Hudson使用参考手册V1.1

作者: hanqunfeng http://hanqunfeng.javaeye.com

本文是关于hudson的一个快速使用手册，本文列出了实际工作中会用到的一些基本功能的使用说明。本文内容hudson的版本为1.386。

目 录

1. 持续集成

1.1 hudson--推荐序

发表时间: 2010-12-30

实话，hudson真的是简单好用，向大家推荐一下。

不过首先说明一下，本人没用过除hudson以外的其它持续集成框架(据说Continuum，CruiseControl也很强大)，以前做持续集成，都是自己写shell脚本实现的，原理很简单，svn更新代码，使用ant编译和打包以及单元测试，当然，这些都是通过shell脚本进行控制，然后也是通过shell进行部署，一个shell脚本就搞定了一切，其实想想，功能也蛮强的，不过就是需要写shell脚本，不是所有人都清楚如何写shell脚本的，另外，跨服务器部署时，为了不输入密码，需要建立ssh密钥认证，总之，需要有一定的linux基础。

最近接触到hudson，发现网上关于hudson的资料还是挺多的，如果对持续集成有了解的话，基本上搞定其常用的配置功能还是很容易的，另外hudson的帮助还是很强大的，如果对某个配置不了解，可以点击其右侧的帮助 ，足以见得其简单。

hudson基于界面进行配置，加上其有着众多的插件做支撑，只需要会编写ant（或者是maven，不过本人对maven不是很熟悉）构建文件，这就够了，其它的交给hudson完成，足见其好用。

hudson的安装很简单，就是一个war包，丢到tomcat中就行，详细的说明请参考下面的链接（推荐大家先看以下的介绍，本文并未对已经介绍的很明白的地方进行赘述）：

英文原文：http://www.javaworld.com/javaworld/jw-12-2008/jw-12-hudson-ci.html?page=1

中文翻译：http://jdonee.javaeye.com/blog/310497

hudson就是一个平台，通过这个平台将一些我们已知的较常用的工具整合起来，以此实现持续集成，hudson的使用并不难掌握，真正需要学习的还是那些基本的工具，比如ant或maven。

后续准备弄一个hudson的专题，介绍我使用hudson的一些经验。

1.2 hudson--中文任务名称

发表时间: 2010-12-30

使用tomcat作为hudson的容器时，选择系统管理后会出现如下提示：

Your container 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 and Tomcat i18n for more details.

意思是tomcat容器没有使用UTF-8编码，所以不能使用 中文作为任务名称，可以在tomcat中进行配置。

```
<Connector port="8989" protocol="HTTP/1.1"
        connectionTimeout="20000"
        redirectPort="8443" URIEncoding="UTF-8" />
```

1.3 hudson--插件管理

发表时间: 2010-12-30

hudson的好用基本体现在其众多而强大的插件上，在“系统管理”中可以找到“插件管理”。

hudson默认自带了如下几个插件，可以在“已安装”中看到：

Hudson CVS Plug-in

Maven 2 Project Plugin

Hudson SSH Slaves plugin

Subversion Plugin

以上4个插件，看字面意思就能明白的，分别提供对不同工具的支持。

为了更好的使用hudson，还需要安装一些插件，在“可选插件”中进行选择：

Deploy to container Plugin：使用其进行war包部署，支持tomcat，jboss，glassfish

Hudson SCP publisher plugin：如果要跨服务器部署，就需要安装该插件，其使用scp的方式将文件从一个服务器拷贝到另一台服务器

Hudson Email Extension Plugin：如果希望部署完成后能将结果以邮件的形式进行通知，可以安装该插件

我只安装了如上3个插件，感觉上基本的功能都够用了，建议使用hudson前先安装如上插件。

hudson通过网络自动下载插件，而且可以通过“更新”进行插件的更新。

如果不能连接外网，可以从已经安装过插件的hudson中将插件拷贝出来，hudson插件的后缀为hpi，然后通过“高级”--“上传插件”进行安装。

hudson默认的插件保存目录为：HUDSON_HOME/plugins，所以也可以直接将插件拷贝到该目录下，重启hudson即可。

HUDSON_HOME：hudson的主目录，默认为当前用户家目录下的.hudson

可以在“系统管理”--“系统设置”中的“主目录”看到具体的路径信息。关于hudson主目录的内容后续介绍。

1.4 hudson--HUDSON_HOME

发表时间: 2010-12-31

Hudson需要一些磁盘空间来执行构建和归档，所以hudson启动后，会自动建立一个HUDSON_HOME目录，该目录用于保存hudson的相关配置信息以及提供一个构建和归档的空间。

默认情况下，HUDSON_HOME会在当前用户的家目录下建立，名称为.hudson，

比如在windows下：C:\Documents and Settings\username\.hudson

在linux下：~/.hudson

你也可以通过如下方式修改HUDSON_HOME的位置：

在hudson的web.xml中找到HUDSON_HOME，默认value为空值，将其设置为你希望的路径，然后重启hudson。

```
<!-- if specified, this value is used as the Hudson home directory -->
  <env-entry>
    <env-entry-name>HUDSON_HOME</env-entry-name>
    <env-entry-type>java.lang.String</env-entry-type>
    <env-entry-value></env-entry-value>
  </env-entry>
```

