|  |  |
| --- | --- |
|  | **2014** |
|  | The Company  Huijsmans, Martien |

|  |
| --- |
| **[Jenkins]** |
| Installation, configuration and management |

Inhoud

[Installation 2](#_Toc394086988)

[Configuration 2](#_Toc394086989)

[Upgrading to new Jenkins 2](#_Toc394086990)

[Plugins 2](#_Toc394086991)

[Developing own Jenkins plugin 3](#_Toc394086992)

[Jobs 4](#_Toc394086993)

[Management 4](#_Toc394086994)

[Workspace 4](#_Toc394086995)

[Execute shell script after successful build 4](#_Toc394086996)

[Jenkins environment variables and Maven 4](#_Toc394086997)

[C++ builds 5](#_Toc394086998)

[XLST to transform CUnit into JUnit XML 5](#_Toc394086999)

# Installation

Guide: <https://wiki.jenkins-ci.org/display/JENKINS/Installing+Jenkins+on+Ubuntu>  
<https://wiki.jenkins-ci.org/display/JENKINS/Installing+Jenkins+on+RedHat+distributions>   
Note: If you install without first 2 lines, you get an old version 1.424.6.

Installation of java (because Jenkins needs it).  
sudo apt-get install openjdk-7-jdk

After installation: access Jenkins from localhost:8080  
In terminal I run top and observed that there is indeed a user Jenkins running java.

Installation from Jenkins also ensure that Jenkins is started at boot.

# Configuration

Under “manage Jenkins”, configured.  
After a re-install of maven the settings where presented. Autodiscovery or from previous installation is unclear.

* Maven  
  Point to installed maven  
  maven\_home found via commandline: “mvn –version”
* Java  
  Point to installed openjdk  
  java\_home found via commandline: “mvn –version”

# Upgrading to new Jenkins

Followed the instruction at  
<http://jenkins-ci.org/> for Ubuntu/Debian  
There are 4 commands. The 2nd one failed (command ‘deb’ was not found; sudo missing? I didn’t try that), but I continued. The upgrade works.  
I observed that Jenkins was stopped and started again as part of the procedure.

# Plugins

* Jenkins comes with a list of installed plugin’s
  + Check the installed list and upgrade.
* Some jobs want to perform a GUI, so they need a X-server running on the machine.  
    
  Installed vnc server as described:  
  <http://vitorbaptista.com/continuous-integration-for-android-apps-with-jenkins-and-maven3/>   
  password: Jenkins  
  After setting password: $ exit  
    
  installed xvnc pluging.  
  Manage Jenkins > Configure System > Xvnc Command line:  
  /path/to/vncserver :$DISPLAY\_NUMBER -geometry 800x600
* Walldisplay:  
  <https://wiki.jenkins-ci.org/display/JENKINS/Wall+Display+Plugin>  
  <http://pelle.io/jenkins-walldisplay/>
* Different settings.xml  
  <http://stackoverflow.com/questions/14558002/specify-a-separate-location-for-settings-xml-in-jenkins>
* Publish Over SSH Plugin on remote Server  
  <https://wiki.jenkins-ci.org/display/JENKINS/Publish+Over+SSH+Plugin>
* Adds a build step to copy artifacts from another project.   
  <https://wiki.jenkins-ci.org/display/JENKINS/Copy+Artifact+Plugin>
* Copy a set of files, from a location somewhere on the master node, to jobs' workspaces. It also allows to copy files back from the workspaces of jobs located on a slave node to their workspaces on the master one.  
  <https://wiki.jenkins-ci.org/display/JENKINS/Copy+To+Slave+Plugin>
* Publish the test results of an execution of a testing tool in Jenkins.   
  <https://wiki.jenkins-ci.org/display/JENKINS/xUnit+Plugin>
* Matrix, for debug and release build, build for different platforms  
  <https://wiki.jenkins-ci.org/display/JENKINS/Building+a+matrix+project>
* Build pipeline  
  <https://wiki.jenkins-ci.org/display/JENKINS/Build+Pipeline+Plugin>
* Scons plugin  
  <https://wiki.jenkins-ci.org/display/JENKINS/SCons+Plugin>
* Promoted builds  
  <https://wiki.jenkins-ci.org/display/JENKINS/Promoted+Builds+Plugin>

# Developing own Jenkins plugin

Ref: <https://wiki.jenkins-ci.org/display/JENKINS/Plugin+tutorial>  
Maven archetype exists that generates an hpi.

# Jobs

* Poll SCM syntax:  
  <http://htmlpreview.github.io/?https://github.com/jenkinsci/jenkins/blob/master/core/src/main/resources/hudson/triggers/TimerTrigger/help-spec.html>

# Management

Manage old build, e.g. keep 5 youngest

Monitoring of CPU and Disk usage.

<http://di388e0fcqllf.cloudfront.net/whitepapers/7WaysToOptimizeJenkins.pdf>

# Workspace

Jenkins has concept of workspace in which it does build. It also keep data from previous builds.

# Execute shell script after successful build

Goal: Jenkins shall execute script /opt/publish\_rpm/publish to copy ceated rpm to local yum   
repository.  
Assumption: /opt/publish\_rpm/publish exists.  
Steps:  
Open sudoers  
$ sudo visudo  
and add  
jenkins ALL = NOPASSWD: /opt/publish\_rpm/publish  
( Ref:  
<http://stackoverflow.com/questions/11880070/how-to-run-the-script-as-root-in-jenkins> )  
comment out: #Default requiretty  
(when present, a sudo command can only be executed using a tty).

Add following shell command in post build script  
$ sudo /opt/publish\_rpm/publish $WORKSPACE/target/rpm/appname/RPMS/noarch/appname-\*.noarch.rpm  
where /opt/publish\_rpm/publish refer to a shell script that has been installed.

Note: WORKSPACE is en Jenkins variable. For complete list see:  
<https://wiki.jenkins-ci.org/display/JENKINS/Building+a+software+project>

# Jenkins environment variables and Maven

Jenkins environment variables can be used in maven pom.xml  
For example, assume that pom.xml has a build.number property (with a value), then this maven property can be set by Jenkins, using Jenkins environment variables.  
e.g. added to a maven build command: –Dbuild.number=${BUILD\_NUMBER}  
Where BUILD\_NUMBER is a jenkins environment variable.  
<http://tedvinke.wordpress.com/2013/04/20/using-jenkins-build-version-with-maven/>

Jenkins environment variables are listed here:  
<https://wiki.jenkins-ci.org/display/JENKINS/Building+a+software+project>

# C++ builds

<https://wiki.jenkins-ci.org/pages/viewpage.action?pageId=65143593>  
This article has a strategy for incremental and daily clean builds.  
  
Example java, C and Python projects that can run under Jenkins  
<https://bitbucket.org/softwaresaved/examples/src/957ea6dfcbe209971e5242576c042a54a76f5110/simple_build_and_test/?at=default>

Scons which also has own Plugin  
<https://wiki.jenkins-ci.org/display/JENKINS/SCons+Plugin>

# XLST to transform CUnit into JUnit XML

<http://www.van-porten.de/blog/2009/05/cunit-tests-in-hudson/>