Guidelines – Jenkins and Sonar Build process

JSAG Suggests Series

Nagarro Software Pvt. Ltd.
Java Software & Architecture Group

Revision History			
Version	Date	Author/Contributor	Comments
0.1	23-Mar-2012	Amit Balwani	Compiled document.
0.1	27-Mar-2012	Archna Aggarwal	Review comments
1.0	29-Mar-2012	Amit Balwani	Baseline Release
1.1	19-Jun-2012	Amit Balwani	Updated Sonar configuration required for Ant.
1.2	28-Dec-2012	Vaibhav Shukla	Migrating from Hudson to Jenkins
1.3	28-Dec-2012	Kuldeep Singh	Reviewed and Updated
1.4	25-Jan-2013	Amit Balwani	Updated a note on SVN credentials to be updated regularly when changed in the domain

Contents

Contents 3
1. Introduction 4
2. Jenkins Setup 5
2.1. Build Job Creation 5
2.1.1. Create a new job 5
2.1.2. Add SVN location 7
2.1.3. Build triggers 8
2.1.4. Build Step using Maven 9
2.1.5. Post-build actions setup 10
2.2. Manual build execution 12
3. Sonar Setup for Jenkins 12
3.1. Sonar configuration in Jenkins project build Job 13
4. Accessing Build and Sonar reports in Jenkins 14
4.1. Accessing Build when built using Maven 14
4.2. Accessing Sonar reports 16

1. Introduction

The purpose of this document is to guide project teams to automate the build process for Java based projects and execute code analyzer tools (PMD, Checkstyle, Findbugs) on a pre-defined schedule. This is enabled through a Jenkins and Sonar installation on centralized JSAG server machines. Project teams can configure and run the build process through an easy to use web interface exposed by Jenkins and Sonar.

Jenkins is an award winning application that monitors executions of repeated jobs, such as building a software project or jobs run by cron. Among those things, current Jenkins focuses on the following two jobs:

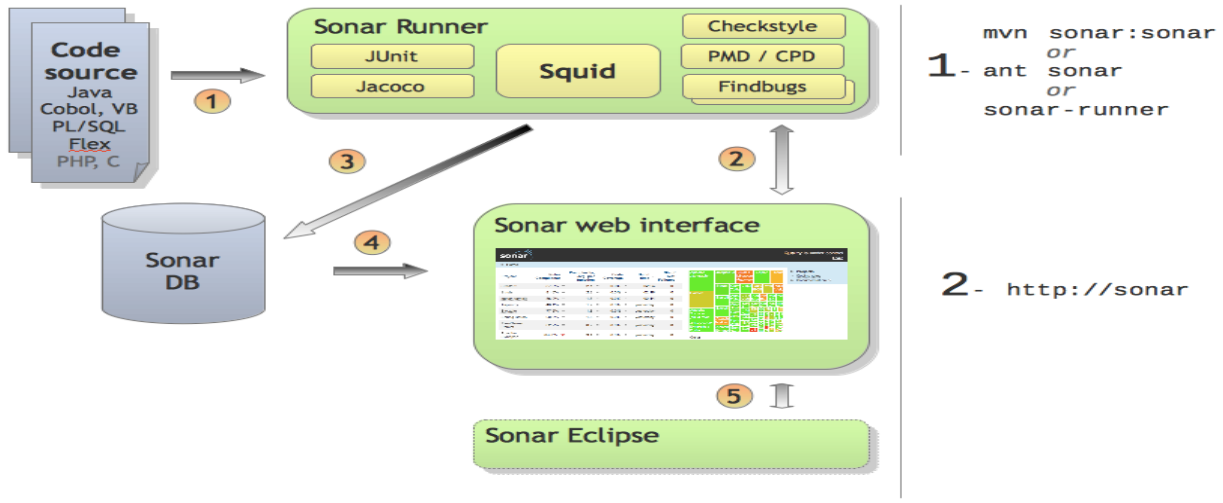
1. **Building/testing software projects continuously**, just like CruiseControl or DamageControl. In a nutshell, Jenkins provides an easy-to-use so-called continuous integration system, making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. The automated, continuous build increases the productivity.

2. **Monitoring executions of externally-run jobs**, such as cron jobs and procmail jobs, even those that are run on a remote machine. For example, with cron, all you receive is regular e-mails that capture the output, and it is up to you to look at them diligently and notice when it broke. Jenkins keeps those outputs and makes it easy for you to notice when something is wrong.

Sonar is an open platform to manage code quality. As such, it covers the 7 axes of code quality:

- Architecture and design
- Comments
- Coding rules
- Potential bugs
- Complexity
- Duplications
- Unit tests

Sonar architecture is explained in the image shown below:



Project teams can benefit from this automated build process and code analysis enabling them to ensure code quality and pre-scheduled builds without any hardware or software installation.

Please note that it is a mandatory requirement for all Java projects to setup their project's build on the JSAG Jenkins and Sonar server. If because of any infrastructure issue, it is not possible to setup build on the JSAG Jenkins and Sonar server, an approval should be taken by sending email to jsag@nagarro.com at the start of the project.

2. Jenkins Setup

Jenkins server web interface can be accessed through the URL http://jsag.ggn.nagarro.com/jenkins/

2.1. Build Job Creation

A build job needs to be setup for a project with project specific settings. Please follow the steps in the sections below to setup a job and configure the build on it.

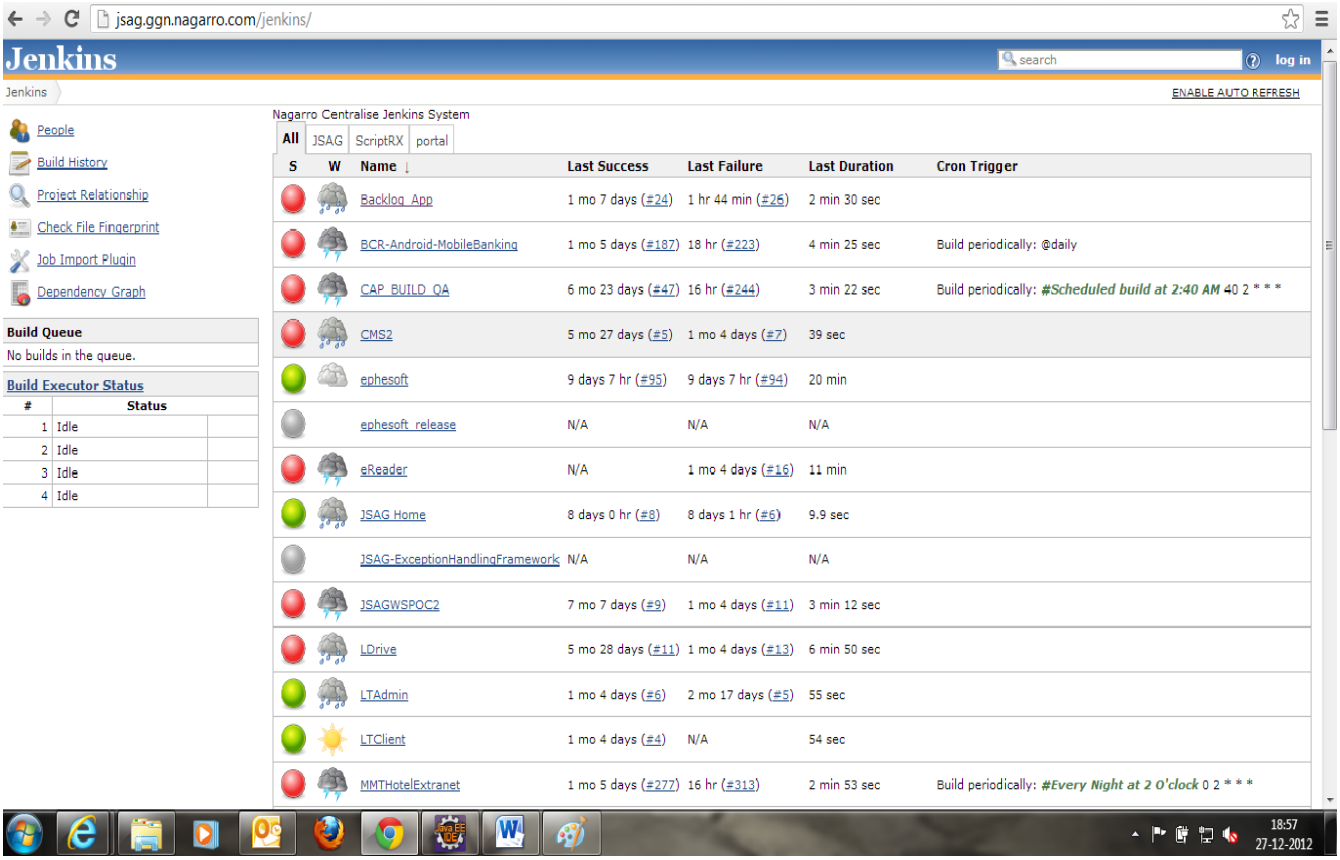
2.1.1. Create a new job

A new job can be created by sending an email to scm@nagarro.com with the following details:

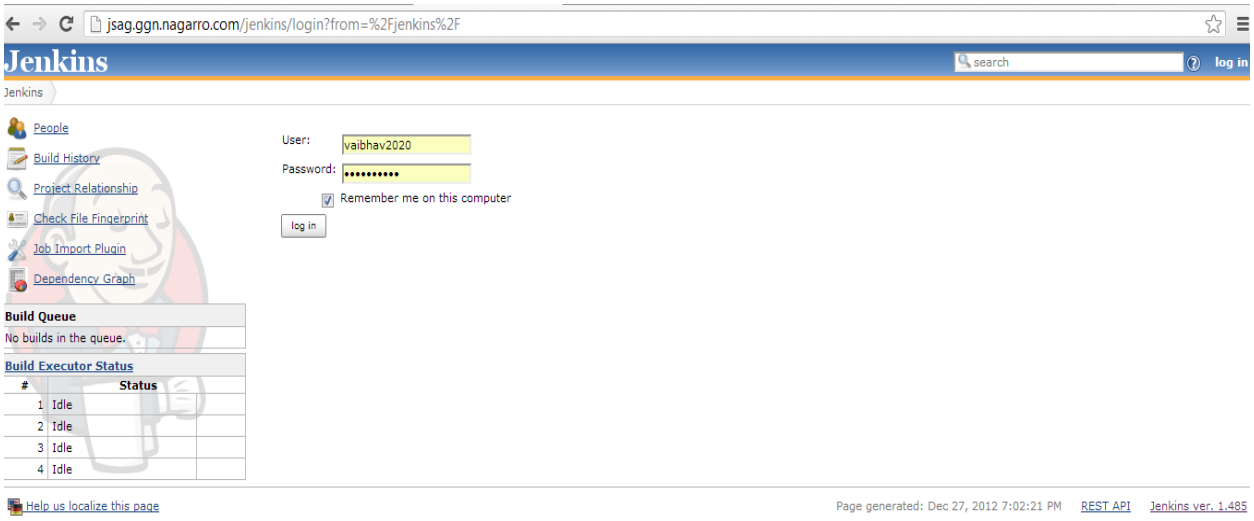
- SVN credentials – *Make sure you change it in the job regularly when you change it on the domain else the build will fail and give "SVN update failed" error.*
- SVN location
- Job name
- Job schedule

A new job can be created by following the steps below using the Jenkins UI interface as well.

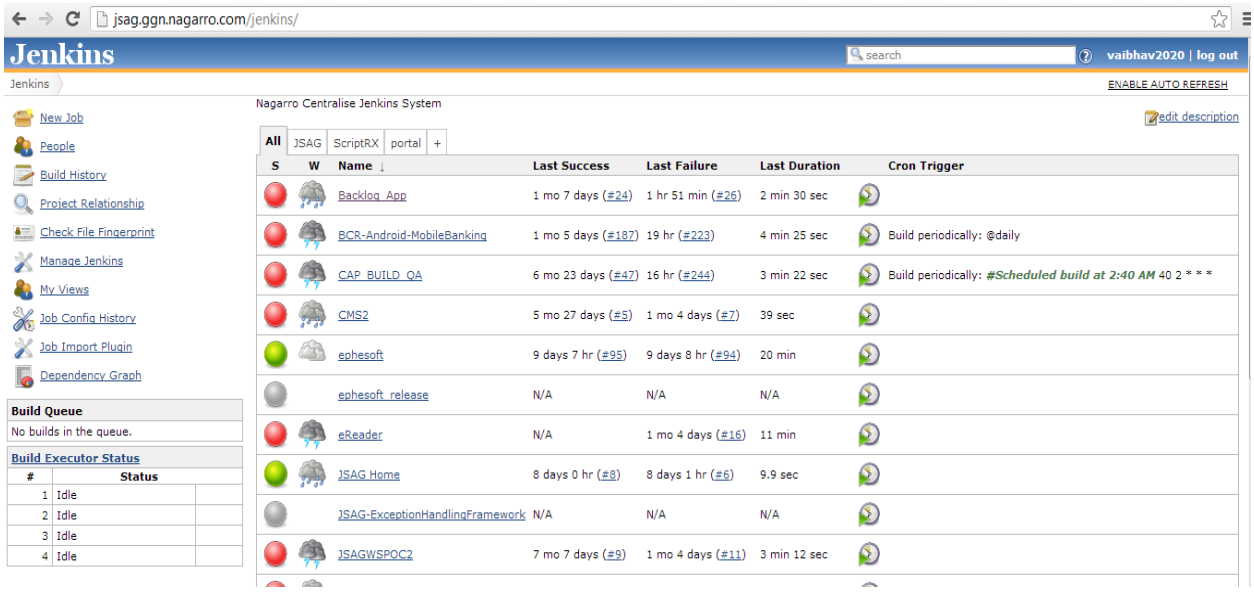
1. Go to the URL http://jsag.ggn.nagarro.com/jenkins/ using your browser, you will see the page shown below.



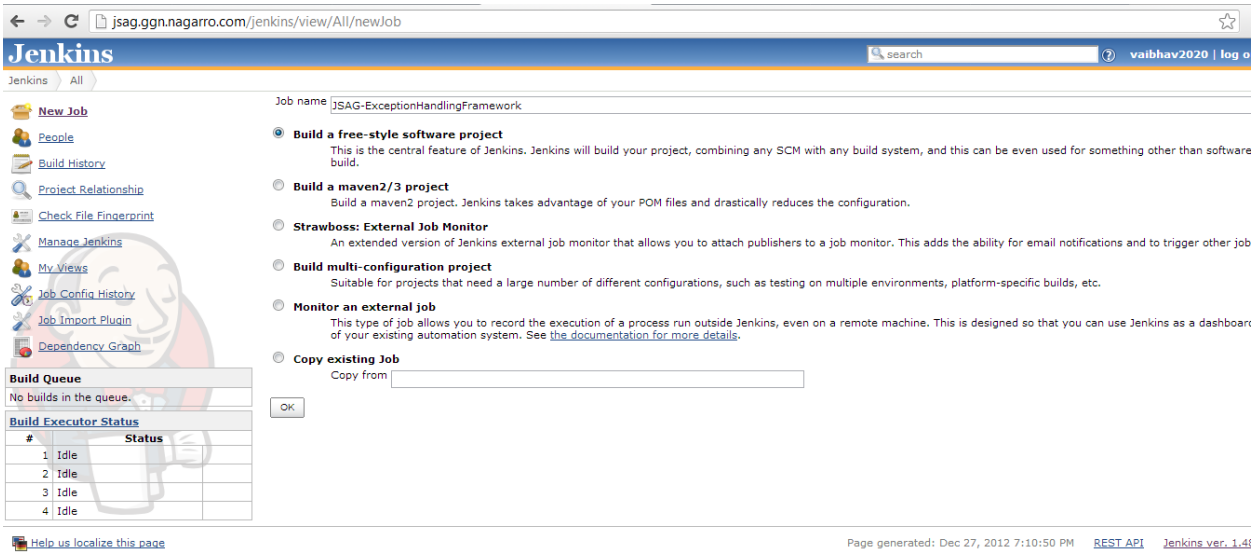
2. On top right, you'll find a "log in" button. Click and enter your domain credentials to login