修改一个已经配置过的Hudson的HOME路径，如果希望保留所有配置信息，这需要彻底停掉Hudson，然后把老的 HUDSON_HOME挪到新HUDSON_HOME下，设置新的HUDSON_HOME，最后重启Hudson。

下面是一个典型的HUDSON_HOME目录结构：

```
HUDSON_HOME


 +- config.xml     (hudson总配置文件)


 +- *.xml          (其它配置文件，比如SVN,Maven,JDK,ANT...)
```

```
+- userContent    (files in this directory will be served under your http://server/hudson/user

+- users   (存储用户目录)

+- plugins        (插件目录)

+- jobs

    +- [JOBNAME]      (每个Job的子目录)

        +- config.xml     (job配置文件)

        +- workspace      (版本控制工作目录)

        +- latest         (最后一次成功构建连接)

        +- builds

            +- [BUILD_ID]     (每次构建ID目录)

                +- build.xml      (构建结果摘要)

                +- log            (日志文件)
```

```
                +- changelog.xml  (更新日志)
```

HUDSON_HOME就是一个目录，所以你可以随时对其进行备份，如果希望hudson恢复到之前的某一次备份，直接使用备份覆盖现有的HUDSON_HOME即可，当然，覆盖后要记得重启hudson。

不同的hudson间，如果其环境和版本相同，也可以使用将一个配置好的hudson的HUDSON_HOME覆盖掉另一个HUDSON_HOME，实现快速配置，如果不希望保留Job信息，可以不复制jobs下的内容。

1.5 hudson--URL选项

发表时间: 2010-12-31

hudson为我们提供了一些通过url快速执行的功能

```
http://[hudson-server]/[command]
```

[command] 可以是：

- exit :关闭Hudson，不建议使用这个功能，它会关闭hudson所在的服务器。
- restart :重启Hudson
- script: 执行Groovy scripts，这个功能挺有用的，如果你熟悉Groovy，倒是可以尝试下。

关于hudson的Groovy scripts，可以参考该地址：http://wiki.hudson-ci.org/display/HUDSON/Hudson+Script+Console

1.6 hudson--系统管理

发表时间: 2010-12-31

hudson的总体配置，都是在“系统管理”中进行配置的，hudson“系统管理”提供了如下功能，除了“系统设置”以外，其它的功能都比较简单：

系统设置

全局设置&路径，这是hudson最核心的功能，jdk，ant，maven，scp，邮件等等，都是在这里设置的。

读取设置

放弃当前内存中所有的设置信息并从配置文件中重新读取 仅用于当您手动修改配置文件时重新读取设置。

管理插件

添加、删除、禁用或启用Hudson功能扩展插件。

系统信息

显示系统环境信息以帮助解决问题。该功能会列出系统属性，环境变量以及Plugins信息。

系统日志

系统日志从`java.util.logging` 捕获Hudson相关的日志信息。

负载统计

检查您的资源利用情况，看看是否需要更多的计算机来帮助您构建。

Hudson CLI

从您命令行或脚本访问或管理您的Hudson。感兴趣的话可以通过命令行尝试一下

脚本命令行

执行用于管理或故障探测或诊断的任意脚本命令。这就是Groovy scripts。

管理节点

添加、删除、控制和监视系统运行任务的节点。

管理用户

创建/删除/修改Hudson用户，系统的登录用户都是在这里建立的。

准备关机

停止执行新的构建任务以安全关闭计算机。

1.7 hudson--配置邮件

发表时间: 2010-12-31

如果希望在构建完成后将构建结果以邮件的形式发送给相关的用户，推荐安装Hudson Email Extension Plugin插件，它比系统默认的邮件设置的功能要强大，可以设置邮件标题及内容的格式，也可以指定触发邮件的情况，该插件安装完成后可以在“系统设置”中看到如下配置项：

Extended E-mail Notification

Override Global Settings 如果勾选上，就会覆盖job中的邮件配置。

SMTP server：邮件服务器地址，可以不填，不填则使用javamail发送，如果在公司内部使用，可以使用内网邮箱。

Default user E-mail suffix：缺省的邮件后缀， 例如,如果这里设定了`@acme.org`,那么用户foo的默认邮件地址为`foo@acme.org`

System Admin E-mail Address：就是发送邮件的用户，这个用户不要求一定存在。

其它配置项默认即可，这里注意一下，在Default Content中可以增加一下内容，比如构建日志$BUILD_LOG，具体的配置详见Content Token Reference后面的那个帮助Help for feature: Content Token Reference

保存后就配置好了。这样在建立job时，可以在job设置里指定邮件的收件人和触发邮件的情况。

具体如下，在job的设置中找到Post-build Actions，在其中找到Editable Email Notification，勾选上。

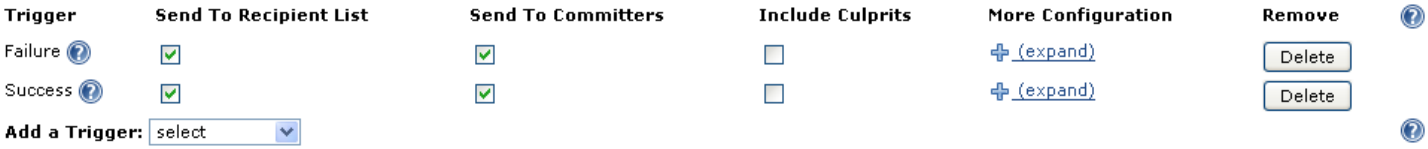
Global Recipient List：收件人列表，以英文逗号分隔。

其它配置项与系统设置中相同。

重点在“Advanced”

Advanced...

