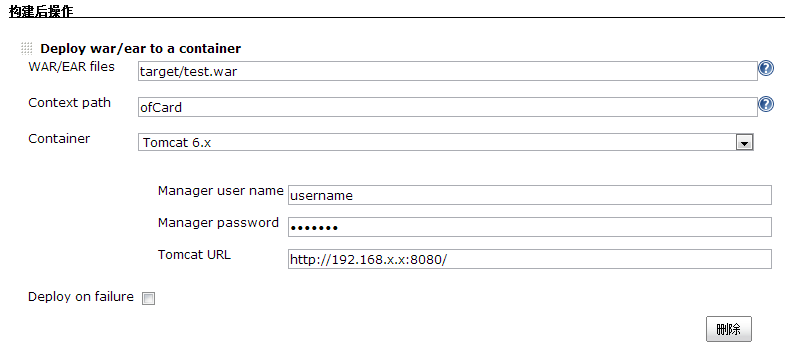
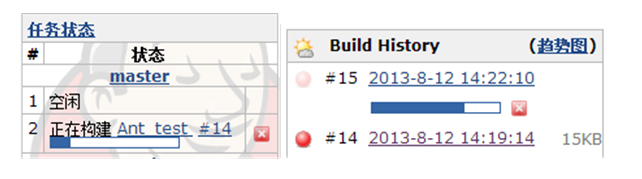
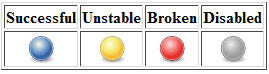
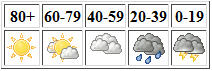
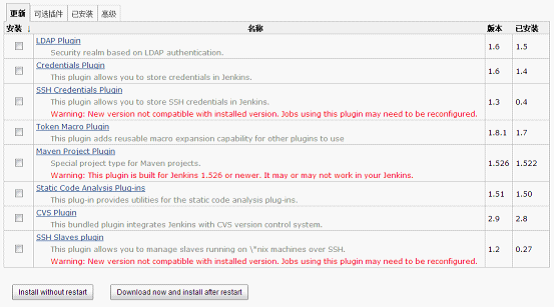
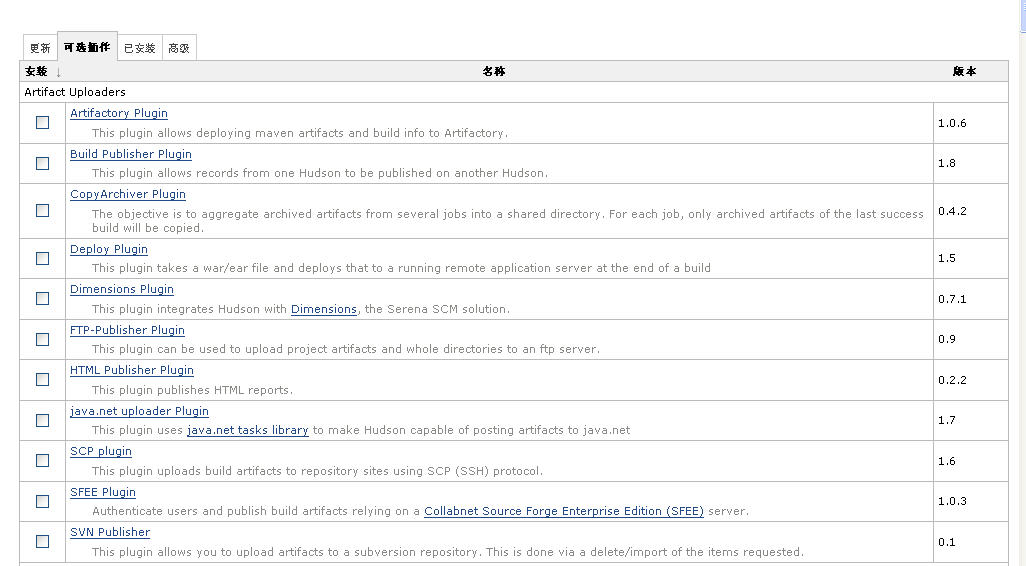
**4.构建项目——左侧“立即构建”**

****

**5.构建之后查看构建结果**

         点击构建历史，点击选择——ConsoleOutput控制台输出，见图：



1. Ps：svn的用户名和密码设置了是没有办法在web界面修改的。如果要修改则先去Jenkins目录删除hudson.scm.SubversionSCM.xml文件（点到为止，毕竟这只是入门教程）
2. **3.2.2.2.2 构建触发器**
3. 在其他项目构建完成后才执行构建：指定的项目完成构建后，触发此项目的构建。
4. Poll SCM ：这是CI 系统中常见的选项。当您选择此选项，您可以指定一个定时作业表达式来定义Jenkins每隔多久检查一下您源代码仓库的变化。如果发现变化，就执行一次构建。例如，表达式中填写0,15,30,45 \* \* \* \*将使Jenkins每隔15分钟就检查一次您源码仓库的变化。
5. Build periodically ：此选项仅仅通知Jenkins按指定的频率对项目进行构建，而不管SCM是否有变化。如果想在这个Job中运行一些测试用例的话，它就很有帮助。
6. **3.2.2.2.3 Maven构建设置**
7. 
