Jenkins - Parameterized builds

-- To configure parameters for your build job that can be enter by user when the build job is triggered or from another build job.

Eg:

-- Deployment job, where you want to choose the target environment in drop-down list when you start the build job.

Target Enviroments like **QA,UAT,PRE-PROD**

-- You may want to specify the version of the application you want to deploy.

Application Version: **1.0.0-SNAPSHOT,1.1.0-SNAPSHOT**

-- when running a build job involving web tests, you might want to specify the browser to run your Selenium or WebDriver tests in.

Jenkins simply provides a user interface for users to enter values for the parameters, and passes these parameters to the build script.

Creating a Parameterized Build Job

1) Goto Job configuration
2) Select “This build is parameterized”
3) click Add Parameter

Add screenshot

-- You can add as many parameters as you want for a build job.

-- To add a parameter to your build job, just pick the parameter type in the drop-down list.

-- You can choose from several different parameter types, such as Strings, Booleans, choice..etc

-- All Parameter types have a name and description and most often a default value.

-- The default value will be initially displayed when Jenkins prompts the user for this parameter, so if the user doesn’t change anything, this value will be used.

NOTE:

Parameterized build jobs are very often started manually.

Adapting Parameters in your build job

-- Once you have added a parameter, that parameter can be use in required sections.

Eg : Your parameter name VERSION then to use this in build job

$VERSION, - In execute shell script

%VERSION%, - In execute windows script

${VERSION} – In others section like post build action, scm section,…etc.

The parameters are available as environment variables. So e.g. a shell ($FOO, %FOO%) or Ant (${env.FOO}) can access these values.

Jenkins – Notification

→ We suppose to provide the status of build to team.Lets team to know when build failed or passed.

→ We Will send the email Notification when

Compilation failed
Test cases failed i.e junits, Integration tests.
Quality related issues code coverage or code quality metrics failed
Deployment failed in Test Enviroment.
Deployment successful in Test Enviroment.

→ By default, Jenkins will send an email for every failed or unstable build.
→ it will also send a new email for the first successful build after a series of failed or unstable builds, to indicate that the issue has been fixed.

"E-mail Notification" section

Goto-> Configure System

Enter the SMTP server name under 'Email Notification'. Click the 'Advanced' button and then click the checkbox next to the 'Use SMTP Authentication' option. Now, set the following fields.

SMTP server name : smtp.gmail.com
User name: user_email_id@gmail.com
Password: 123456
Use SSL : Checked
SMTP Port: 456

Job Configuration:

In Post build action

→ Just activate email notification in Jenkins by ticking the E-mail Notification checkbox and providing the list of email addresses of the people who need to be notified.

Editable Email Notification

→ We can customize emails with different messages and recipient lists for different events

Steps:

Click "Add post-build action"
Click "Editable Email Notification"
Click "Advanced Settings…"
Click "Add Trigger" → Click "Always" → Save

→ We can use predefined tokens to create your own customized message title and body.
→ You include a token in your message template using the familiar dollar notation
→ Some of the tokens accept parameters, which you can specify using a name=value format

${DEFAULT_SUBJECT}
The default email subject configured in the Jenkins system configuration page
${DEFAULT_CONTENT}
The default email content configured in the Jenkins system configuration page
${PROJECT_NAME}
The project's name
${BUILD_NUMBER}
Current build number
${BUILD_STATUS}
Current build status (failing, success, etc.)
${CAUSE}
The cause of the build
200
${BUILD_URL}
A link to the corresponding build job page on Jenkins
${FAILED_TESTS}
Shows information about failing unit tests, if any have failed
${CHANGES}
Displays the changes made since the last build
${CHANGES_SINCE_LAST_SUCCESS}
All the changes made since the last successful build

Tiggers

→ Triggers determine when email notification messages should be sent out.

→ The supported triggers include the following:

Failure = Any time the build fails.
Still Failing = Any successive build failures.
Unstable = Any time a build is unstable.
Still Unstable = Any successive unstable builds.
Success = Any successful build.
Fixed = When the build changes from Failure or Unstable to Successful.
Before Build = Sent before every build begins.

Email Build Template

SUBJECT:
Build status -PROJECTNAME-JOB_NAME-BUILD_STATUS

CONTENTS:

Hi all,

Build Status : $BUILD_STATUS
JOB NAME : $JOB_NAME
BUILD_NUMBER : $BUILD_NUMBER

Build is created successfully & required artifacts uploaded
onto destination servers.

