

Jenkins

Lab Book

Copyright © 2011 IGATE Corporation. All rights reserved. No part of this publication shall be reproduced in any way, including but not limited to photocopy, photographic, magnetic, or other record, without the prior written permission of iGATE Corporation.

IGATE Corporation considers information included in this document to be Confidential and Proprietary.

Document Revision History

Date	Revision No.	Author	Summary of Changes
24/05/2013	0.1	Rathnajothi Perumalsamy	First version

Table of Contents

<i>Document Revision History</i>	2
<i>Table of Contents</i>	3
<i>Getting Started</i>	5
<i>Overview</i>	5
<i>Setup Checklist for Jenkins</i>	5
<i>Instructions</i>	5
<i>Learning More (Bibliography if applicable)</i>	5
<i>Lab 1. Installing and running Jenkins</i>	6
1.1: <i>Installing Jenkins</i>	6
1.1.1: <i>As a standalone application</i>	6
1.1.2: <i>As a windows service</i>	7
1.2: <i><<TODO></i>	7
<i>Lab 2. Jenkins Configuration</i>	8
2.1: <i>Configuring Jenkins</i>	8
2.1.1: <i>Configuring JDK</i>	8
2.1.2: <i>Configuring ANT</i>	9
2.1.3: <i>Email Configuration in Jenkins</i>	9
<i>Lab 3. Build Job creation in Jenkins</i>	11
3.1: <i>Creating build job in Jenkins</i>	11
3.2: <i>Configure build job in Jenkins</i>	11
3.3: <i>Configure SCM</i>	12
3.3: <i>Scheduling job execution</i>	13
3.4: <i>Invoke Ant</i>	13
<<TODO>>.....	15
<i>Lab 4. Automated deployment and continuous delivery</i>	16
4.1: <i>Installation of Deploy plug-in</i>	16
4.2: <i>Automate deployment of an application</i>	16
<<TODO>>.....	18
<i>Lab 5. Automating Testing using Jenkins</i>	19
5.1: <i>Automate unit testing of an application</i>	19
<<TODO>>.....	20
<i>Lab 6. Securing Jenkins</i>	21
6.1: <i>Activating Securing in Jenkins</i>	21
6.2: <i>Creating new users in Jenkins</i>	21

6.3: Authorizing users in Jenkins.....	22
<<TODO>>.....	23
Lab 7. Code Quality.....	24
7.1: Ensuring code quality in Jenkins.....	24
<<TODO>>.....	25
Appendices.....	26
Appendix A: Table of Figures	26

Getting Started

Overview

This lab book is a guided tour for learning Jenkins. It comprises examples and 'To Do' assignments. Follow the steps provided in the examples and work out the 'To Do' assignments given.

Setup Checklist for Jenkins

Here is what is expected on your machine in order for the lab to work.

Minimum System Requirements

- Intel Pentium 90 or higher (P166 recommended)
- Microsoft Windows 95, 98, or NT 4.0, 2k, XP.
- Memory: 32MB of RAM (64MB or more recommended)
- Internet Explorer 6.0 or higher
- MS-Access/Connectivity to Oracle database
- Apache Tomcat Version 5.0.

Please ensure that the following is done:

- Eclipse 3.5 or above with SVN or CVS plugin is installed.
- JDK 1.5 or above is installed. (This path is henceforth referred as <java_install_dir>)
- ANT OR MAVEN is installed
- Connected to SVN or CVS

Instructions

- Create a directory by your name in drive <drive>. In this directory, create a subdirectory Jenkins_assgn. For each lab exercise create a directory as lab <lab number>.

Learning More (Bibliography if applicable)

- Jenkins Continuous Integration Cookbook by Alan Mark Berg
- Jenkins – The Definitive Guide by John Ferguson Smart

Lab 1. Installing and running Jenkins

Goals	• Learn to install and start Jenkins
Time	30 minutes

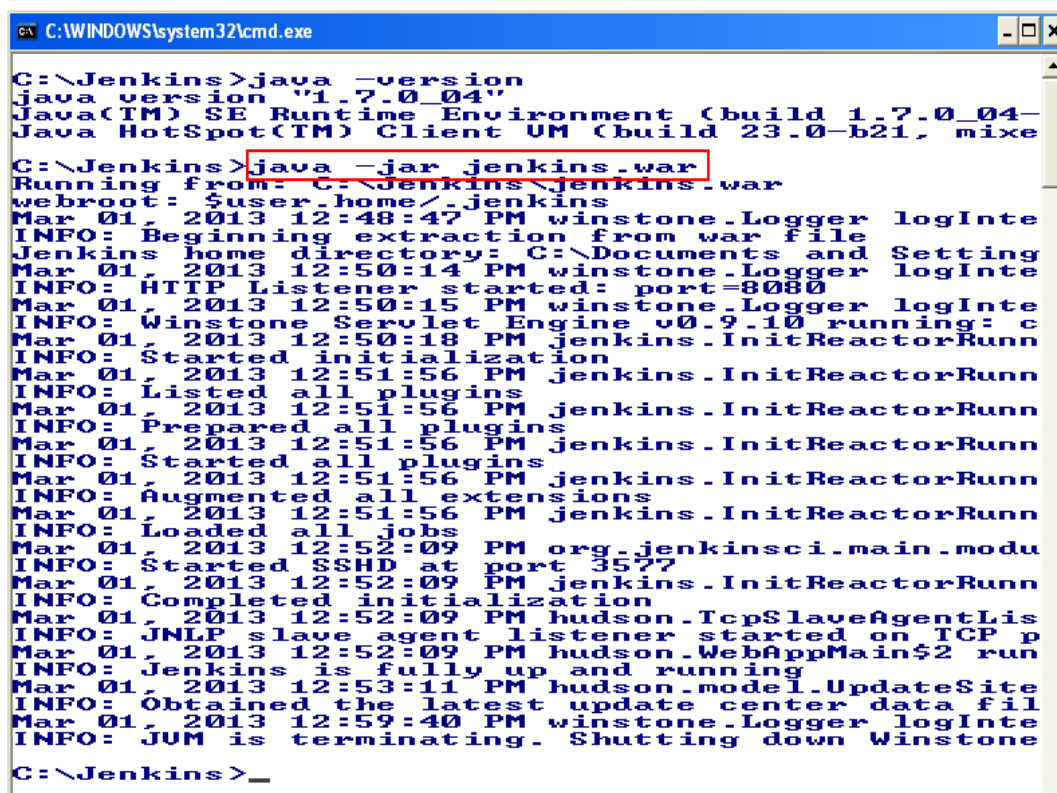
1.1: Installing Jenkins

1.1.1: As a standalone application

Step 1: Download jenkins.war file from <http://jenkins-ci.org> and save it into your local disk.

