



# jenkins

**tutorialspoint**  
SIMPLY EASY LEARNING

[www.tutorialspoint.com](http://www.tutorialspoint.com)



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

## About the Tutorial

---

Jenkins is a powerful application that allows continuous integration and continuous delivery of projects, regardless of the platform you are working on. It is a free source that can handle any kind of build or continuous integration. You can integrate Jenkins with a number of testing and deployment technologies. In this tutorial, we would explain how you can use Jenkins to build and test your software projects continuously.

## Audience

---

This tutorial is going to help all those software testers who would like to learn how to build and test their projects continuously in order to help the developers to integrate the changes to the project as quickly as possible and obtain fresh builds.

## Prerequisites

---

Jenkins is a popular tool for performing continuous integration of software projects. This is a preliminary tutorial that covers the most fundamental concepts of Jenkins. Any software professional having a good understanding of Software Development Life Cycle should benefit from this tutorial.

## Disclaimer & Copyright

---

© Copyright 2015 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute, or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness, or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at [contact@tutorialspoint.com](mailto:contact@tutorialspoint.com)

## Table of Contents

---

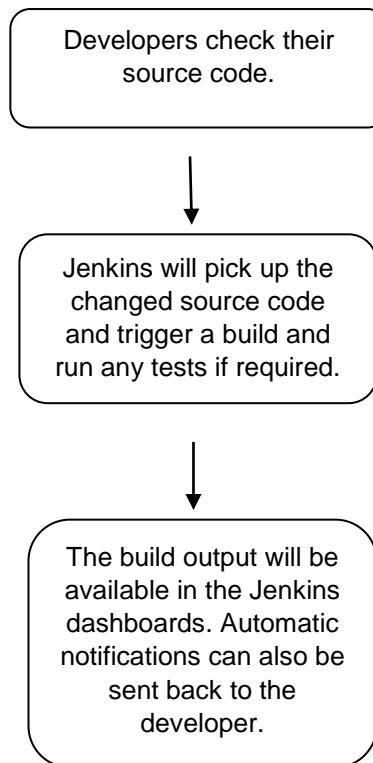
About the Tutorial .....	i
Audience .....	i
Prerequisites .....	i
Disclaimer & Copyright.....	i
Table of Contents .....	ii
 1. JENKINS – OVERVIEW .....	1
Why Jenkins?.....	1
What is Continuous Integration?.....	1
System Requirements .....	2
 2. INSTALLING JENKINS.....	3
Download Jenkins .....	3
Starting Jenkins .....	4
Accessing Jenkins .....	5
 3. JENKINS – TOMCAT SETUP.....	6
Step 3: Download Tomcat .....	7
 4. JENKINS – GIT SETUP .....	10
 5. JENKINS – MAVEN SETUP .....	17
 6. JENKINS – CONFIGURATION .....	24
 7. JENKINS – MANAGEMENT .....	28
Configure System .....	29
Reload Configuration from Disk.....	29
Manage Plugins .....	30
System Information.....	30

8. JENKINS – SETUP BUILD JOBS .....	33
9. JENKINS – UNIT TESTING .....	44
Example of a Junit Test in Jenkins.....	46
10. JENKINS – AUTOMATED TESTING .....	54
11. JENKINS – NOTIFICATION.....	62
12. JENKINS – REPORTING .....	65
13. JENKINS – CODE ANALYSIS.....	66
14. JENKINS – DISTRIBUTED BUILDS .....	67
15. JENKINS – AUTOMATED DEPLOYMENT .....	72
16. JENKINS – METRICS AND TRENDS .....	75
17. JENKINS – SERVER MAINTENANCE.....	89
URL Options .....	89
Backup Jenkins Home.....	89
18. JENKINS – CONTINUOUS DEPLOYMENT.....	91
19. JENKINS – MANAGING PLUGINS .....	107
Uninstalling Plugins .....	108
Installing another Version of a Plugin .....	109
20. JENKINS – SECURITY.....	110
21. JENKINS – BACKUP PLUGIN.....	116
22. JENKINS – REMOTE TESTING.....	125

# 1. Jenkins – Overview

## Why Jenkins?

Jenkins is a software that allows **continuous integration**. Jenkins will be installed on a server where the central build will take place. The following flowchart demonstrates a very simple workflow of how Jenkins works.



Along with Jenkins, sometimes, one might also see the association of **Hudson**. Hudson is a very popular open-source Java-based continuous integration tool developed by Sun Microsystems which was later acquired by Oracle. After the acquisition of Sun by Oracle, a fork was created from the Hudson source code, which brought about the introduction of Jenkins.

## What is Continuous Integration?

Continuous Integration is a development practice that requires developers to integrate code into a shared repository at regular intervals. This concept was meant to remove the problem of finding later occurrence of issues in the build lifecycle. Continuous integration requires the developers to have frequent builds. The common practice is that whenever a code commit occurs, a build should be triggered.

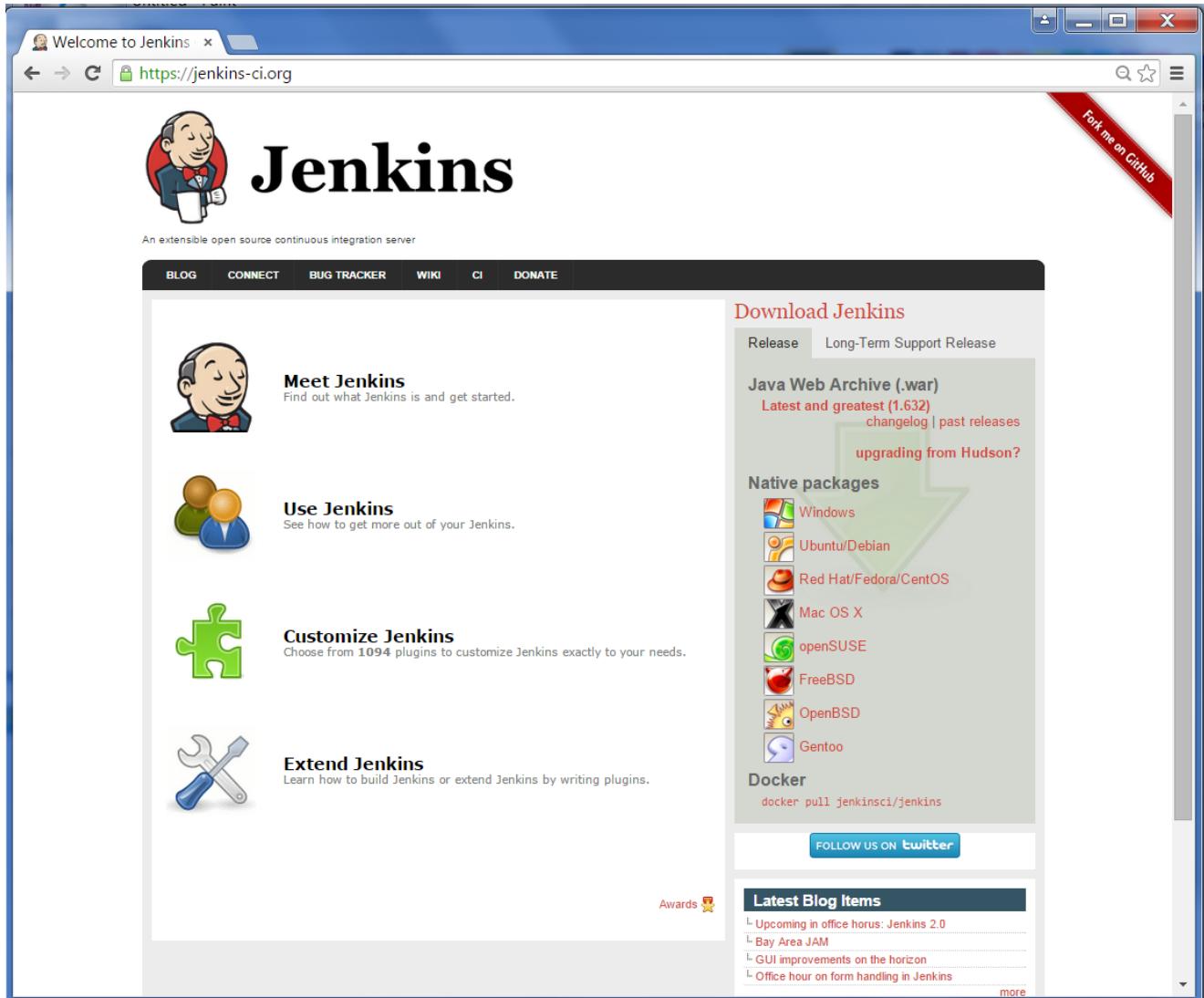
## System Requirements

JDK	JDK 1.5 or above
Memory	2 GB RAM (recommended)
Disk Space	No minimum requirement. Note that since all builds will be stored on the Jenkins machines, it has to be ensured that sufficient disk space is available for build storage.
Operating System Version	Jenkins can be installed on Windows, Ubuntu/Debian, Red Hat/Fedora/CentOS, Mac OS X, openSUSE, FreeBSD, OpenBSD, Gentoo.
Java Container	The WAR file can be run in any container that supports Servlet 2.4/JSP 2.0 or later.(An example is Tomcat 5).

## 2. Installing Jenkins

### Download Jenkins

The official website for Jenkins is <https://jenkins-ci.org/>. If you click the given link, you can get the home page of the Jenkins official website as shown below.



By default, the latest release and the Long-Term support release will be available for download. The past releases are also available for download. Click the Long-Term Support Release tab in the download section.

Click the link "Older but stable version" to download the Jenkins war file.

## Starting Jenkins

Open the command prompt. From the command prompt, browse to the directory where the jenkins.war file is present. Run the following command

```
D:\>Java -jar Jenkins.war
```

After the command is run, various tasks will run, one of which is the extraction of the war file which is done by an embedded webserver called winstome.

```
D:\>Java -jar Jenkins.war
Running from: D:\jenkins.war
Webroot: $user.home/.jenkins
```

```
Sep 29, 2015 4:10:46 PM winstone.Logger logInternal  
INFO: Beginning extraction from war file
```

Once the processing is complete without major errors, the following line will come in the output of the command prompt.

```
INFO: Jenkins is fully up and running
```

## Accessing Jenkins

Once Jenkins is up and running, one can access Jenkins from the link – <http://localhost:8080>

This link will bring up the Jenkins dashboard.

The screenshot shows the Jenkins dashboard running on a local host at port 8080. The browser title bar reads "Dashboard [Jenkins]". The main content area features the Jenkins logo and the heading "Welcome to Jenkins!". Below it, a message says "Please [create new jobs](#) to get started." On the left sidebar, there are links for "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". Under "Build Queue", it says "No builds in the queue.". Under "Build Executor Status", it lists "1 Idle" and "2 Idle". At the bottom, there are links for "Help us localize this page", "Page generated: Oct 6, 2015 10:40:33 PM", "REST API", and "Jenkins ver. 1.609.3".

### 3. Jenkins – Tomcat Setup

The following prerequisites must be met for Jenkins Tomcat setup.

#### Step 1: Verifying Java Installation

To verify Java installation, open the console and execute the following java command.

OS	Task	Command
Windows	Open command console	\>java -version
Linux	Open command terminal	\$java -version

If Java has been installed properly on your system, then you should get one of the following outputs, depending on the platform you are working on.

OS	Output
Windows	Java version "1.7.0_60" Java (TM) SE Run Time Environment (build 1.7.0_60-b19) Java Hotspot (TM) 64-bit Server VM (build 24.60-b09, mixed mode)
Linux	java version "1.7.0_25" Open JDK Runtime Environment (rhel-2.3.10.4.el6_4-x86_64) Open JDK 64-Bit Server VM (build 23.7-b01, mixed mode)

We assume the readers of this tutorial have Java 1.7.0\_60 installed on their system before proceeding for this tutorial.

In case you do not have Java JDK, you can download it from the link  
<http://www.oracle.com/technetwork/java/javase/downloads/dk7-downloads-1880260.html>

#### Step 2: Verifying Java Installation

Set the JAVA\_HOME environment variable to point to the base directory location where Java is installed on your machine. For example,

OS	Output
Windows	Set Environmental variable JAVA_HOME to C:\ProgramFiles\java\jdk1.7.0_60
Linux	export JAVA_HOME=/usr/local/java-current

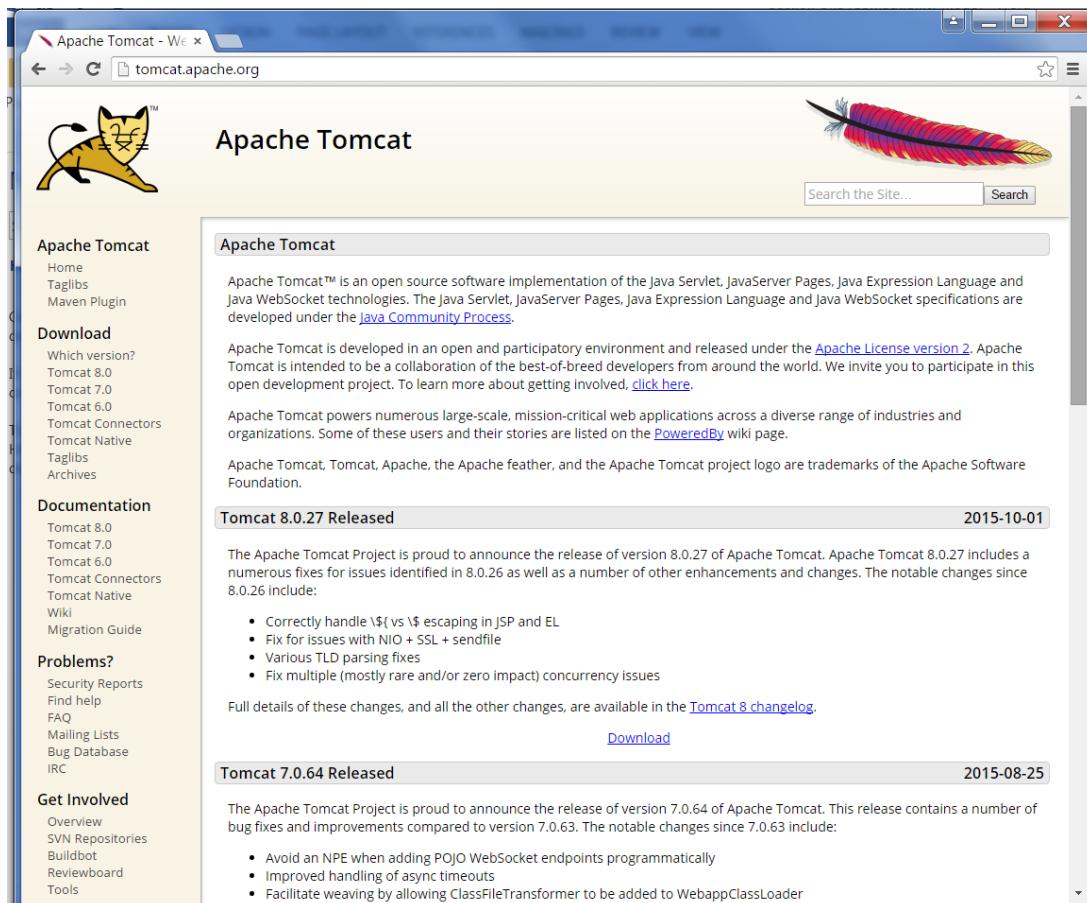
Append the full path of the Java compiler location to the System Path.

OS	Output
Windows	Append the String; C:\Program Files\Java\jdk1.7.0_60\bin to the end of the system variable PATH.
Linux	export PATH=\$PATH:\$JAVA_HOME/bin/

Verify the command java-version from command prompt as explained above.

### Step 3: Download Tomcat

The official website for tomcat is <http://tomcat.apache.org/>. If you click the given link, you can get the home page of the tomcat official website as shown below.



Browse to the link <https://tomcat.apache.org/download-70.cgi> to get the download for tomcat.

Go to the 'Binary Distributions' section. Download the 32-bit Windows zip file.

Then unzip the contents of the downloaded zip file.

## Step 4: Jenkins and Tomcat Setup

Copy the Jenkins.war file which was downloaded from the previous section and copy it to the webapps folder in the tomcat folder.

Now open the command prompt. From the command prompt, browse to the directory where the tomcat7 folder is located. Browse to the bin directory in this folder and run the start.bat file

```
E:\Apps\tomcat7\bin>startup.bat
```

Once the processing is complete without major errors, the following line will come in the output of the command prompt.

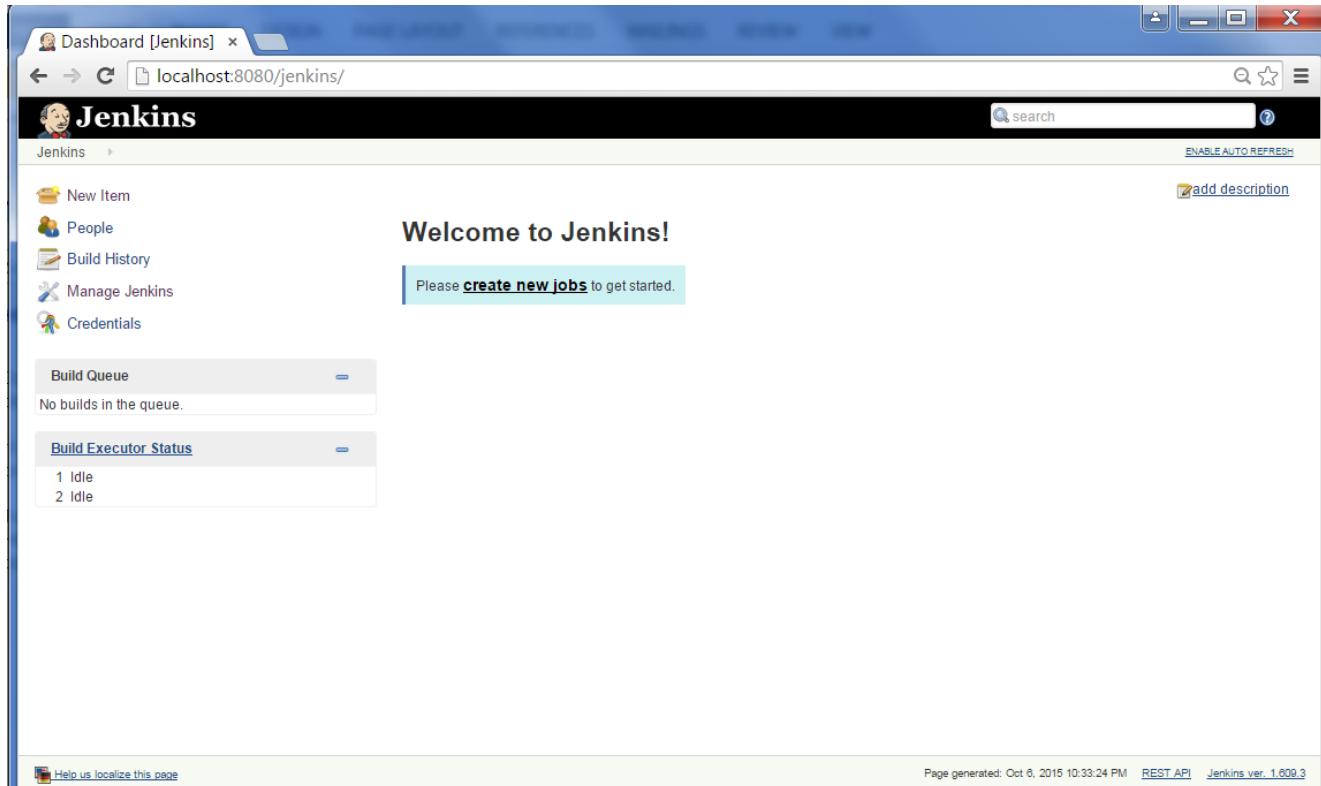
INFO: Server startup in 1302 ms

Open the browser and go to the link – <http://localhost/jenkins>. Jenkins will be up and running on tomcat.

The screenshot shows the Jenkins dashboard at <http://localhost:8080/jenkins>. The title bar says "Dashboard [Jenkins]". The main content area features a large "Welcome to Jenkins!" heading and a message: "Please [create new jobs](#) to get started." On the left sidebar, there are links for "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". Below the sidebar, two sections are displayed: "Build Queue" (which shows "No builds in the queue.") and "Build Executor Status" (which shows "1 Idle" and "2 Idle"). At the bottom of the page, there are links for "Help us localize this page", "Page generated: Oct 6, 2015 10:33:24 PM", "REST API", and "Jenkins ver. 1.609.3".

## 4. Jenkins – Git Setup

For this exercise, you have to ensure that Internet connectivity is present from the machine on which Jenkins is installed. In your Jenkins Dashboard (Home screen), click the Manage Jenkins option on the left hand side.



In the next screen, click the 'Manage Plugins' option.

The screenshot shows the Jenkins 'Manage Jenkins' interface. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below these are two collapsed sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'Manage Jenkins' and contains several configuration options. At the top of this section, there are three warning messages:

- 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 18n](#) for more details.
- Unsecured Jenkins allows anyone on the network to launch processes on your behalf.
- Consider at least enabling authentication to discourage misuse.

Below these messages are buttons for 'Setup Security' and 'Dismiss'. The main list of management options includes:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. ([updates available](#))
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from `java.util.logging` related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Manage Credentials**: Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
- About Jenkins**: See the version and license information.
- Manage Old Data**: Scrub configuration files to remove remnants from old plugins and earlier versions.
- In-process Script Approval**: Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.
- Prepare for Shutdown**: Stops executing new builds, so that the system can eventually shut down safely.

In the next screen, click the Available tab. This tab will give a list of plugins which are available for downloading. In the 'Filter' tab type 'Git plugin'

The screenshot shows the Jenkins Plugin Manager interface. At the top, there's a navigation bar with links for 'Back to Dashboard' and 'Manage Jenkins'. Below this is a search bar with the placeholder 'search' and a filter input field containing 'Git plugin'. The main area has tabs for 'Updates', 'Available' (which is selected), 'Installed', and 'Advanced'. A 'Install' button is at the top left of the list. The list table has columns for 'Name' and 'Version'. The 'Available' tab displays several Git-related plugins:

Name	Version
Git Parameter Plug-In	0.4.0
UserContent in Git.plugin	1.4
Alternative build chooser	1.1
Team Concert Git Plugin	1.0.10
Tracking Git Plugin	1.0
GIT plugin	2.4.0

At the bottom of the page, there are buttons for 'Install without restart' and 'Download now and install after restart'. A status message says 'Update information obtained: 1 hr 0 min ago'. On the far right, there's a 'Check now' button. The footer includes links for 'Help us localize this page', 'Page generated: Oct 6, 2015 10:30:10 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

The list will then be filtered. Check the Git Plugin option and click on the button 'Install without restart'

The screenshot shows the Jenkins Update Center interface. The URL in the browser is `localhost:8080/jenkins/pluginManager/available`. The page title is "Update Center [Jenkins]". The main content area is titled "Plugin Manager" with a "Available" tab selected. A search bar at the top right contains the text "search" and a filter input with the value "Git plugin". Below the search bar, there is a "Filter:" dropdown set to "Git plugin". The plugin list table has columns for "Name" and "Version". The "GIT plugin" row is highlighted with a blue background and has a checked checkbox in its "Install" column. Other visible plugins include "Git Parameter Plug-in", "UserContent in Git plugin", "Alternative build chooser", "Team Concert Git Plugin", and "Tracking Git Plugin". At the bottom of the table, there are two buttons: "Install without restart" and "Download now and install after restart". To the right of these buttons, a message says "Update information obtained: 1 hr 0 min ago". On the far right, there is a "Check now" button. The footer of the page includes links for "Help us localize this page", "Page generated: Oct 6, 2015 10:30:16 PM", "REST API", and "Jenkins ver. 1.809.3".

Install	Name	Version
<input type="checkbox"/>	<a href="#">Git Parameter Plug-in</a> This plugin allows you to choose between Git tags or sha1 of your SCM repository so Git Plugin installed is required.	0.4.0
<input type="checkbox"/>	<a href="#">UserContent in Git plugin</a> This plugin exposes \$JENKINS_HOME/userContent as Git repository.	1.4
<input type="checkbox"/>	<a href="#">Alternative build chooser</a> An alternative build chooser plugin for the Jenkins git plugin.	1.1
<input type="checkbox"/>	<a href="#">Team Concert Git Plugin</a> Integrates Jenkins with <a href="#">Rational Team Concert</a> for Jenkins Builds which use Git as source control. This plugin will create traceability links from a Jenkins build to Rational Team Concert <a href="#">Work Items</a> and <a href="#">build</a> results. This plugin adds traceability links from a Jenkins build to an RTC build result. It also publishes links to work items and annotates the change log generated by Jenkins with links to RTC Work Items; It leverages the current RTC features and workflows that users are already familiar with such as, emails, toaster popups, reporting, dashboards, etc.	1.0.10
<input type="checkbox"/>	<a href="#">Tracking Git Plugin</a> Lets one project track the Git revisions that are built for another project.	1.0
<input checked="" type="checkbox"/>	<a href="#">GIT plugin</a> This plugin allows use of <a href="#">Git</a> as a build SCM. A recent Git runtime is required (1.7.9 minimum, 1.8.x recommended). Plugin is only tested on official <a href="#">git client</a> . Use exotic installations at your own risks.	2.4.0

The installation will then begin and the screen will be refreshed to show the status of the download.

A screenshot of a web browser displaying the Jenkins Update Center. The URL in the address bar is `localhost:8080/jenkins/updateCenter/`. The page title is "Update Center [Jenkins]". The main content area is titled "Installing Plugins/Upgrades". On the left sidebar, there are links for "Back to Dashboard", "Manage Jenkins", and "Manage Plugins". The central content area shows a list of plugins with their current status as "Pending": Credentials Plugin, SSH Credentials Plugin, GIT client plugin, SCM API Plugin, Mailer Plugin, and GIT plugin. A note at the bottom says "Restart Jenkins when installation is complete and no jobs are running". At the bottom of the page, there is a link "Help us localize this page" and footer text "Page generated: Oct 6, 2015 10:42:52 PM REST API Jenkins ver. 1.609.3".

Once all installations are complete, restart Jenkins by issue the following command in the browser.

<http://localhost:8080/jenkins/restart>

After Jenkins is restarted, Git will be available as an option whilst configuring jobs. To verify, click on New Item in the menu options for Jenkins. Then enter a name for a job, in the following case, the name entered is 'Demo'. Select 'Freestyle project' as the item type. Click the Ok button.