点击后，在**Add a Trigger中增加触发邮件的情况。**



1.8 hudson--scp设置

发表时间: 2010-12-31

如果是远程部署，你就会需要一个scp插件，可以在hudson的插件管理中进行安装，这个插件就是Hudson SCP publisher plugin。

插件安装完成后，我们就可以在“系统管理”--“系统设置”中进行配置，具体如下：

找到SCP repository hosts

点击add，出现SCP sites配置界面

Hostname：远程服务器IP

Port：端口

Root Repository Path ：根目录，这个目录必须是存在的，比如：/usr/local，该项可以不填

User Name：登录远程服务器的用户名称

Password/Passphrase：密码

Keyfile：也可以不填用户名和密码，而使用一个密钥文件

ok，这样就配置完成了。

可以通过add，配置多个远程站点。

之后，在job设置中，我们就可以进行如下配置：

在Post-build Actions中找到
Publish artifacts to SCP Repository，勾选上。

SCP site：选择scp站点

点击add

Source：要发送的文件，该文件的路径必须相对于job的工作区目录，既HUDSON_HOME/jobs/$jobname/workspace下的目录和文件，比如**/build/test-reports/*.xml

Destination：目的地目录，在linux环境下，如果以“/”开头，则不会在前面增加**Root Repository Path**，否则就会增加。如果该目录不存在会自动创建。

可以配置多个Source/Destination。

点击保存后完成配置。

1.9 hudson--构建执行顺序

发表时间: 2010-12-31

hudson的构建顺序与普通的持续集成一样，遵循着先更新代码-->构建-->部署的顺序，下面我们看一个hudson的构建报告

说明：我使用的svn+ant+tomcat

```
Started by user hanqunfeng
Updating https://svn.netqin.local/netqin/boss2/BOSS_NQCP/trunk/BOSS_NQCP
At revision 11035
no change for https://svn.netqin.local/netqin/boss2/BOSS_NQCP/trunk/BOSS_NQCP since the previou
No emails were triggered.
[BOSS_NQCP] $ /usr/local/ant/bin/ant
Buildfile: /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build.xml
     [echo] nqcp ant

delete:
     [echo] delete run
   [delete] Deleting directory /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build
     [echo] delete completed!

init:
     [echo] init run
    [mkdir] Created dir: /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/WebCor
    [mkdir] Created dir: /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/test-c
     [echo] init completed!

compile:
     [echo] compile run
    [javac] /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build.xml:79: warning: 'i
    [javac] Compiling 123 source files to /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_
     [copy] Copying 13 files to /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build
     [copy] Copying 21 files to /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build
     [copy] Copied 18 empty directories to 1 empty directory under /usr/local/hudson_home/jobs/
     [echo] Compile completed!
```

```
prepare-release:
     [echo] prepare-release run
     [copy] Copying 4 files to /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/
     [echo] prepare-release completed!

war:
     [echo] war run
     [copy] Copying 209 files to /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/buil
      [war] Building war: /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/nqcp.
     [echo] war completed!
     [echo] Files built to: /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/Web

build-release:
     [echo] ======  Build for RELEASE environment completed!!  ======

BUILD SUCCESSFUL
Total time: 8 seconds
Archiving artifacts
Deploying /usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/nqcp.war to container
  Redeploying [/usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/nqcp.war]
  Undeploying [/usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/nqcp.war]
  Deploying [/usr/local/hudson_home/jobs/boss_nqcp/workspace/BOSS_NQCP/build/nqcp.war]
[SCP] Connecting to 192.168.12.80
[SCP] Trying to create /usr/local//usr/local/tem
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP/config
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP/config/product
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP/config/product/context
[SCP] uploading file: '/usr/local//usr/local/tem/BOSS_NQCP/config/product/context/applicationCo
[SCP] uploading file: '/usr/local//usr/local/tem/BOSS_NQCP/config/product/context/applicationCo
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP/config/release
[SCP] Trying to create /usr/local//usr/local/tem/BOSS_NQCP/config/release/context
[SCP] uploading file: '/usr/local//usr/local/tem/BOSS_NQCP/config/release/context/applicationCo
[SCP] uploading file: '/usr/local//usr/local/tem/BOSS_NQCP/config/release/context/applicationCo
Email was triggered for: Success
Sending email for trigger: Success
```

```
Sending email to: hanqunfeng@netqin.local
Finished: SUCCESS
```

很清楚了吧，顺序如下：

svn更新代码---->ant编译和打包---->部署到tomcat---->远程scp文件传输---->发动邮件通知

1.10 hudson--JOB_WORKSPACE

发表时间: 2010-12-31

JOB_WORKSPACE：就是构建时使用的目录，也就是从代码仓库中取得的代码所要保存的路径，默认路径为HUDSON_HOME/jobs/$jobname/workspace

job的配置中，所有涉及到路径的配置，都是基于JOB_WORKSPACE基础上的，配置时都是使用相对路径，也就是从这个路径开始计算。

可以在job设置中修改这个路径，找到Advanced Project Options，点击“Advanced”，勾选Use custom workspace，并设置一个绝对路径，比如/usr/local/hudson/jobname/workspace