Step 2: Open Command Prompt and change working directory where jenkins.war file exists.

Step 3: Install Jenkins by executing the command as highlighted in the below screen.



```
C:\WINDOWS\system32\cmd.exe
C:\Jenkins>java -version
java version "1.7.0_04"
Java(TM) SE Runtime Environment (build 1.7.0_04-
Java HotSpot(TM) Client VM (build 23.0-b21, mixe
C:\Jenkins>java -jar jenkins.war
Running from: C:\Jenkins\jenkins.war
webroot: $user.home/.jenkins
Mar 01, 2013 12:48:47 PM winstone.Logger logInte
INFO: Beginning extraction from war file
Jenkins home directory: C:\Documents and Setting
Mar 01, 2013 12:50:14 PM winstone.Logger logInte
INFO: HTTP Listener started: port=8080
Mar 01, 2013 12:50:15 PM winstone.Logger logInte
INFO: Winstone Servlet Engine v0.9.10 running: c
Mar 01, 2013 12:50:18 PM jenkins.InitReactorRunn
INFO: Started initialization
Mar 01, 2013 12:51:56 PM jenkins.InitReactorRunn
INFO: Listed all plugins
Mar 01, 2013 12:51:56 PM jenkins.InitReactorRunn
INFO: Prepared all plugins
Mar 01, 2013 12:51:56 PM jenkins.InitReactorRunn
INFO: Started all plugins
Mar 01, 2013 12:51:56 PM jenkins.InitReactorRunn
INFO: Augmented all extensions
Mar 01, 2013 12:51:56 PM jenkins.InitReactorRunn
INFO: Loaded all jobs
Mar 01, 2013 12:52:09 PM org.jenkinsci.main.modu
INFO: Started SSHD at port 3577
Mar 01, 2013 12:52:09 PM jenkins.InitReactorRunn
INFO: Completed initialization
Mar 01, 2013 12:52:09 PM hudson.TcpSlaveAgentLis
INFO: JNLP slave agent listener started on TCP p
Mar 01, 2013 12:52:09 PM hudson.WebAppMain$2 run
INFO: Jenkins is fully up and running
Mar 01, 2013 12:53:11 PM hudson.model.UpdateSite
INFO: Obtained the latest update center data fil
Mar 01, 2013 12:59:40 PM winstone.Logger logInte
INFO: JVM is terminating. Shutting down Winstone
C:\Jenkins>
```

Figure 1: Jenkins Installation

Step 4: Once Jenkins is started, the Jenkins dash board can be accessed by giving the following link in the browser

<http://localhost:8080/>

1.1.2: As a windows service

Step 1: Start Jenkins as a standalone application.

Step 2: Open Jenkins Dashboard by requesting the URL <http://localhost:8080>

Step 3: Click "Manage Jenkins" link.

Step 4: Click "Install as Windows Service" button.

Step 5: Select the installation directory of Jenkins and click "Install".

Step 6: Once Jenkins is successfully installed as a windows service, Jenkins dashboard will always be accessible by requesting the URL <http://localhost:8080>

1.2: <<TODO>

Install and run Jenkins as a windows Service

Lab 2. Jenkins Configuration

Goals	At the end of this lab session, you will be able to understand: <ul style="list-style-type: none"> How to configure Jenkins
Time	30 minutes

2.1: Configuring Jenkins

Step 1: Open Jenkins Dashboard by requesting the URL <http://localhost:8080>

Step 2: Click “Manage Jenkins” link.

Step 3: Click on “Configure System” Link and fill the field values as shown below:

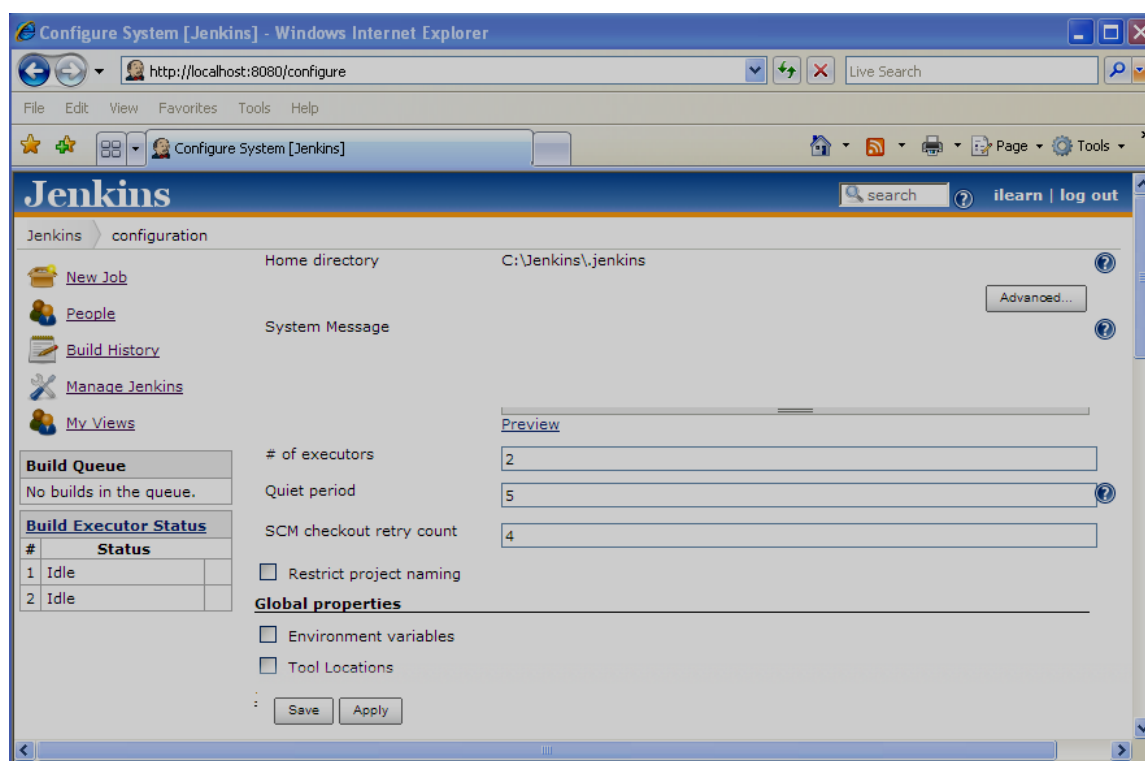


Figure 2: Configure System

2.1.1: Configuring JDK

Step 1: Type name of the JDK. For example, JENKINS_JDK

Step 2: Specify the JDK installation directory



JDK installation can be automated by following the below steps

1. Check “Install Automatically” option in JDK section
2. Select the option to download required JDK version and install.