New Item [Jenkins] x localhost:8080/jenkins/view/All/newJob search

Jenkins All

New Item Item name Demo

Freestyle project This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

External Job This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See [the documentation for more details](#).

Multi-configuration project Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Build Queue No builds in the queue.

Build Executor Status 1 Idle 2 Idle

OK

Help us localize this page Page generated: Oct 6, 2015 10:45:17 PM REST API Jenkins ver. 1.609.3

In the next screen, if you browse to the Source code Management section, you will now see 'Git' as an option.

The screenshot shows the Jenkins configuration interface for a job named 'Demo'. The 'Source Code Management' section is expanded, displaying the following options:

- Source Code Management: Radio buttons for None, CVS, CVS Projectset, Git, and Subversion. 'None' is selected.
- Build Triggers:
  - Build after other projects are built
  - Build periodically
  - Poll SCM
- Build:
  - Add build step
- Post-build Actions:
  - Add post-build action

At the bottom are 'Save' and 'Apply' buttons.

# 5. Jenkins – Maven Setup

## Step 1: Downloading and Setting Up Maven

The official website for maven is <https://maven.apache.org/download.cgi>. If you click the given link, you can get the home page of the maven official website as shown below.

The screenshot shows a web browser window displaying the Apache Maven Project download page at <https://maven.apache.org/download.cgi>. The page features the Apache feather logo and the text "Apache Maven Project" and "Maven™". A sidebar on the left contains links for MAIN (Welcome, License, Download, Install, Configure, Run, IDE Integration), ABOUT MAVEN (What is Maven?, Features, FAQ, Support and Training), DOCUMENTATION (Maven Plugins, Index (category), Running Maven, User Centre, Plugin Developer Centre, Maven Repository Centre, Maven Developer Centre, Books and Resources). The main content area is titled "Downloading Apache Maven 3.3.3" and includes a note about the latest release. It lists system requirements for Java Development Kit (JDK), Memory, Disk, and Operating System. Below this is a section on "Files" with information about Maven distribution formats and verification. At the bottom, there are download links for "Link", "Checksum", and "Signature".

While browsing to the site, go to the Files section and download the link to the Binary.zip file.

	<b>Link</b>	<b>Checksum</b>	<b>Signature</b>
Binary tar.gz archive	<a href="#">apache-maven-3.3.3-bin.tar.gz</a>	<a href="#">apache-maven-3.3.3-bin.tar.gz.md5</a>	<a href="#">apache-maven-3.3.3-bin.tar.gz.asc</a>
Binary zip archive	<a href="#">apache-maven-3.3.3-bin.zip</a>	<a href="#">apache-maven-3.3.3-bin.zip.md5</a>	<a href="#">apache-maven-3.3.3-bin.zip.asc</a>
Source tar.gz archive	<a href="#">apache-maven-3.3.3-src.tar.gz</a>	<a href="#">apache-maven-3.3.3-src.tar.gz.md5</a>	<a href="#">apache-maven-3.3.3-src.tar.gz.asc</a>
Source zip archive	<a href="#">apache-maven-3.3.3-src.zip</a>	<a href="#">apache-maven-3.3.3-src.zip.md5</a>	<a href="#">apache-maven-3.3.3-src.zip.asc</a>

- [Release Notes](#)
- [Reference Documentation](#)
- [Apache Maven Website As Documentation Archive](#)
- [All sources \(plugins, shared libraries,...\) available at <http://www.apache.org/dist/maven/>](#)
- [Distributed under the Apache License, version 2.0](#)

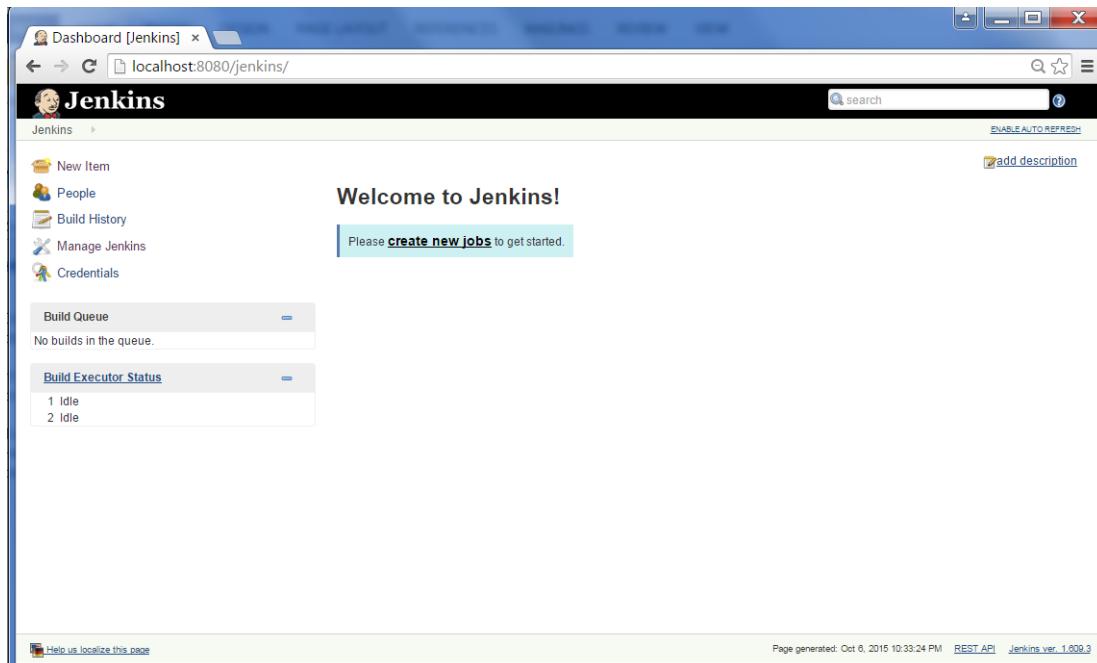
## Previous Releases

It is strongly recommended to use the latest release version of Apache Maven to take advantage of newest features and bug fixes. If you still want to use an old version you can find more information in the [Maven Releases History](#) and can download files from the [archives](#) for versions 3.0.4+ and [legacy archives](#) for earlier releases.

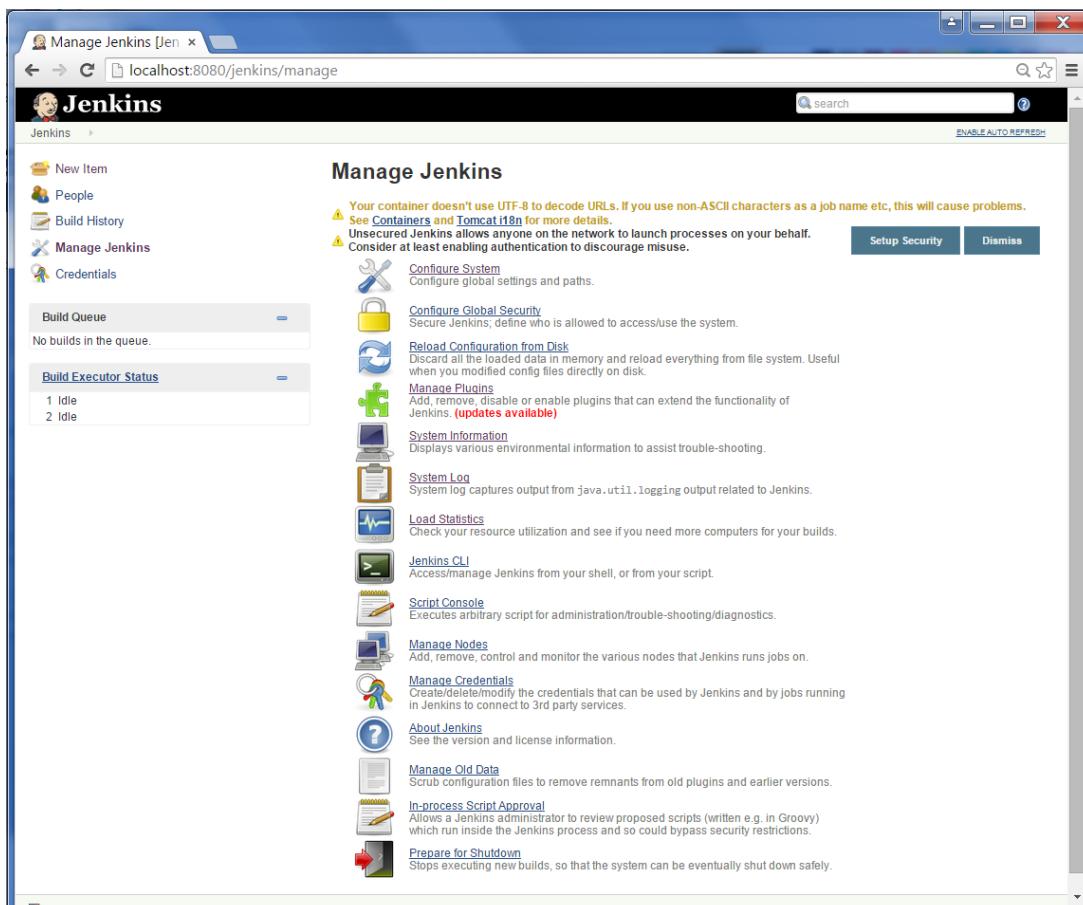
Once the file is downloaded, extract the files to the relevant application folder. For this purpose, the maven files will be placed in E:\Apps\apache-maven-3.3.3.

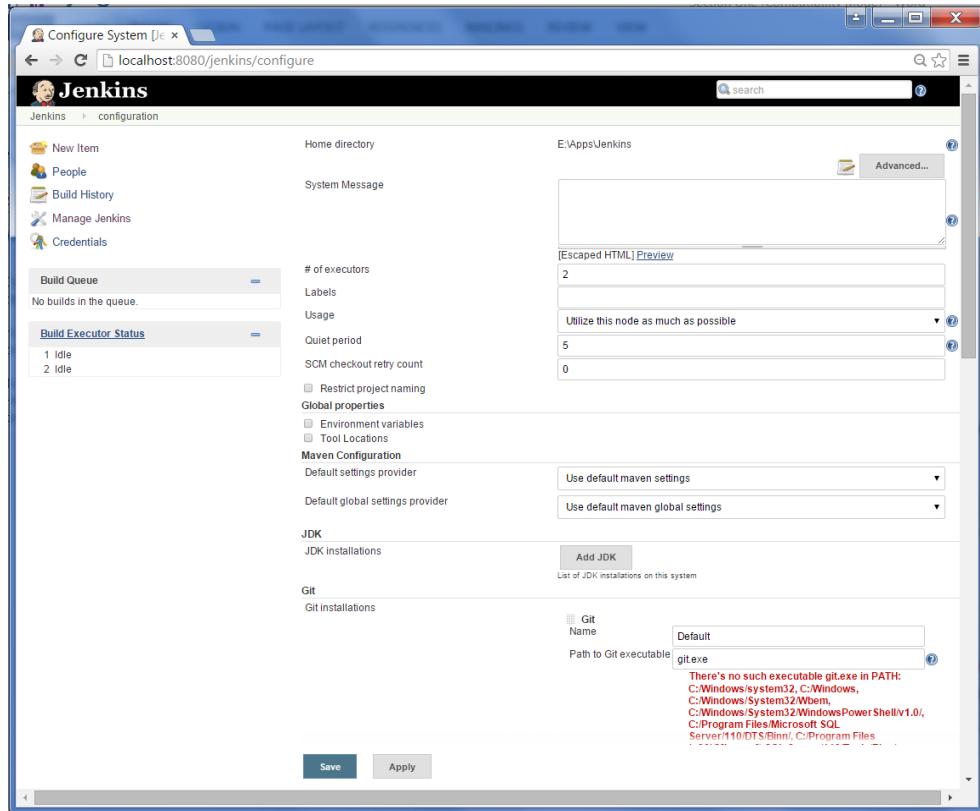
## Step 2: Setting up Jenkins and Maven

In the Jenkins dashboard (Home screen), click Manage Jenkins from the left-hand side menu.

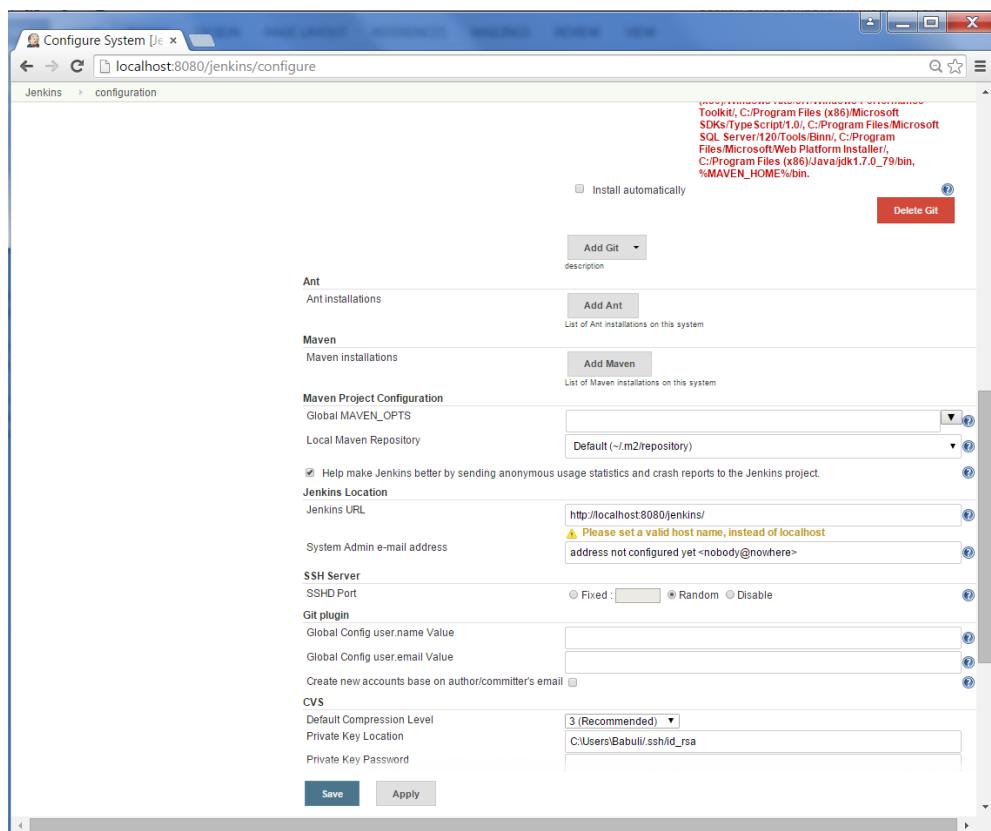


Then, click on 'Configure System' from the right hand side.





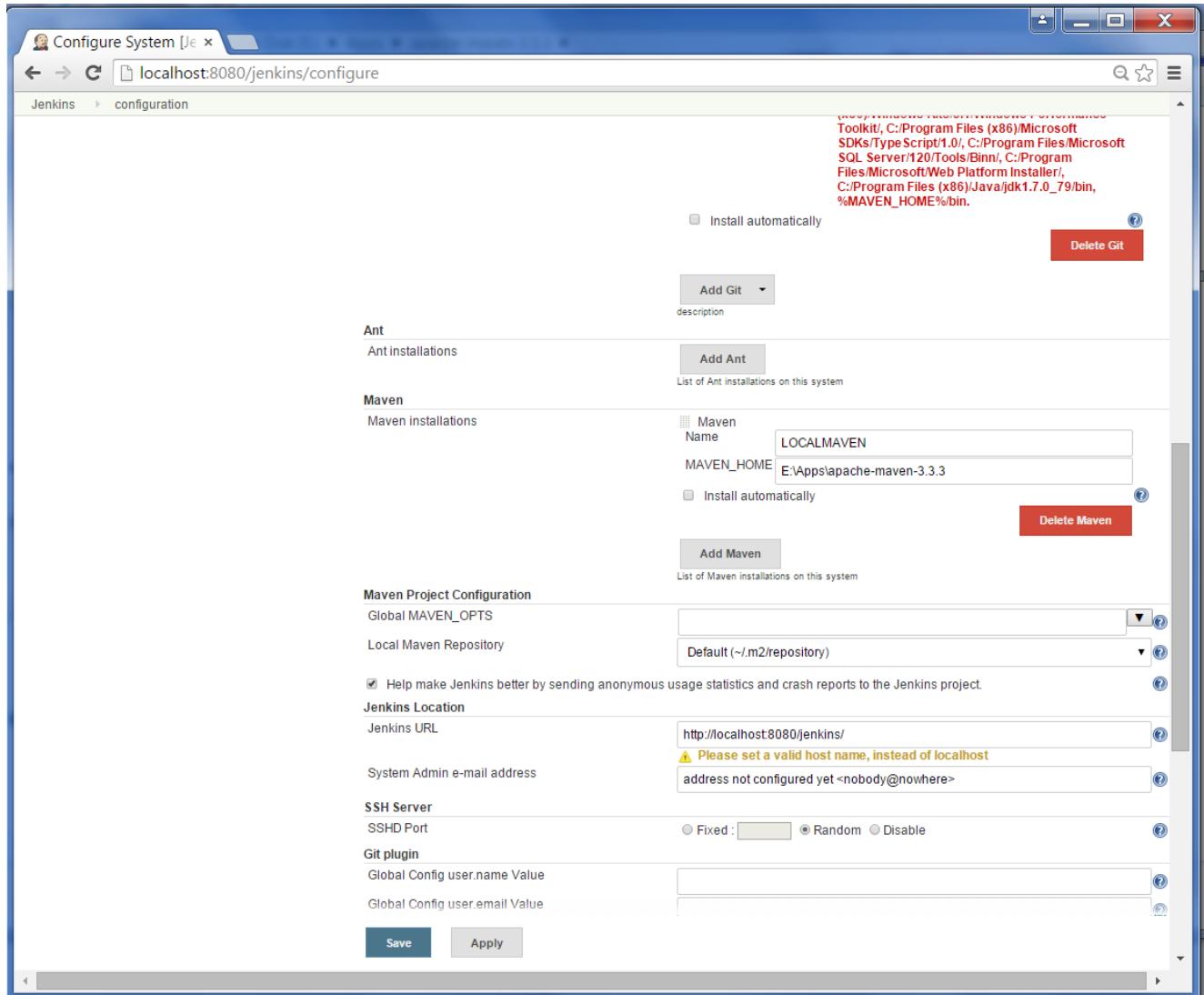
In the Configure system screen, scroll down till you see the Maven section and then click on the 'Add Maven' button.



Uncheck the 'Install automatically' option.

Add any name for the setting and the location of the MAVEN\_HOME.

Then, click on the 'Save' button at the end of the screen.



You can now create a job with the 'Maven project' option. In the Jenkins dashboard, click the New Item option.

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). The left sidebar includes links for New Item, People, Build History, Manage Jenkins, and Credentials. The main area features a table for managing jobs, with columns for Status (S), Workstation (W), Name (sorted), Last Success, Last Failure, and Last Duration. A single job named 'Demo' is listed with N/A values across all metrics. Below the table, there's a legend for RSS feeds and a note about build counts (S M L). The bottom of the page includes links for localization help and page generation details.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		Demo	N/A	N/A	N/A

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

Help us localize this page Page generated: Oct 6, 2015 10:55:57 PM REST API Jenkins ver. 1.609.3

New Item [Jenkins] x

localhost:8080/jenkins/view/All/newJob

Jenkins

search

New Item

Item name: MavenDemo

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

External Job

This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See [the documentation for more details](#).

Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Copy existing Item

Copy from:

Build Queue

No builds in the queue.

Build Executor Status

1 Idle  
2 Idle

OK

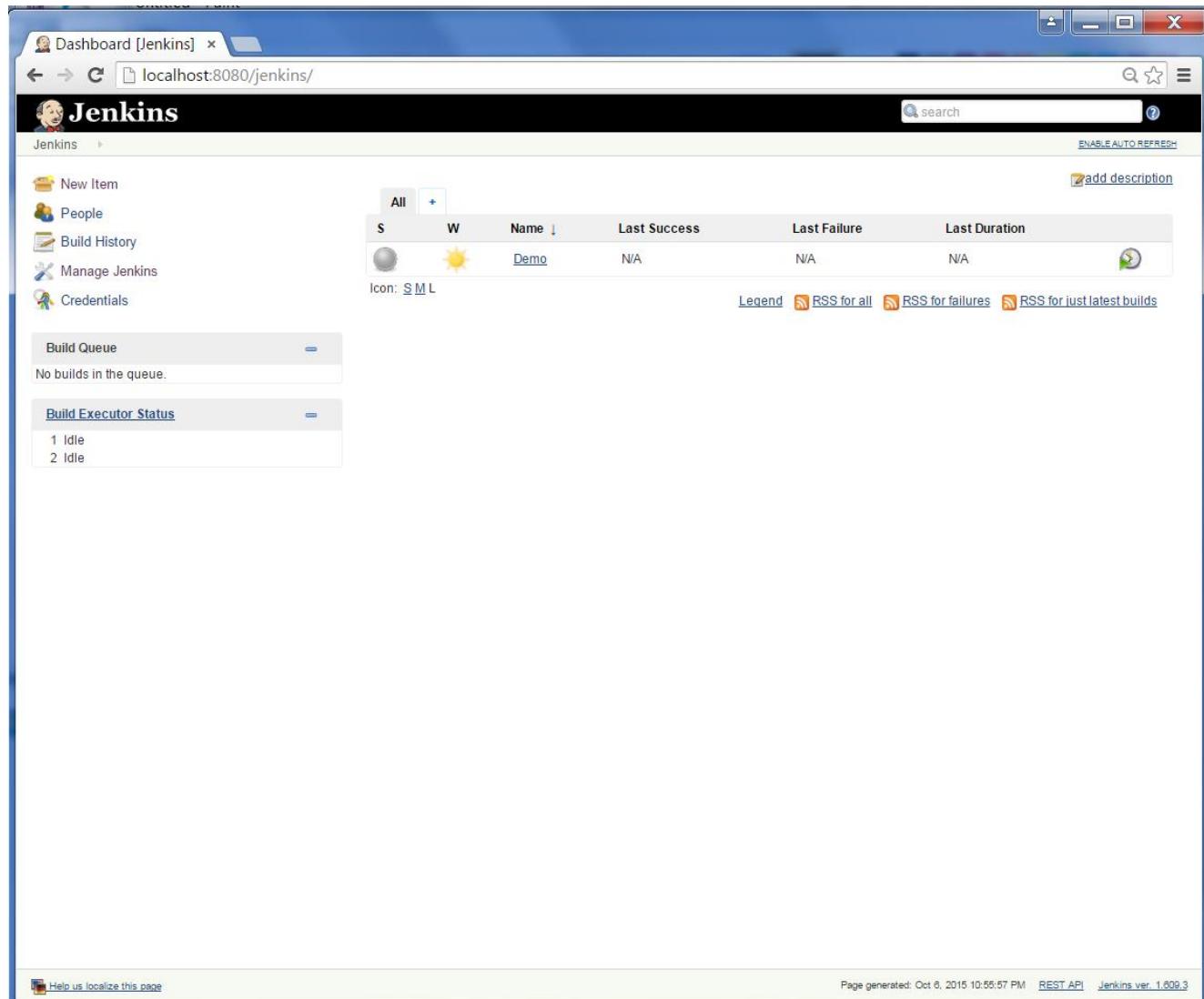
Help us localize this page

Page generated: Oct 6, 2015 10:56:50 PM REST API Jenkins ver. 1.609.3

# 6. Jenkins – Configuration

You probably would have seen a couple of times in the previous exercises wherein we had to configure options within Jenkins. The following shows the various configuration options in Jenkins.

So one can get the various configuration options for Jenkins by clicking the 'Manage Jenkins' option from the left hand menu side.



The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). The 'Manage Jenkins' option is highlighted in the left sidebar. The main area displays a table of build items, with one entry for 'Demo'. The table includes columns for Status (S), Warning (W), Name (Name), Last Success, Last Failure, and Last Duration. Below the table, there are sections for 'Build Queue' (empty) and 'Build Executor Status' (2 Idle). The bottom right corner shows page generation details: 'Page generated: Oct 6, 2015 10:55:57 PM REST API Jenkins ver. 1.609.3'.

You will then be presented with the following screen:

Click on Configure system. Discussed below are some of the Jenkins configuration settings which can be carried out.

## Jenkins Home Directory

Jenkins needs some disk space to perform builds and keep archives. One can check this location from the configuration screen of Jenkins. By default, this is set to `~/.jenkins`, and this location will initially be stored within your user profile location. In a proper environment, you need to change this location to an adequate location to store all relevant builds and archives. Once can do this in the following ways

- Set "JENKINS\_HOME" environment variable to the new home directory before launching the servlet container.
- Set "JENKINS\_HOME" system property to the servlet container.
- Set JNDI environment entry "JENKINS\_HOME" to the new directory.

The following example will use the first option of setting the "JENKINS\_HOME" environment variable.

First create a new folder E:\Apps\Jenkins. Copy all the contents from the existing ~/.jenkins to this new directory.

Set the JENKINS\_HOME environment variable to point to the base directory location where Java is installed on your machine. For example,

OS	Output
Windows	Set Environmental variable JENKINS_HOME to you're the location you desire. As an example you can set it to E:\Apps\Jenkins
Linux	export JENKINS_HOME =/usr/local/Jenkins or the location you desire.

In the Jenkins dashboard, click Manage Jenkins from the left hand side menu. Then click on 'Configure System' from the right hand side.

In the Home directory, you will now see the new directory which has been configured.

The screenshot shows the Jenkins 'Configure System' configuration page. In the 'Home directory' field, the value 'E:\Apps\Jenkins' is entered. On the right side of the page, there is a 'Git' section where the 'Path to Git executable' field contains 'git.exe'. A red error message at the bottom of this section states: 'There's no such executable git.exe in PATH: C:/Windows/system32, C:/Windows, C:/Windows/System32/Wbem, C:/Windows/System32/WindowsPowerShell/v1.0/, C:/Program Files/Microsoft SQL Server/110/DTS/Binn/, C:/Program Files/...'. Below the configuration fields are 'Save' and 'Apply' buttons.

## # of executors

This refers to the total number of concurrent job executions that can take place on the Jenkins machine. This can be changed based on requirements. Sometimes the recommendation is to keep this number the same as the number of CPU on the machines for better performance.

## Environment Variables

This is used to add custom environment variables which will apply to all the jobs. These are key-value pairs and can be accessed and used in Builds wherever required.

