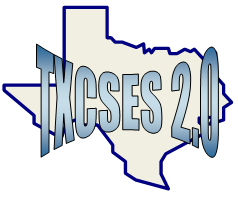
Texas Office of the Attorney General



SonarQube Installation and Setup Documentation

Jan 20, 2016

Table of Contents

[1 Introduction 4](#_Toc444171018)

[2 SonarQube setup topology 4](#_Toc444171019)

[3 MySQL Installation 5](#_Toc444171020)

[3.1 Installing MySQL 6](#_Toc444171021)

[3.2 Creating the SonarQube Database 7](#_Toc444171022)

[4 SonarQube Installation 8](#_Toc444171023)

[4.1 Database configuration 8](#_Toc444171024)

[4.2 Web server configuration 8](#_Toc444171025)

[4.3 Installing SonarQube as a window service 9](#_Toc444171026)

[4.4 Starting the server 9](#_Toc444171027)

[5 SonarQube Initial Setup 11](#_Toc444171028)

[5.1 Configuring the Default Quality Profile 13](#_Toc444171029)

[6 SonarQube Update Centre 15](#_Toc444171030)

[7 Installing and Configuring the Jenkins SonarQube Plugin 16](#_Toc444171031)

[8 Installing and configuring the Eclipse SonarQube plugin 18](#_Toc444171032)

[8.1 Installing the Eclipse SonarQube plugin 18](#_Toc444171033)

[8.1.1 Getting SonarQube using Eclipse Marketplace 18](#_Toc444171034)

[8.1.2 Installing SonarQube using the Eclipse update center 18](#_Toc444171035)

[8.1.3 Installation 19](#_Toc444171036)

[8.2 Configuring the Eclipse SonarQube plugin 20](#_Toc444171037)

[8.3 Executing local analysis 23](#_Toc444171038)

[8.3.1 Configuring global security settings 23](#_Toc444171039)

[8.3.2 Adding properties for the Preview Mode. 25](#_Toc444171040)

[9 Maven Script updates for Sonar Analysis 26](#_Toc444171041)

[10 References 26](#_Toc444171042)

[Document Control 27](#_Toc444171043)

[10.1 Change History 27](#_Toc444171044)

[10.2 Open Issues 28](#_Toc444171045)

# Introduction

SonarQube is a Continuous Quality Assurance engine that consolidates static and dynamic quality metrics leveraging known tools as Checkstyle, PMD, FindBugs, JUnit, Cobertura, JDepend or Architecture Rules. SonarQube integrates with Maven builds or with Jenkins jobs, automatically running all the tools, gathering all the relevant data and collating them all in a single dashboard.

SonarQube is the recommended tool in Java Blueprint for adopting *Automated Solution Quality* practices in the server-side using SonarQube for *Build Quality Dashboard*.

Also since Java Automation Blueprint 3.2 also SonarQube is the recommended tool for *Automated Code / Configuration Review* in developer workstations using the Eclipse plugin for SonarQube.

This document contains detailed insights on how to install SonarQube and setup it for first use. The companion document, ***SonarQube Usage Guide***, includes information for end users on how to analyse sources and work with the SonarQube dashboard.

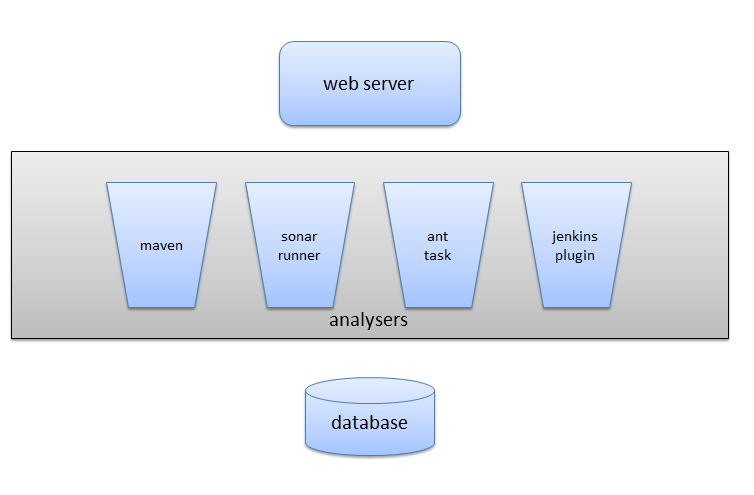
Although this guide will cover in detail, step by step, the process of installing SonarQube inside an Apache Tomcat web container, we strongly recommend leveraging the **Java Build Accelerator for Continuous Integration** that can be downloaded from the [Accenture Portal for Java](https://go.accenture.com/javaportal). The Java Build Accelerator will install a pre-configured SonarQube dashboard instance, along Jenkins for Continuous Integration, quickly and easily.

You can directly download the **Java Build Accelerator** (JBA) by accessing it at <http://go.accenture.com/jba>.There is also a specific documentation for the JBA, the ***Java Build Accelerator Installation and Setup Guide*** that can be also found from the Java Build Accelerator download section within the Developer Centre at the [Accenture Portal for Java](https://go.accenture.com/javaportal).

# SonarQube setup topology

The topology that is used for the setup of SonarQube is based on three main elements:

* **Web server:** To deploy the SonarQube dashboard that allows configuring projects, rules and plugins, and also where to browse through the analysed projects.
* **Analysers**: The components that will take the responsibility to analyse projects and populate results at database.
* **Database**: Where the quality, users, and projects configuration are stored. Also the information about the project analysed are warehoused.



# MySQL Installation

SonarQube uses a relational database to store the configuration and metrics for configured projects. Although SonarQube ships with an embedded Apache Derby database, for the best performance and user experience it is recommended to use a more robust database engine.