2.1.2: Configuring ANT

Step 1: Type name of the ANT installation in Name field under ANT section. For example, JENKINS_ANT

Step 2: Specify the ANT installation directory.



ANT installation can be automated by following the below steps

1. Check “Install Automatically” option in ANT section
2. Select the option to download required ANT version and install.

More than one instance of JDK and ANT can be specified.

2.1.3: Email Configuration in Jenkins

To send a feedback to the developers during build failure, E-mail notification configuration is mandatory.

Step 1: Specify details such as SMTP server name, sender credentials, SMTP port in E-mail notification section as shown below:

E-mail Notification

SMTP server: localhost

Default user e-mail suffix:

☒ Use SMTP Authentication

User Name: user2@ilearn.com

Password:

Use SSL: ☐

SMTP Port: 25

Reply-To Address:

Charset: UTF-8

☐ Test configuration by sending test e-mail

Save Apply

From : EMail Id and password.

Figure 3: E-mail Configuration

Step 2: Test Email configuration by checking “Test Configuration by sending test e-mail” field and type “Recipient Address” as shown below:

E-mail Notification

SMTP server: localhost

Default user e-mail suffix:

☒ Use SMTP Authentication

User Name: user2@ilearn.com

Password: ••••••

Use SSL: ☐

SMTP Port: 25

Reply-To Address:

Charset: UTF-8

☒ Test configuration by sending test e-mail

Test e-mail recipient: user1@ilearn.com

Email was successfully sent

Test configuration

Save Apply

Figure 4: Testing E-mail configuration

Step 3: Test Email will be sent successfully, if the configuration is done correctly.

Lab 3. Build Job creation in Jenkins

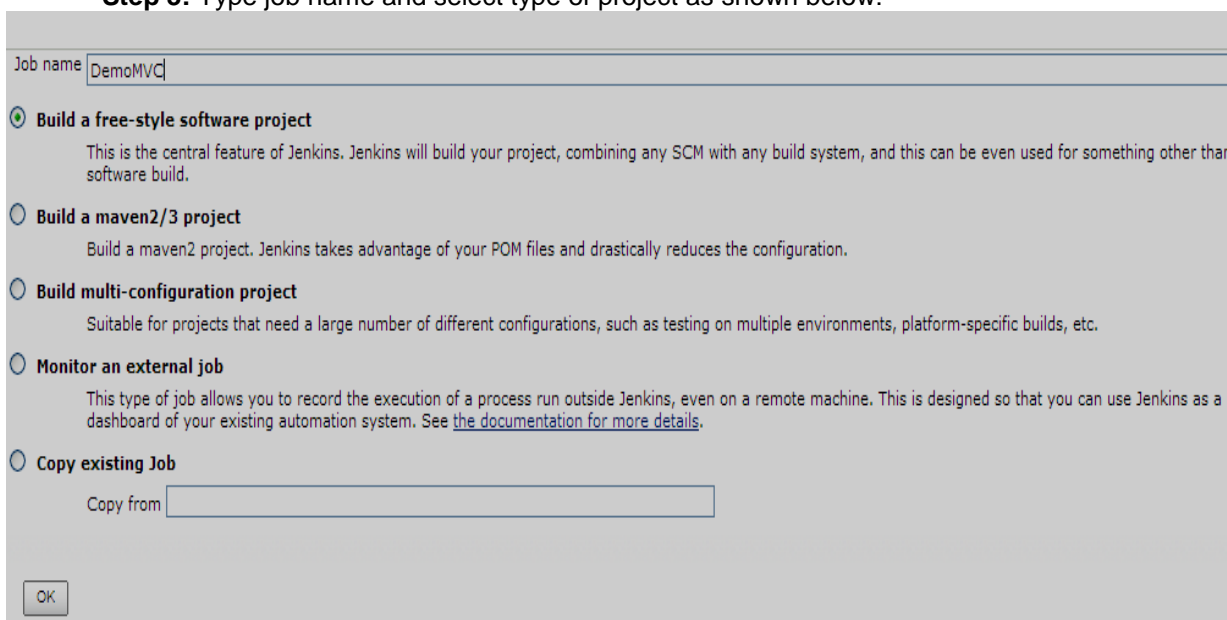
Goals	At the end of this lab session, you will be able to understand: <ul style="list-style-type: none">• How to create build job and configure build job• Scheduling build job• Execution of project in jenkins
Time	120 minutes

3.1: Creating build job in Jenkins

Step 1: Open Jenkins Dashboard by requesting the URL <http://localhost:8080>

Step 2: Click “New Job” link.

Step 3: Type job name and select type of project as shown below:



Job name

☒ **Build a free-style software 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.

☐ **Build a maven2/3 project**
Build a maven2 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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

☐ **Monitor an 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](#).