## Jenkins URL

By default, the Jenkins URL points to localhost. If you have a domain name setup for your machine, set this to the domain name else overwrite localhost with IP of machine. This will help in setting up slaves and while sending out links using the email as you can directly access the Jenkins URL using the environment variable JENKINS\_URL which can be accessed as \${JENKINS\_URL}.

## Email Notification

In the Email Notification area, you can configure the SMTP settings for sending out emails. This is required for Jenkins to connect to the SMTP mail server and send out emails to the recipient list.

## 7. Jenkins – Management

To manage Jenkins, click on the 'Manage Jenkins' option from the left hand menu side.

So one can get the various configuration options for Jenkins by clicking the 'Manage Jenkins' option from the left hand menu side.

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). On the left, there's a sidebar with links: New Item, People, Build History, Manage Jenkins (which is selected), and Credentials. The main area has a search bar and a table for managing items. The table has columns: All, S, W, Name (sorted down), Last Success, Last Failure, and Last Duration. One item is listed: Demo, with an icon of a sun, Last Success: N/A, Last Failure: N/A, and Last Duration: N/A. Below the table, there's a legend and links for RSS feeds. At the bottom, there's a link to help localize the page and footer information: Page generated: Oct 6, 2015 10:55:57 PM | REST API | Jenkins ver. 1.609.3.

You will then be presented with the following screen:

Some of the management options are as follows:

## Configure System

This is where one can manage paths to the various tools to use in builds, such as the JDKs, the versions of Ant and Maven, as well as security options, email servers, and other system-wide configuration details. When plugins are installed, Jenkins will add the required configuration fields dynamically after the plugins are installed.

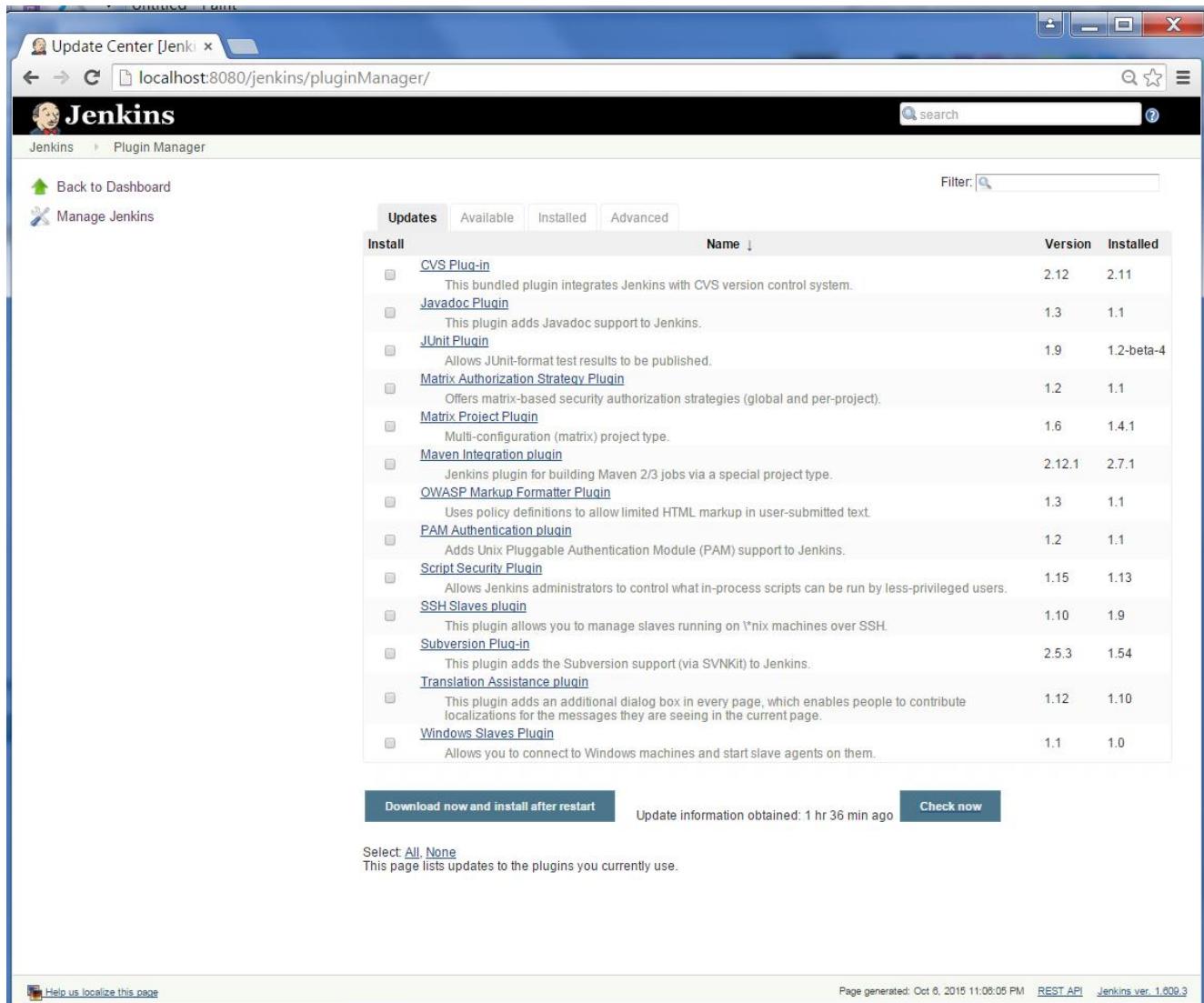
## Reload Configuration from Disk

Jenkins stores all its system and build job configuration details as XML files which is stored in the Jenkins home directory. Here also all of the build history is stored. If you are migrating build jobs from one Jenkins instance to another, or archiving old build jobs, you will need to add or remove the corresponding build job directories to Jenkins's *builds* directory. You don't need to take

Jenkins offline to do this—you can simply use the “Reload Configuration from Disk” option to reload the Jenkins system and build job configurations directly.

## Manage Plugins

Here one can install a wide variety of third-party plugins right from different Source code management tools such as Git, Mercurial or ClearCase, to code quality and code coverage metrics reporting. Plugins can be installed, updated and removed through the Manage Plugins screen.



The screenshot shows the Jenkins Manage Plugins interface. At the top, there's a navigation bar with links for 'Back to Dashboard' and 'Manage Jenkins'. Below that is a search bar and a filter button. The main area has tabs for 'Updates', 'Available', 'Installed', and 'Advanced', with 'Updates' currently selected. A table lists various plugins with their names, descriptions, current version, and installed status. At the bottom, there are buttons for 'Download now and install after restart' and 'Check now'.

Install	Name	Version	Installed
<a href="#">CVS Plug-in</a>	This bundled plugin integrates Jenkins with CVS version control system.	2.12	2.11
<a href="#">Javadoc Plugin</a>	This plugin adds Javadoc support to Jenkins.	1.3	1.1
<a href="#">JUnit Plugin</a>	Allows JUnit-format test results to be published.	1.9	1.2-beta-4
<a href="#">Matrix Authorization Strategy Plugin</a>	Offers matrix-based security authorization strategies (global and per-project).	1.2	1.1
<a href="#">Matrix Project Plugin</a>	Multi-configuration (matrix) project type.	1.6	1.4.1
<a href="#">Maven Integration plugin</a>	Jenkins plugin for building Maven 2/3 jobs via a special project type.	2.12.1	2.7.1
<a href="#">OWASP Markup Formatter Plugin</a>	Uses policy definitions to allow limited HTML markup in user-submitted text.	1.3	1.1
<a href="#">PAM Authentication plugin</a>	Adds Unix Pluggable Authentication Module (PAM) support to Jenkins.	1.2	1.1
<a href="#">Script Security Plugin</a>	Allows Jenkins administrators to control what in-process scripts can be run by less-privileged users.	1.15	1.13
<a href="#">SSH Slaves plugin</a>	This plugin allows you to manage slaves running on Unix machines over SSH.	1.10	1.9
<a href="#">Subversion Plug-in</a>	This plugin adds the Subversion support (via SVNKit) to Jenkins.	2.5.3	1.54
<a href="#">Translation Assistance plugin</a>	This plugin adds an additional dialog box in every page, which enables people to contribute localizations for the messages they are seeing in the current page.	1.12	1.10
<a href="#">Windows Slaves Plugin</a>	Allows you to connect to Windows machines and start slave agents on them.	1.1	1.0

## System Information

This screen displays a list of all the current Java system properties and system environment variables. Here one can check exactly what version of Java Jenkins is running in, what user it is running under, and so forth.

The following screenshot shows some of the name-value information available in this section.

The screenshot shows the Jenkins System Information page at [localhost:8080/jenkins/systemInfo](http://localhost:8080/jenkins/systemInfo). The left sidebar includes links for New Item, People, Build History, Manage Jenkins, and Credentials. Under Build Queue and Build Executor Status, there are no builds in the queue or running. The main content area is titled "System Properties" and displays a table of system properties with their values. Key entries include catalina.base (E:\Apps\tomcat7), catalina.home (E:\Apps\tomcat7), catalina.useNaming (true), common.loader (\$\{catalina.base\}/lib\$\{catalina.base\}/lib/\* jar\$\{catalina.home\}/lib\$\{catalina.home\}/lib/\* jar), file.encoding (Cp1252), file.separator (\), java.awt.graphicsenv (sun.awt.Win32GraphicsEnvironment), java.awt.printerjob (sun.awt.windows.WPrinterJob), java.class.path (E:\Apps\tomcat7\bin\bootstrap.jar;E:\Apps\tomcat7\bin\tomcat-juli.jar), java.class.version (51.0), java.endorsed.dirs (E:\Apps\tomcat7\endorsed), java.ext.dirs (C:\Program Files (x86)\Java\jdk1.7.0\_79\jre\lib\ext;C:\Windows\SunJava\lib\ext), java.home (C:\Program Files (x86)\Java\jdk1.7.0\_79\jre), java.io.tmpdir (E:\Apps\tomcat7\temp), and java.library.path (C:\Program Files (x86)\Java\jdk1.7.0\_79\bin;C:\Windows\SunJava\bin;C:\Windows\System32;C:\Windows\system32;C:\Windows;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Program Files\Microsoft SQL Server\110\Tools\Binn\;C:\Program Files\Microsoft SQL Server\110\Tools\Binn\ManagementStudio\;C:\Program Files\Microsoft SQL Server\110\Tools\Binn\ManagementStudio\;C:\Program Files\Microsoft Visual Studio 10.0\Common7\IDE\PrivateAssemblies\;C:\Program Files (x86)\Microsoft SQL Server\110\Tools\Binn\;C:\Program Files (x86)\Windows Kits\8.1\Windows Performance Toolkit\;C:\Program Files (x86)\Microsoft SDKs\TypeScript\1.0\;C:\Program Files\Microsoft SQL Server\120\Tools\Binn\;C:\Program Files\Microsoft\Windows Platform\Installer\;C:\Program Files (x86)\Java\jdk1.7.0\_79\bin;%MAVEN\_HOME%\bin). Other properties listed include java.naming.factory.initial (org.apache.naming.java.javaURLContextFactory), java.naming.factory.url.pkgs (org.apache.naming), java.runtime.name (Java(TM) SE Runtime Environment), java.runtime.version (1.7.0\_79-b15), java.specification.name (Java Platform API Specification), java.specification.vendor (Oracle Corporation), java.specification.version (1.7), java.util.logging.config.file (E:\Apps\tomcat7\conf\logging.properties), java.util.logging.manager (org.apache.juli.ClassLoaderLogManager), java.vendor (Oracle Corporation), java.vendor.url (<http://java.oracle.com/>), java.vendor.url.bug (<http://bugreport.sun.com/bugreport/>), java.version (1.7.0\_79), and java.vm.info (mixed mode, sharing).

## System Log

The System Log screen is a convenient way to view the Jenkins log files in real time. Again, the main use of this screen is for troubleshooting.

## Load Statistics

This page displays graphical data on how busy the Jenkins instance is in terms of the number of concurrent builds and the length of the build queue which gives an idea of how long your builds need to wait before being executed. These statistics can give a good idea of whether extra capacity or extra build nodes is required from an infrastructure perspective.

## Script Console

This screen lets you run Groovy scripts on the server. It is useful for advanced troubleshooting since it requires a strong knowledge of the internal Jenkins architecture.

## Manage nodes

Jenkins is capable of handling parallel and distributed builds. In this screen, you can configure how many builds you want. Jenkins runs simultaneously, and, if you are using distributed builds, set up build nodes. A build node is another machine that Jenkins can use to execute its builds.

## Prepare for Shutdown

If there is a need to shut down Jenkins, or the server Jenkins is running on, it is best not to do so when a build is being executed. To shut down Jenkins cleanly, you can use the Prepare for Shutdown link, which prevents any new builds from being started. Eventually, when all of the current builds have finished, one will be able to shut down Jenkins cleanly.

## 8. Jenkins – Setup Build Jobs

For this exercise, we will create a job in Jenkins which picks up a simple HelloWorld application, builds and runs the java program.

**Step 1 :** Go to the Jenkins dashboard and Click on New Item

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). The title bar says "Dashboard [Jenkins]". The main content area features a "Welcome to Jenkins!" message with a call to action: "Please [create new jobs](#) to get started." On the left sidebar, there are links for "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". Below the sidebar, two expandable sections are visible: "Build Queue" (which shows "No builds in the queue.") and "Build Executor Status" (which shows "1 Idle" and "2 Idle"). At the bottom of the page, there are links for "Help us localize this page", "Page generated: Oct 10, 2015 12:40:44 AM", "REST API", and "Jenkins ver. 1.609.3".

**Step 2 :** In the next screen, enter the Item name, in this case we have named it Helloworld. Choose the 'Freestyle project option'

New Item [Jenkins] ×

localhost:8080/jenkins/view/All/newJob

**Jenkins**

search

New Item

People

Build History

Manage Jenkins

Credentials

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Item name **Helloworld**

Freestyle project  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Maven project  
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

External Job  
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system. See [the documentation for more details](#).

Multi-configuration project  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

OK

Help us localize this page

Page generated: Oct 10, 2015 12:41:39 AM REST API Jenkins ver. 1.609.3

**Step 3 :** The following screen will come up in which you can specify the details of the job.

The screenshot shows the Jenkins configuration interface for a job named 'Helloworld'. The left sidebar contains links like Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, and Configure. The main panel has fields for 'Project name' (set to 'Helloworld') and 'Description'. It includes sections for 'Discard Old Builds', 'GitHub project', and 'Advanced Project Options' (with 'Execute concurrent builds if necessary' checked). Under 'Source Code Management', 'None' is selected. In the 'Build Triggers' section, 'Poll SCM' is checked. The 'Build' section has an 'Add build step' button. At the bottom are 'Save' and 'Apply' buttons.

**Step 4 :** We need to specify the location of files which need to be built. In this example, we will assume that a local git repository(E:\Program) has been setup which contains a 'HelloWorld.java' file. Hence scroll down and click on the Git option and enter the URL of the local git repository.

**Note –** If your repository is hosted on GitHub, you can also enter the URL of that repository here. In addition to this, you would need to click on the Add button for the credentials to add a user name and password to the GitHub repository so that the code can be picked up from the remote repository.

The screenshot shows the Jenkins configuration interface for the 'Helloworld' job. The left sidebar includes links for Changes, Workspace, Build Now, Delete Project, and Configure. The main area displays project settings under 'Escaped HTML Preview'. Advanced options like Discard Old Builds, GitHub project, and build parameters are shown. The 'Source Code Management' section is set to Git, with a repository URL of 'E:\Program' and credentials dropdown. Branches to build are specified as '\*/\*master'. A 'Repository browser' is set to '(Auto)'. At the bottom are 'Save' and 'Apply' buttons.

**Step 5 :** Now go to the Build section and click on Add build step->Execute Windows batch command

The screenshot shows the Jenkins configuration page for a job named "Helloworld Config". The "Build" section is expanded, showing the "Add build step" dropdown menu. The "Execute Windows batch command" option is highlighted with a blue selection bar. Other options visible in the list include "Execute shell", "Invoke Ant", "Invoke top-level Maven targets", and "Set build status to 'pending' on GitHub commit". The rest of the page displays repository settings, build triggers, and additional behaviors.

**Step 6 :** In the command window, enter the following commands and then click on the Save button.

```
Java HelloWorld.java  
javac HelloWorld.java
```

The screenshot shows the Jenkins configuration page for a job named "Helloworld". The URL is [localhost:8080/jenkins/job/Helloworld/configure](http://localhost:8080/jenkins/job/Helloworld/configure). The "Build" section contains a single step: "Execute Windows batch command" with the command:  
`javac HelloWorld.java  
java HelloWorld`

**Step 7 :** Once saved, you can click on the Build Now option to see if you have successfully defined the job.

The screenshot shows the Jenkins interface for the 'Helloworld' project. At the top, there's a navigation bar with links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', and 'Configure'. Below this is a search bar and a 'Project Helloworld' title. To the right are buttons for 'add description' and 'Disable Project'. A sidebar on the left lists 'Build History' and 'RSS feeds' for all and failures. The main content area contains links for 'Workspace' and 'Recent Changes'. At the bottom, there's a footer with a localization link and build information: 'Page generated: Oct 10, 2015 12:51:53 AM REST API Jenkins ver. 1.609.3'.

**Step 8 :** Once the build is scheduled, it will run. The following Build history section shows that a build is in progress.

The screenshot shows the Jenkins interface for the 'Helloworld' project. At the top, there's a navigation bar with links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', and 'Configure'. Below this is a 'Project Helloworld' header. To the right of the header are buttons for 'Add description' and 'Disable Project'. A sidebar on the left contains a 'Build History' section with a single entry for build #1, which was run on Oct 10, 2015 at 12:52 AM. There are also links for 'Recent Changes' and 'Permalinks'. At the bottom of the page, there are links for 'RSS for all' and 'RSS for failures', along with a link to help localize the page and footer information about the page's generation date and Jenkins version.

**Step 9 :** Once the build is completed, a status of the build will show if the build was successful or not. In our case, the following build has been executed successfully. Click on the #1 in the Build history to bring up the details of the build.

The screenshot shows the Jenkins interface for the 'Helloworld' project. The title bar says 'Helloworld [Jenkins]'. The URL in the address bar is 'localhost:8080/jenkins/job/Helloworld/'. The main content area is titled 'Project Helloworld'. On the left, there's a sidebar with links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', and 'Configure'. On the right, there are buttons for 'add description' and 'Disable Project'. Below these are two icons: 'Workspace' (a folder icon) and 'Recent Changes' (a notebook icon). A 'Permalinks' section shows a build history entry for '#1 Oct 10, 2015 12:52 AM'. At the bottom, there are links for 'RSS for all' and 'RSS for failures'. The footer includes links for 'Help us localize this page', 'Page generated: Oct 10, 2015 12:51:53 AM', 'REST API', and 'Jenkins ver. 1.609.3'.

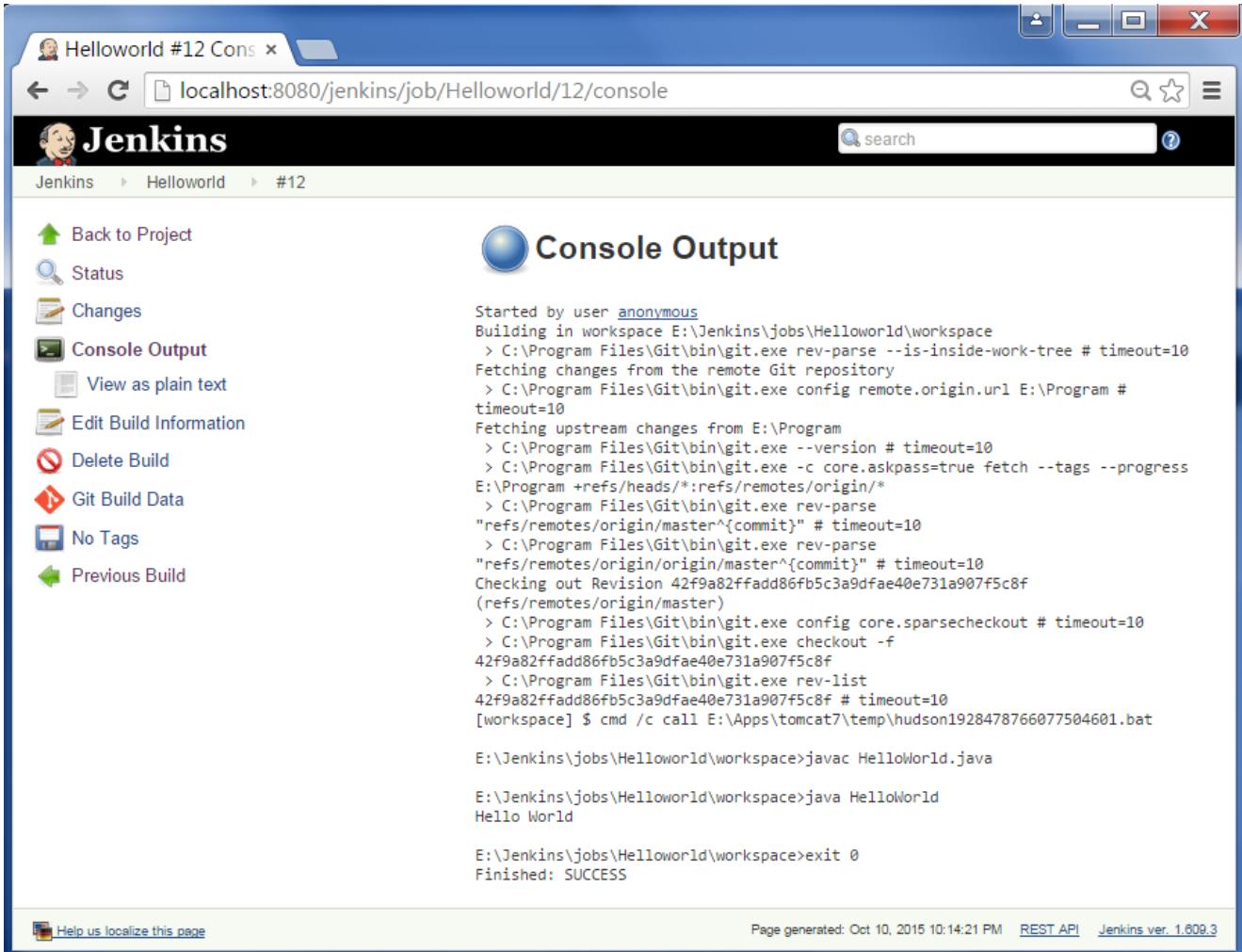
**Step 10 :** Click on the Console Output link to see the details of the build

The screenshot shows a Jenkins build page for the project 'Helloworld'. The build number is '#1', and it was started on Oct 10, 2015, at 12:52:50 AM. The build took 4.7 seconds and completed 4 minutes and 40 seconds ago. There are no changes, and it was started by an anonymous user. The revision is 42f9a82ffadd86fb5c3a9dfae40e731a907f5c8f, with the master branch. A 'Console Output' link is visible on the left sidebar.

Build #1 (Oct 10, 2015 12:52:50 AM)  
Started 4 min 40 sec ago  
Took 4.7 sec

No changes.  
Started by anonymous user  
Revision: 42f9a82ffadd86fb5c3a9dfae40e731a907f5c8f  
refs/remotes/origin/master

Help us localize this page Page generated: Oct 10, 2015 12:57:31 AM REST API Jenkins ver. 1.609.3



The screenshot shows the Jenkins interface for a build named "Helloworld #12 Cons". The main content area is titled "Console Output". The output log is as follows:

```

Started by user anonymous
Building in workspace E:\Jenkins\jobs\Helloworld\workspace
> C:\Program Files\Git\bin\git.exe rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> C:\Program Files\Git\bin\git.exe config remote.origin.url E:\Program #
timeout=10
Fetching upstream changes from E:\Program
> C:\Program Files\Git\bin\git.exe --version # timeout=10
> C:\Program Files\Git\bin\git.exe -c core.askpass=true fetch --tags --progress
E:\Program +refs/heads/*:refs/remotes/origin/*
> C:\Program Files\Git\bin\git.exe rev-parse
"refs/remotes/origin/master^{commit}" # timeout=10
> C:\Program Files\Git\bin\git.exe rev-parse
"refs/remotes/origin/origin/master^{commit}" # timeout=10
Checking out Revision 42f9a82ffadd86fb5c3a9dfa40e731a907f5c8f
(refs/remotes/origin/master)
> C:\Program Files\Git\bin\git.exe config core.sparsecheckout # timeout=10
> C:\Program Files\Git\bin\git.exe checkout -f
42f9a82ffadd86fb5c3a9dfa40e731a907f5c8f
> C:\Program Files\Git\bin\git.exe rev-list
42f9a82ffadd86fb5c3a9dfa40e731a907f5c8f # timeout=10
[workspace] $ cmd /c call E:\Apps\tomcat7\temp\hudson1928478766077504601.bat

E:\Jenkins\jobs\Helloworld\workspace>javac HelloWorld.java

E:\Jenkins\jobs\Helloworld\workspace>java HelloWorld
Hello World

E:\Jenkins\jobs\Helloworld\workspace>exit 0
Finished: SUCCESS

```

At the bottom left is a link "Help us localize this page". At the bottom right are links for "Page generated: Oct 10, 2015 10:14:21 PM", "REST API", and "Jenkins ver. 1.809.3".

Apart from the steps shown above there are just so many ways to create a build job, the options available are many, which what makes Jenkins such a fantastic continuous deployment tool.

## 9. Jenkins – Unit Testing

Jenkins provides an out of box functionality for Junit, and provides a host of plugins for unit testing for other technologies, an example being MSTest for .Net Unit tests. If you go to the link <https://wiki.jenkins-ci.org/display/JENKINS/xUnit+Plugin> it will give the list of Unit Testing plugins available.

The screenshot shows the Jenkins xUnit Plugin page. The left sidebar has a 'Documents' section with links like Meet Jenkins, Use Jenkins, Extend Jenkins, Plugins, Servlet Container Notes, etc. The main content area has a 'Plugin Information' section with tabs for 'Changes' (In Latest Release, Since Latest Release), 'Source Code' (GitHub), 'Issue Tracking' (Open Issues), 'Pull Requests', and 'Maintainer(s)' (Gregory Boissinot). It also shows a 'Usage' chart titled 'xunit - installations' with data from Oct 2014 to Sep 2015. Below the chart is a note: 'This plugin makes it possible to publish the test results of an execution of a testing tool in Jenkins.' At the bottom, there's a 'CppUnit output' section with a progress bar.