点击保存，重新执行构建即可。

1.11 hudson--部署设置

发表时间: 2010-12-31

需要为hudson安装Deploy to container Plugin插件。安装完成后，job设置Post-build Actions中会找到Deploy war/ear to a container配置项，勾选上。

该插件只支持war包或ear包的部署。

具体配置如下：

WAR/EAR files：war或ear文件的路径，这个路径是相对于JOB_WORKSPACE的。注意：如果是第一次构建，hudsno会提示你路径不存在，原因就是war包还没有创建，如果是这样，可以不用管它，保存即可。

Container：选择使用的容器

Manager user name：容器管理员，如果使用的是tomcat，可以参考tomcat管理账号配置

Manager password：密码

XXX URL：容器访问路径，hudson支持跨服务器部署，所以该路径可以与hudson不在同一台服务器。（个人认为，这是hudson比较强大的地方）

ok，这样就配置完成了。执行构建时会将指定的包部署到指定的服务器中。比如tomcat，就是部署到其webapps中。

1.12 hudson--junit测试报告

发表时间: 2010-12-31

如果你的项目在构建时进行了单元测试，可以在hudson中直接查看测试报告。

在job设置中的Post-build Actions找到 Publish JUnit test result report，勾选上。

Test report XMLs：指定junit测试结果目录，就是那一堆Test*.xml所在的目录，该目录必须是相对于JOB_WORKSPACE的。注意：如果是第一次构建，hudsno会提示你路径不存在，原因就是测试结果还没有创建，如果是这样，可以不用管它，保存即可。

ok，配置完成。

执行构建。在job信息页中会看到，增加了如下内容：

Latest Test Result (4 failures / ±0)

同时还会看到Test Result Trend的一个图表。

点击图表或Latest Test Result连接，就可以看到测试结果报告了。

1.13 hudson--安全设置

发表时间: 2011-01-01

hudson默认是不需要登录既可以使用的，如果希望只有登录用户我们才能访问，可以在“系统管理”--“系统设置”中进行配置。

找到**启用安全**，勾选上。

JNLP节点代理的TCP端口:选择禁用

访问控制--安全域

选择“**Hudson专有用户数据库**”：使用Hudson自己的用户列表验证, 而不是外部系统代理. 这适用于没有用户数据库小范围的设定. 基本上这个就比较好用，建立的用户配置信息保存在HUDSON_HOUME/users下。个人感觉这种配置方式是比较方便的。

允许用户注册：如果勾选上，则表示允许用户自己注册一个新账号,通过点击页面右上角的"注册"链接进行操作. 但如果你想禁止任意注册新账号,而使用严格的方式控制账号创建,那么取消这个选框。 当这个选框被取消,就必须使用系统管理员来创建账号.

这里说明一下，如果**授权策略**选择的是“**安全矩阵**”或者“**项目矩阵授权策略**”，在注册用户时，如果与注册用户名同名的角色没有事先创建，该用户是没有相应的访问权限的。

接着说一下**授权策略**，这里只说一下“**安全矩阵**”和“**项目矩阵授权策略**”，其它的都比较好理解，同时授权也过于简单，可以根据实际情况进行选择。

如果选择**安全矩阵**，出现如下视图：

默认会有一个匿名用户角色，表示未登录用户的访问权限，你可以不授权或只授予read权限。

你可以为角色设置全局，job，视图等的不同权限。

添加用户/组：输入一个名称，比如admin，然后点击添加，这里实际上是先建立一个角色，因为这时用户并不存在，如下所示：

用户/组	Overall	Slave			Job						View			SCM
	Administer	Read	Configure	Delete	Create	Delete	Configure	Read	Build	Workspace	Create	Delete	Configure	Tag
匿名用户	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
admin	☑	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

这时admin前面有一个红色的减号图标，表示用户尚不存在。

这里我们授予admin管理员权限，注意，必须先建立一个管理员角色。接着点击保存。

如果我们没有勾选“允许用户注册”，则在保存后会进入到用户注册页面，如果勾选了“允许用户注册”，则会立刻退出系统，这时我们可以在页面右上角找到“注册”，自己注册一个账号。

注意，无论哪种方式，我们这里必须要填注册一个admin用户（与角色名称相同），否则我们将无法管理hudson。

注册完成后会自动登录，我们再进入“系统管理”--“系统设置”中，会看到admin前面的变成了一个“小人图标”，表示用户已经创建完成。

再次强调一下，无论是否允许用户自己注册，这里都需要管理员先建立好角色，否则是无法正常使用系统的。

管理员可以通过“系统管理”中的“用户管理”进行创建。

接着说一下“项目矩阵授权策略”：这个授权模型扩展自"安全矩阵",允许把下面的ACL(访问控制列表)矩阵附加到每个项目定义中(在Job配置页面). 就是说，除了可以在这里进行全局授权外，还可以在JOB中进行配置。