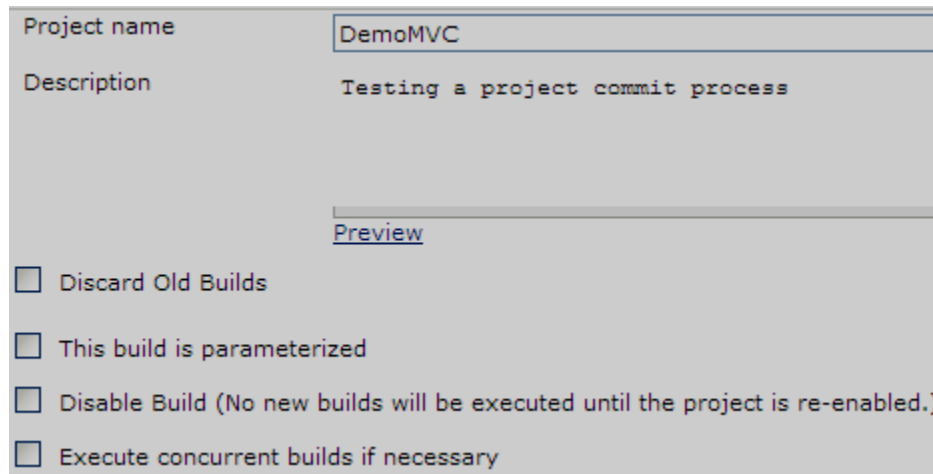
☐ **Copy existing Job**
Copy from

Figure 5: Build job Creation

Step 4: Click ok

3.2: Configure build job in Jenkins

Step 1: Type the description of the project in the “Project Configuration” page (Which will be displayed once build job is created at first time otherwise manually this page should be opened) as shown below:



Project name	DemoMVC
Description	Testing a project commit process
Preview	
<input type="checkbox"/>	Discard Old Builds
<input type="checkbox"/>	This build is parameterized
<input type="checkbox"/>	Disable Build (No new builds will be executed until the project is re-enabled.)
<input type="checkbox"/>	Execute concurrent builds if necessary

Figure 6: Project Configuration

3.3: Configure SCM

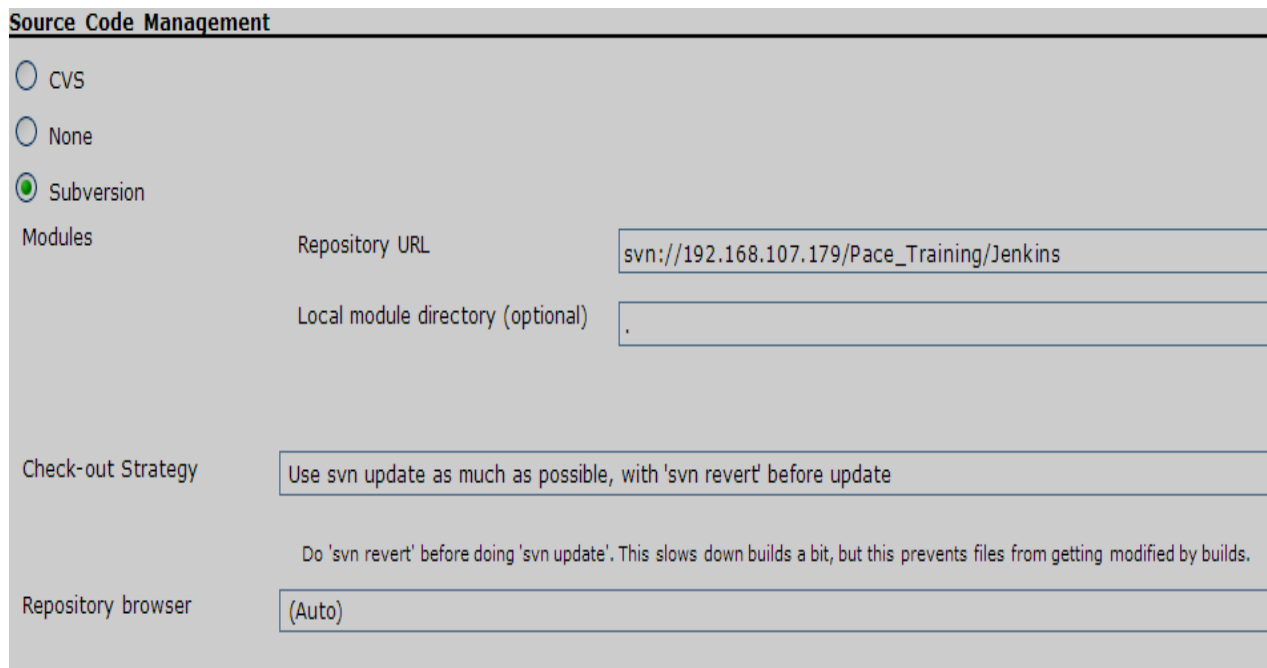
Step 1: Open “Project Configuration” page

Step 2: Select “Subversion” in SCM section

Step 3: Specify Repository URL

Step 4: Type “.” in Local module directory field to avoid creation of new directory for source in Local workspace as shown below:

Step 5: Select Check-out strategy as “Use ‘svn update’ as much as possible, with ‘svn revert’ before update” (which will systematically run svn revert before running svn update)



Source Code Management	
<input type="radio"/> CVS	
<input type="radio"/> None	
<input checked="" type="radio"/> Subversion	
Modules	Repository URL
	svn://192.168.107.179/Pace_Training/Jenkins
	Local module directory (optional)
	.
Check-out Strategy	Use svn update as much as possible, with 'svn revert' before update
	Do 'svn revert' before doing 'svn update'. This slows down builds a bit, but this prevents files from getting modified by builds.
Repository browser	(Auto)

Figure 7: SCM Configuration

3.3: Scheduling job execution

Step 1: Open “Project Configuration” page

Step 2: Select “Poll SCM” in Build Triggers section and type * * * * * in the schedule as shown below: (which polls SCM for every change commits and triggers build execution)

Build Triggers

☐ Build after other projects are built

☐ Trigger builds remotely (e.g., from scripts)

☐ Build periodically

☒ Poll SCM

Schedule * * * * *

Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "0 * * * *"

Ignore post-commit hooks ☐

Figure 8: Build scheduling



To build the job periodically, check Build periodically field and type relevant expression in schedule field. For an Example, to build job every day at 8 hours, the expression should be “0 **8** * * *”

Cron expression should be created to schedule the job trigger.