Plugin ID	xunit	Changes	In Latest Release Since Latest Release
Latest Release	<a href="#">1.98 (archives)</a>	Source Code	<a href="#">GitHub</a>
Latest Release Date	Oct 09, 2015	Issue Tracking	<a href="#">Open Issues</a>
Required Core Dependencies	<a href="#">1.580.1</a> <a href="#">junit (version:1.6)</a>	Pull Requests	<a href="#">Pull Requests</a>
Usage	xunit - installations	Maintainer(s)	<a href="#">Gregory Boissinot (id: gboissinot)</a>
		Installations	2014-Oct 11692 2014-Nov 11557 2014-Dec 11631 2015-Jan 12106 2015-Feb 12262 2015-Mar 12891 2015-Apr 12694 2015-May 12716 2015-Jun 13143 2015-Jul 13470 2015-Aug 13192 2015-Sep 13563

This plugin makes it possible to publish the test results of an execution of a testing tool in Jenkins.

CppUnit output

xUnit Plugin - Jenkins

<https://wiki.jenkins-ci.org/display/JENKINS/xUnit+Plugin>

## Features

- Records xUnit tests
- Mark the build unstable or fail according to threshold values

## Supported tools

### Embedded tools

- \* JUnit itself
- \* AUnit
- \* MSTest (imported from [MSTest Plugin](#))
- \* NUnit (imported from [NUnit Plugin](#))
- \* UnitTest++
- \* Boost Test Library
- \* PHPUnit
- \* Free Pascal Unit
- \* CppUnit
- \* MbUnit
- \* GoogleTest
- \* EmbUnit
- \* gtester/glib
- \* QTestLib

### Other plugins as an extension of the xUnit plugin:

- \* Gallio ([Gallio plugin](#))
- \* Parasoft C++ Test tool ([Cpptest Plugin](#))
- \* JSUnit ([JSUnit Plugin](#))
- \* JBehave
- \* TestComplete ([TestComplete xUnit Plugin](#))

### External contributions

## Example of a JUnit Test in Jenkins

The following example will consider

- A simple HelloWorldTest class based on Junit.
- Ant as the build tool within Jenkins to build the class accordingly.

**Step 1 :** Go to the Jenkins dashboard and Click on the existing HelloWorld project and choose the Configure option

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). On the left sidebar, there are links for New Item, People, Build History, Manage Jenkins, and Credentials. Below these are sections for Build Queue (No builds in the queue) and Build Executor Status (master has 1 idle and 2 idle executors; build\_slave is offline). In the center, a table lists projects: Helloworld (Last Success: 5 sec - #11, Last Failure: 2 days 23 hr - #10, Last Duration: 3.2 sec). An 'Icon' dropdown menu is open over the Helloworld row, showing options: Changes, Workspace, Build Now, Delete Project, and Configure (which is highlighted with a blue border). At the bottom of the page, the URL is localhost:8080/jenkins/job/.../configure and the page footer indicates it was generated on Oct 15, 2015 at 10:23:01 PM, with REST API and Jenkins ver. 1.609.3 links.

**Step 2 :** Browse to the section to Add a Build step and choose the option to Invoke Ant.

The screenshot shows the Jenkins configuration page for a job named "Helloworld". In the "Build" section, there is a step defined with the command:

```
javac HelloWorld.java  
java HelloWorld
```

An "Add build step" dropdown menu is open, showing the following options:

- Execute Windows batch command
- Execute shell
- Invoke Ant** (This option is highlighted with a blue selection bar)
- Invoke top-level Maven targets
- Set build status to "pending" on GitHub commit

**Step 3 :** Click on the Advanced button.

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. At the top, there's a header bar with the job name 'Helloworld Config' and a back arrow. Below it is a breadcrumb navigation: Jenkins > Helloworld > configuration. The main content area is titled 'Additional Behaviours' with an 'Add' dropdown menu. Under 'Build Triggers', there are four options: 'Subversion' (radio button selected), 'Build after other projects are built', 'Build periodically', 'Build when a change is pushed to GitHub', and 'Poll SCM'. In the 'Build' section, there are two entries: 'Execute Windows batch command' with a command box containing 'javac HelloWorld.java' and 'java HelloWorld'; and 'Invoke Ant' with an 'Ant Version' dropdown set to 'Default' and a 'Targets' dropdown. At the bottom of the configuration page are 'Save' and 'Apply' buttons.

**Step 4 :** In the build file section, enter the location of the build.xml file.

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. The 'Build' section contains a 'Execute Windows batch command' step with the command:

```
javac HelloWorld.java  
java HelloWorld
```

The 'Post-build Actions' section contains a 'Publish JUnit test result report' step with the 'Test report XMLs' field set to 'Reports\\*.xml'. There are 'Save' and 'Apply' buttons at the bottom.

**Step 5 :** Next click the option to Add post-build option and choose the option of “Publish JUnit test result report”

The screenshot shows the Jenkins configuration interface for a job named "Helloworld". The "Build" section contains a command box with the following content:

```
javac HelloWorld.java  
java HelloWorld
```

A dropdown menu is open under the "Add post-build action" button, showing various options. The "Publish JUnit test result report" option is highlighted with a blue selection bar.

At the bottom of the screen, the URL "localhost:8080/jenkins/job/Helloworld/configure#" is visible.

**Step 6 :** In the Test reports XML's, enter the location as shown below. Ensure that Reports is a folder which is created in the HelloWorld project workspace. The “\*.xml” basically tells Jenkins to pick up the result xml files which are produced by the running of the Junit test cases. These xml files which then be converted into reports which can be viewed later.

Once done, click the Save option at the end.

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. The top navigation bar shows 'Helloworld Config' and the URL 'localhost:8080/jenkins/job/Helloworld/configure'. The main content area is divided into sections:

- Build Step:** An 'Invoke Ant' step is configured with 'Ant Version' set to 'NewHome', 'Targets' empty, 'Build File' set to 'E:\Java\HelloWorldTest\build.xml', 'Properties' empty, and 'Java Options' empty. A red 'Delete' button is visible.
- Post-build Actions:** A 'Publish JUnit test result report' action is present. It has 'Test report XMLs' set to 'Reports\\*.xml'. A tooltip explains that this fileset includes generated raw XML report files like 'myproject/target/test-reports/\*.xml'. There is also a checked checkbox for 'Retain long standard output/error' and a value '1.0' for 'Health report amplification factor'. A note states '1% failing tests scores as 99% health. 5% failing tests scores as 95% health'. A red 'Delete' button is visible.
- Buttons:** At the bottom are 'Save' and 'Apply' buttons.

**Step 7 :** Once saved, you can click on the Build Now option.

Once the build is completed, a status of the build will show if the build was successful or not. In the Build output information, you will now notice an additional section called Test Result. In our case, we entered a negative Test case so that the result would fail just as an example.

The screenshot shows the Jenkins web interface for a project named "Helloworld". The main title bar says "Helloworld #4 [Jenki...]" and the URL is "localhost:8080/jenkins/job/Helloworld/4/". The page header includes the Jenkins logo, a search bar, and an "ENABLE AUTO REFRESH" link. On the left, there's a sidebar with links like "Back to Project", "Status", "Changes", "Console Output", "Edit Build Information", "Delete Build", "Git Build Data", "No Tags", "Test Result", "Previous Build", and "Next Build". The main content area displays the build summary for "Build #4 (Oct 12, 2015) at 8:33:16 PM". It shows a yellow success icon, the start time, and duration ("Started 3 days 1 hr ago Took 3.9 sec on master"). Below this, it says "No changes." with a note about being started by an anonymous user and provides the git revision and ref. Under "Test Result", it indicates "1 failure" with a link to "HelloWorldTestCase.initializationError". At the bottom, there are links for "Help us localize this page", "Page generated: Oct 15, 2015 10:24:38 PM", "REST API", and "Jenkins ver. 1.609.3".

One can go to the Console output to see further information. But what's more interesting is that if you click on Test Result, you will now see a drill down of the Test results.

The screenshot shows the Jenkins Test Result page for the Helloworld project, build #4. The left sidebar contains links for Back to Project, Status, Changes, Console Output, Edit Build Information, History, Git Build Data, No Tags, Test Result, and Previous Build. The main content area has a title 'Test Result' and a message '1 failures'. Below this is a summary table:

Test Name	Duration	Age
<a href="#">HelloWorldTestCase.initializationError</a>	10 ms	1

Below the summary is a section titled 'All Failed Tests' with a single entry:

Test Name
<a href="#">HelloWorldTestCase.initializationError</a>

Finally, there is a section titled 'All Tests' with a summary table:

Package	Duration	Fail	(diff)	Skip	(diff)	Pass	(diff)	Total	(diff)
<a href="#">(root)</a>	10 ms	1	+1	0	0	0	1	1	+1

At the bottom of the page are links for 'Help us localize this page', 'Page generated: Oct 12, 2015 8:45:49 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

# 10. Jenkins – Automated Testing

One of the basic principles of Continuous Integration is that a build should be verifiable. You have to be able to objectively determine whether a particular build is ready to proceed to the next stage of the build process, and the most convenient way to do this is to use automated tests. Without proper automated testing, you find yourself having to retain many build artifacts and test them by hand, which is hardly in the spirit of Continuous Integration. The following example shows how to use Selenium to run automated web tests.

**Step 1 :** Go to Manage Plugins.

The screenshot shows the Jenkins Manage Jenkins page. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is selected), and 'Credentials'. Below that are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'Manage Jenkins' and contains several configuration options. At the top right of this area, there are two buttons: 'Setup Security' and 'Dismiss'. A warning message is displayed: '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.' Below the warning, there are ten items listed with icons:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins, define who is allowed to access/use the system.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. (updates available)
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from java.util.logging output related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Manage Credentials**: Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
- About Jenkins**: See the version and license information.

**Step 2 :** Find the Hudson Selenium Plugin and choose to install. Restart the Jenkins instance.

The screenshot shows the Jenkins Plugin Manager interface. The title bar says "Update Center [jenki...]" and the address bar says "localhost:8080/jenkins/pluginManager/available". The main area has a "Jenkins" logo and a "Plugin Manager" link. A search bar at the top right contains the text "selenium". Below it, there are four tabs: "Updates", "Available" (which is selected), "Installed", and "Advanced". A "Filter" input field also contains "selenium". The main table lists several Jenkins plugins:

Install	Name	Version
<input type="checkbox"/>	Selenium Auto Exec Server(AES) plugin	0.5
<input type="checkbox"/>	This plugin is for continuous regression test by <a href="#">Selenium Auto Exec Server (AES)</a> .	
<input checked="" type="checkbox"/>	Hudson Seleniumhq plugin	0.4
<input type="checkbox"/>	This plugin allows you to run and load HTML Selenese suite result generate by Selenium Server from <a href="#">Seleniumhq</a> . Jenkins will generate the trend report of test result. The Seleniumhq plug in can be <a href="#">downloaded here</a> .	
<input type="checkbox"/>	Selenium HTML report	0.94
<input type="checkbox"/>	This plugin visualizes the results of selenium tests.	
<input type="checkbox"/>	TestingBot plugin	
<input type="checkbox"/>	This plugin allows for integration of <a href="#">TestingBot</a> Selenium in Jenkins. TestingBot provides <a href="#">cross browser testing</a> in the cloud.	1.11
<input type="checkbox"/>	TestLink Plugin	
<input type="checkbox"/>	This plug-in integrates Jenkins and <a href="#">TestLink</a> and generates reports on automated test execution. With this plug-in you can manage your tests in TestLink, schedule and control in Jenkins, and execute using your favorite test execution tool (TestPartner, Selenium, TestNG, Perl modules, PHPUnit, among others).	3.10
<input type="checkbox"/>	Nerrvana Plugin for Jenkins	
<input type="checkbox"/>	The Nerrvana Jenkins plugin allows you to automate functional and cross browser Selenium testing of your web applications in <a href="#">Nerrvana cloud</a> .	1.02.06
<input type="checkbox"/>	Sauce OnDemand plugin	
<input type="checkbox"/>	This plugin allows you to integrate <a href="#">Sauce Selenium Testing</a> with Jenkins.	1.141
<input type="checkbox"/>	Selenium Builder plugin	
<input type="checkbox"/>	Invokes <a href="#">Selenium Builder</a> scripts from a Jenkins build	1.14
<input type="checkbox"/>	SeleniumRC plugin	

At the bottom, there are two buttons: "Install without restart" and "Download now and install after restart". A status message "Update information obtained" is shown next to the download button.

**Step 3 :** Go to Configure system.

The screenshot shows the Jenkins 'Manage Jenkins' interface at the URL `localhost:8080/jenkins/manage`. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below that are sections for 'Build Queue' (empty) and 'Build Executor Status' (2 Idle). The main content area is titled 'Manage Jenkins' and contains several configuration links with icons:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. **(updates available)**
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from `java.util.logging` related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

At the top right, there are 'Setup Security' and 'Dismiss' buttons. A search bar and an 'ENABLE AUTO REFRESH' link are also present.

**Step 4 :** Configure the selenium server jar and click on the Save button.

The screenshot shows the Jenkins 'Configure System' page at [localhost:8080/jenkins/configure](http://localhost:8080/jenkins/configure). The 'Selenium Remote Control' section is highlighted, showing the 'htmlSuite Runner' field set to 'E:\Apps\selenium-server-standalone-2.48.2.jar'. Other configuration options like CVS, Subversion, and E-mail Notification are also visible.

**Selenium Remote Control**

htmlSuite Runner: E:\Apps\selenium-server-standalone-2.48.2.jar

**Shell**

Shell executable:

**E-mail Notification**

SMTP server:

Default user e-mail suffix:

Test configuration by sending test e-mail

**Buttons:** Save, Apply, Advanced...

**Note:** The selenium jar file can be downloaded from the location <http://www.seleniumhq.org/download/>

Click on the download for the Selenium standalone server.

The screenshot shows a web browser window with the URL [www.seleniumhq.org/download/](http://www.seleniumhq.org/download/) in the address bar. The page is titled "SeleniumHQ Browser Automation". The main content area is titled "Downloads" and contains information about the Selenium Server. It states: "Below is where you can find the latest releases of all the Selenium components. You can also find a list of [previous releases](#), [source code](#), and additional information for [Maven users](#) (Maven is a popular Java build tool)." Below this, there is a section for the "Selenium Standalone Server" which says: "The Selenium Server is needed in order to run either Selenium RC style scripts or Remote Selenium WebDriver ones. The 2.x server is a drop-in replacement for the old Selenium RC server and is designed to be backwards compatible with your existing infrastructure." It provides a link to "Download version 2.48.2". Further down, it says: "To use the Selenium Server in a Grid configuration [see the wiki page](#)." Another section, "The Internet Explorer Driver Server", is mentioned with a note about its requirement for the WebDriver InternetExplorerDriver. It links to "Download version 2.48.0 for (recommended) [32 bit Windows IE](#) or [64 bit Windows IE](#)". The "CHangelog" link is also provided. At the bottom, there is a section for "Selenium Client & WebDriver Language Bindings" with a note about language bindings for other languages. The page footer features the "BrowserStack" logo.

**Step 5 :** Go back to your dashboard and click on the Configure option for the HelloWorld project.

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. The main area displays a table of projects. One row for 'Helloworld' is selected, showing its status: 'Last Success' was 23 hr - #12, 'Last Failure' was 23 hr - #10, and 'Last Duration' was 3.7 sec. Below the table is a legend with four RSS feed icons: 'RSS for all', 'RSS for failures', 'RSS for just latest builds', and another unlabeled one. A context menu is open over the 'Helloworld' row, with options: 'Changes', 'Workspace', 'Build Now', 'Delete Project', and 'Configure'. The 'Configure' option is highlighted with a blue border. At the bottom of the page, the URL 'localhost:8080/jenkins/job/Helloworld/configure' is shown in the address bar, along with page generation details: 'Page generated: Oct 11, 2015 9:16:04 PM REST API Jenkins ver. 1.609.3'.

**Step 6 :** Click on Add build step and choose the optin of "SelecniumHQ htmlSuite Run"

The screenshot shows the Jenkins configuration interface for a job named "Helloworld Config". The "Build" section contains a command box with the following content:

```
javac HelloWorld.java  
java HelloWorld
```

An "Add build step" dropdown menu is open, showing several options. The "SeleniumHQ htmlSuite Run" option is highlighted with a blue selection bar.

**Step 7 :** Add the necessary details for the selenium test. Here the suiteFile is the TestSuite generated by using the Selenium IDE. Click on Save and execute a build. Now the post build will launch the selenium driver, and execute the html test.

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. The URL is [localhost:8080/jenkins/job/Helloworld/configure](http://localhost:8080/jenkins/job/Helloworld/configure). The configuration is divided into sections:

- Build**:
  - Execute Windows batch command**:  
Command: `javac HelloWorld.java  
java HelloWorld`
  - SeleniumHQ htmlSuite Run**:  
browser: `firefox`  
startURL: `https://www.google.ae`  
suiteFile: `E:\Apps\NewSample.html`  
resultFile: `E:\Jenkins\jobs\Helloworld\workspace\Reports\Results.html`  
other: [empty]
- Add build step**: A dropdown menu.
- Post-build Actions**:
  - Add post-build action**: A dropdown menu.
- Buttons**: **Save** (dark blue), **Apply** (light gray)

# 11. Jenkins – Notification

Jenkins comes with an out of box facility to add an email notification for a build project.

**Step 1 :** Configuring an SMTP server. Goto Manage Jenkins->Configure System. Go to the E-mail notification section and enter the required SMTP server and user email-suffix details.

The screenshot shows the Jenkins 'Configure System' page at [localhost:8080/jenkins/configure](http://localhost:8080/jenkins/configure). The 'E-mail Notification' section is highlighted, showing the following configuration:

- SMTP server: 192.168.0.100
- Default user e-mail suffix: @emirates.com
- Test configuration by sending test e-mail

At the bottom, there are 'Save' and 'Apply' buttons.

**Step 2 :** Configure the recipients in the Jenkins project - When you configure any Jenkins build project, right at the end is the ability to add recipients who would get email notifications for unstable or broken builds. Then click on the Save button.

Apart from the default, there are also notification plugin's available in the market. An example is the notification plugin from Tikal Knowledge which allows sending Job Status notifications in JSON and XML formats. This plugin enables end-points to be configured as shown below.

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. On the left, there's a sidebar with links like 'Changes', 'Workspace', 'Build Now', 'Delete Project', and 'Configure'. The main area is titled 'Job Notifications' under 'Notification Endpoints'. It shows a 'Delete' button and fields for 'Format' (set to 'JSON'), 'Protocol' (set to 'HTTP'), 'Event' (set to 'All Events'), 'URL' (set to 'http://dxbmem30/1xe4567h'), 'Timeout' (set to '30000'), and 'Log' (set to '0'). Below these fields are three checkboxes: 'This build is parameterized', 'Disable Build (No new builds will be executed until the project is re-enabled.)', and 'Execute concurrent builds if necessary'. At the bottom are 'Save' and 'Apply' buttons.

Here are the details of each option:

- "**Format**" : This is the notification payload format which can either be JSON or XML.
- "**Protocol**" : protocol to use for sending notification messages, HTTP, TCP or UDP.
- "**Event**" : The job events that trigger notifications: Job Started, Job Completed, Job Finalized or All Events (the default option).
- "**URL**" : URL to send notifications to. It takes the form of "<http://host>" for HTTP protocol, and "host:port" for TCP and UDP protocols.
- "**Timeout**" : Timeout in milliseconds for sending notification request, 30 seconds by default.

# 12. Jenkins – Reporting

As demonstrated in the earlier section, there are many reporting plugins available with the simplest one being the reports available for jUnit tests.

In the Post-build action for any job, you can define the reports to be created. After the builds are complete, the Test Results option will be available for further drill-down.

The screenshot shows the Jenkins configuration page for a job named 'Helloworld'. In the 'Build' section, there is a 'Execute Windows batch command' step with the command:  
javac HelloWorld.java  
java HelloWorld

A 'Post-build Actions' dropdown menu is open, showing various options:

- Execute Windows batch command
- SeleniumHQ htmlSuite Run
- Publish JUnit test result report (selected)
- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Build now
- Checkstyle
- Copy artifact
- Git Publisher
- E-mail Notification
- Set build status on GitHub commit
- Add post-build action

At the bottom of the configuration page, there are 'Save' and 'Apply' buttons.

# 13. Jenkins – Code Analysis

Jenkins has a host of Code Analysis plugin. The various plugins can be found at <https://wiki.jenkins-ci.org/display/JENKINS/Static+Code+Analysis+Plugins>

The screenshot shows the Jenkins Static Code Analysis Plug-ins page. On the left is a sidebar with links like Home, Mailing lists, Source code, Bugtracker, Security, etc. The main content area has a title 'Static Code Analysis Plug-ins' with a sub-section for 'analysis-core'. It shows the plugin ID, latest release (1.74), latest release date (Sep 07, 2015), required core dependencies (ant, token-macro, maven-plugin, matrix-project, dashboard-view), and maintainers (Ulli Hafner). Below this is a 'Usage' section with a line graph titled 'analysis-core - installations' showing the number of installations from October 2014 to September 2015. The graph shows a steady increase from approximately 25,000 to over 30,000 installations.

This plugin provides utilities for the static code analysis plugins. Jenkins can parse the results file from various Code Analysis tools such as CheckStyle, FindBugs, PMD etc. For each corresponding code analysis tool, a plugin in Jenkins needs to be installed.

Additionally the add-on plugin [Static Analysis Collector](#) is available that combines the individual results of these plugins into a single trend graph and view.

The plugins can provide information such as

- The total number of warnings in a job
- A showing of the new and fixed warnings of a build
- Trend Reports showing the number of warnings per build
- Overview of the found warnings per module, package, category, or type
- Detailed reports of the found warnings optionally filtered by severity (or new and fixed)

# 14. Jenkins – Distributed Builds

Sometimes many build machines are required if there are instances wherein there are a larger and heavier projects which get built on a regular basis. And running all of these builds on a central machine may not be the 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.

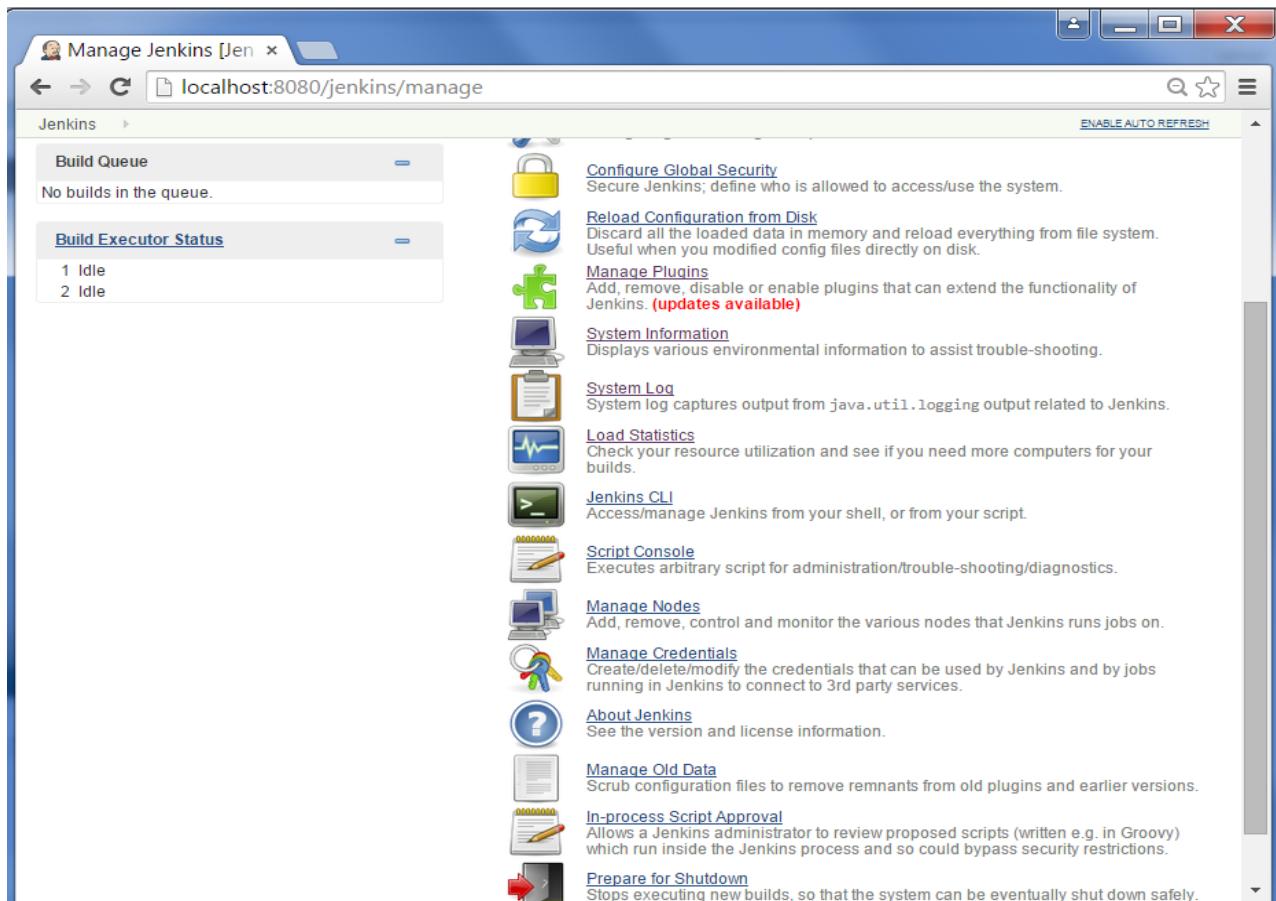
Sometimes you might also need several different environments to test your builds. In this case using a slave to represent each of your required environments is almost a must.

A slave is a computer that is set up to offload build projects from the master and once setup this distribution of tasks is fairly automatic. The exact delegation behavior depends on the configuration of each project; some projects may choose to "stick" to a particular machine for a build, while others may choose to roam freely between slaves.

Since each slave runs a separate program called a "slave agent" there is no need to install the full Jenkins (package or compiled binaries) on a slave. There are various ways to start slave agents, but in the end the slave agent and Jenkins master needs to establish a bi-directional communication link (for example a TCP/IP socket.) in order to operate.

To set up slaves/nodes in Jenkins follow the steps given below.