3. After login you'll see page as below



4. On top left, you'll see a "New Job" button. Click on New job, and you will get the page shown below

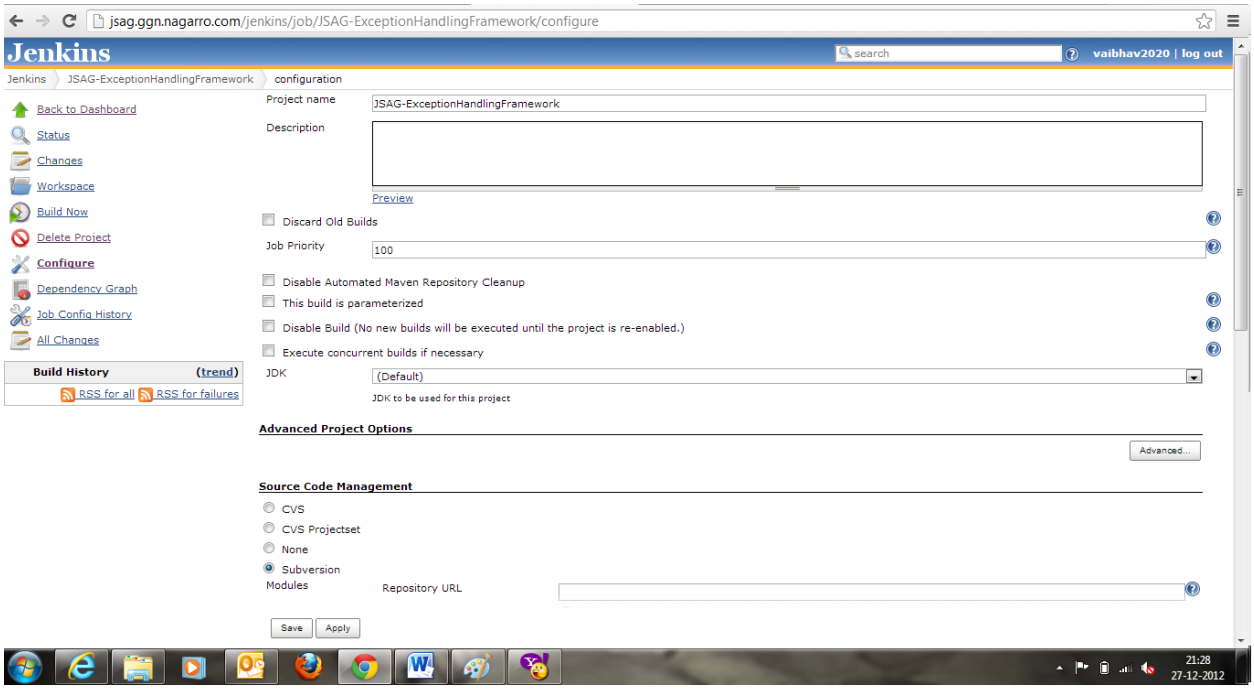


5. Enter the job name as shown in the image above
6. Select "Build a free-style software project" and click on OK button

2.1.2. Add SVN location

Please follow the steps below to configure SVN on Jenkins.

1. On clicking OK after entering the job name and type of project, the following page will come

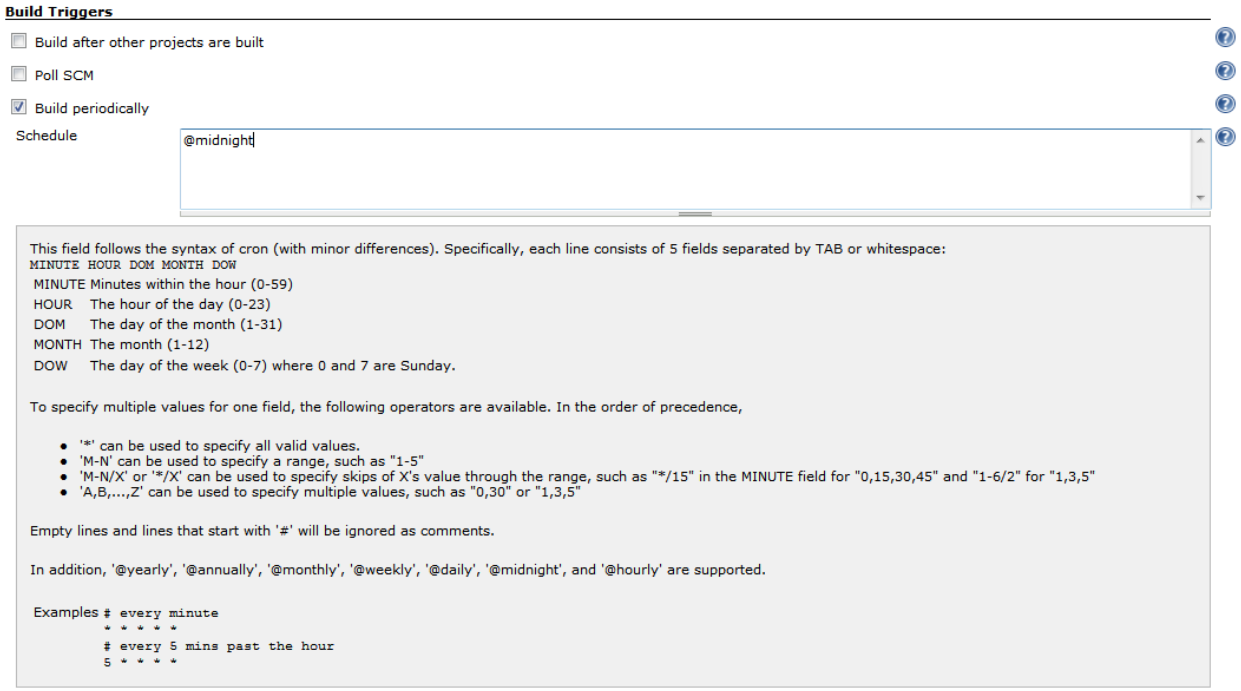


2. You can select discard builds option if you do not want to keep history on Jenkins server
3. Under section “Source code Management”, select subversion to enter the project root URL

2.1.3. Build triggers

Follow the steps below to configure Build triggers:

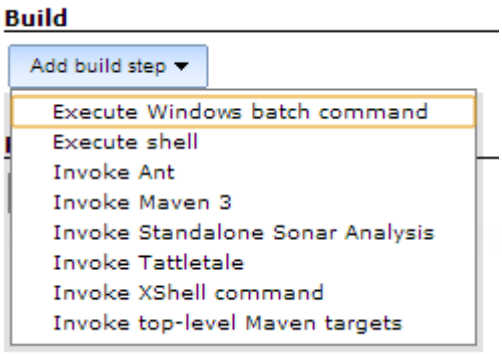
1. Select the “Build Periodically” option and enter the schedule in the cron syntax or short cut keys supported as shown in the screenshot below. You can click on the help icon shown besides each option to know more details about the option.



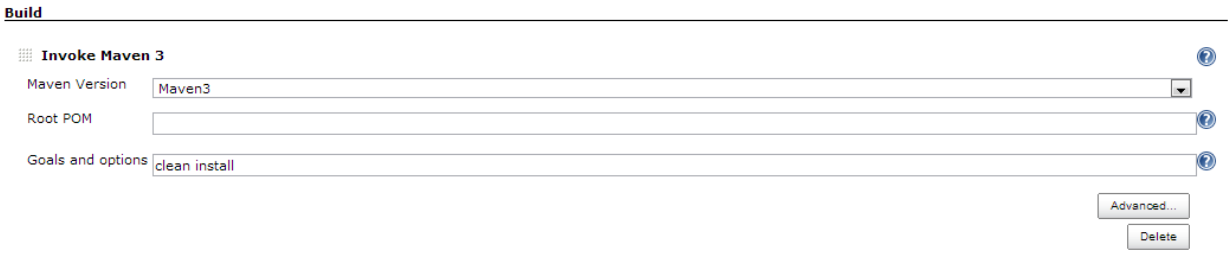
2.1.4. Build Step using Maven

Follow the steps given below to configure build steps when using Maven. Your project should have a pom.xml checked in your project's SVN under the project directory for Maven to build your project as per the dependencies and other configuration given in your projects' pom.xml file.