我们新建一个JOB，在设置中我们找到“启用项目安全”，勾选上，然后我们就会看到下面的列表：

这里只能对当前的JOb进行授权配置，同样的，这里也是创建角色，用户需要自己注册或管理员进行创建。

如果“系统设置”与“JOB设置”中存在相同的角色，则权限取并集。

有时候会出现这样的情况，就是虽然创建了角色，但是没有勾选上“允许用户注册”，这时就无法控制hudson了，这时可以通过如下修改，开启“允许用户注册”。

进入HUDSON_HOME，编辑config.xml ，找到如下内容：

```
<securityRealm class="hudson.security.HudsonPrivateSecurityRealm">
    <disableSignup>true</disableSignup>
</securityRealm>
```

修改disableSignup为false，然后重启hudson，这时就可以进行注册了。

ok。

1.14 hudson--build配置

发表时间: 2011-01-04

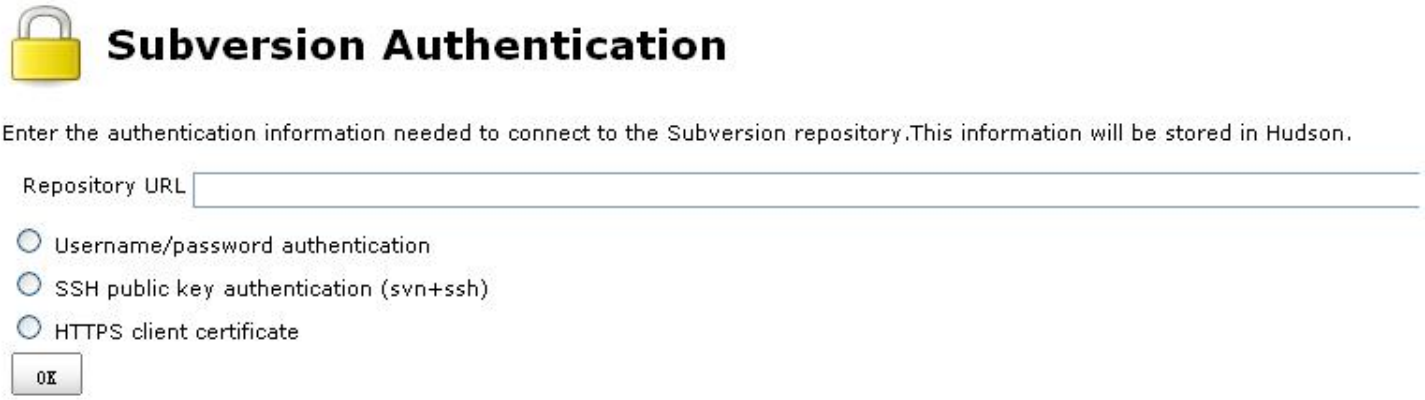
在每个job中，我们可以指定其构建方式，比如是通过ant或maven构建，还是通过shell或bat脚本构建，亦或是他们组合在一起完成一次构建，我们可以指定他们的构建顺序。

举个例子，我使用的集成方式：svn+ant+tomcat。

一。指定svn地址

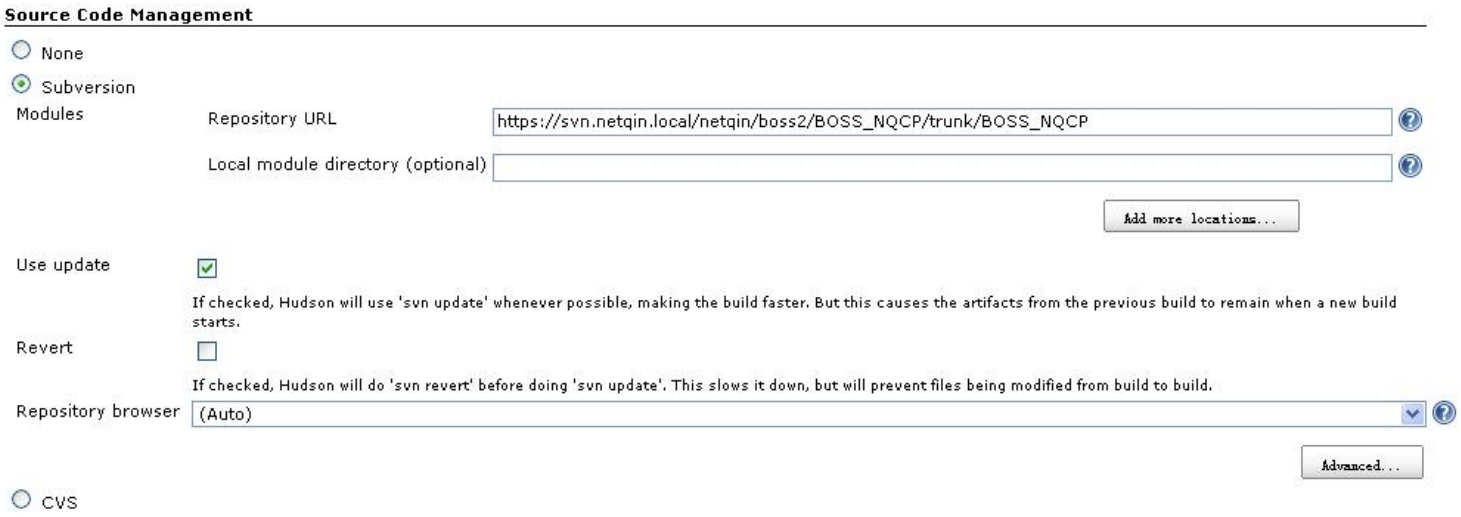
在Source Code Management中，选择 Subversion

我使用Repository URL：指定代码的url地址。代码会下载到JOB_WORKSPACE下。如果svn地址是需要安全认证的，可以点击“Repository URL”右侧的帮助Help for feature: Repository URL ，从帮助中找到“this link ”并点击，出现如下配置界面：