**Step 1 :** Go to the Manage Jenkins section and scroll down to the section of Manage Nodes.



**Step 2:** Click on New Node

The screenshot shows the Jenkins 'Nodes' page at [localhost:8080/jenkins/computer/](http://localhost:8080/jenkins/computer/). The page title is 'Nodes [Jenkins]'. On the left, there's a sidebar with links: 'Back to Dashboard', 'Manage Jenkins', 'New Node' (which is highlighted in blue), and 'Configure'. Below the sidebar are two collapsed sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area displays a table of nodes. The table has columns: S, Name (sorted by name), Architecture, Clock Difference, Free Disk Space, Free Swap Space, and Free. There is one row for the 'master' node, which is currently 'In sync'. The table shows the following data:

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free
	master	Windows 7 (x86)	In sync	229.89 GB	12.13 GB	
		Data obtained	3 min 11 sec	3 min 12 sec	3 min 12 sec	3 min 12 sec

A blue 'Refresh status' button is located at the bottom right of the table. At the bottom of the page, there are links for 'Help us localize this page', 'Page generated: Oct 12, 2015 9:23:44 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

**Step 3 :** Give a name for the node, choose the Dumb slave option and click on Ok.

The screenshot shows the Jenkins web interface at [localhost:8080/jenkins/computer/new](http://localhost:8080/jenkins/computer/new). The left sidebar has links for Back to Dashboard, Manage Jenkins, New Node (which is selected), and Configure. The main area shows a 'Node name' input field containing 'build\_slave'. A radio button labeled 'Dumb Slave' is selected, with a tooltip explaining it adds a plain, dumb slave to Jenkins. Below the input fields are sections for 'Build Queue' (empty) and 'Build Executor Status' (1 Idle, 2 Idle). At the bottom right are 'OK' and 'Cancel' buttons, and footer links for Help, REST API, and Jenkins version.

**Step 4 :** Enter the details of the node slave machine. In the below example, we are considering the slave machine to be a windows machine, hence the option of “Let Jenkins control this Windows slave as a Windows service” was chosen as the launch method. We also need to add the necessary details of the slave node such as the node name and the login credentials for the node machine. Click the Save button. The Labels for which the name is entered as “New\_Slave” is what can be used to configure jobs to use this slave machine.

The screenshot shows the Jenkins 'build\_slave' configuration page. The 'Name' field is set to 'build\_slave'. The 'Labels' field contains 'New\_Slave'. The 'Launch method' dropdown is set to 'Let Jenkins control this Windows slave as a Windows service'. A note below the launch method field states: 'This launch method relies on DCOM and is often associated with [subtle problems](#). Consider using [Launch slave agents using Java Web Start instead](#), which also permits installation as a Windows service but is generally considered more reliable.' The 'Administrator user name' is 'admin', and the 'Host' is 'dxbmem30'. The 'Run service as' dropdown is set to 'Use Local System User'. The 'Availability' section says 'Keep this slave on-line as much as possible'. At the bottom, there are 'Node Properties' sections for 'Environment variables' and 'Tool Locations', and a 'Save' button.

Once the above steps are completed, the new node machine will initially be in an offline state, but will come online if all the settings in the previous screen were entered correctly. One can at any time make the node slave machine as offline if required.

The screenshot shows the Jenkins 'Nodes' page at [localhost:8080/jenkins/computer/](http://localhost:8080/jenkins/computer/). The left sidebar includes links for Back to Dashboard, Manage Jenkins, New Node, and Configure. The main content displays a table of nodes:

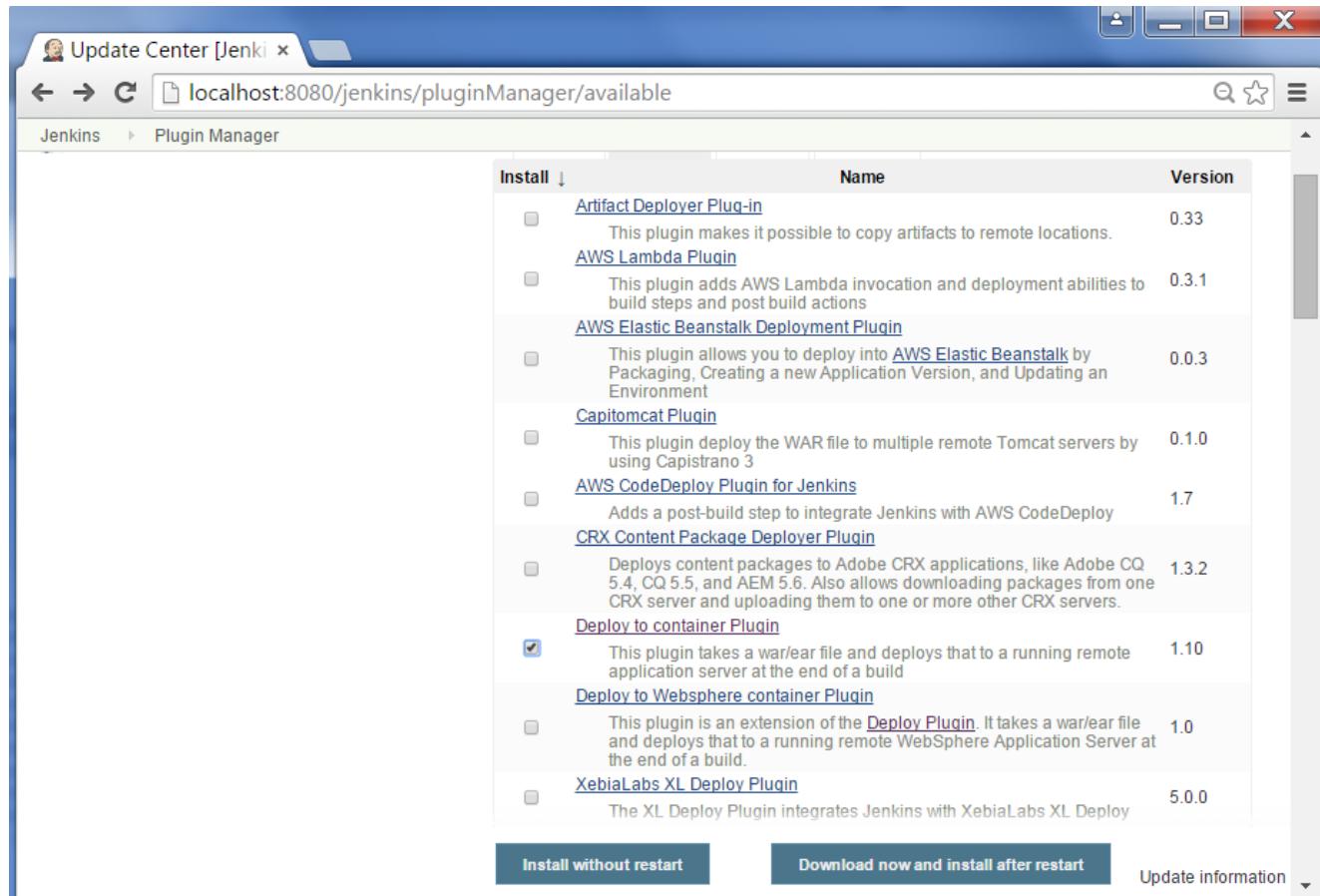
S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp S
	build_slave		N/A	N/A	N/A	
	master	Windows 7 (x86)	In sync	229.89 GB	12.13 GB	229.8
		Data obtained	3 ms	2 ms	1 ms	11 min

Below the table, there are sections for Build Queue (No builds in the queue) and Build Executor Status (master: 1 Idle, 2 Idle; build\_slave: (offline)). A 'Refresh status' button is located in the top right of the main content area. At the bottom, there are links for Help us localize this page, Page generated: Oct 12, 2015 9:31:43 PM, REST API, and Jenkins ver. 1.609.3.

# 15. Jenkins – Automated Deployment

There are many plugins available which can be used to transfer the build files after a successful build to the respective application/web server. One example is the “Deploy to container Plugin”. To use this follow the steps given below.

**Step 1:** Go to Manage Jenkins->Manage Plugins. Go to the Available section and find the plugin “Deploy to container Plugin” and install the plugin. Restart the Jenkins server.



The screenshot shows the Jenkins Plugin Manager interface. The URL in the address bar is `localhost:8080/jenkins/pluginManager/available`. The 'Available' tab is selected. A table lists various plugins, with the 'Deploy to container Plugin' checked for installation. The table columns are 'Install', 'Name', and 'Version'. The 'Deploy to container Plugin' is version 1.10 and is described as taking a war/ear file and deploying it to a running remote application server at the end of a build.

Install	Name	Version
<input type="checkbox"/>	<a href="#">Artifact Deployer Plug-in</a> This plugin makes it possible to copy artifacts to remote locations.	0.33
<input type="checkbox"/>	<a href="#">AWS Lambda Plugin</a> This plugin adds AWS Lambda invocation and deployment abilities to build steps and post build actions	0.3.1
<input type="checkbox"/>	<a href="#">AWS Elastic Beanstalk Deployment Plugin</a> This plugin allows you to deploy into <a href="#">AWS Elastic Beanstalk</a> by Packaging, Creating a new Application Version, and Updating an Environment	0.0.3
<input type="checkbox"/>	<a href="#">Capitomcat Plugin</a> This plugin deploy the WAR file to multiple remote Tomcat servers by using Capistrano 3	0.1.0
<input type="checkbox"/>	<a href="#">AWS CodeDeploy Plugin for Jenkins</a> Adds a post-build step to integrate Jenkins with AWS CodeDeploy	1.7
<input type="checkbox"/>	<a href="#">CRX Content Package Deployer Plugin</a> Deploys content packages to Adobe CRX applications, like Adobe CQ 5.4, CQ 5.5, and AEM 5.6. Also allows downloading packages from one CRX server and uploading them to one or more other CRX servers.	1.3.2
<input checked="" type="checkbox"/>	<a href="#">Deploy to container Plugin</a> This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build	1.10
<input type="checkbox"/>	<a href="#">Deploy to Websphere container Plugin</a> This plugin is an extension of the <a href="#">Deploy Plugin</a> . It takes a war/ear file and deploys that to a running remote WebSphere Application Server at the end of a build.	1.0
<input type="checkbox"/>	<a href="#">Xebialabs XL Deploy Plugin</a> The XL Deploy Plugin integrates Jenkins with Xebialabs XL Deploy	5.0.0

**Buttons at the bottom:**

- Install without restart
- Download now and install after restart
- Update information

This plugin takes a war/ear file and deploys that to a running remote application server at the end of a build.

Tomcat 4.x/5.x/6.x/7.x

JBoss 3.x/4.x

Glassfish 2.x/3.x

**Step 2 :** Go to your Build project and click the Configure option. Choose the option "Deploy war/ear to a container"

The screenshot shows the Jenkins configuration page for the 'Helloworld' job. The 'Configure' tab is selected. A context menu is open over the 'Post-build Actions' section, with the 'Deploy war/ear to a container' option highlighted.

Configuration details visible in the main pane:

- Invoke Ant
- Ant Version: NewHome
- Targets: (empty)
- Advanced...
- Delete

Post-build actions listed in the context menu (with 'Deploy war/ear to a container' highlighted):

- Publish FindBugs analysis results
- Publish combined analysis results
- Aggregate downstream test results
- Archive the artifacts
- Build other projects
- Publish JUnit test result report
- Publish Javadoc
- Publish Selenium Report
- Record fingerprints of files to track usage
- Git Publisher
- Deploy war/ear to a container** (highlighted)
- E-mail Notification
- Set build status on GitHub commit

Buttons at the bottom:

- Add post-build action ▾
- Save
- Apply

Page footer:

localhost:8080/jenkins/job/Helloworld/configure# Page generated: Oct 12, 2015 9:43:02 PM REST API Jenkins ver. 1.609.3

**Step 3 :** In the Deploy war/ear to a container section, enter the required details of the server on which the files need to be deployed and click on the Save button. These steps will now ensure that the necessary files get deployed to the necessary container after a successful build.

The screenshot shows the Jenkins job configuration page for 'Demo'. Under 'Post-build Actions', there is a 'Deploy war/ear to a container' step. The 'WAR/EAR files' field contains 'target/spare-1.0.war'. The 'Context path' field contains 'spare'. Under 'Containers', there is a 'Tomcat 7.x' configuration with 'Manager user name' set to 'S112233', 'Manager password' masked as '.....', and 'Tomcat URL' set to 'http://dxbmem10:8080'. There are 'Delete' buttons for both the 'Invoke Ant' and 'Deploy war/ear to a container' sections. At the bottom, there are 'Save' and 'Apply' buttons.

# 16. Jenkins – Metrics and Trends

There are various plugins which are available in Jenkins to showcase metrics for builds which are carried out over a period of time. These metrics are useful to understand your builds and how frequently they fail/pass over time. As an example, let's look at the 'Build History Metrics plugin'.

This plugin calculates the following metrics for all of the builds once installed

- Mean Time To Failure (MTTF)
- Mean Time To Recovery (MTTR)
- Standard Deviation of Build Times

**Step 1 :** Go to the Jenkins dashboard and click on Manage Jenkins

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). The main header says "Jenkins". On the left, there's a sidebar with links: "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". Below that are two sections: "Build Queue" (No builds in the queue) and "Build Executor Status" (1 Idle, 2 Idle). The central area has a "Welcome to Jenkins!" message and a call to action: "Please [create new jobs](#) to get started." At the bottom, there are links for "Help us localize this page", "Page generated: Oct 10, 2015 12:40:44 AM", "REST API", and "Jenkins ver. 1.609.3".

**Step 2 :** Go to the Manage Plugins option.

The screenshot shows the Jenkins Manage Jenkins page. On the left, there's a sidebar with links like New Item, People, Build History, Manage Jenkins (which is selected), and Credentials. Below that are sections for Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle). The main content area is titled "Manage Jenkins". It features a warning message about URL decoding and security, with "Manage Plugins" highlighted in red. A list of management options follows:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. **(updates available)**
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from java.util.logging output related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Manage Credentials**: Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
- About Jenkins**: See the version and license information.

**Step 3 :** Go to the Available tab and search for the plugin 'Build History Metrics plugin' and choose to 'install without restart'.

The screenshot shows the Jenkins Plugin Manager interface. The browser address bar displays 'localhost:8080/jenkins/pluginManager/available'. The main title is 'Jenkins' with a subtitle 'Plugin Manager'. On the left, there are links for 'Back to Dashboard' and 'Manage Jenkins'. The top navigation bar has tabs: 'Updates', 'Available' (which is selected), 'Installed', and 'Advanced'. A search bar at the top right contains the text 'build-history-metrics-plugin'. Below the tabs is a table with columns 'Name' and 'Version'. One row in the table is highlighted, showing 'Build History Metrics plugin' and '1.2'. A note below the table states 'Provides build metrics that encompass the history of all the runs'. At the bottom of the table are two buttons: 'Install without restart' (highlighted in blue) and 'Download now and install after restart'. To the right of these buttons is the text 'Update information obtained: 2 mi'. At the very bottom of the page, there are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 3:53:24 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

**Step 4:** The following screen shows up to confirm successful installation of the plugin. Restart the Jenkins instance.

The screenshot shows the Jenkins Update Center interface. The title bar says "Update Center [Jenki x]" and the address bar shows "localhost:8080/jenkins/updateCenter/". The main header is "Jenkins" with a search bar and an "ENABLE AUTO REFRESH" link. On the left, there's a sidebar with links: "Back to Dashboard", "Manage Jenkins", and "Manage Plugins". The main content area has a title "Installing Plugins/Upgrades". Under "Preparation", it lists: "Checking internet connectivity", "Checking update center connectivity", and "Success". Below that, it shows "Build History Metrics plugin" with a blue circular icon labeled "Success". At the bottom, there are two green arrows pointing right with instructions: "Go back to the top page (you can start using the installed plugins right away)" and "Restart Jenkins when installation is complete and no jobs are running". At the very bottom, there's a link "Help us localize this page" and footer text "Page generated: Oct 24, 2015 3:53:57 PM REST API Jenkins ver. 1.609.3".

When you go to your Job page, you will see a table with the calculated metrics. Metric's are shown for the last 7 days, last 30 days and all time.

The screenshot shows the Jenkins interface for the 'Helloworld' project. On the left, there's a sidebar with links: Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, and Configure. Below that is the 'Build History' section, which lists builds from #1 to #12 with their respective dates. At the bottom of this section are links for 'RSS for all' and 'RSS for failures'. To the right of the sidebar is a large table displaying performance metrics: MTTR, MTTF, and Standard Deviation, each with three rows for Last 7 Days, Last 30 Days, and All Time. Below the table is a section titled 'Permalinks' containing a bulleted list of links to various build logs. The top right of the page has a search bar, an 'ENABLE AUTO REFRESH' link, and buttons for 'add description' and 'Disable Project'.

	Last 7 Days	Last 30 Days	All Time
MTTR	0 ms	23 hr	23 hr
MTTF	0 ms	2 days 4 hr	2 days 4 hr
Standard Deviation	0 ms	52 sec	52 sec

To see overall trends in Jenkins, there are plugins available to gather information from within the builds and Jenkins and display them in a graphical format. One example of such a plugin is the 'Hudson global-build-stats plugin'. So let's go through the steps for this.

**Step 1 :** Go to the Jenkins dashboard and click on Manage Jenkins

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). The main header says "Dashboard [Jenkins]". On the left sidebar, there are links for "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". The "Manage Jenkins" link is currently selected. The central area has a "Welcome to Jenkins!" message with a call to action: "Please [create new jobs](#) to get started." Below this, there are two expandable sections: "Build Queue" (which shows "No builds in the queue.") and "Build Executor Status" (which shows "1 Idle" and "2 Idle"). At the bottom, there are links for "Help us localize this page", "Page generated: Oct 10, 2015 12:40:44 AM", "REST API", and "Jenkins ver. 1.609.3".

**Step 2 : Go to the Manage Plugins option**

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links like New Item, People, Build History, Manage Jenkins (which is selected), and Credentials. The main area is titled "Manage Jenkins". It displays a warning message about URL encoding and security, with "Manage Plugins" highlighted in red. Below the warning, there's a list of management options with icons:

- Configure System
- Configure Global Security
- Reload Configuration from Disk
- Manage Plugins
- System Information
- System Log
- Load Statistics
- Jenkins CLI
- Script Console
- Manage Nodes
- Manage Credentials
- About Jenkins

**Step 3 :** Go to the Available tab and search for the plugin 'Hudson global-build-stats plugin' and choose to 'install without restart'.

The screenshot shows the Jenkins Plugin Manager interface. At the top, there's a navigation bar with links for 'Back to Dashboard' and 'Manage Jenkins'. Below it, a search bar contains the text 'global-build-stats'. The main area has tabs for 'Updates', 'Available' (which is selected), 'Installed', and 'Advanced'. A table lists a single plugin:

Install ↓	Name	Version
<input checked="" type="checkbox"/>	Hudson global-build-stats plugin Global build stats plugin will allow to gather and display global build result statistics. It is a useful tool allowing to display global hudson build trend over time.	1.3

At the bottom of the table are three buttons: 'Install without restart' (highlighted in blue), 'Download now and install after restart', and 'Update inform'.

**Step 4:** The following screen shows up to confirm successful installation of the plugin. Restart the Jenkins instance.

The screenshot shows a web browser window for the Jenkins Update Center. The URL is `localhost:8080/jenkins/updateCenter/`. The main title is "Installing Plugins/Upgrades". On the left sidebar, there are links: "Back to Dashboard", "Manage Jenkins", and "Manage Plugins". The central content area has a section titled "Preparation" with a bulleted list: "Checking internet connectivity", "Checking update center connectivity", and "Success". Below this, it says "Hudson global-build-stats plugin" followed by a blue circular icon with a white checkmark and the word "Success". At the bottom, there are two green arrows pointing right: one pointing to "Go back to the top page" (you can start using the installed plugins right away) and another pointing to "Restart Jenkins when installation is complete and no jobs are running". The footer includes a link to "Help us localize this page", a timestamp "Page generated: Oct 24, 2015 4:01:04 PM", and "REST API Jenkins ver. 1.609.3".

To see the Global statistics, please follow the Step 5 through 8.

**Step 5:** Go to the Jenkins dashboard and click on Manage Jenkins. In the Manage Jenkins screen, scroll down and now you will now see an option called 'Global Build Stats'. Click on this link.

The screenshot shows the Jenkins 'Manage Jenkins' interface at the URL [localhost:8080/jenkins/manage](http://localhost:8080/jenkins/manage). The page lists several management functions with corresponding icons:

- Manage Plugins**: Adds, removes, disables or enables plugins that can extend the functionality of Jenkins. (updates available)
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from `java.util.logging` related to Jenkins.
- Load Statistics**: Checks resource utilization to determine if more computers are needed for builds.
- Jenkins CLI**: Accesses Jenkins from the shell or script.
- Script Console**: Executes arbitrary scripts for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Adds, removes, controls, and monitors various nodes Jenkins runs jobs on.
- Manage Credentials**: Creates/deletes/modifies credentials used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
- About Jenkins**: Sees version and license information.
- Manage Old Data**: Scrubs configuration files to remove remnants from old plugins and earlier versions.
- Global Build Stats**: Displays stats about daily build results.
- In-process Script Approval**: Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.
- Prepare for Shutdown**: Stops executing new builds so the system can be safely shut down.

At the bottom of the page, there are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 4:02:40 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

**Step 6 :** Click on the button 'Initialize stats'. What this does is that it gathers all the existing records for builds which have already been carried out and charts can be created based on these results.

The screenshot shows the Jenkins Global Build Stats page at the URL [localhost:8080/jenkins/plugin/global-build-stats/](http://localhost:8080/jenkins/plugin/global-build-stats/). The page title is "Global Build Stats". On the left sidebar, there are links: "Back to Dashboard", "Create new chart", "Manage retention strategies", and "Data Initialization". The main content area has a section titled "Statistics" with the message "No chart configured for the moment ... [Create a new chart configuration](#)". Below this is a section titled "Build Results retention strategies" with three checkboxes: "Automatically discard results older than 365 days", "Do not keep build results when they are discarded", and "Keep existing job results only". A "Update retention strategies" button is present. At the bottom, there is a section titled "Data Initialization" with the instruction "Click button below to initialize build statistics. Job results read will be merged with already recorded job results." and a large "Initialize stats" button.

**Step 7 :** Once the data has been initialized, it's time to create a new chart. Click on the 'Create new chart' link.

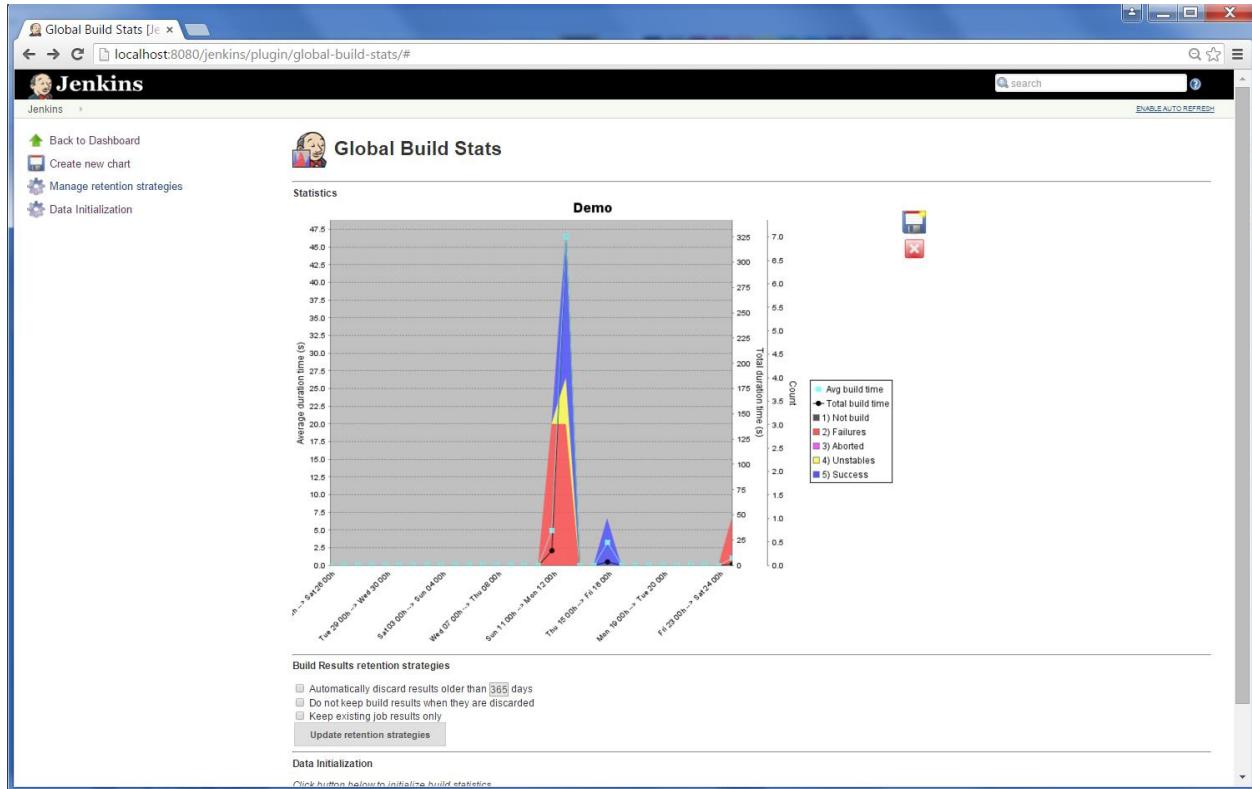
The screenshot shows the Jenkins Global Build Stats page at [localhost:8080/jenkins/plugin/global-build-stats/#Initialize](http://localhost:8080/jenkins/plugin/global-build-stats/#Initialize). The left sidebar has links for Back to Dashboard, Create new chart, Manage retention strategies, and Data Initialization. The main content area is titled "Global Build Stats". It shows a "Statistics" section with a message: "No chart configured for the moment ... [Create a new chart configuration](#)". Below it is a "Build Results retention strategies" section with three checkboxes: "Automatically discard results older than [365] days", "Do not keep build results when they are discarded", and "Keep existing job results only". A "Update retention strategies" button is present. At the bottom is a "Data Initialization" section with instructions: "Click button below to initialize build statistics. Job results read will be merged with already recorded job results." A green success message "Data successfully initialized!" is displayed above a "Initialize stats" button. The footer includes links for "Help us localize this page", "Page generated: Oct 24, 2015 4:03:17 PM", "REST API", and "Jenkins ver. 1.609.3".