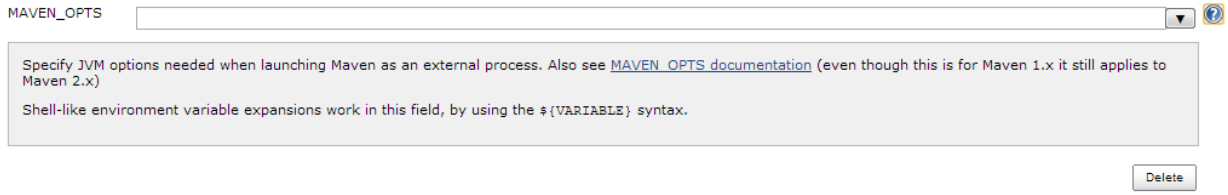
1. Under the "Build" section, click on the button "Add build step". An Options menu will be displayed as shown below:



2. Click on "Invoke Maven 3" to configure Maven settings. Settings shown below are for the "Invoke Maven 3" option.

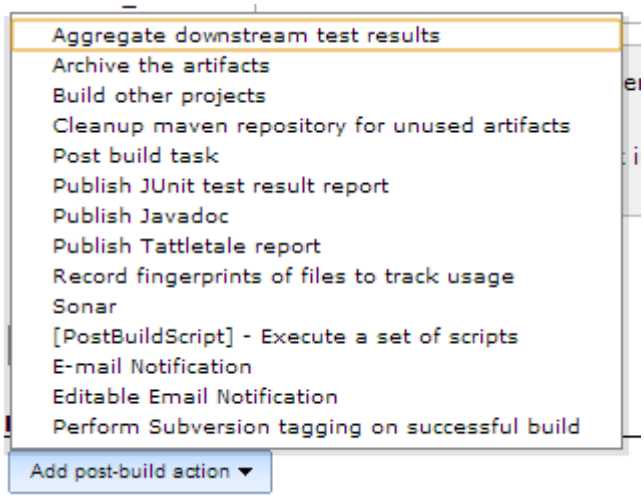


3. Enter “clean install” in the Goals field
4. Click on “Advanced” button to add MAVEN_OPTS entries. You can click on the help icon shown besides each option to know more details about the option.



2.1.5. Post-build actions setup

You can select the settings from “Post-build Actions” as per your project requirements, sample settings are show below:



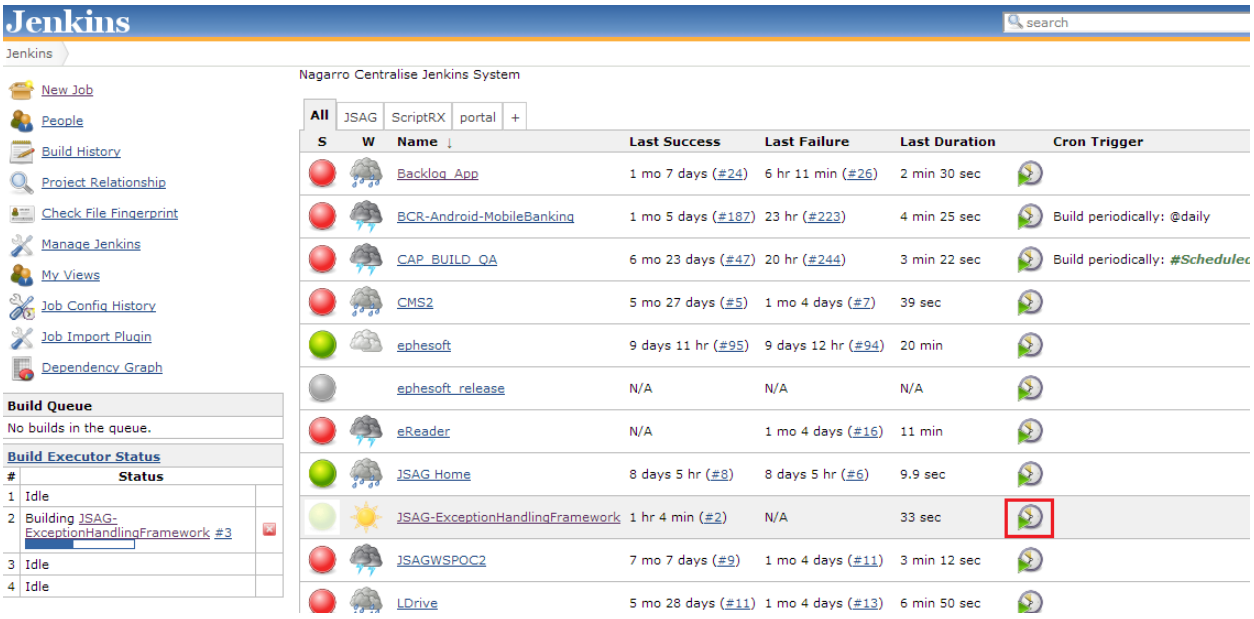
1. Click on “Save” button once all the settings have been selected.
2. Your job setup is complete

3. You can test your build process by clicking on the "Build Now" icon as shown below

4. You can view the progress of the build by selecting the console output as shown below

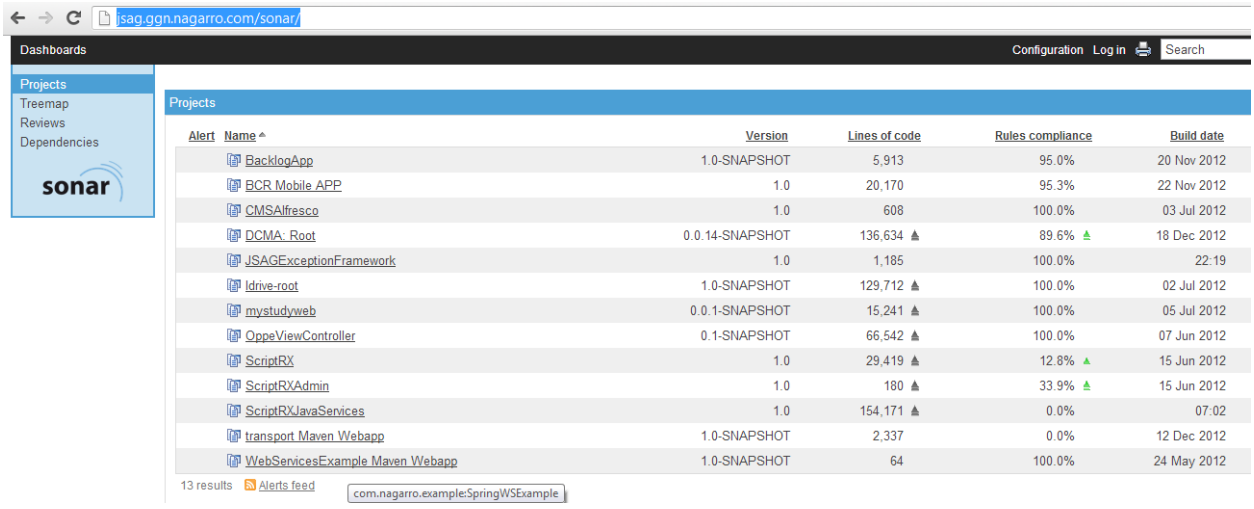
2.2. Manual build execution

A manual build can be executed by clicking on the build now link when inside the project link or by clicking the schedule build as shown below.