8. 图22 Maven构建配置界面
9. 2013-08-22补充Goals and options ：clean install  -Dmaven.test.skip=true　　#加入了跳过测试的代码
10. Root POM:填写你项目的pom.xml文件的位置，注意：是相对位置，如果该文件不存在，会有红色字提示。
11. 部署请参考：[war文件部署](http://www.cnblogs.com/zz0412/p/jenkins02.html#_War文件部署)章节
12. **3.2.2.2.4 构建maven项目的心得**
13. 使用Jenkins构建maven项目的一点小心得：
14. maven项目的构建是比较麻烦的，如果你的项目是下图这种结构。那么恭喜你！你新建一个job就可以了，因为只有一个根。如果你的svn地址是：https://192.xxx/Pe\_Project/root-pom，那么Root POM只需要保持默认就行了，因为Jenkins可以再workspace目录下面找到pom.xml文件
15. 如果你的svn地址是：https://192.xxx/Pe\_Project，那么Root POM需要指定为root-pom/pom.xml，因为Jenkins可以再workspace/root-pom目录下面找到pom.xml文件
16. 
17. 图23 Maven项目结构界面1
18. 上面这种方法打包的时候非常简单，但是用eclipse开发的时候你就不右键run as —>tomca启动了，如果你想使用这种方式，将tomcat换成jetty即可。
20. 如果你的项目是下图这种结构，那么非常悲剧的告诉你，你要建立好几个job来构建这一个项目，因为这个项目有4个根。
21. 
22. 图24 Maven项目结构界面2
23. 上面这种方法打包的时候比较麻烦，但是用eclipse开发的时候你就可以使用右键run as —>tomca启动了
25. **3.2.3 邮件通知设置**
26. 
27. 图25 构建后操作界面
28. 选择Add post-build action，然后选择E-mail Notification，如下图：
29. 
30. 图26 收件人列表界面
31. 在Recipients中输入收件人邮件地址，如果用多个收件人用“,”英文逗号隔开
32. **3.2.4 War文件部署设置**
33. 首先你必须安装好[Deploy Plugin](https://wiki.jenkins-ci.org/display/JENKINS/Deploy+Plugin)插件，然后在tomcat的conf目录配置tomcat-users.xml文件，在<tomcat-users>节点里添加如下内容：
34. <role rolename="manager-gui"/>
35. <role rolename="manager-script"/>
36. <role rolename="manager-jmx"/>
37. <role rolename="manager-status"/>
38. <user username="username" password="password" roles="manager-gui,manager-script,manager-jmx,manager-status"/>
39. 引号里的username和password可以随便替换，待会要用的。
40. 好了，回到Jenkins项目配置页面：
41. 
42. 图27 构建后操作界面
43. 选择Add post-build action，然后选择Deploy war/ear to a container，如下图：
44. 
45. 图28 远程部署配置界面
46. WAR/EAR files：war文件的存放位置，如：target/test.war 注意：相对路径，target前是没有/的。
47. Context path：访问时需要输入的内容，如ofCard访问时如下：http://192.168.x.x:8080/ofCard/如果为空，默认是war包的名字。
48. Container：选择你的web容器，如tomca 6.x
49. Manager user name：填入tomcat-users.xml配置的username内容
50. Manager password：填入tomcat-users.xml配置的password内容
51. Tomcat URL：填入http://192.168.x.x:8080/
52. Deploy on failure：构建失败依然部署，一般不选择
53. 注意：虽然这种部署方法可能会导致tomcat加载时出现卡死的现象。但是也是最简单的部署方式。如果卡死了重启下就好了，将tomcat的java内存参数调高可以解决这个问题。
54. 最后不要忘记点击保存喔。
55. 好了！到此一个项目的获取源码，打包，远程部署，邮件通知就完成了。
56. **3.3 监控**
57. 当任务一旦运行，您将会看到这个任务正在队列中的仪表板和当前工作主页上运行。这两种显示如下。
58. 