**Step 8 :** A pop-up will come to enter relevant information for the new chart details. Enter the following mandatory information

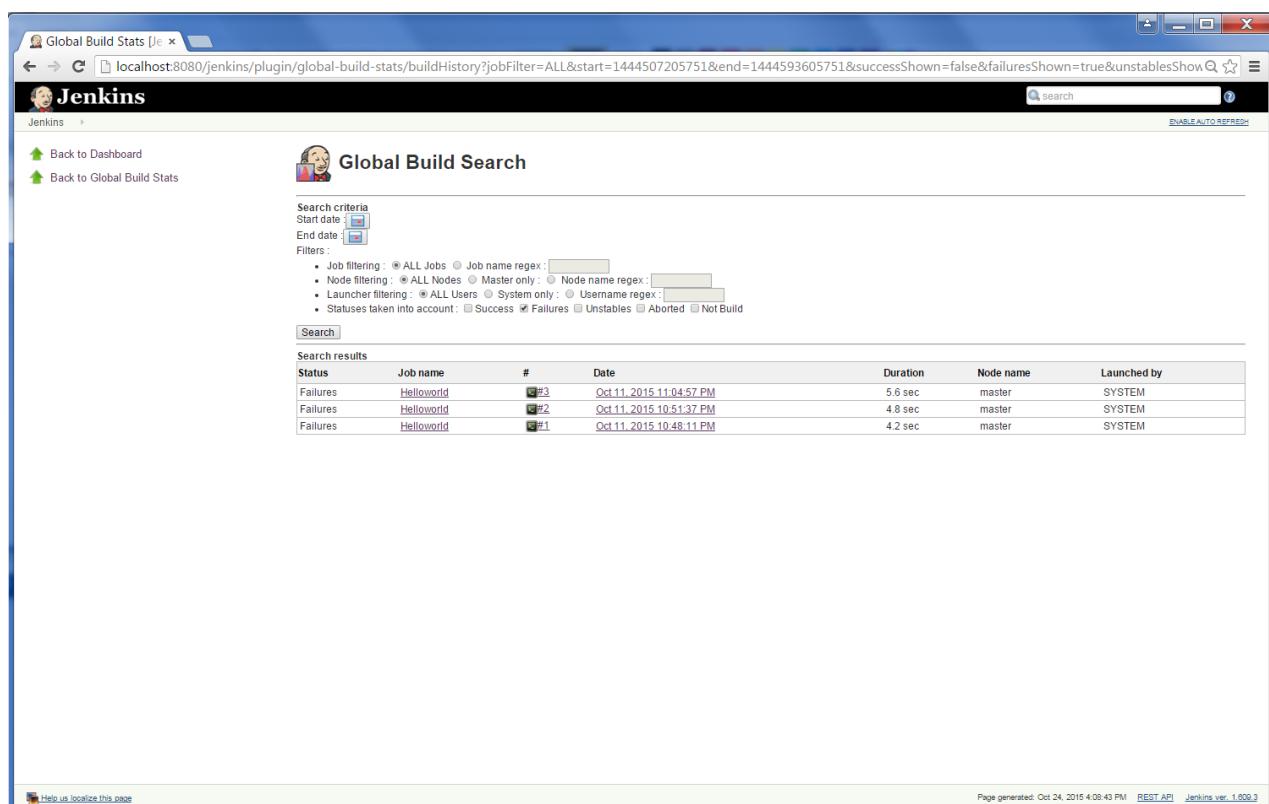
- Title – Any title information, for this example is given as 'Demo'
- Chart Width – 800
- Chart Height – 600
- Chart time scale – Daily
- Chart time length – 30 days

The rest of the information can remain as it is. Once the information is entered, click on Create New chart.

You will now see the chart which displays the trends of the builds over time.



If you click on any section within the chart, it will give you a drill down of the details of the job and their builds.



# 17.Jenkins – Server Maintenance

The following are some of the basic activities you will carry out, some of which are best practices for Jenkins server maintenance

## URL Options

The following commands when appended to the Jenkins instance URL will carry out the relevant actions on the Jenkins instance.

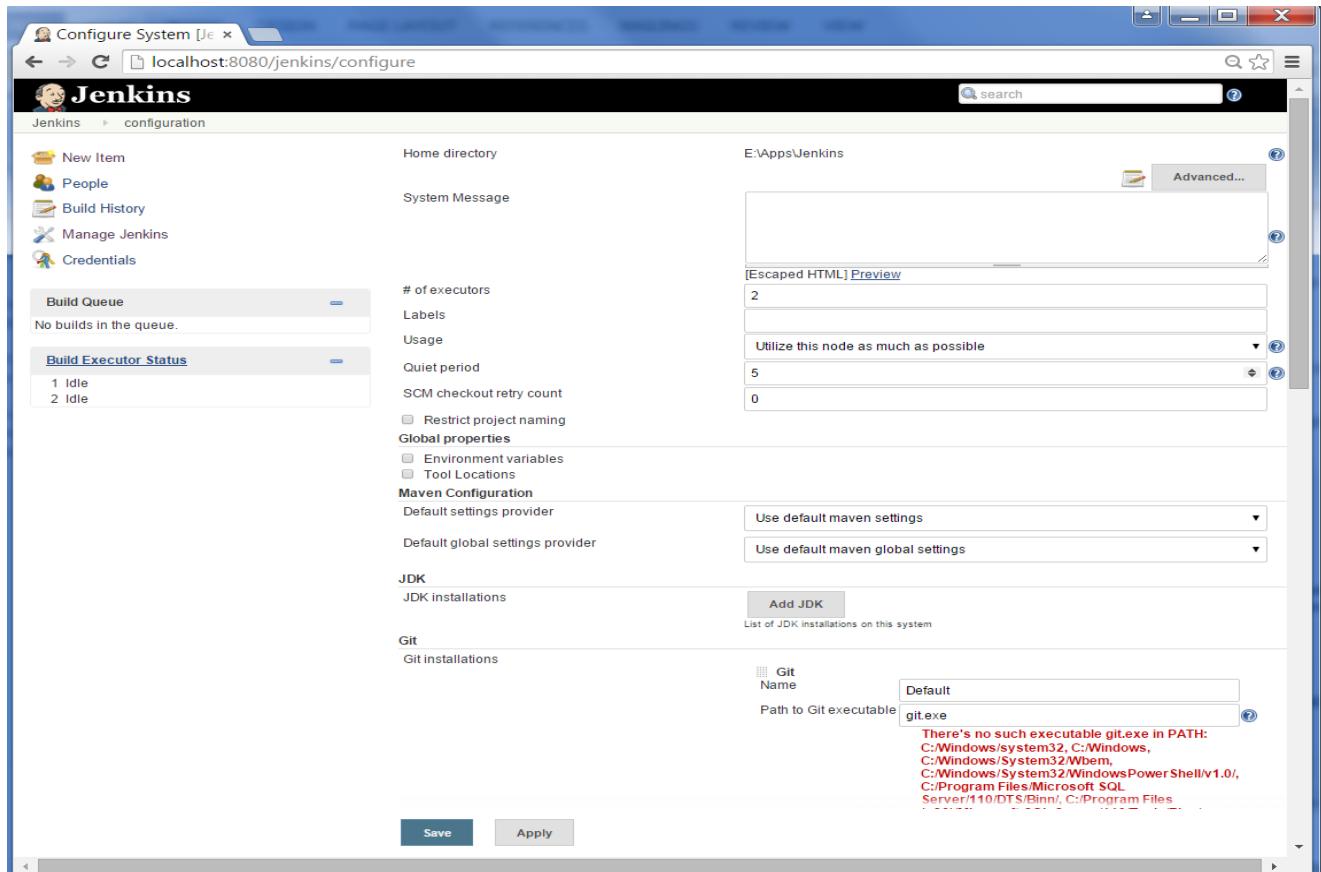
<http://localhost:8080/jenkins/exit> - shutdown jenkins

<http://localhost:8080/jenkins/restart> - restart jenkins

<http://localhost:8080/jenkins/reload> - to reload the configuration

## Backup Jenkins Home

The Jenkins Home directory is nothing but the location on your drive where Jenkins stores all information for the jobs, builds etc. The location of your home directory can be seen when you click on Manage Jenkins->Configure system.



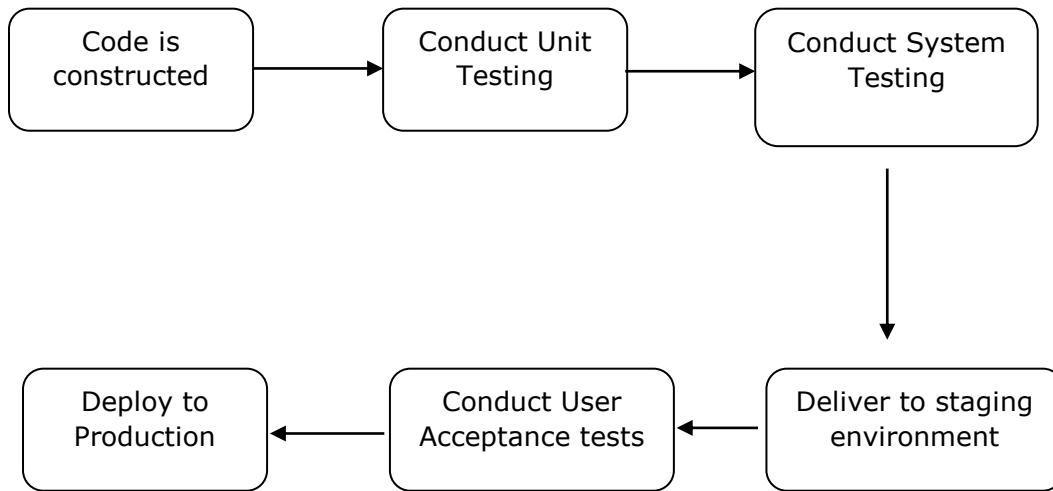
The screenshot shows the Jenkins 'Configure System' page. The 'Home directory' field is set to 'E:\Apps\Jenkins'. Other settings visible include the number of executors (2), quiet period (5), and Maven configuration options. A warning message at the bottom right indicates that 'git.exe' is not found in the PATH.

Set up Jenkins on the partition that has the most free disk-space – Since Jenkins would be taking source code for the various jobs defined and doing continuous builds, always ensure that Jenkins is setup on a drive that has enough hard disk space. If you hard disk runs out of space, then all builds on the Jenkins instance will start failing.

Another best practice is to write cron jobs or maintenance tasks that can carry out clean-up operations to avoid the disk where Jenkins is setup from becoming full.

# 18. Jenkins – Continuous Deployment

Jenkins provides good support for providing continuous deployment and delivery. If you look at the flow of any software development through deployment, it will be as shown below.



The main part of Continuous deployment is to ensure that the entire process which is shown above is automated. Jenkins achieves all of this via various plugins, one of them being the "Deploy to container Plugin" which was seen in the earlier lessons.

The screenshot shows the Jenkins configuration page for a job named "Helloworld". The main section displays an "Invoke Ant" step with an "Ant Version" set to "NewHome" and a "Targets" field containing an empty list. Below this is a list of available post-build actions, with "Deploy war/ear to a container" highlighted in blue. Other actions listed include Publish FindBugs analysis results, Publish combined analysis results, Aggregate downstream test results, Archive the artifacts, Build other projects, Publish JUnit test result report, Publish Javadoc, Publish Selenium Report, Record fingerprints of files to track usage, Git Publisher, E-mail Notification, and Set build status on GitHub commit. At the bottom of the configuration page are "Save" and "Apply" buttons.

There are plugins available which can actually give you a graphical representation of the Continuous deployment process. But first lets create another project in Jenkins, so that we can see best how this works.

Let's create a simple project which emulates the QA stage, and does a test of the Helloworld application.

**Step 1 :** Go to the Jenkins dashboard and click on New Item. Choose a 'Freestyle project' and enter the project name as 'QA'. Click on the Ok button to create the project.

The screenshot shows the Jenkins 'New Item' dialog box. In the top left, there's a 'New Item [Jenkins]' tab and a back/forward navigation bar. The main title is 'Jenkins'. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below the sidebar are two collapsed sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The right side of the dialog has a form for creating a new item. It includes a 'Item name' field with 'QA' typed in, a radio button for 'Freestyle project' (which is selected), and a detailed description of what it means. There are also options for 'Maven project', 'External Job', 'Multi-configuration project', and 'Copy existing Item'. At the bottom right is an 'OK' button.

**Step 2 :** In this example, we are keeping it simple and just using this project to execute a test program for the Helloworld application.

The screenshot shows the Jenkins configuration page for a job named "QA Config". The URL is [localhost:8080/jenkins/job/QA/configure](http://localhost:8080/jenkins/job/QA/configure). The configuration is set up as follows:

- Build Triggers:** No triggers are selected.
- Build Environment:** No environment variables are listed.
- Build:**
  - Execute Windows batch command:** This step is selected.
  - Command:** The command entered is:

```
javac HelloWorldTest.java  
java HelloWorldTest
```
- Post-build Actions:** No actions are selected.

At the bottom, there are "Save" and "Apply" buttons, along with links for localization and page information.

So our project QA is now setup. You can do a build to see if it builds properly.

The screenshot shows the Jenkins Project QA page for the QA job. The left sidebar includes links for Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, and Configure. The main content area is titled "Project QA" and contains sections for "Build History" (with four recent builds listed), "MTTR" (Mean Time To Repair) metrics, "MTTF" (Mean Time To Failure) metrics, and "Standard Deviation" metrics. A "Recent Changes" link is also present. On the right, there are buttons for "add description" and "Disable Project". Below the metrics is a "Permalinks" section with links to the last build and the last stable build.

**Step 3 :** Now go to your Helloworld project and click on the Configure option

The screenshot shows the Jenkins Dashboard. The left sidebar lists New Item, People, Build History, Manage Jenkins, and Credentials. The main area displays a table of projects, with the Helloworld project selected. The table columns are S (Status), W (Last Warning), Name, Last Success, Last Failure, and Last Duration. The Helloworld project row shows a blue circle icon, a cloud and rain icon, the name "Helloworld", the last success at "12 days - #15", the last failure at "12 days - #14", and a duration of "6.6 sec". Below the table are links for Changes, Workspace, Build Now, Delete Project, and Configure. The "Configure" link is highlighted with a blue box. At the bottom of the dashboard, there are sections for Build Queue (empty) and Build Executor Status (1 Idle).

**Step 4 :** In the project configuration, choose the 'Add post-build action' and choose 'Build other projects'

The screenshot shows the Jenkins configuration page for a job named 'Helloworld'. In the 'Post-build Actions' section, a dropdown menu is open, showing various options like 'Aggregate downstream test results', 'Archive the artifacts', and 'Build other projects'. The 'Build other projects' option is highlighted with a blue selection bar. Below this menu, there is a detailed description of the 'Build other projects' action, which includes a 'Basedir' field set to 'myproject/target/test-reports/\*.xml'. There are also sections for 'Test threshold' and 'Email notifications'. At the bottom of the configuration page are 'Save' and 'Apply' buttons.

**Step 5 :** In the 'Project to build' section, enter QA as the project name to build. You can leave the option as default of 'Trigger only if build is stable'. Click on the Save button.

This screenshot shows the Jenkins configuration page for the 'Helloworld' job. In the 'Build' section, there is a 'Execute Windows batch command' step with the command: E:\javac HelloWorld.java & java HelloWorld. In the 'Post-build Actions' section, there is a 'Build other projects' step. The 'Projects to build' field contains 'QA'. Below this field are three radio button options: 'Trigger only if build is stable' (selected), 'Trigger even if the build is unstable', and 'Trigger even if the build fails'. At the bottom of the configuration page are 'Save' and 'Apply' buttons.

**Step 6 :** Build the Helloworld project. Now if you see the Console output, you will also see that after the Helloworld project is successfully built, the build of the QA project will also happen.

The screenshot shows the Jenkins interface for the 'Helloworld' job, specifically build #14. The left sidebar lists options like 'Back to Project', 'Status', 'Changes', 'Console Output' (which is currently selected), 'View as plain text', 'Edit Build Information', 'Delete Build', and 'Previous Build'. The main content area is titled 'Console Output' and displays the following log:

```
Started by user anonymous
Building in workspace E:\Jenkins\jobs\Helloworld\workspace
[workspace] $ cmd /c call E:\Apps\tomcat7\temp\hudson3970974123969609693.bat

E:\Jenkins\jobs\Helloworld\workspace>E:\
'E:' is not recognized as an internal or external command,
operable program or batch file.

E:\Jenkins\jobs\Helloworld\workspace>javac HelloWorld.java

E:\Jenkins\jobs\Helloworld\workspace>java HelloWorld
Hello World

E:\Jenkins\jobs\Helloworld\workspace>exit 0
Warning: you have no plugins providing access control for builds, so falling back to legacy behavior of
permitting any downstream builds to be triggered
Triggering a new build of QA
Finished: SUCCESS
```

At the bottom of the page, there are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 5:22:02 PM', 'REST API', and 'Jenkins ver. 1.800.3'.

**Step 7 :** Let now install the Delivery pipeline plugin. Go to Manage Jenkins->Manage Plugin's. In the available tab, search for 'Delivery Pipeline Plugin'. Click On Install without Restart. Once done, restart the Jenkins instance.

Install ↓	Name	Version
<input type="checkbox"/>	<a href="#">ontrack Jenkins plug-in</a>	2.15.0
<input type="checkbox"/>	<a href="#">Fail The Build Plugin</a>	1.0
<input type="checkbox"/>	<a href="#">Runscope plugin</a>	1.44
<input type="checkbox"/>	<a href="#">Build Graph View Plugin</a>	1.1.1
<input type="checkbox"/>	<a href="#">Deployment Sphere</a>	0.1.105
<input type="checkbox"/>	<a href="#">CloudBees Docker Hub Notification</a>	1.0.2
<input type="checkbox"/>	<a href="#">This plugin provides integration between Jenkins and Docker Hub, utilizing a Docker Hub hook to trigger one (or more) Jenkins job(s). This allows you to implement continuous delivery pipelines based on Docker in Jenkins.</a>	0.17.0
<input checked="" type="checkbox"/>	<a href="#">Seed Jenkins plug-in</a>	0.9.7
<input checked="" type="checkbox"/>	<a href="#">The Seed project aims to help automating the generation and management of pipelines for branches of a project in Jenkins.</a>	0.9.7
<input checked="" type="checkbox"/>	<a href="#">Delivery Pipeline Plugin</a>	0.9.7

Update information obtained: 1 hr 36 min ago

[Install without restart](#)   [Download now and install after restart](#)

**Step 8 :** To see the Delivery pipeline in action, in the Jenkins Dashboard, click on the + symbol in the Tab next to the 'All' Tab.

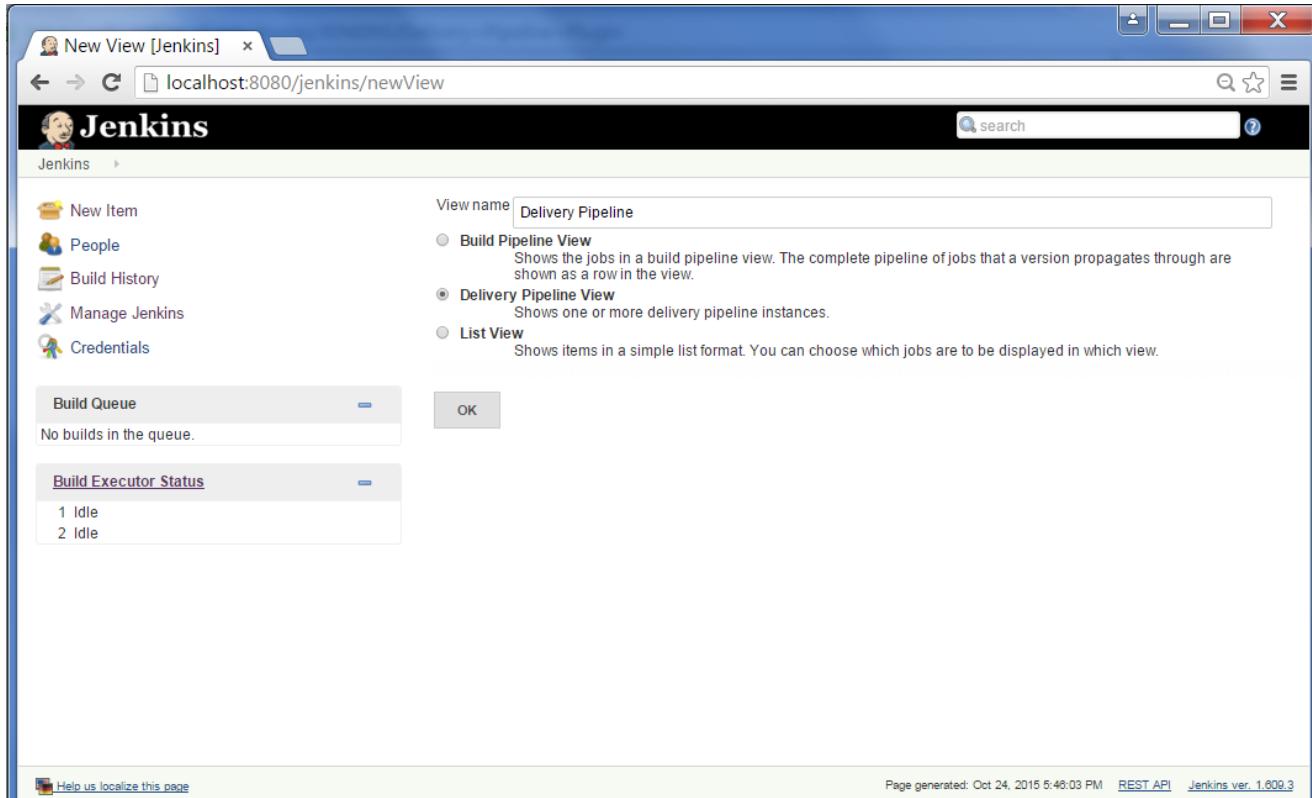
All	+	S	W	Name ↓	Last Success	Last Failure	Last Duration
		Helloworld	25 min - #14	1 hr 49 min - #12	1.4 sec		
		QA	25 min - #5	28 min - #2	1.4 sec		

Icon: [S](#) [M](#) [L](#)

Legend [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Page generated: Oct 24, 2015 5:46:52 PM REST API Jenkins ver. 1.609.3

**Step 9 :** Enter any name for the View name and choose the option 'Delivery Pipeline View'.



**Step 10 :** In the next screen, you can leave the default options. One can change the following settings:

- Ensure the option 'Show static analysis results' is checked.
- Ensure the option 'Show total build time' is checked.
- For the Initial job – Enter the Helloworld project as the first job which should build.
- Enter any name for the Pipeline
- Click the OK button.

Edit View [Jenkins] x localhost:8080/jenkins/view/Delivery%20Pipeline/configure

## Jenkins

Jenkins > Delivery Pipeline >

New Item

People

Build History

Edit View

Delete View

View Fullscreen

Manage Jenkins

Credentials

Build Queue

No builds in the queue.

Build Executor Status

1 Idle  
2 Idle

Name: Delivery Pipeline

View settings

Number of pipeline instances per pipeline: 3

Display aggregated pipeline for each pipeline:

Number of columns: 1

Sorting: None

Update interval: 2

Enable start of new pipeline build:

Enable manual triggers:

Enable rebuild:

Show avatars:

Show commit messages:

Show job description:

Show job promotions:

Show junit results:

Show static analysis results:

Show total build time:

URL for custom CSS file:

URL for custom CSS file (fullscreen):

Pipelines

Components

Name: Firstjob

Initial Job: Helloworld

Final Job (optional):

Add

Regular Expression

Add

OK Apply

Help us localize this page Page generated: Oct 24, 2015 5:47:28 PM REST API Jenkins ver. 1.609.3

You will now see a great view of the entire delivery pipeline and you will be able to see the status of each project in the entire pipeline.

The screenshot shows the Jenkins interface for a 'Delivery Pipeline'. On the left, there's a sidebar with various Jenkins management links like 'New Item', 'People', 'Build History', etc. The main area is titled 'Delivery Pipeline' and shows three sequential build steps:

- Firstjob**:
  - #14 triggered by user anonymous started 29 minutes ago
  - Total build time: 2 sec
  - Build flow: Helloworld (green bar, 29 minutes ago, 1 sec) → QA (green bar, 28 minutes ago, 1 sec)
- #13 triggered by user anonymous started an hour ago**
  - Total build time: 3 sec
  - Build flow: Helloworld (green bar, an hour ago, 3 sec) → QA (green bar)
- #12 triggered by user anonymous started 2 hours ago**
  - Total build time: 1 sec
  - Build flow: Helloworld (red bar, 2 hours ago, 1 sec) → QA (white bar)

At the bottom, there are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 5:49:35 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

Another famous plugin is the **build pipeline plugin**. Let's take a look at this.

**Step 1** : Go to Manage Jenkins->Manage Plugin's. In the available tab, search for 'Build Pipeline Plugin'. Click On Install without Restart. Once done, restart the Jenkins instance.

The screenshot shows the Jenkins Plugin Manager interface. The URL in the browser is `localhost:8080/jenkins/pluginManager/available`. The 'Available' tab is selected. A search bar at the top right contains the text 'Build pipeline'. Below the tabs, there is a table with columns: 'Install ↓', 'Name', and 'Version'. The 'Build Pipeline Plugin' is listed with a checked checkbox in the 'Install' column. Its details are shown: 'This plugin provides a Build Pipeline View of upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.' The version is 1.4.8. Other plugins listed include 'Fail The Build Plugin' (version 1.0), 'Runscope plugin' (version 1.44), 'Build Graph View Plugin' (version 1.1.1), and 'Delivery Pipeline Plugin' (version 0.9.7). At the bottom, there are two buttons: 'Install without restart' and 'Download now and install after restart'. The status 'Update info' is shown to the right of the buttons. The footer includes links for localization and page information.

Install ↓	Name	Version
<input checked="" type="checkbox"/>	<a href="#">Build Pipeline Plugin</a>	1.4.8
<input type="checkbox"/>	<a href="#">Fail The Build Plugin</a>	1.0
<input type="checkbox"/>	<a href="#">Runscope plugin</a>	1.44
<input type="checkbox"/>	<a href="#">Build Graph View Plugin</a>	1.1.1
<input type="checkbox"/>	<a href="#">Delivery Pipeline Plugin</a>	0.9.7

[Install without restart](#)    [Download now and install after restart](#)

Update info

[Help us localize this page](#)    Page generated: Oct 24, 2015 4:46:09 PM    [REST API](#)    Jenkins ver. 1.609.3

**Step 2 :** To see the Build pipeline in action, in the Jenkins Dashboard, click on the + symbol in the Tab next to the 'All' Tab.