3. Sonar Setup for Jenkins

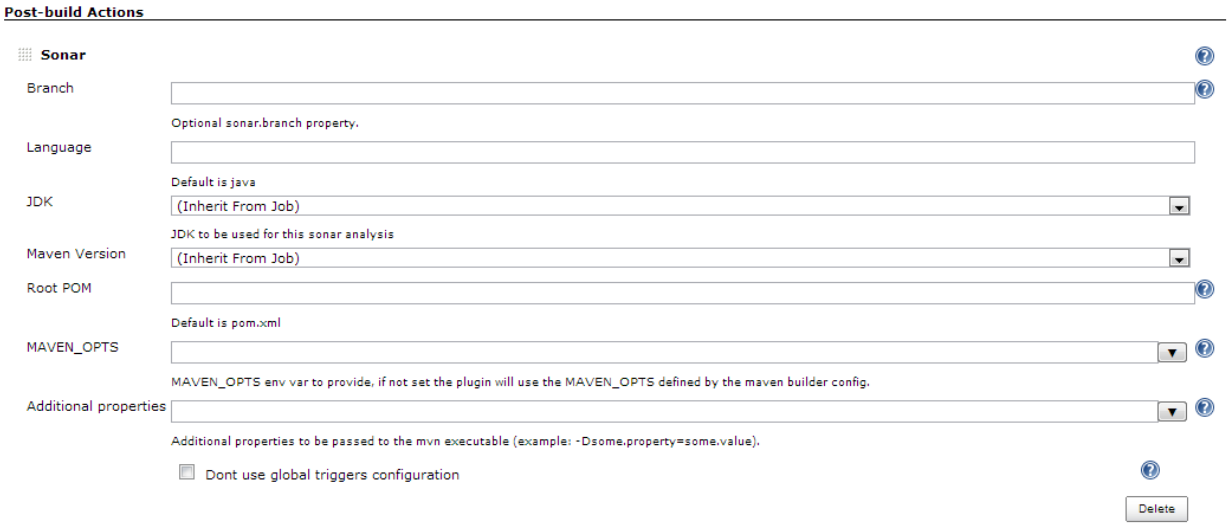
Sonar is installed on the JSAG server which can be accessed through the URL http://jsag.ggn.nagarro.com/sonar as shown below.



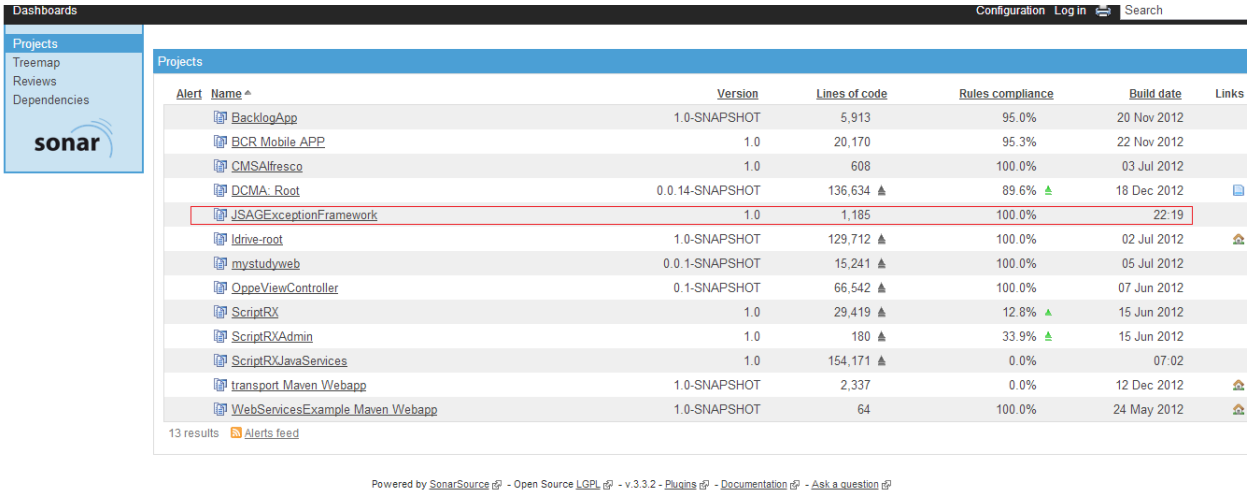
Follow the steps below to setup Sonar to be integrated with your automated project builds.

3.1. Sonar configuration in Jenkins project build Job

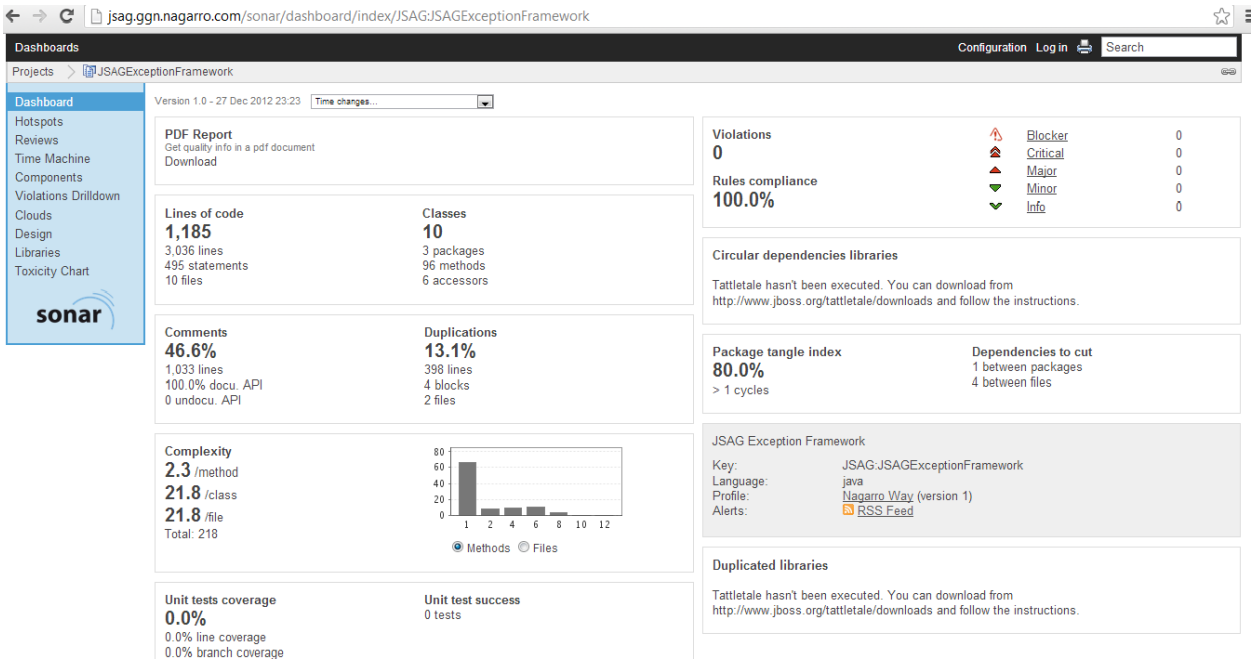
While setting up your project build, select the "Sonar" option in the post-build actions. If not already selected during the project build setup, you can edit the settings by selecting configure option from the Jenkins interface. The settings shown below can be selected for most of the requirements.



Once your project is successfully built with Sonar settings, it will start showing up in the Sonar project list. You can view the dashboard by clicking on the project name as shown below:



The dashboard will be available as shown below.