59. 图29 主页监控（左），项目监控（右）
60. 一旦构建完成后，完成后的任务将会有三个地方进行显示。
61. 你可以在Jenkins的控制面板上看到它，如下图。
62. http://images.cnitblog.com/blog/529896/201308/12164320-7fa46061ee2147d3a66655ce878609ed.png
63. 图30 主页项目界面
65. 在上面展示的截图中，您将注意到有两个图标描述当前作业的状态。S栏目代表着“最新构建状态”，W栏目代表着“构建稳定性”。Jenkins使用这两个概念来介绍一个作业的总体状况：
66. 构建状态:下图中分级符号概述了一个Job新近一次构建会产生的四种可能的状态：
67. Successful:完成构建，且被认为是稳定的。
68. Unstable:完成构建，但被认为不稳定。
69. Failed:构建失败。
70. Disabled:构建已禁用。
71. 
72. 图31 构建状态界面
73. 构建稳定性: 当一个Job中构建已完成并生成了一个未发布的目标构件，如果您准备评估此次构建的稳定性，Jenkins会基于一些后处理器任务为构建发布一个稳健指数 (从0-100 )，这些任务一般以插件的方式实现。它们可能包括单元测试(JUnit)、覆盖率(Cobertura )和静态代码分析(FindBugs)。分数越高，表明构建越稳定。下图中分级符号概述了稳定性的评分范围。任何构建作业的状态(总分100)低于80分就是不稳定的。
74. 
75. 图32 构建稳定性界面
76. 你也可以在当前Job主界面上看到它，如下图左下部分
77. 
78. 图33 项目主界面
79. 当前作业主页上还包含了一些有趣的条目。左侧栏的链接主要控制Job的配置、删除作业、构建作业。右边部分的链接指向最新的项目报告和构件。
80. 通过点击构建历史（Build History）中某个具体的构建链接，您就能跳转到Jenkins为这个构建实例而创建的构建主页上。如下图
81. 
82. 图34 构建历史界面
83. 如果你想通过视图输出界面来监控当前任务的进展情况。你可以单击Console Output（控制台输出）。如果工作已完成，这将显示构建脚本产生的静态输出；如果作业仍然在运行中，Jenkins将不断刷新网页的内容，以便您可以看到它运行时的输出。如下图：
84. 
85. 图35 控制台输出界面
86. 4 Jenkins插件
87. 从Jenkins现有的功能扩展或开发者们为Jenkins提供的新功能都可以称之为Jenkins插件。有些插件可以无缝添加到您的构建过程，而其它，诸如除CVS和Subversion的SCM插件则需要源代码控制系统的支持。
88. **4.1 Jenkins插件安装**
89. Jenkins 插件管理器允许您安装新的插件，和更新您Jenkins服务器上的插件。管理者将连接到联机资料库，检索可用的和已更新的插件。如果您的Jenkins服务器 无法直接连接到外部资源，您可以从Jenkins网站上下载。
90. 在已运行的Jenkins主页中，点击左侧的系统管理—>管理插件进入如下界面：
91. 
92. 图36 插件管理界面
93. 它包含四个标签：
94. 更新:清单中列示了Jenkins为某些插件搜索到了可用的更新。列出的每个插件可以被选择并应用更新。
95. 可选安装:清单中列示了可用于安装（而不是目前已安装的）的所有插件。列出的每个插件都可以被选择并安装。
96. 已安装:清单中列示了已经安装的插件。
97. 高级:允许您通过设定HTTP代理的方式使Jenkins与在线插件库建立连接。此外，还提供了一个上传设备，可以安装你在Jenkins以外已下载的那些插件。
98. 由上图可知，Jenkins缺省集成了maven2插件，并且一旦插件有新版本，会提示更新新版本插件。
99. 如果想安装新的插件，可以点击tab分页中的可选插件。如下图：
100. 
101. 图37 可选插件界面
102. 从图可知，各种Jenkins插件根据之前所记述的类型进行分门别类。可勾选任意想安装的Jenkins插件，点击Install without restart按钮进行安装。安装后，所有插件以hpi作为后缀名放置在plugins文件夹下。如果是高级用户还可以自行开发插件方便具体项目使用。
103. 注意：安装完成后需要重启Jenkins部署的容器。这样才能使用新装的插件。