在这里输入svn地址，并选择你的认证类型。点击ok即可完成配置。

选择Use update：这表示在构建时使用代码更新的方式，而不是重新下载全部代码，这样可以减少构建的时间。



二。指定Build Triggers

这里我选择Poll SCM：5 * * * *

每隔5分钟检查一次svn，如果有代码更新则进行自动构建。

三。Build配置

设置Invoke Ant，如果没有该项，可以在“Add build step”中选择，通过该按钮可以添加ant或maven以及shell或bat脚本，其添加顺序，即为构建顺序。

Ant Version：选择我们要使用的ant的版本，这可以在系统设置中进行设置。执行时会自动从项目根目录下查找build.xml文件，这里就是指的JOB_WORKSPACE/BOSS_NQCP下。

Targets：指定要执行的任务，如果不指定，则默认执行ant中的default。如果要一次执行多个任务，可以每行指定一个，不过这里要注意一下，每个任务都是独立执行的，比如要指定A和B两个任务，先执行A后执行B，如果B依赖于A，则执行B时，A会被再执行一次。

anvanced：如果ant文件不在项目根目录下或者名称不为build.xml，或者执行ant时需要传入参数，则可以点击advanced按钮。

Build File：指定ant文件路径

Properties：指定参数，每行配置一个参数，格式为name=value。

这样在执行ant时会自动加上这些参数：

```
ant -Dname=value  build-release
```

Java Options：配置java选项，如内存上限-Xmx512m。

如果，我希望在ant执行完成之后，执行一个shell命令，可以单击“Add build step”，选择“Execute shell”，在Command中输入要执行的命令，这里我们只打印出“hello world”：

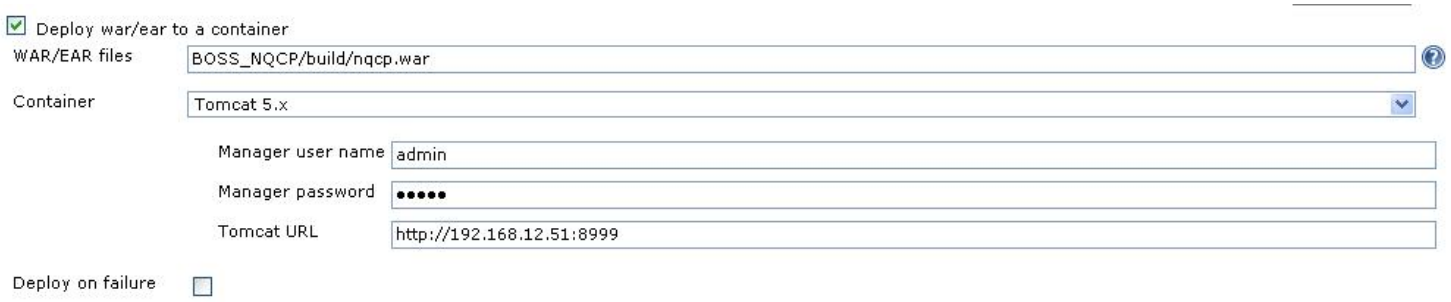
控制台输出结果：

```
Started by user hanqunfeng
.....................
.....................
BUILD SUCCESSFUL
Total time: 16 seconds
[workspace] $ /bin/sh -xe /usr/local/tomcat7/temp/hudson4224991464384646624.sh + echo 'hello wo
hello world
```

实际上我们输入的命令会在hudson所在tomcat下生成一个临时脚本文件。

四。部署 发布

之后Deploy war/ear to a container 配置tomcat发布即可。



ok，这样我们就完成了一个简单的构建配置。

1.15 hudson--环境变量

发表时间: 2011-01-04

在执行构建时，我们可以为每个JOB指定一些环境变量，方法如下：

在job的设置中，勾选“This build is parameterized”

点击“Add Parameter” ，选择我们需要的类型，hudson提供对如下类型的支持：

boolean value：布尔值，勾选Default Value，则默认值为true

choice：一组待选择的值，每行设置一个，第一行为其默认值

string Parameter：字符串

password Parameter：字符串，但是加密过的

run Parameter：运行时参数，需要制定job，不过没太搞明白具体该怎么使用

file Parameter：指定一个文件的保存路径及文件名称，构建时可以在本地选择一个文件，该文件会被保存在这个设置的

路径下，并使用指定的文件名称。比如这里设置~/temp/123.jpg，构建时在本地选择的文件名称为456.txt，则456.txt会被上传到~/temp/下，并重新命名为123.jpg。