The screenshot shows the Jenkins dashboard at [localhost:8080/jenkins/](http://localhost:8080/jenkins/). On the left, there's a sidebar with links for 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. The main area has a table titled 'All' showing two builds: 'Helloworld' and 'QA'. The 'Helloworld' build was last successful 25 min - #14 ago, failed 1 hr 49 min - #12 ago, and took 1.4 sec. The 'QA' build was last successful 25 min - #5 ago, failed 28 min - #2 ago, and took 1.4 sec. Below the table are links for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). At the bottom, there's a link to 'Help us localize this page' and footer text: 'Page generated: Oct 24, 2015 5:46:52 PM REST API Jenkins ver. 1.609.3'.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		<a href="#">Helloworld</a>	25 min - #14	1 hr 49 min - #12	1.4 sec
		<a href="#">QA</a>	25 min - #5	28 min - #2	1.4 sec

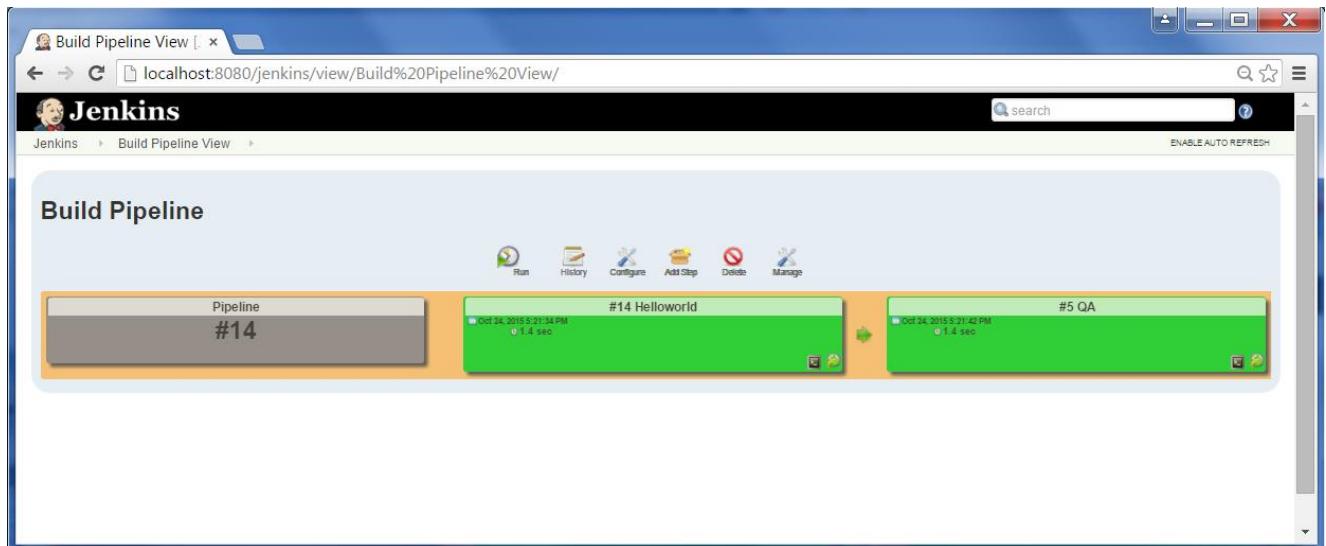
**Step 3 :** Enter any name for the View name and choose the option 'Build Pipeline View'.

The screenshot shows the Jenkins 'New View' configuration dialog. The title bar says 'New View [Jenkins]'. The URL in the address bar is 'localhost:8080/jenkins/newView'. The main area has a 'Jenkins' logo and a search bar. On the left is a sidebar with links: 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. The main configuration panel has a 'View name' field set to 'Build Pipeline View'. Below it are three radio button options: 'Build Pipeline View' (selected), 'Delivery Pipeline View', and 'List View'. The 'Build Pipeline View' description states: 'Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.' The 'Delivery Pipeline View' description states: 'Shows one or more delivery pipeline instances.' The 'List View' description states: 'Shows items in a simple list format. You can choose which jobs are to be displayed in which view.' At the bottom right of the configuration panel is an 'OK' button. Below the configuration panel are two collapsed sections: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing '1 Idle' and '2 Idle'). At the bottom of the page are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 5:50:44 PM', 'REST API', and 'Jenkins ver. 1.809.3'.

**Step 4 :** Accept the default settings, just in the Selected Initial job, ensure to enter the name of the Helloworld project. Click on the Ok button.

The screenshot shows the Jenkins 'Edit View [Jenkins]' configuration page for 'Build Pipeline View'. The 'Name' field is set to 'Build Pipeline View'. Under 'Select Initial Job', 'Helloworld' is selected. The 'Based on upstream/downstream relations' dropdown is open, showing options for triggering builds based on upstream/downstream relationships. The 'No Of Displayed Builds' dropdown is set to 1. The 'Console Output Link Style' dropdown is set to 'Lightbox'. At the bottom, there are 'OK' and 'Apply' buttons.

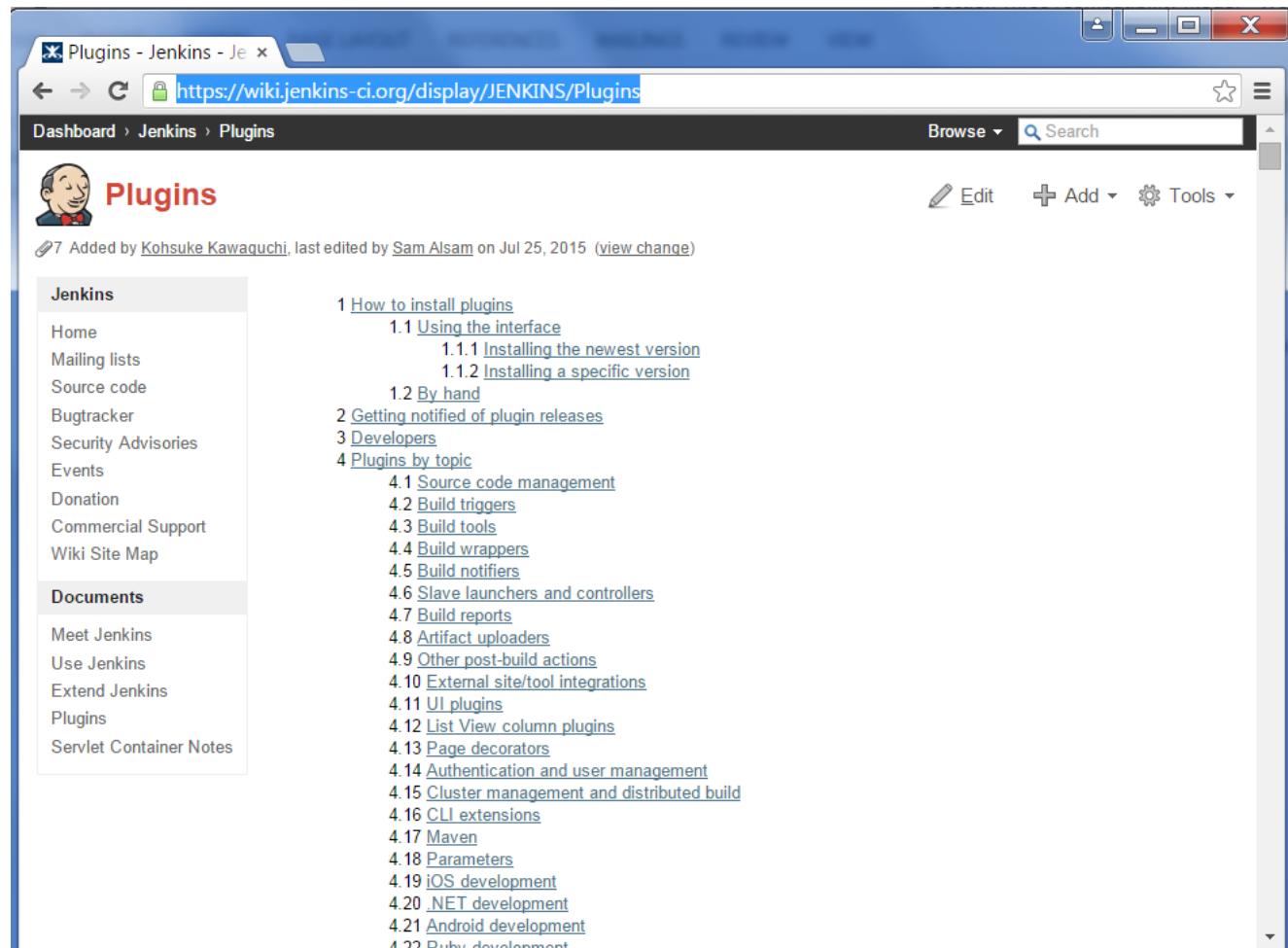
You will now see a great view of the entire delivery pipeline and you will be able to see the status of each project in the entire pipeline.



# 19. Jenkins – Managing Plugins

To get the list of all plugins available within Jenkins, one can visit the link -

<https://wiki.jenkins-ci.org/display/JENKINS/Plugins>



The screenshot shows a web browser window displaying the Jenkins Plugins page. The URL in the address bar is <https://wiki.jenkins-ci.org/display/JENKINS/Plugins>. The page title is "Plugins". On the left, there is a sidebar with sections for "Jenkins" (Home, Mailing lists, Source code, Bugtracker, Security Advisories, Events, Donation, Commercial Support, Wiki Site Map) and "Documents" (Meet Jenkins, Use Jenkins, Extend Jenkins, Plugins, Servlet Container Notes). The main content area lists various plugin categories and their sub-topics:

- 1 [How to install plugins](#)
  - 1.1 [Using the interface](#)
    - 1.1.1 [Installing the newest version](#)
    - 1.1.2 [Installing a specific version](#)
  - 1.2 [By hand](#)- 2 [Getting notified of plugin releases](#)
- 3 [Developers](#)
- 4 [Plugins by topic](#)
  - 4.1 [Source code management](#)
  - 4.2 [Build triggers](#)
  - 4.3 [Build tools](#)
  - 4.4 [Build wrappers](#)
  - 4.5 [Build notifiers](#)
  - 4.6 [Slave launchers and controllers](#)
  - 4.7 [Build reports](#)
  - 4.8 [Artifact uploaders](#)
  - 4.9 [Other post-build actions](#)
  - 4.10 [External site/tool integrations](#)
  - 4.11 [UI plugins](#)
  - 4.12 [List View column plugins](#)
  - 4.13 [Page decorators](#)
  - 4.14 [Authentication and user management](#)
  - 4.15 [Cluster management and distributed build](#)
  - 4.16 [CLI extensions](#)
  - 4.17 [Maven](#)
  - 4.18 [Parameters](#)
  - 4.19 [iOS development](#)
  - 4.20 [.NET development](#)
  - 4.21 [Android development](#)
  - 4.22 [Ruby development](#)

We've already seen many instances for installing plugins, let's look at some other maintenance tasks with regards to plugins

## Uninstalling Plugins

To uninstall a plugin, Go to Manage Jenkins->Manage plugins. Click on the Installed tab. Some of the plugins will have the Uninstall option. You can click these buttons to uninstall the plugins. Ensure to restart your Jenkins instance after the uninstallation.

Enabled	Name	Version	Previously installed version	Pinned	Uninstall
<input checked="" type="checkbox"/>	<a href="#">Ant Plugin</a> This plugin adds Apache Ant support to Jenkins.	<a href="#">1.2</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Build History Metrics plugin</a> This plugin is used to provide reliability metrics for a job	<a href="#">1.2</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Build Pipeline Plugin</a> This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.	<a href="#">1.4.8</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Credentials Plugin</a> This plugin allows you to store credentials in Jenkins.	<a href="#">1.23</a>	<a href="#">Downgrade to 1.18</a>	<a href="#">Unpin</a>	<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">CVS Plug-in</a> Integrates Jenkins with CVS version control system using a modified version of the Netbeans cvscclient.	<a href="#">2.11</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Delivery Pipeline Plugin</a> This plugin visualize Delivery Pipelines (Jobs with upstream/downstream dependencies)	<a href="#">0.9.7</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Deploy to container Plugin</a> This plugin allows you to deploy a war to a container after a successful build. Glassfish 3.x remote deployment	<a href="#">1.10</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">External Monitor Job Type Plugin</a> Adds the ability to monitor the result of externally executed jobs.	<a href="#">1.4</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">Extreme Notification Plugin</a> This plugin is a sample to explain how to write a Jenkins plugin.	<a href="#">1.1</a>			<a href="#">Uninstall</a>
<input checked="" type="checkbox"/>	<a href="#">FindBugs Plug-in</a> This plug-in collects the FindBugs analysis results of the project modules and visualizes the found warnings.	<a href="#">4.62</a>			<a href="#">Uninstall</a>

## Installing another Version of a Plugin

Sometimes it may be required to install an older version of a plugin, in such a case, you can download the plugin from the relevant plugin page on the Jenkins web site. You can then use the **Upload** option to upload the plugin manually.

The screenshot shows the Jenkins Update Center interface at [localhost:8080/jenkins/pluginManager/advanced](http://localhost:8080/jenkins/pluginManager/advanced). The top navigation bar includes 'Updates', 'Available', 'Installed', and 'Advanced' tabs, with 'Advanced' being the active tab. The main content area is divided into three sections:

- HTTP Proxy Configuration:** Fields for Server, Port, User name, Password, and No Proxy Host, each with a help icon (question mark).
- Upload Plugin:** A 'Submit' button and a file input field labeled 'File: Choose File' with the message 'No file chosen'. Below it is an 'Upload' button.
- Update Site:** A 'Submit' button and a URL input field containing 'http://updates.jenkins-ci.org/update-center.json'. To the right, a status message says 'Update information obtained: 2 hr 5 min ago' and a 'Check now' button.

# 20. Jenkins – Security

In Jenkins you have the ability to setup users and their relevant permissions on the Jenkins instance. By default you will not want everyone to be able to define jobs or other administrative tasks in Jenkins. So Jenkins has the ability to have a security configuration in place.

To configure Security in Jenkins, follow the steps given below.

**Step 1 :** Click on Manage Jenkins and choose the 'Configure Global Security' option.

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below that are sections for 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'Manage Jenkins'. It features several configuration options with icons: 'Configure System' (gear icon), 'Configure Global Security' (padlock icon), 'Reload Configuration from Disk' (refresh icon), 'Manage Plugins' (puzzle piece icon), 'System Information' (monitor icon), 'System Log' (document icon), 'Load Statistics' (graph icon), 'Jenkins CLI' (terminal icon), 'Script Console' (script icon), and 'Manage Nodes' (computer icon). A warning message at the top right says: '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.' Another message says: 'New version of Jenkins (1.625.1) is available for download ([changelog](#)).'. At the bottom right of the main area, there are 'Setup Security' and 'Dismiss' buttons. A search bar is at the top right, and a 'ENABLE AUTO REFRESH' link is near the top center.

**Step 2 :** Click on Enable Security option. As an example, let's assume that we want Jenkins to maintain its own database of users, so in the Security Realm, choose the option of 'Jenkins' own user database'.

By default you would want a central administrator to define users in the system, hence ensure the 'Allow users to sign up' option is unselected. You can leave the rest as it is for now and click the Save button.

The screenshot shows the Jenkins 'Configure Global Security' configuration page. Key settings include:

- Enable security:** Checked
- TCP port for JNLP slave agents:** Fixed: [ ] (disabled)
- Disable remember me:** Unchecked
- Access Control:** Jenkins' own user database (selected), Allow users to sign up (unchecked)
- Security Realm:** Jenkins' own user database (selected)
- Authorization:** Anyone can do anything (selected)
- Markup Formatter:** Escaped HTML (selected)
- Escaped HTML:** Treats all input as plain text. HTML unsafe characters like < and & are escaped to their respective character entities.
- Prevent Cross Site Request Forgery exploits:** Unchecked
- Use browser for metadata download:** Unchecked

At the bottom, there are 'Save' and 'Apply' buttons, with 'Save' being the active button.

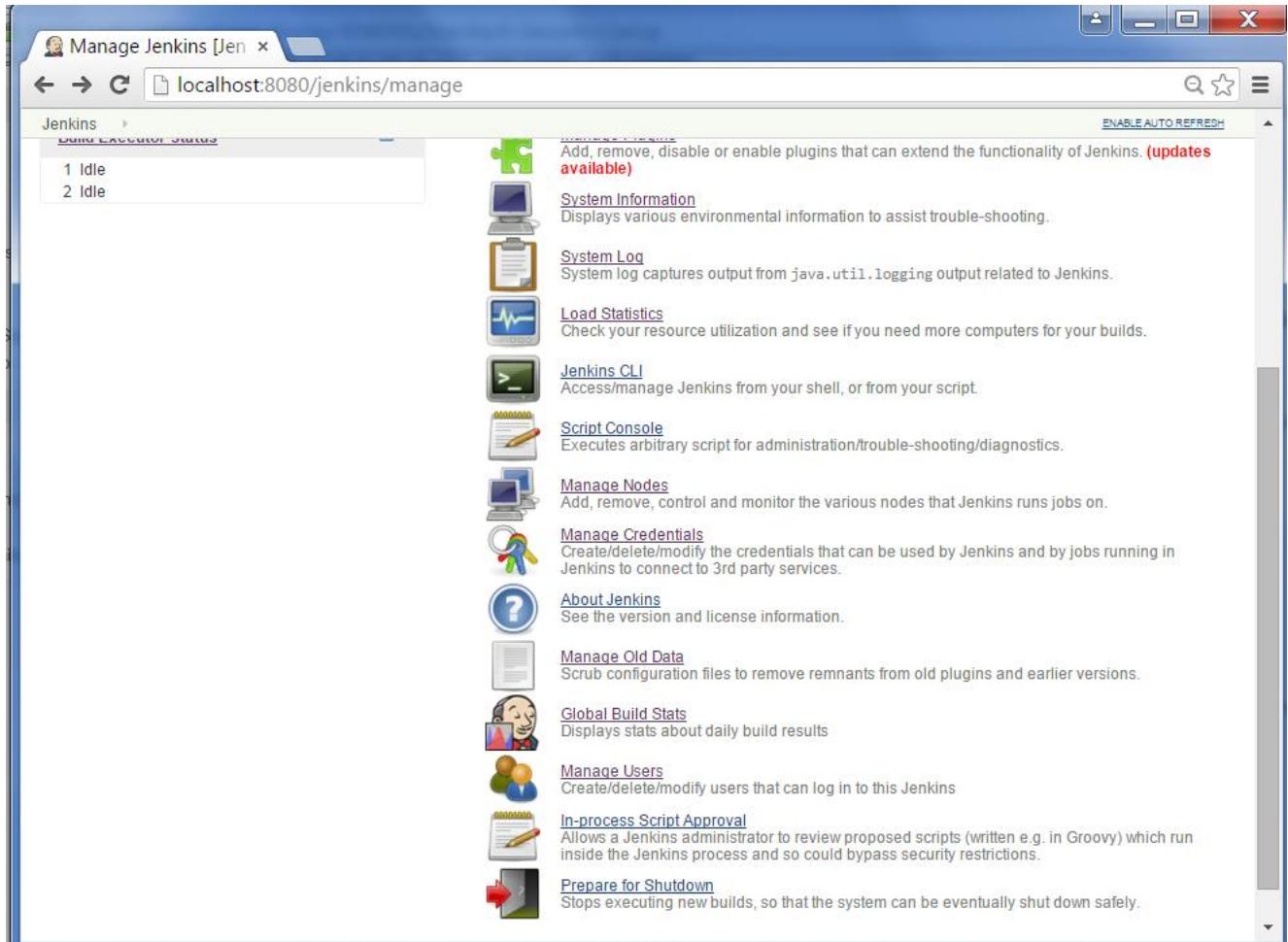
**Step 3 :** You will be prompted to add your first user. As an example, we are setting up an admin users for the system.

The screenshot shows the Jenkins 'Sign up' page. The URL in the browser is `localhost:8080/jenkins/securityRealm/firstUser`. The page title is 'Sign up [Jenkins]'. On the left sidebar, there are links: 'Back to Dashboard', 'Manage Jenkins', and 'Create User'. The main content area has a heading 'Sign up'. It contains five input fields with the following values:

Username:	admin
Password:	.....
Confirm password:	.....
Full name:	Administrator
E-mail address:	al@gmail.com

A blue 'Sign up' button is located below the input fields. At the bottom of the page, there is a link 'Help us localize this page' and footer text 'Page generated: Oct 24, 2015 6:18:01 PM REST API Jenkins ver. 1.609.3'.

**Step 4 :** It's now time to setup your users in the system. Now when you go to Manage Jenkins, and scroll down, you will see a 'Manage Users' option. Click this option.



**Step 5 :** Just like you defined your admin user, start creating other users for the system. As an example, we are just creating another user called 'user'.

The screenshot shows a web browser window for the Jenkins 'Sign up' page. The URL in the address bar is `localhost:8080/jenkins/securityRealm/addUser`. The page title is 'Sign up [Jenkins]'. The main content is a 'Sign up' form with the following fields:

Username:	user
Password:	.....
Confirm password:	.....
Full name:	user
E-mail address:	user@gmail.com

A blue 'Sign up' button is located below the form. On the left side of the page, there is a sidebar with links: 'Back to Dashboard', 'Manage Jenkins', and 'Create User'. At the bottom of the page, there is a link 'Help us localize this page' and footer information: 'Page generated: Oct 24, 2015 6:20:59 PM REST API Jenkins ver. 1.609.3'.

**Step 6 :** Now it's time to setup your authorizations, basically who has access to what. Go to Manage Jenkins->Configure Global Security.

Now in the Authorization section, click on 'Matrix based security'

The screenshot shows the Jenkins 'Configure Global Security' page. At the top, there are options for 'Enable security', 'TCP port for JNLP slave agents' (set to 'Random'), and 'Access Control'. Below these are sections for 'Security Realm' (set to 'Jenkins' own user database') and 'Authorization' (set to 'Matrix-based security'). The main area is a large grid table where users can assign permissions to specific Jenkins features. A row for 'User/group' contains a dropdown with 'Anonymous' and 'user' selected, and an 'Add' button. A column for 'Overall' permissions includes checkboxes for 'Administer', 'Configure', 'Update Center', 'Read', 'Run Scripts', 'Upload Plugins', 'Create', 'Delete', 'Manage Domains', 'Update View', 'Build', 'Configure', 'Connect', 'Create', 'Delete', 'Disconnect', 'Build', 'Cancel', 'Configure', 'Create', 'Delete', 'Discover', 'Read', 'Workspace', 'Delete', 'Update', 'Configure', 'Create', 'Delete', 'Read', and 'Tag'. Other columns represent 'Credentials', 'Slave', 'Job', 'Run', 'View', and 'SCM' with similar sets of checkboxes.

**Step 7 :** If you don't see the user in the user group list, enter the user name and add it to the list. Then give the appropriate permissions to the user.

Click on the Save button once you have defined the relevant authorizations.

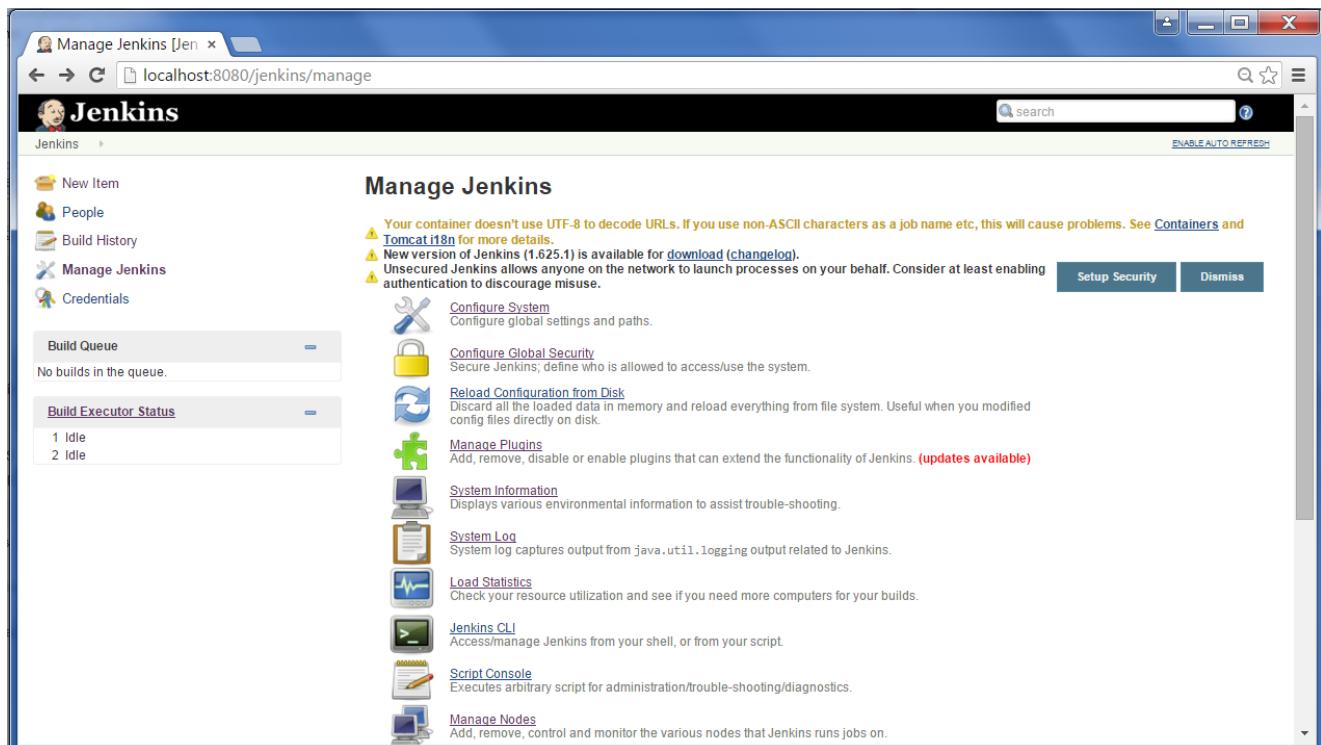
Your Jenkins security is now setup.

**Note :** For Windows AD authentication, one has to add the Active Directory plugin to Jenkins.

# 21. Jenkins – Backup Plugin

Jenkins has a backup plugin which can be used to backup critical configuration settings related to Jenkins. Follow the steps given below to have a backup in place.

**Step 1 :** Click on Manage Jenkins and choose the 'Manage Plugins' option.



The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below that are 'Build Queue' and 'Build Executor Status' sections. The main area is titled 'Manage Jenkins' and contains several configuration options with icons. One of the options is 'Manage Plugins', which is highlighted. At the top right, there are buttons for 'Setup Security' and 'Dismiss'.