* * * * *

1st Field - MINUTES Minutes in one hour (0-59)

2nd Field - HOURS Hours in one day (0-23)

3rd Field - DAYMONTH Day in a month (1-31)

4th Field - MONTH Month in a year (1-12)

5th Field - DAYWEEK Day of the week (0-7) where 0 and 7 are Sunday

To schedule job every minute, use cron expression * * * * *

3.4: Invoke Ant

Step 1: Once SCM is configured, choose “Invoke Ant option” from “Add build step” drop down list in build section as shown below:

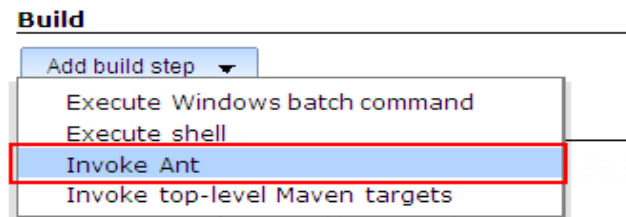


Figure 9: Invoking Ant

Step 2: Specify the Ant version and type target name (which need to be executed from your build targets) in Targets field as shown below:

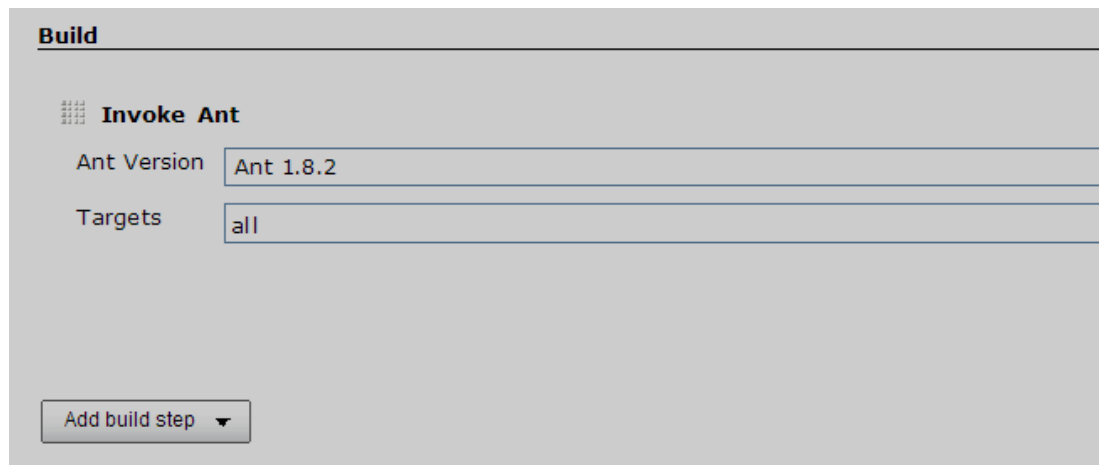


Figure 10: Selecting target in build

Step 3: Click on save

Step 4: Schedule the build to be executed immediately and view the console output as shown below:

```
Started by user anonymous
Building in workspace C:\Jenkins\.jenkins\jobs\DemoMVC\workspace
Reverting C:\Jenkins\.jenkins\jobs\DemoMVC\workspace\
Updating svn://192.168.107.179/Pace_Training/Jenkins
At revision 1556
[workspace] $ cmd.exe /C ""C:\Material-iGATE Format\Developer Workbench\Demos-ANT\apache-ant-1.8.2-bin\apa
Buildfile: C:\Jenkins\.jenkins\jobs\DemoMVC\workspace\build.xml

init:

compile:
[javac] C:\Jenkins\.jenkins\jobs\DemoMVC\workspace\build.xml:16: warning: 'includeantruntime' was not s

war:

BUILD SUCCESSFUL
Total time: 0 seconds
Finished: SUCCESS
```

Figure 11: Build execution output

<<TODO>>

Create a new project in Jenkins for an application, which should build the project automatically after every 24 hours.

Lab 4. Automated deployment and continuous delivery

Goals	At the end of this lab session, you will be able to: <ul style="list-style-type: none">Automate the deployment of an application to a tomcat server
Time	60 minutes

4.1: Installation of Deploy plug-in

Step 1: Open Jenkins Dashboard by requesting the URL <http://localhost:8080>

Step 2: Click “Manage Jenkins” link.

Step 3: Click on “Manage Plugins” Link

Step 4: Select “Available” tabs, check “Deploy to container Plugin” and click on Download and install button.

Alternate Approach:

Step 1: Download deploy.hpi file

Step 2: Follow steps 1 to 3 given in Installation of deploy plug-in

Step 2: Install plug-in manually by selecting “Advanced tab” and upload plug-in as shown below:

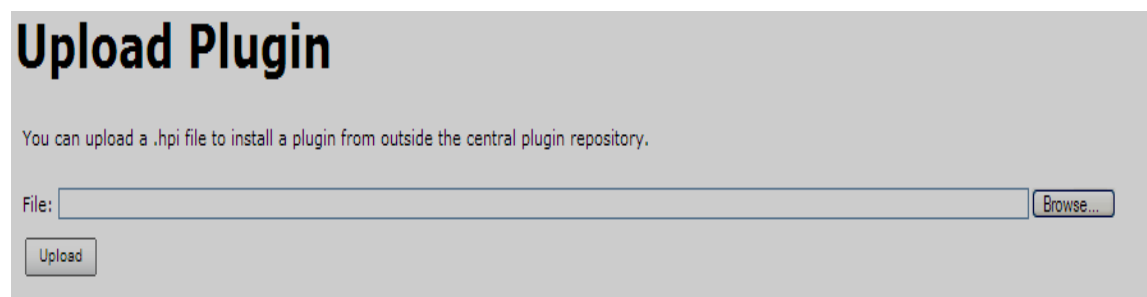


Figure 12: Install plug-in manually

4.2: Automate deployment of an application

Step 1: Open Project (build job) configuration.

Step 2: In Post-build Actions, choose “Deploy war/ear to a container” from “Add post-build action” drop down list

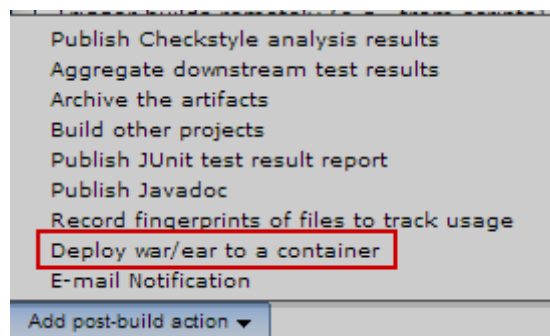


Figure 13: Deploying war file to tomcat server

Step 3: Configure Tomcat server details as shown below:

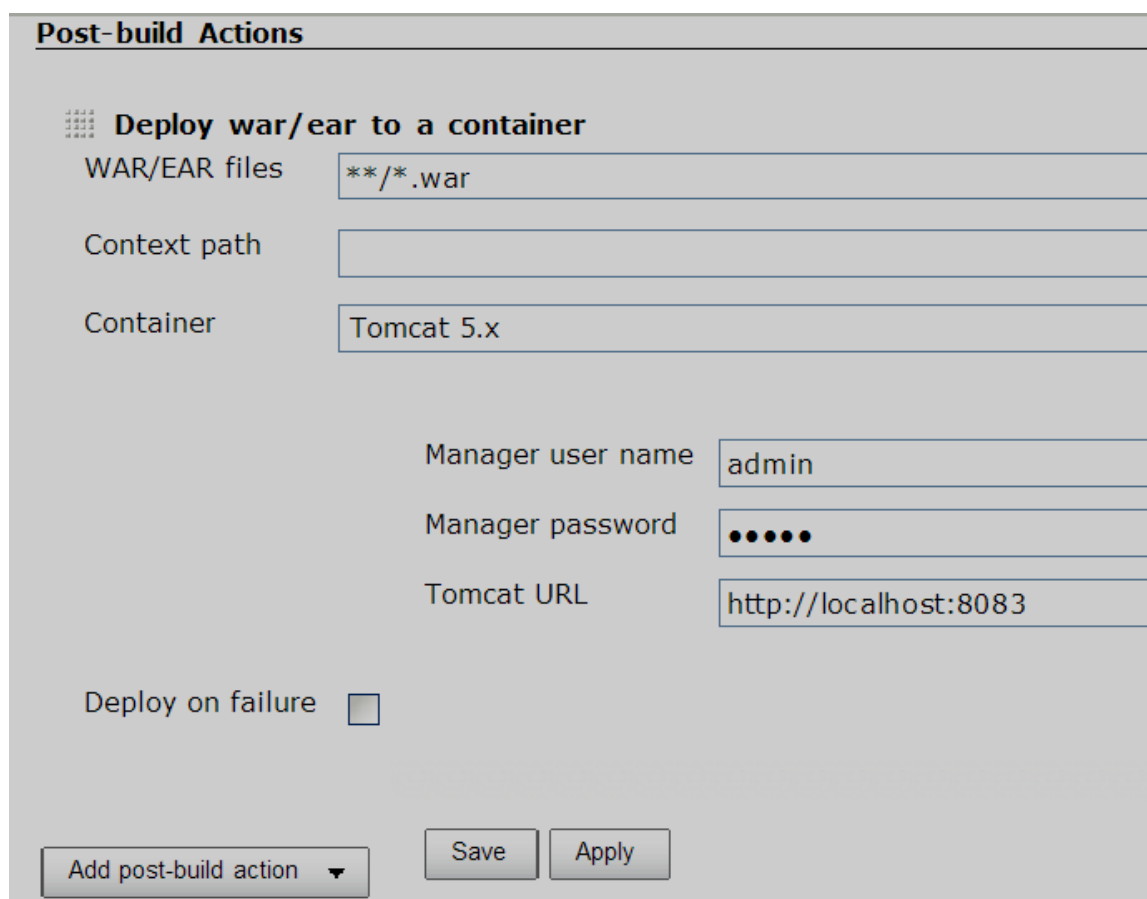


Figure 14: Deployment server configuration details

Step 4: Check Deploy on failure field, in order to send feedback to the developers once build/deployment is failed.

Step 5: choose “E-mail Notification” from “Add post-build action” drop down list as shown below:

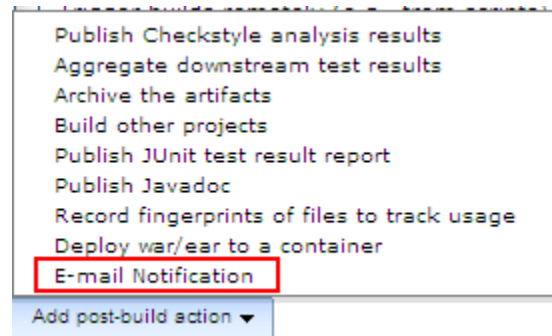


Figure 15: E-mail notification

Step 6: Type recipient mail id and check “Send e-mail for every unstable build.”

Step 7: Click on save and schedule the job execution immediately.

<<TODO>>

Create a new project in Jenkins for an application, which should automate the deployment of project into tomcat server.

Lab 5. Automating Testing using Jenkins

Goals	At the end of this lab session, you will be able to: <ul style="list-style-type: none">Automate the unit testing of an application
Time	60 minutes

5.1: Automate unit testing of an application

Step 1: In application build.xml file, ensure target is included to invoke unit test class execution.

Step 2: When build file is executed, automatically test class execution is also invoked.

Step 2: Open Project (build job) configuration.

Step 3: In Post-build Actions, choose “Publish JUnit test result report” from “Add post-build action” drop down list to display test results.

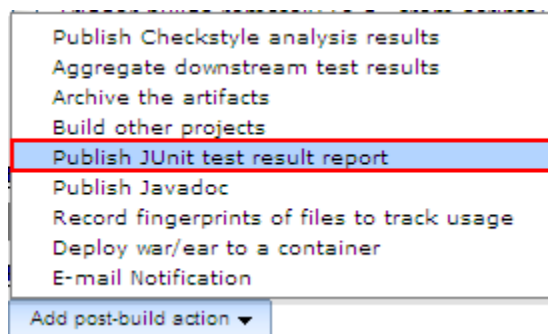


Figure 16: JUnit test result report

Step 4: To generate and publish JUnit test result report, configure report generation as shown below:



Figure 17: Publishing JUnit test report

Step 5: Once test is build, “Test result Trend” will be displayed in the Project home page as shown below:

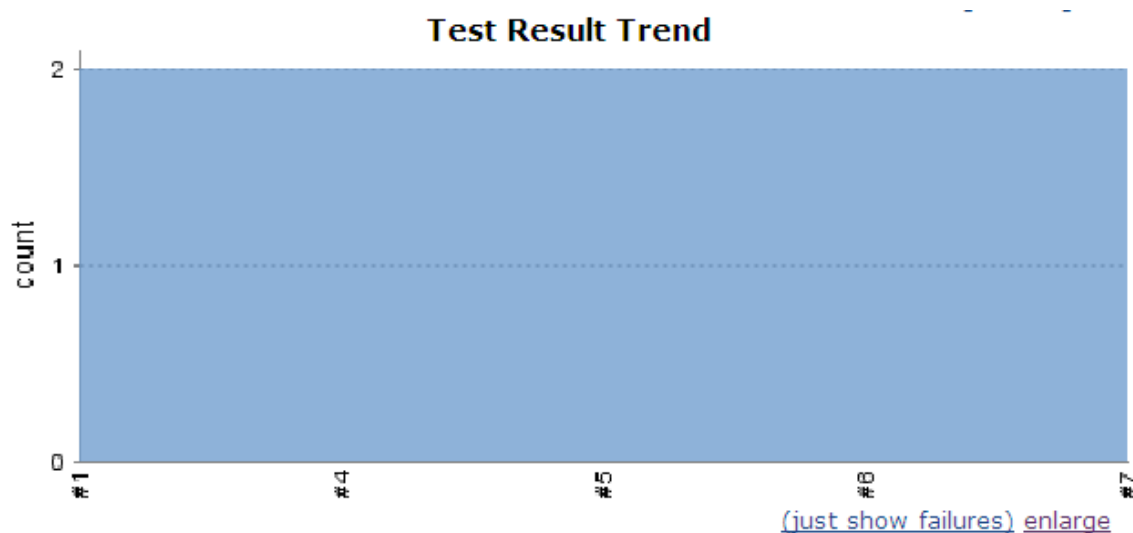


Figure 18: Test Result Trend

Alternate Approach:

Step 5: Once test is build, Click on "Latest Test Result" link in the Project home page as shown below. Detailed test result will be displayed with duration, number of tests failed, etc...

Project JenkinsJUnitAnt

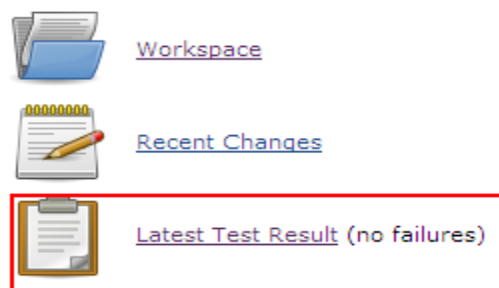


Figure 19: Test Result

<<TODO>>

Create a new project in Jenkins for an application, which should automate unit testing of an application.

Lab 6. Securing Jenkins

Goals	At the end of this lab session, you will be able to: <ul style="list-style-type: none"> • Enable more security in Jenkins
Time	60 minutes

6.1: Activating Securing in Jenkins

Step 1: Click “Configure Global Security” link in “Manage Jenkins” screen.

Step 2: Check “Enable Security” Field

Step 3: Click on “Jenkins own user database” and select “Allow users to sign up” field under Access Control (To authenticate users based on existing credentials in Jenkins own database and to support new users sign up) as shown below:

The screenshot shows the Jenkins Global Security Configuration interface. On the left, a sidebar lists sections: 'Enable security' (checked), 'TCP port for JNLP slave agents', 'Markup Formatter', and 'Access Control'. The main area shows the 'Access Control' configuration. Under 'Security Realm', 'Jenkins's own user database' is selected with a radio button, and 'Allow users to sign up' is checked with a checkbox. Other options like 'Delegate to servlet container', 'LDAP', and 'Unix user/group database' are unselected. Under 'Authorization', 'Logged-in users can do anything' is selected with a radio button, while 'Anyone can do anything', 'Legacy mode', 'Matrix-based security', and 'Project-based Matrix Authorization Strategy' are unselected. Each option has a help icon (question mark) to its right.

Figure 20: Activating security in Jenkins

6.2: Creating new users in Jenkins

Step 1: Open Jenkins Dashboard

Step 2: Click “sign up” link to create new user.

Step 3: Fill the user’s details and click on Sign up.

Sign up

Username:	<input type="text" value="eDude"/>
Password:	<input type="password" value="•••••"/>
Confirm password:	<input type="password" value="•••••"/>
Full name:	<input type="text" value="eDude"/>
E-mail address:	<input type="text" value="edude@igate.com"/>
<input type="button" value="Sign up"/>	

Figure 21: Creating new User

Step 4: View list of created users details by clicking on “Manage Users” link in “Manage Jenkins” screen.









Users		
These users can log into Jenkins. This is a sub set of this list , which also contains auto-created users who really just made some commits on some access.		
User Id	Name	
 admin	Administrator	 
 eDude	eDude	 
 ilearn	ilearn	

Figure 22: List of existing users

6.3: Authorizing users in Jenkins


Step 1: Click “Configure Global Security” link in “Manage Jenkins” screen.

Step 2: Click “Matrix-based security” under Authorization to authorize the users based on username and assign permissions as shown below.

(As default, Logged-in users can do anything is selected under Authorization).

Authorization

☐ Anyone can do anything
☐ Legacy mode
☐ Logged-in users can do anything
☒ Matrix-based security

User/group	Overall				Slave				Job				Run	View		SCM									
	Administer	Read	RunScripts	Configure	Configure	Delete	Create	Disconnect	Connect	Create	Delete	Configure	Read	Discover	Build	Workspace	Cancel	Delete	Update	Create	Delete	Configure	Read	Tag	
 ilearn	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Anonymous	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

User/group to add:

☐ Project-based Matrix Authorization Strategy

Figure 23: Authorizing Users

Step 3: Click “Project-based Matrix Authorization Strategy” under Authorization to authorize the users based on projects.

<<TODO>>

1. Create a new user in Jenkins. Authenticate and authorize new user to schedule a specific job execution. Other users should be allowed to view the job.
2. Implement authorization based on project-matrix.

Lab 7. Code Quality

Goals	At the end of this lab session, you will be able to: <ul style="list-style-type: none"> • Ensure quality of the code.
Time	60 minutes

7.1: Ensuring code quality in Jenkins

Step 1: Install Checkstyle, PMD, and FindBugs plugins.

Step 2: Open Project Configuration screen.

Step 3: In Post-build Actions, choose “Publish Checkstyle analysis report” from “Add post-build action” drop down list as shown below.