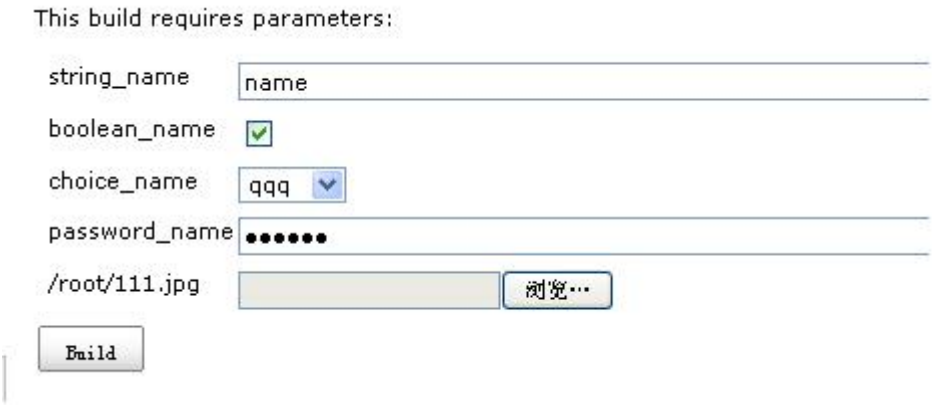
以上参数在构建时会被加入到环境变量中，可以通过${parameterName}或$parameterName进行访问，如下所示：

ant：${parameterName}，比如：<echo message="string_name = ${string_name}" />

shell：$parameterName，比如：echo string_name=$string_name

自动构建时，hudson会使用参数中配置的默认值进行构建。

如果是手动构建，就是在hudson中点击“立即构建”时，hudson会先显示如下页面：



你可以手动修改变量值，然后点击“Build”执行构建。

在**“系统管理”--“系统设置”** 中也可以设置环境变量：

找到**“全局属性”** ，勾选“Environment variables” ，点击“Add”进行添加。

这里配置的变量是全局的，对每个job都有效，构建时不会提示你进行修改。

注意：ant里不能取得该值，shell里可以使用。

1.16 hudson--构建依赖

发表时间: 2011-01-04

hudson一个比较好的功能是提供了构建依赖，就是说，一个job构建完成可以接着进行其它job的构建，这在实际工作中非常有用。

比如说：Ajob负责发布，Bjob负责执行某些部署后脚本，B必须在A完成部署后才能执行，这时候我们可以配置两个job间的依赖关系。

要实现这种配置构建依赖有两种方法：

1.进入A的设置，在“Post-build Actions”中找到“Build other projects ”，勾选上。

在“Projects to build”中填上要接着执行的job名称，多个job以逗号分隔。这里我们填上B的名称即可。

如果希望A在不稳东构建后依然可以执行后续的构建，则勾选“Trigger even if the build is unstable ”，

但是，如果A构建失败，则不会执行后续的构建。

2.也可以再B的设置中进行配置，找到“Build Triggers”，选择“Build after other projects are built”，在“Projects names”中填写要在B执行前执行的job名称，多个job以逗号分隔。

1和2两种方式效果一下，只要A构建执行完成，B构建就会自动执行。

1.17 hudson--findbugs报告

发表时间: 2011-01-06

如果我们的ant构建文件中有findbugs的任务，那么可以使用hudson来查看结果报告。

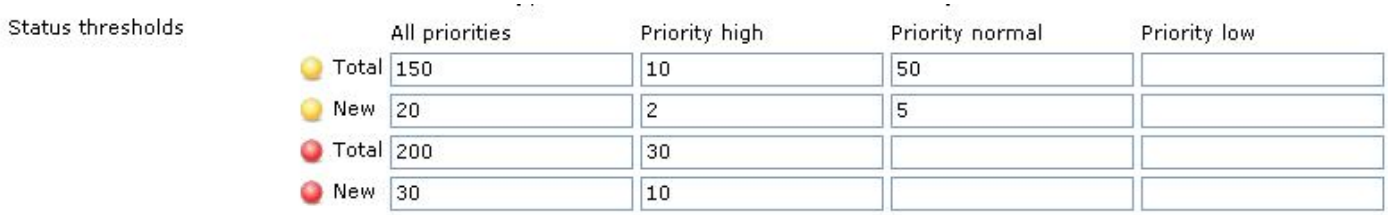
打开job设置，在Post-build Actions中找到Publish FindBugs analysis results，勾选上。

FindBugs results：findbugs报告的路径，这里要求是xml格式。注意：如果是第一次构建，hudson会提示路径错误，不用管它，保存即可。

基本上配置好“FindBugs results”就可以了，如果希望更为个性的配置，可以点击“Advanced”。
Run always：默认只有稳定的构建后才会执行findbugs报告的生成，如果勾选，则表示无论构建成功还是失败，都生出报告。
Health thresholds：健康指数。比如：