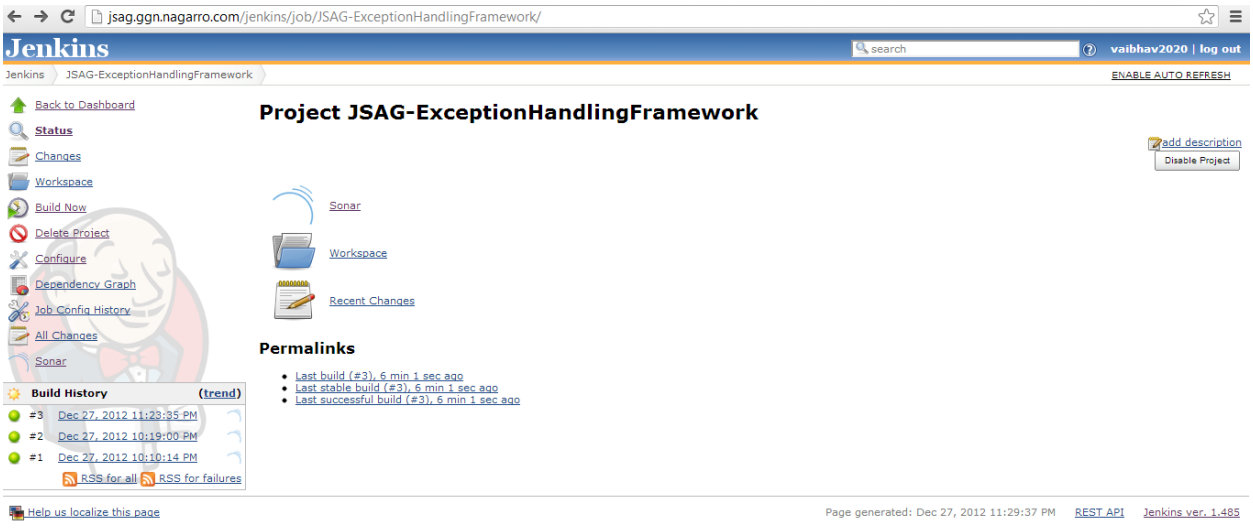


4. Accessing Build and Sonar reports in Jenkins

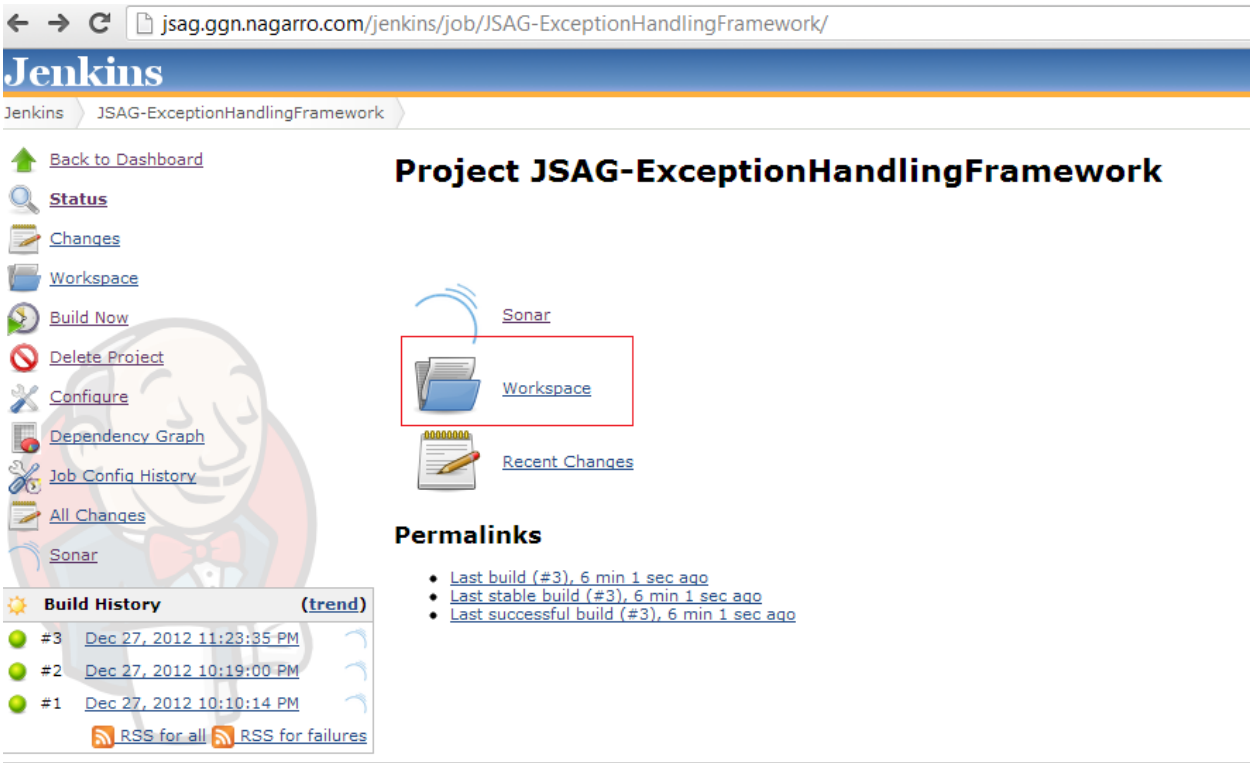
4.1. Accessing Build when built using Maven

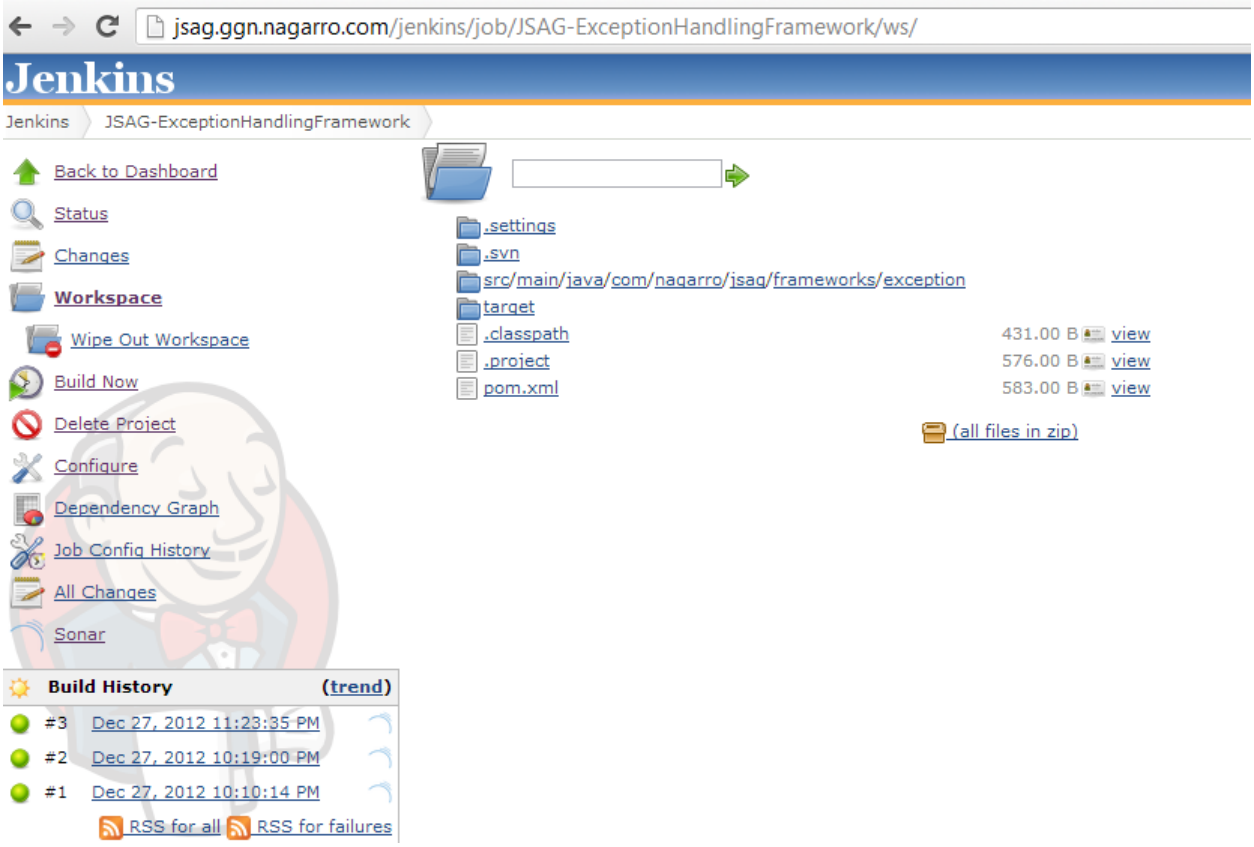
Please follow the steps below to access the build files for a project built using Maven.