**Step 2 :** In the available tab, search for 'Backup Plugin'. Click On Install without Restart. Once done, restart the Jenkins instance

The screenshot shows the Jenkins Update Center interface. The URL is `localhost:8080/jenkins/pluginManager/available`. The 'Available' tab is selected. A search bar at the top right contains the text 'backup'. A table lists several plugins:

Install	Name	Version
<input checked="" type="checkbox"/>	<a href="#">Backup plugin</a>	1.6.1
<input type="checkbox"/>	<a href="#">Backup and interrupt job plugin</a>	1.0
<input type="checkbox"/>	<a href="#">Install CloudBees Jenkins Enterprise</a>	15.05.1
<input type="checkbox"/>	<a href="#">CloudBees Free Enterprise Plugins</a>	5.0
<input type="checkbox"/>	<a href="#">Periodic Backup</a>	1.3
<input type="checkbox"/>	<a href="#">ThinBackup</a>	1.7.4

At the bottom, there are two buttons: 'Install without restart' (highlighted in blue) and 'Download now and install after restart'.

The screenshot shows the Jenkins Update Center interface. The URL is `localhost:8080/jenkins/updateCenter/`. The title is 'Installing Plugins/Upgrades'. The 'Preparation' section lists:

- Checking internet connectivity
- Checking update center connectivity
- Success

The 'Backup plugin' status is shown as 'Success' with a green circular icon. Below the status are two green arrows pointing right:

- Go back to the top page (you can start using the installed plugins right away)
- Restart Jenkins when installation is complete and no jobs are running

At the bottom, there are links for 'Help us localize this page' and 'Page generated: Oct 24, 2015 6:28:30 PM REST API Jenkins ver. 1.609.3'.

**Step 3 :** Now when you go to Manage Jenkins, and scroll down you will see 'Backup Manager' as an option. Click on this option.

The screenshot shows the Jenkins 'Manage Jenkins' interface at the URL [localhost:8080/jenkins/manage](http://localhost:8080/jenkins/manage). The page lists several management options with corresponding icons:

- System Log**: System log captures output from `java.util.logging` related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.
- Jenkins CLI**: Access/manage Jenkins from your shell, or from your script.
- Script Console**: Executes arbitrary script for administration/trouble-shooting/diagnostics.
- Manage Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
- Manage Credentials**: Create/delete/modify the credentials that can be used by Jenkins and by jobs running in Jenkins to connect to 3rd party services.
- About Jenkins**: See the version and license information.
- Manage Old Data**: Scrub configuration files to remove remnants from old plugins and earlier versions.
- Global Build Stats**: Displays stats about daily build results.
- Manage Users**: Create/delete/modify users that can log in to this Jenkins.
- In-process Script Approval**: Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.
- Backup manager**: Backup or Restore Jenkins configuration files.
- Prepare for Shutdown**: Stops executing new builds, so that the system can be eventually shut down safely.

At the bottom of the page, there are links for localization help, page generation details, and REST API access.

**Step 4 :** Click on Setup.

The screenshot shows the Jenkins Backup manager interface. On the left, there's a sidebar with links: New Item, People, Build History, Manage Jenkins, and Credentials. Below these are two expandable sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'Backup manager' and contains three items: 'Setup' (with a wrench icon), 'Backup Hudson configuration' (with a blue arrow icon), and 'Restore Hudson configuration' (with a blue arrow icon). At the bottom of the page, there are links for 'Help us localize this page', 'Page generated: Oct 24, 2015 7:11:02 PM', 'REST API', and 'Jenkins ver. 1.609.3'.

**Step 5 :** Here, the main field to define is the directory for your backup. Ensure it's on another drive which is different from the drive where your Jenkins instance is setup. Click on the Save button.

The screenshot shows the Jenkins interface for managing backup settings. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below that are sections for 'Build Queue' (empty) and 'Build Executor Status' (two idle executors). The main content area is titled 'Backup config files' and contains a 'Backup configuration' section. It shows the 'Hudson root directory' as E:\Jenkins and the 'Backup directory' as D:\Backup. The 'Format' is set to 'zip' and the 'File name template' is 'backup\_@date@. @extension@'. Under 'Custom exclusions', there are three checkboxes: 'Verbose mode', 'Configuration files (.xml) only', and 'No shutdown'. In the 'Backup content' section, four checkboxes are available: 'Backup job workspace', 'Backup builds history', 'Backup maven artifacts archives', and 'Backup fingerprints'. A 'Save' button is located at the bottom right of this section. At the very bottom of the page, there's a link to help localize the page and a footer note about the page being generated on Oct 24, 2015, at 7:17:46 PM.

**Step 6 :** Click on the 'Backup Hudson configuration' from the Backup manager screen to initiate the backup.

The screenshot shows the Jenkins interface with the URL `localhost:8080/jenkins/backup/` in the browser address bar. The page title is "Backup manager". On the left, there is a sidebar with links: "New Item", "People", "Build History", "Manage Jenkins", and "Credentials". Below these are two expandable sections: "Build Queue" (No builds in the queue) and "Build Executor Status" (1 Idle, 2 Idle). The main content area is titled "Backup manager" and contains three items: "Setup" (with a wrench icon), "Backup Hudson configuration" (with a blue arrow icon), and "Restore Hudson configuration" (with a grey arrow icon). At the bottom of the page, there is a link "Help us localize this page" and footer text "Page generated: Oct 24, 2015 7:11:02 PM REST API Jenkins ver. 1.609.3".

The next screen will show the status of the backup.

The screenshot shows the Jenkins interface for managing backups. The left sidebar includes links for New Item, People, Build History, Manage Jenkins, and Credentials. The main area displays a log message indicating a successful backup process:

```
Jenkins is going to shut down
[ INFO] Backup started at [10/24/15 19:19:31]
[ INFO] Setting hudson in shutdown mode to avoid files corruptions.
[ INFO] Waiting all jobs end...
[ INFO] Number of running jobs detected : 0
[ INFO] All jobs finished.
[ INFO] Full backup file name : D:\Backup\backup_20151024_1919.zip
[ INFO] Saved files : 911
[ INFO] Number of errors : 0
[ INFO] Cancel hudson shutdown mode
[ INFO] Backup end at [10/24/15 19:19:50]
[ INFO] [19.524s]
```

Below the log, sections for Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle) are shown. At the bottom, there are links for Help us localize this page, Page generated: Oct 24, 2015 7:19:31 PM, REST API, and Jenkins ver. 1.609.3.

To recover from a backup, go to the Backup Manager screen, click on Restore Hudson configuration.

The screenshot shows the Jenkins interface at [localhost:8080/jenkins/backup/](http://localhost:8080/jenkins/backup/). The left sidebar includes links for New Item, People, Build History, Manage Jenkins, and Credentials. Below these are sections for Build Queue (empty) and Build Executor Status (1 Idle, 2 Idle). The main content area is titled "Backup manager" and contains three items: "Setup" (with a wrench icon), "Backup Hudson configuration" (with a blue arrow icon), and "Restore Hudson configuration" (with a grey arrow icon). A footer bar at the bottom includes a link to help localize the page, a timestamp (Page generated: Oct 24, 2015 7:11:02 PM), and API/Jenkins version information (REST API Jenkins ver. 1.609.3).

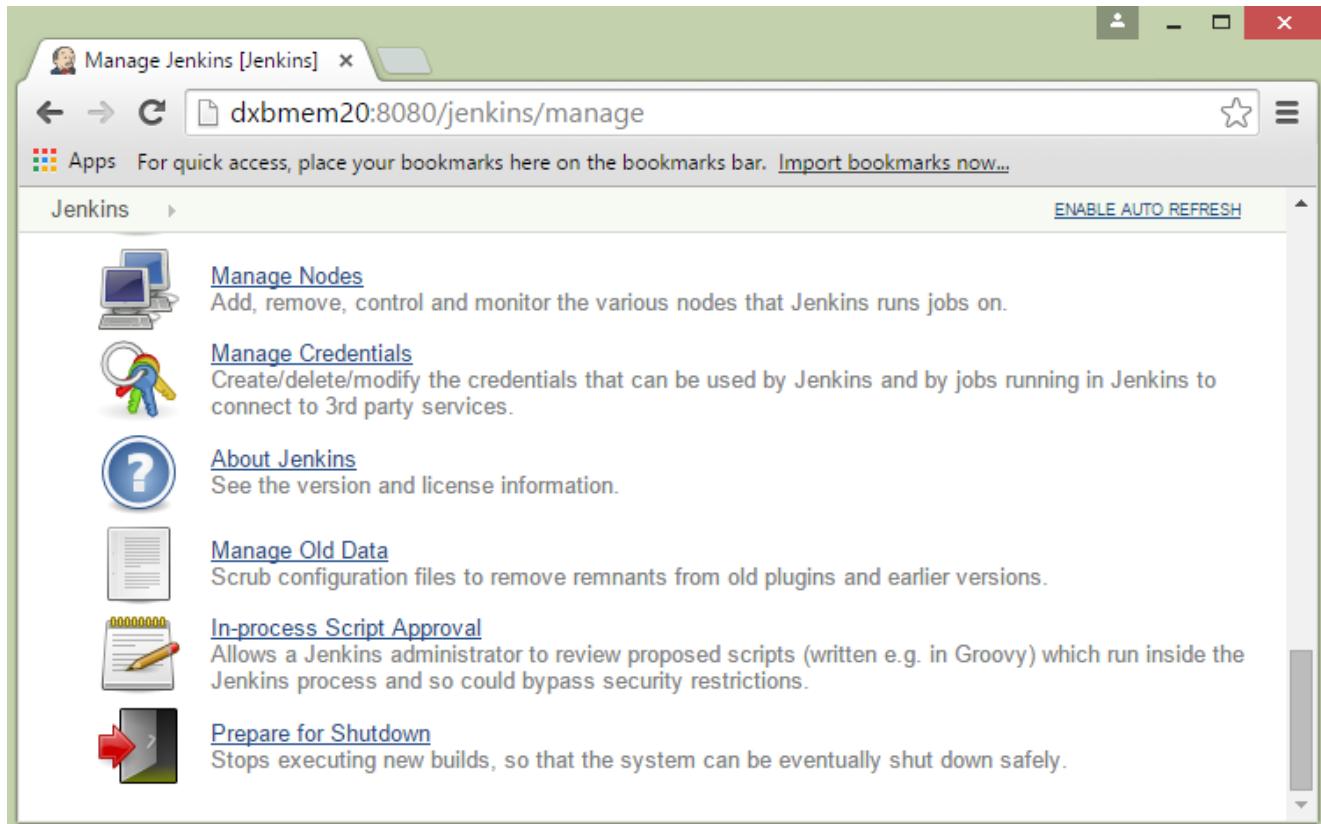
The list of backup's will be shown, click on the appropriate one to click on Launch Restore to begin the restoration of the backup.

The screenshot shows the Jenkins Backup manager interface. At the top, there is a navigation bar with links for 'Jenkins' and 'Backup manager'. On the right side of the header are search and login buttons. The main content area is titled 'Backup manager' and displays the message 'Available backup in D:\Backup :'. Below this, a radio button is selected next to the file name 'backup\_20151024\_1919.zip'. A blue 'Launch restore' button is positioned directly below the file name. To the left of the main content, there are two expandable sections: 'Build Queue' (which shows 'No builds in the queue.') and 'Build Executor Status' (which shows '1 Idle' and '2 Idle'). At the bottom of the page, there is a link to 'Help us localize this page' and footer information indicating the page was generated on Oct 24, 2015 at 7:20:45 PM, and it includes links for 'REST API' and 'Jenkins ver. 1.609.3'.

## 22. Jenkins – Remote Testing

Web tests such as selenium tests can be run on remote slave machines via the master slave and selenium suite plugin installation. The following steps show how to run remote tests using this configuration.

**Step 1 :** Ensuring your master slave configuration is in place. Got to your master Jenkins server. Go to Manage Jenkins->Manage Nodes.



The screenshot shows the Jenkins management interface. The title bar says "Manage Jenkins [Jenkins]". The URL in the address bar is "dxbmeme20:8080/jenkins/manage". Below the address bar, there's a "Apps" section with a link to "Import bookmarks now...". The main content area is titled "Jenkins". On the right, there's a "ENABLE AUTO REFRESH" link. The left sidebar has several options with icons: "Manage Nodes" (monitoring icon), "Manage Credentials" (key and lock icon), "About Jenkins" (question mark icon), "Manage Old Data" (document icon), "In-process Script Approval" (script icon), and "Prepare for Shutdown" (red arrow icon). The "Manage Nodes" option is currently selected and highlighted in blue.

In our node list, the DXBMEM30 label is the slave machine. In this example, both the master and slave machines are windows machines.

The screenshot shows the Jenkins 'Nodes' page. At the top, there's a header bar with a user icon, a search bar containing 'dxbmeme20:8080/jenkins/computer/', and various navigation icons. Below the header, a message says 'For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...'. The main content area has a breadcrumb trail: 'Jenkins > nodes >'. On the right, there's a link 'ENABLE AUTO REFRESH' and a 'Refresh status' button. The central part of the page displays a table of nodes:

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Sp
1	<a href="#">DXBMEM30</a>	Windows Server 2012 (x86)	In sync	112.79 GB	4.54 GB	13 min
2	<a href="#">master</a>	Windows Server 2012 (x86)	In sync	94.20 GB	3.85 GB	94.20
	Data obtained	13 min	13 min	13 min	13 min	1

At the bottom, there's a footer bar with the URL 'dxbmeme20:8080/jenkins/computer/#'.

**Step 2 :** Click on configure for the DXBMEM30 slave machine.

The screenshot shows the Jenkins 'Nodes' configuration page. At the top, there's a header bar with the Jenkins logo and a 'Nodes [Jenkins]' tab. Below the header is a toolbar with icons for back, forward, search, and other functions. The main content area has a breadcrumb navigation path: Jenkins > nodes > DXBMEM30. On the left, there's a sidebar with sections for 'Idle' and 'DXBMEM30'. The 'DXBMEM30' section shows one idle node. Below this is a table with columns: S, Name, Architecture, Clock Difference, Free Disk Space, Free Swap Space, and Free Temp Sp. The first row for 'DXBMEM30' is selected. A context menu is open over this row, with the 'Configure' option highlighted in blue. Other options in the menu include 'Delete Slave' and 'Build History'. At the bottom right of the table, there's a 'Refresh status' button. The URL in the browser is 'dxbmem20:8080/jenkins/computer/'.

**Step 3 :** Ensure the launch method is put as 'Launch slave agents via Java Web Start'

The screenshot shows the configuration page for the 'DXBMEM30' slave machine. The title bar says 'DXBMEM30 Configuration'. The page has a breadcrumb path: Jenkins > nodes > DXBMEM30. It contains several configuration fields: 'Name' (DXBMEM30), 'Description' (empty), '# of executors' (1), 'Remote root directory' (C:\users\administrator.EMIRATES\jenkins), 'Labels' (empty), 'Usage' (Utilize this node as much as possible), and 'Launch method' (set to 'Launch slave agents via Java Web Start'). There's also an 'Advanced...' button and a 'Save' button at the bottom. The URL in the browser is 'dxbmem20:8080/jenkins/computer/DXBMEM30/configure'.

**Step 4 :** Now go to your slave machine and from there, open a browser instance to your Jenkins master instance. Then go to Manage Jenkins->Manage Nodes. Go to DXBMEM30 and click on

The screenshot shows the Jenkins dashboard for a slave node named 'dxbmem30:8080/jenkins/'. The main menu includes 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'Credentials'. Below the menu, sections for 'Build Queue' (empty) and 'Build Executor Status' (showing 1 Idle and 2 Idle executors) are displayed. A 'Manage Nodes' link is visible in the top right corner of the dashboard area.

**Step 5 :** Click on the DXBMEM30 instance.

The screenshot shows the Jenkins 'Nodes' page. The URL in the browser is `dxbmem20:8080/jenkins/computer/`. The page displays the 'Build Executor Status' for the 'master' node and the 'DXBMEM30' node. The 'DXBMEM30' node is listed as '(offline)'. Below this, a table provides detailed information for each node:

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space
	<a href="#">DXBMEM30</a>	Windows Server 2012 (x86)	In sync	112.79 GB	4.54 GB
	<a href="#">master</a>	Windows Server 2012 (x86)	In sync	94.20 GB	3.85 GB
	Data obtained	45 min	45 min	45 min	45 min

A blue button labeled 'Refresh status' is visible at the bottom right of the table area.

**Step 6 :** Scroll down and you will see the Launch option which is the option to Start 'Java Web Start'

The screenshot shows the Jenkins interface for managing a slave node named 'DXBMEM30'. At the top, there's a header bar with a logo, the node name, and standard window controls. Below the header, the URL in the address bar is `dxbmemo20:8080/jenkins/computer/DXBMEM30/`. The main content area has a breadcrumb navigation: 'Jenkins > nodes > DXBMEM30'. To the right of the breadcrumb is a link to 'ENABLE AUTO REFRESH'. The page displays a log entry:

```
at org.jenkinsci.remoting.nio.NioChannelHub.run(NioChannelHub.java:561)
... 6 more
```

Below the log, instructions for connecting the slave to Jenkins are provided:

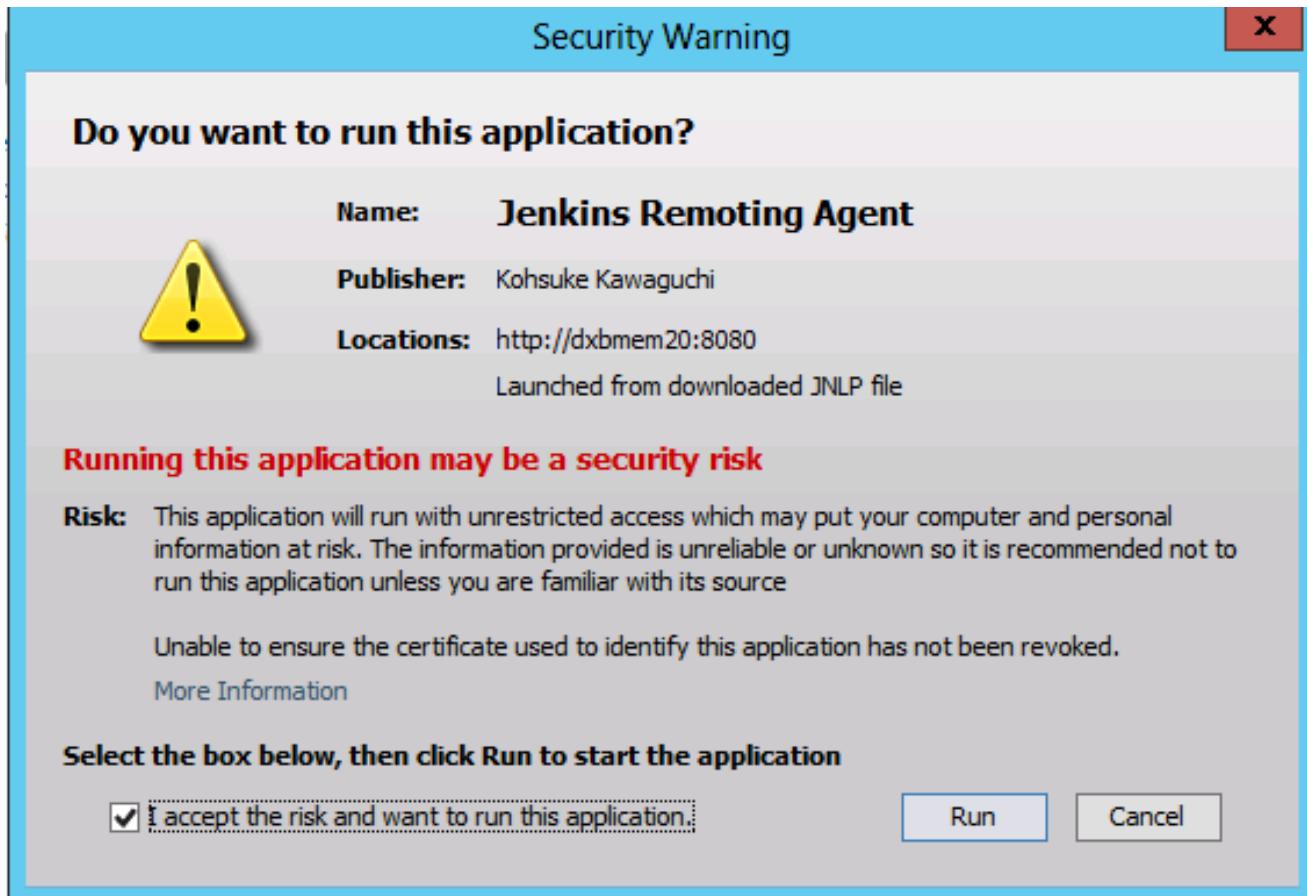
- **Launch** Launch agent from browser on slave
- Run from slave command line:  
`javaws http://localhost:8080/jenkins/computer/DXBMEM30/slave-agent.jnlp`
- Or if the slave is headless:  
`java -jar slave.jar -jnlpUrl http://localhost:8080/jenkins/computer/DXBMEM30/slave-agent.jnlp`

At the bottom left, it says 'Created by anonymous user'. On the right side, there's a section titled 'Projects tied to DXBMEM30' with a table showing build status for a single project:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		<a href="#">HelloWorld</a>	43 min - #12	41 min - #13	7.3 sec

Below the table, there are icons for 'Icon: S M L' and links for 'Legend', 'RSS for all', 'RSS for failures', and 'RSS for just latest builds'.

**Step 7 :** You will be presented with a Security Warning. Click on the Acceptance checkbox and click on run.



You will now see a Jenkins Slave window opened and now connected.

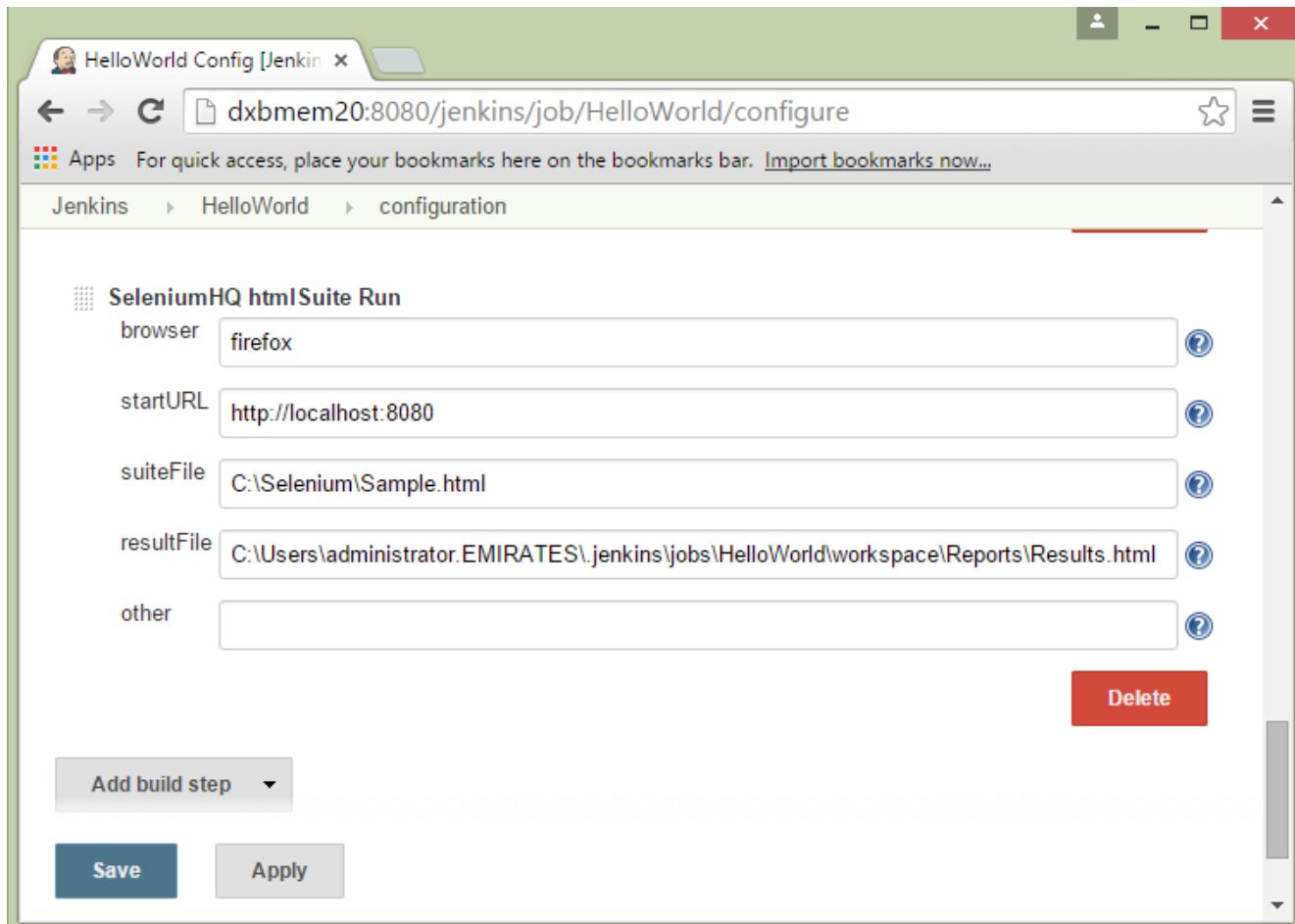


**Step 8 :** Configuring your tests to run on the slave. Here, you have to ensure that the job being created is meant specifically to only run the selenium tests.

In the job configuration, ensure the option 'Restrict where this project can be run' is selected and in the Label expression put the name of the slave node.

The screenshot shows the Jenkins configuration interface for a job named 'HelloWorld'. In the 'Advanced Project Options' section, the 'Restrict where this project can be run' checkbox is checked, and the 'Label Expression' field contains 'DXBMEM30'. The 'Save' and 'Apply' buttons are visible at the bottom left, and an 'Advanced...' button is located in the top right corner of the configuration area.

**Step 9 :** Ensure the selenium part of your job is configured. You have to ensure that the Sample.html file and the selenium-server.jar file is also present on the slave machine.



The screenshot shows the Jenkins configuration interface for the 'HelloWorld' job. The browser is set to 'firefox', the start URL is 'http://localhost:8080', the suite file is 'C:\Selenium\Sample.html', and the result file is 'C:\Users\administrator.EMIRATES\jenkins\jobs\HelloWorld\workspace\Reports\Results.html'. There is a 'Delete' button at the bottom right of the configuration section. Below the configuration section, there is a 'Save' button and an 'Apply' button.

Once you have followed all of the above steps, and click on Build, this project will run the Selenium test on the slave machine as expected.