上图表示，如果问题数量小于100个，则显示太阳图标，如果问题数量超过200个，则使用暴雨图片显示。

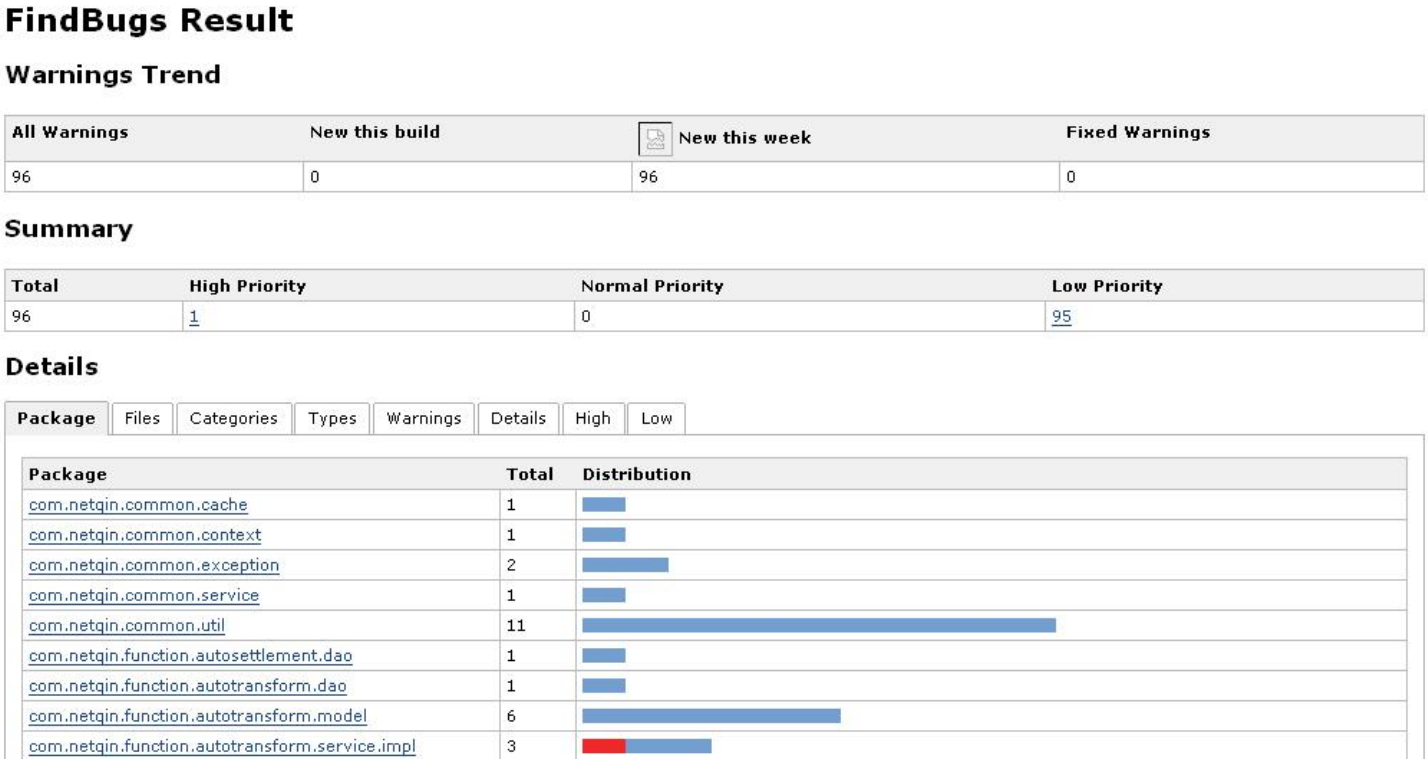
Health priorities：确定构建过程中关注的优先级。findbugs有三个级别：high，normal，low。一般只关注normal以上的级别即可。
Status thresholds：状态指数。在每个级别下配置相应的数量，构建时，如果超过配置的数量，则job就会显示对应的状态图标。比如：



上图表示：如果150<问题总数<200，或10<高级别的问题<30，或50<普通级别的问题，则job状态使用黄色的不稳定图标表示，超过则使用红色的失败图标。Total：所有的问题数量。New：本次构建新发现的问题数量。
Use delta for new warnings：如果勾选，表示新的问题数量使用增量更新的方式。
Default Encoding：生成报告的编码方式。缺省使用hudson所在平台的编码。

ok，这样就配置完成了。

执行构建后，可以在JOB状态页中会看到“FindBugs Trend”图表。“同时，右侧的菜单中会看到 FindBugs Warnings”连接，点击则可以进行查看findbugs报告。



FindBugs Result

Warnings Trend

All Warnings	New this build	New this week	Fixed Warnings
96	0	96	0

Summary

Total	High Priority	Normal Priority	Low Priority
96	1	0	95

Details

Package | Files | Categories | Types | Warnings | Details | High | Low

Package	Total	Distribution
com.netqin.common.cache	1	
com.netqin.common.context	1	
com.netqin.common.exception	2	
com.netqin.common.service	1	
com.netqin.common.util	11	
com.netqin.function.autosettlement.dao	1	
com.netqin.function.autotransform.dao	1	
com.netqin.function.autotransform.model	6	
com.netqin.function.autotransform.service.impl	3	

关于findbugs的介绍，可以参考如下地址：

http://blog.csdn.net/ansel13/archive/2009/12/17/5024238.aspx

http://hanqunfeng.javaeye.com

Hudson使用参考手册V1.1

作者: hanqunfeng
http://hanqunfeng.javaeye.com