1. After a successful build, the build files generated can be accessed by going to Jenkins URL http://jsag.ggn.nagarro.com/jenkins/, clicking on your project from the home page. The project's home page will look like the one shown below.

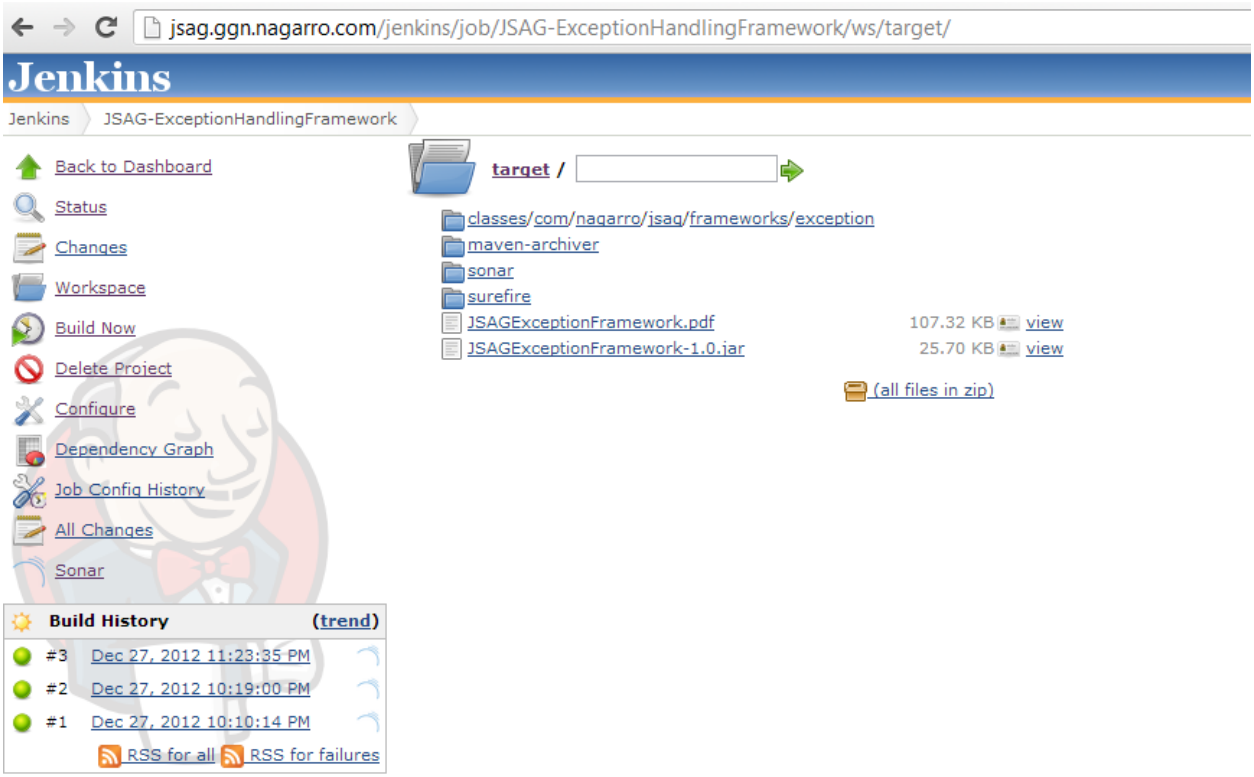


2. Click on the workspace link to enter the Jenkins workspace for the selected project as shown below.


jsag.ggn.nagarro.com/jenkins/job/JSAG-ExceptionHandlingFramework/
Jenkins
Jenkins JSAG-ExceptionHandlingFramework
Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Project
Configure
Dependency Graph
Job Config History
All Changes
Sonar
Build History (trend)
#3 Dec 27, 2012 11:23:35 PM
#2 Dec 27, 2012 10:19:00 PM
#1 Dec 27, 2012 10:10:14 PM
RSS for all RSS for failures
Project JSAG-ExceptionHandlingFramework
Sonar
Workspace
Recent Changes
Permalinks
Last build (#3), 6 min 1 sec ago
Last stable build (#3), 6 min 1 sec ago
Last successful build (#3), 6 min 1 sec ago

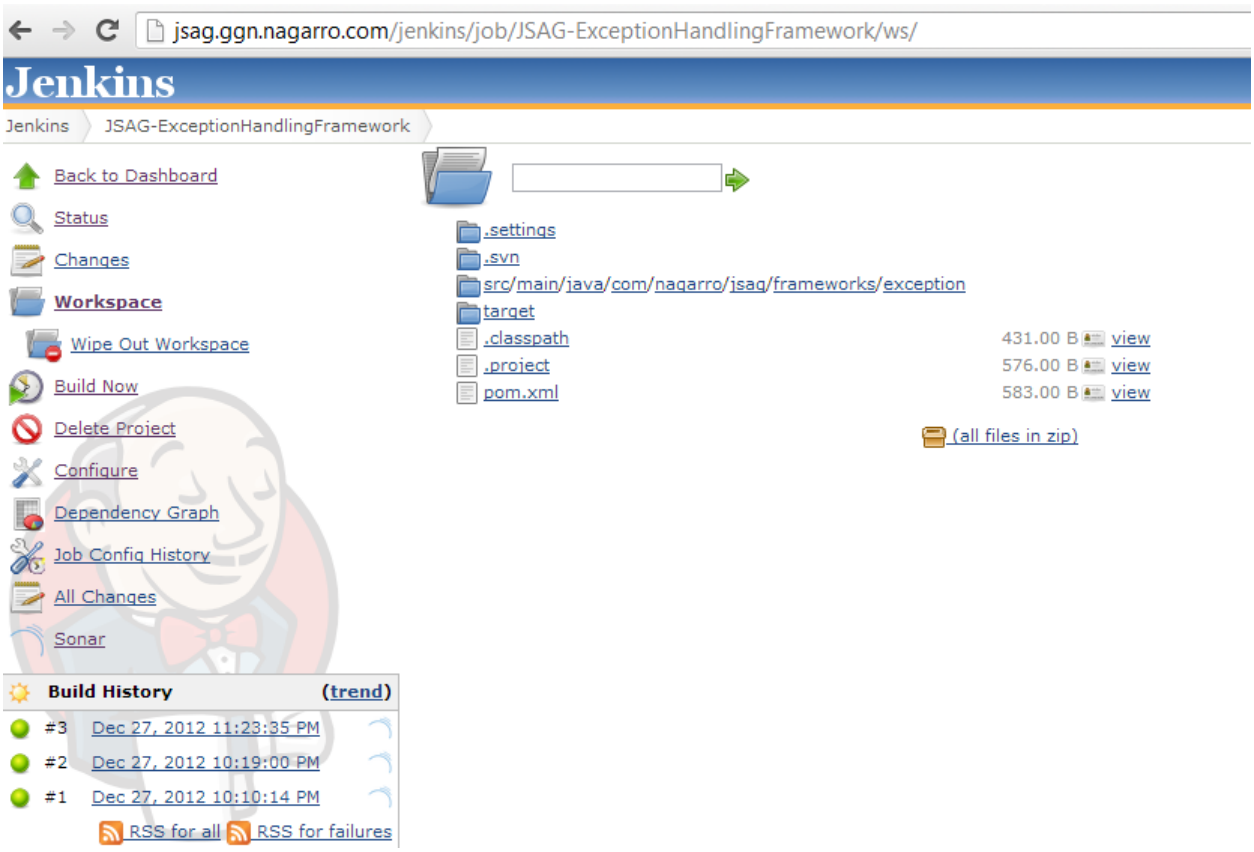

jsag.ggn.nagarro.com/jenkins/job/JSAG-ExceptionHandlingFramework/ws/
Jenkins
Jenkins JSAG-ExceptionHandlingFramework
Back to Dashboard
Status
Changes
Workspace
Wipe Out Workspace
Build Now
Delete Project
Configure
Dependency Graph
Job Config History
All Changes
Sonar
Build History (trend)
#3 Dec 27, 2012 11:23:35 PM
#2 Dec 27, 2012 10:19:00 PM
#1 Dec 27, 2012 10:10:14 PM
RSS for all RSS for failures
.settings
.svn
src/main/java/com/nagarro/jsag/frameworks/exception
target
.classpath 431.00 B view
.project 576.00 B view
pom.xml 583.00 B view
(all files in zip)

3. If built using Maven, the build files will be created inside the target folder as shown below. The files can be downloaded by doing right click → save target as or by clicking (all files in zip) which will download a zip file containing all the files being shown in the current view.