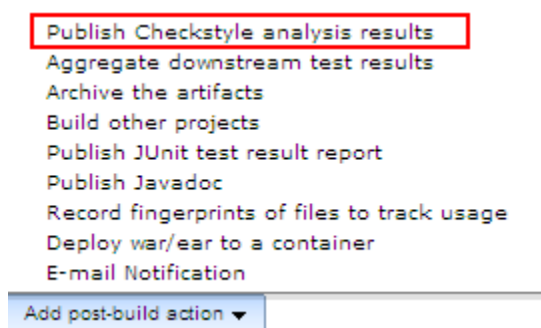


Figure 24: Checkstyle configuration

Step 3: To generate and publish Checkstyle analysis result, configure report generation as shown below:

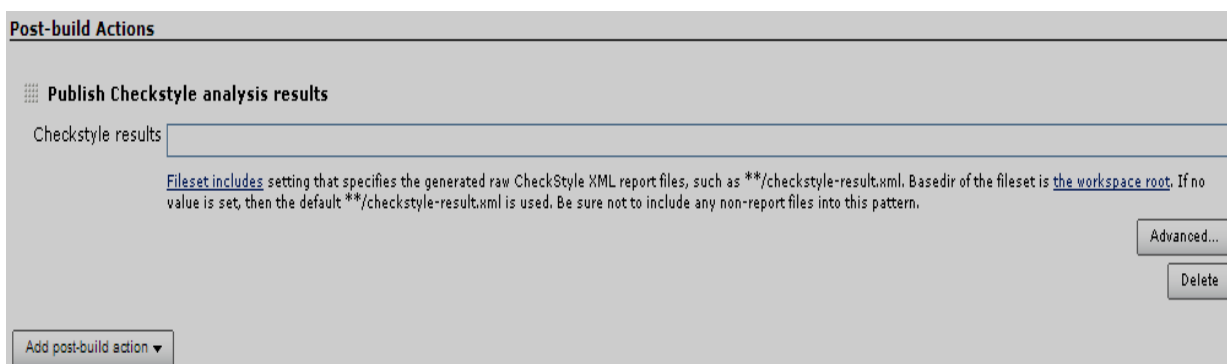


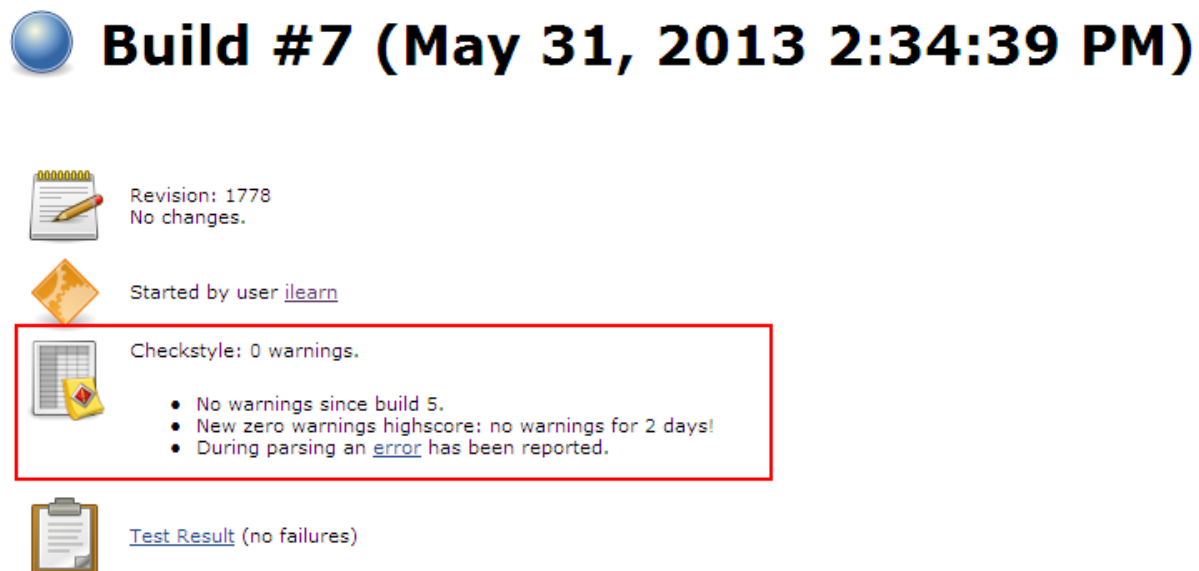
Figure 25: Checkstyle result configuration

(As default, `**/Checkstyle-result.xml` used as value in Checkstyle results field.)

Step 4: Once job is build, “Checkstyle Trend” will be displayed in the Project home page

Alternate Approach:

Step 4: Once job is executed, the following output displays in build home page.



The screenshot shows the Jenkins build home page for 'Build #7 (May 31, 2013 2:34:39 PM)'. The page includes several status indicators: a 'Revision: 1778' with 'No changes.'; a 'Started by user [jlearn](#)'; and a 'Checkstyle: 0 warnings.' section highlighted with a red border. This section contains a list of messages: 'No warnings since build 5.', 'New zero warnings highscore: no warnings for 2 days!', and 'During parsing an [error](#) has been reported.' Below this, there is a 'Test Result (no failures)' link.

Build #7 (May 31, 2013 2:34:39 PM)

Revision: 1778
No changes.

Started by user [jlearn](#)

Checkstyle: 0 warnings.

- No warnings since build 5.
- New zero warnings highscore: no warnings for 2 days!
- During parsing an [error](#) has been reported.

[Test Result](#) (no failures)

Figure 26: Checkstyle result

<<TODO>>

1. Check the quality of the code by configuring checkstyle in existing job.

Appendices

Appendix A: Table of Figures

Figure 1: Jenkins Installation	6
Figure 2: Configure System.....	8
Figure 3: E-mail Configuration.....	9
Figure 4: Testing E-mail configuration	10
Figure 5: Build job Creation	11
Figure 6: Project Configuration	12
Figure 7: SCM Configuration	12
Figure 8: Build scheduling.....	13
Figure 9: Invoking Ant	14
Figure 10: Selecting target in build	14
Figure 11: Build execution output.....	15
Figure 12: Install plug-in manually	16
Figure 13: Deploying war file to tomcat server.....	17
Figure 14: Deployment server configuration details	17
Figure 15: E-mail notification	18
Figure 16: JUnit test result report	19
Figure 17: Publishing JUnit test report	19
Figure 18: Test Result Trend	20
Figure 19: Test Result	20
Figure 20: Activating security in Jenkins.....	21
Figure 21: Creating new User	22
Figure 22: List of existing users	22
Figure 23: Authorizing Users	23
Figure 24: Checkstyle configuration	24
Figure 25: Checkstyle result configuration	24
Figure 26: Checkstyle result	25