check changes output at $BUILD URL to view the results

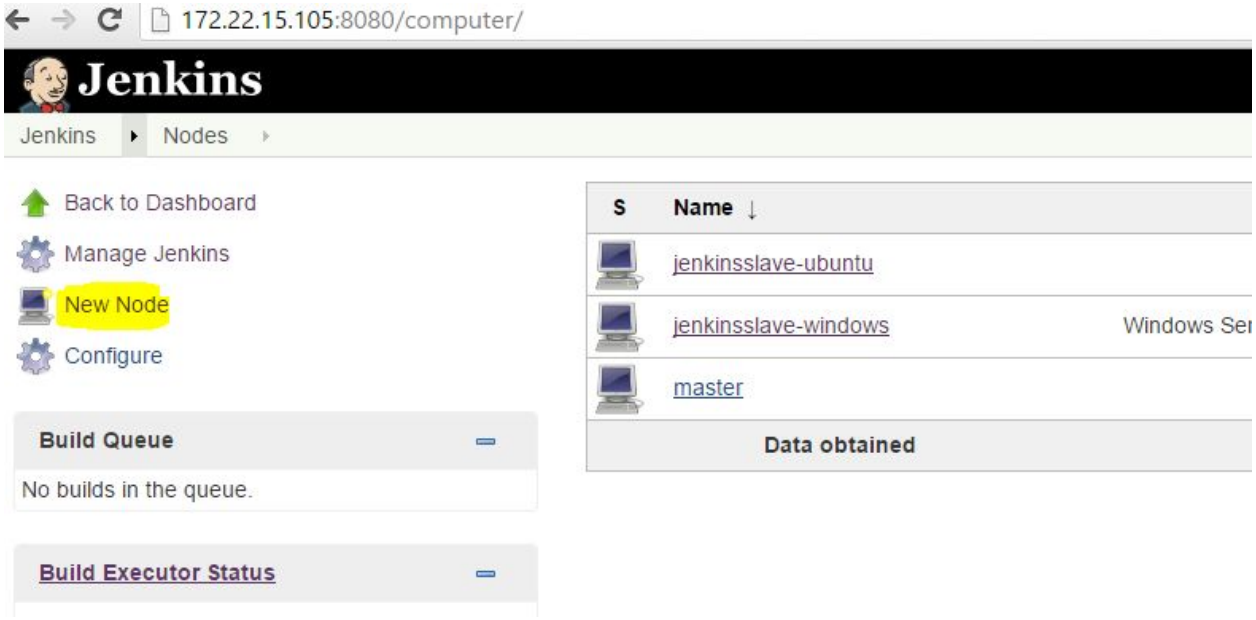
Please contact apps@.com in case of any discrepancy with the
above info

Thanks & Regards,
Java Home Technologies

Jenkins – Distributed Builds

→ Dispatch build jobs across multiple machines.
→ Sometimes many builds are required for large projects which gets built on regular basis.
→ And running all build jobs in one maching may not be a best option.In such a scenario, one can configure other Jenkins machines to be slave machines to take the load off the master Jenkins server.
→ Sometimes you might also need several different environments to test your builds in this case also configure the jenkins
→ For example, you may need to run particular build jobs on a particular machine or operating system.
i.e if you need to run web tests using Internet Explorer, you will need to be use a Windows machine.

Configure Jenkins Slave and Connect to Master




Node name
Jenkins Demo Slave
Permanent Agent
Adds a plain, permanent agent to Jenkins. This is called
for example such as when you are adding a physical con
Copy Existing Node
Copy from
OK


Name
Jenkins Demo Slave
Description
# of executors
1
Remote root directory
/opt/jenkins/
Labels
Usage
Use this node as much as possible
Launch method
Launch agent via Java Web Start
Advanced...
Availability
Keep this agent online as much as possible
Node Properties
Environment variables
Tool Locations
Save


Mark this node temporarily offline
Agent Jenkins Demo Slave
This agent is offline because Jenkins failed to launch the agent process on it. See log for more details
Connect agent to Jenkins one of these ways:
Launch
Launch agent from browser
Run from agent command line:
java -jar slave.jar -jnlpUrl http://172.22.15.105:8080/computer/Jenkins%20Demo%20Slave/slave-agent.jnlp -secret e78c9bde06ca1ada782b3b222cc19baaece0fc48b1ae7a1d40b2cae716b485de
Projects tied to Jenkins Demo Slave
None

Nexus
-- Its an Artifactory Server
-- Repository Manager
-- Use to Store Compile code Binaries - project war
-- Deployable binaries

snapshot binaries- work in progress version i.e 1.0.0-SNAPSHOT
release
binaries -- stable vesions i.e 1.0.0 or 1.0.0-releases
http://localhost:8082/nexus/service/local/artifact/maven/redirect?r=snapshots&g=in.mphasis&a=DevOpsDemoProject&v=LATEST&e=war

Commands:
Mvn deploy
Mvn deploy -Modulename

Download Link:

https://www.sonatype.com/download-oss-sonatype

Url Info:
http://localhost:8081
Cred: admin/admin123

Repository details:

Released artifacts
http://localhost:8082/nexus/content/repositories/releases/

Snapshot artifacts
http://localhost:8082/nexus/content/repositories/snapshots/

Nexus Configurations:

In.m2/Setting.xml:

```
<servers>
   <server>
     <id>deployment</id>
     <username>admin</username>
     <password>admin123</password>
   </server>
 </servers>
```

In pom.xml:

```
<distributionManagement>
      <repository>
         <id>deployment</id>
         <name>Internal Releases</name>
         <url>http://localhost:8082/nexus/content/repositories/releases/</url>
      </repository>
      <snapshotRepository>
         <id>deployment</id>
         <name>Internal Snapshot Releases</name>
         <url>http://localhost:8082/nexus/content/repositories/snapshots/</url>
      </snapshotRepository>
   </distributionManagement>
```