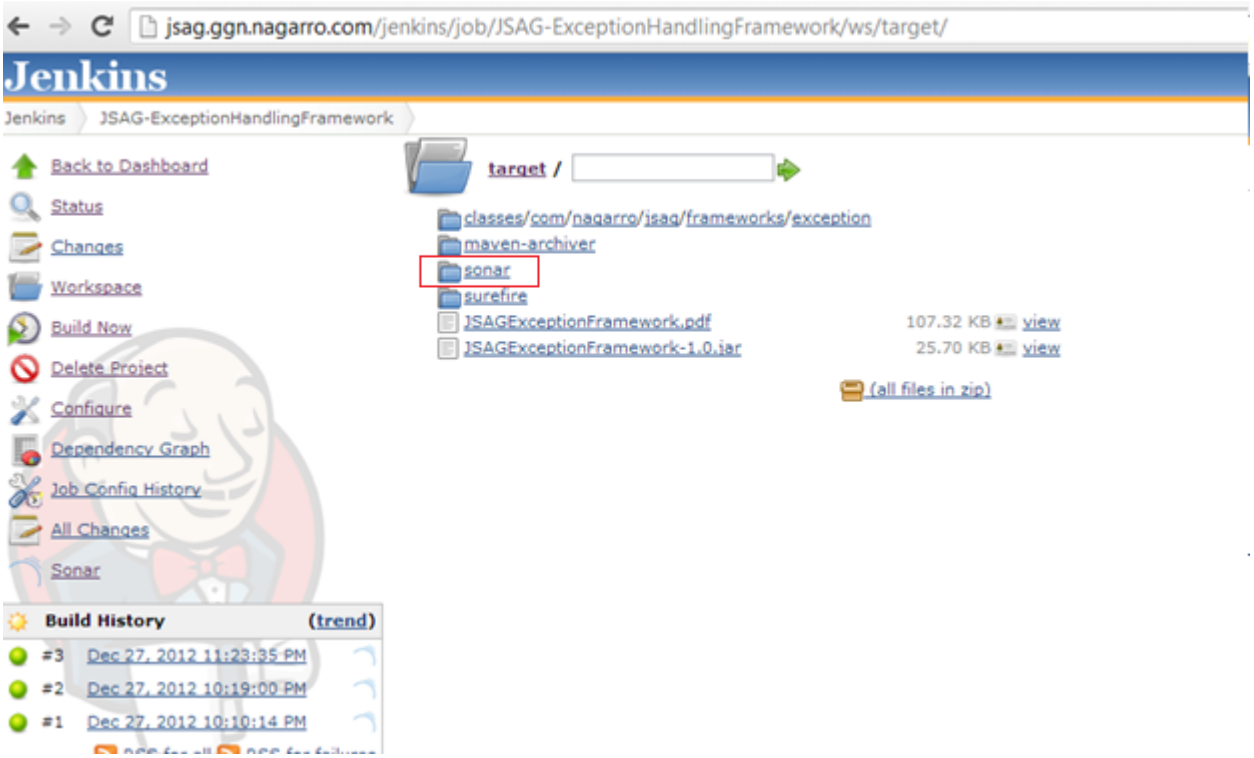


4.2. Accessing Sonar reports

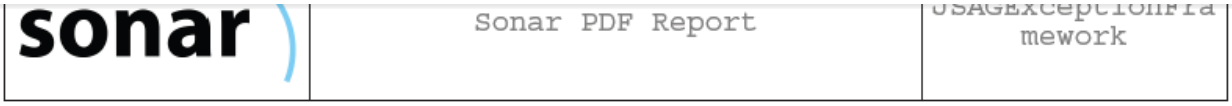
Sonar reports can be accessed by clicking on workspace from the Project home page on Jenkins as shown below.



Click on the target folder to see the files and folders inside it as shown below. Sonar creates a pdf report which is shown highlighted below (bcrmobile.pdf).



The report gives details of the code analysis performed by Sonar as shown below:



1. JSAGExceptionFramework

This chapter presents an overview of the project measures. This dashboard shows the most important measures related to project quality, and it provides a good starting point for identifying problems in source code.

1.1. Report Overview

Static Analysis

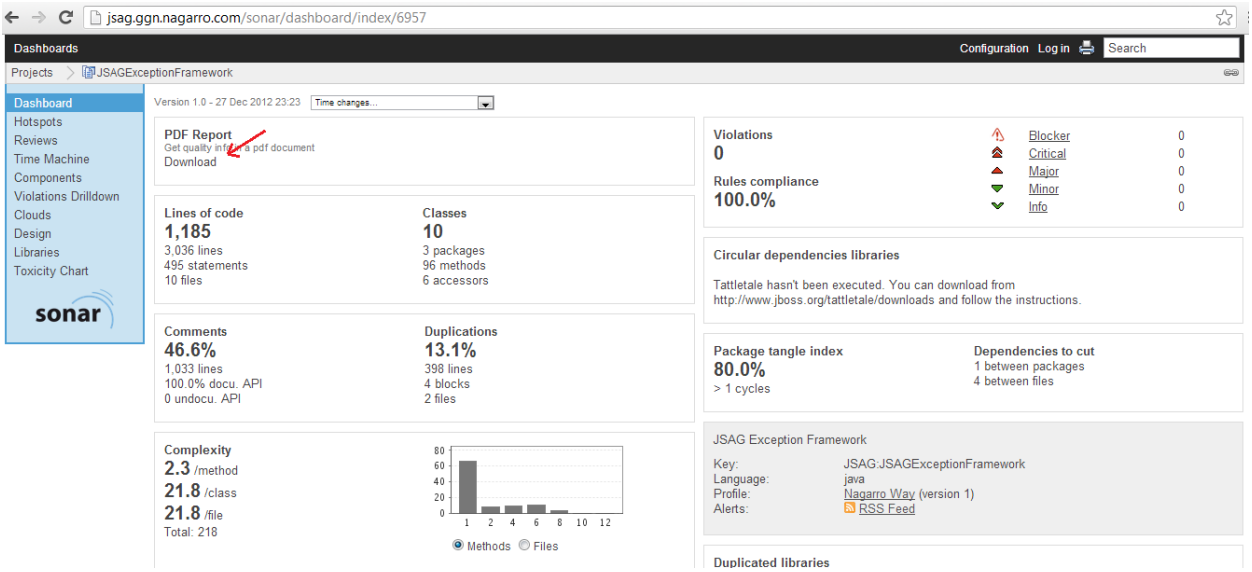
Lines of code	Comments	Complexity
1,185	46.6%	2.3
3 packages	1,033 comment lines	21.8 /class
10 classes		218 decision points
96 methods		
13.1% duplicated lines		

sonar	Sonar PDF Report	JSAGExceptionFramework

Most complex files	
ExceptionHelper	92
ExceptionUtil	53
BaseException	27
TechnicalRuntimeException	26
BusinessException	6

Most duplicated files	
TechnicalRuntimeException	210
BaseException	188

The same details can be accessed from the Sonar Web interface (http://jsag.ggn.nagarro.com/sonar).



Independent violation details can be accessed by clicking on the violation or the count reported.

Independent results generated by PMD and findbugs can be accessed by clicking on the sonar directory inside the target directory as shown below.