In this section we will describe how to install and configure a MySQL database for its use with SonarQube. MySQL is Open Source software and a very widely used database engine. Oracle (including the free edition XE), Microsoft SQL Server and PostgreSQL are also supported.

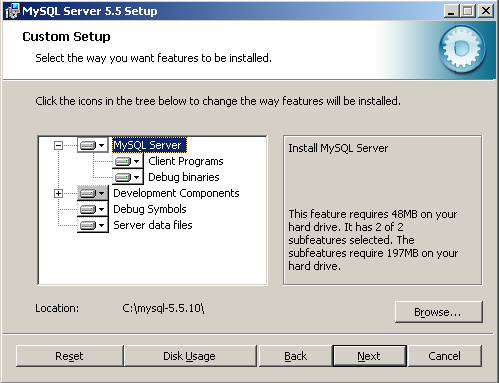
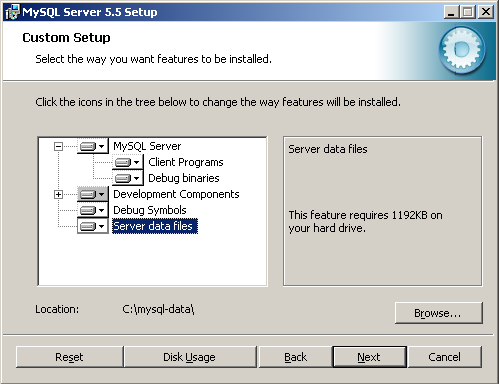
Recommended MySQL version is 5.5.10. Depending on the OS version in the target box, you may use the 32-bits and 64-bits version of MySQL. Installers can be downloaded from these URLs: <http://downloads.mysql.com/archives/mysql-5.5/mysql-5.5.10-win32.msi> and <http://downloads.mysql.com/archives/mysql-5.5/mysql-5.5.10-winx64.msi>

Alternatively you may download and install MySQL from a Zipped distribution. However, Zipped distributions do not come with an installation program and will need additional manual steps to complete the installation. Read MySQL documentation for more information on how to install MySQL from Zipped distributions.

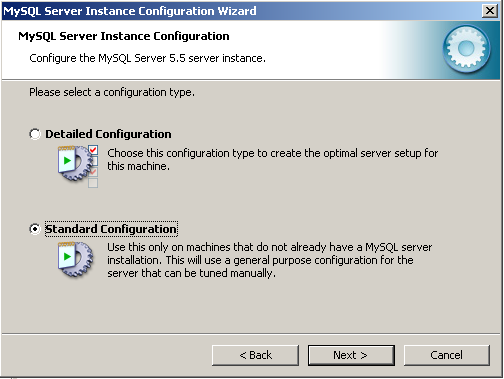
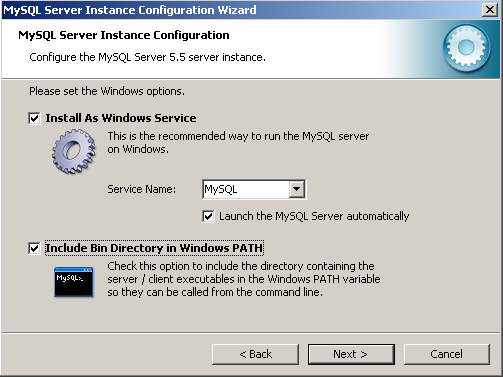
## Installing MySQL

To being MySQL installation, execute the .msi file corresponding to the OS version in the target box. For the rest of the guide we will assume that we are installing MySQL 32-bits versions. Therefore, execute the file **mysql-5.5.10-win32.msi**.

The installation wizard is pretty straightforward. We suggest a Custom Setup to be able to choose some settings as the destination folder for MySQL engine and the destination folder for databases. For the rest of the guide we will assume that **c:\jba\ci\mysql-5.5.10** folder is where MySQL engine is installed, and **c:\jba\ci\mysql-data** folder is the default location for databases:

Once all files are copied, launch the MySQL Instance Configuration Wizard. If you forget to select the option before finishing the installer, you may launch the wizard from the Programs Menu in Windows. Once the wizard is launched, select **Standard Configuration** mode. In the next wizard screen, we recommend selecting the **Install As Windows Service** and **Include Bin Directory in Windows PATH** check-boxes. We also recommend to start the MySQL service automatically when Windows starts:

In the next wizard screen you should enter the administrator account (root) password. We will use **adminadmin** as with the other software we are installing. The wizard will create the configuration file, **c:\jba\ci\mysql-5.5.10\my.ini**, and start the MySQL service. By default, MySQL is listening on port 3306.

## Creating the SonarQube Database

Now that MySQL is installed and running, we will create and prepare a database for SonarQube use.

Execute the MySQL Command Line Client, either through the Programs Menu or through a console (**mysql** command).

In the console, issue the following commands to create the **sonarqube** user (using password **sonarsonar**) and **sonar** database. We need only an empty database as SonarQube will create the schema and populate it with needed data on first run. The commands needed are:

CREATE DATABASE sonar;

CREATE USER 'sonarqube'@'localhost' IDENTIFIED BY 'sonarsonar';

GRANT ALL PRIVILEGES ON sonar.\* TO 'sonarqube'@'localhost' WITH GRANT OPTION;

CREATE USER 'sonarqube'@'%' IDENTIFIED BY 'sonarsonar';

GRANT ALL PRIVILEGES ON sonar.\* TO 'sonarqube'@'%' WITH GRANT OPTION;

Once finished, issue the **exit** command to close the client.

# SonarQube Installation

This section of the guide will explain how to install SonarQube inside the Apache Tomcat instance and using the MySQL instance installed on previous sections.

The Java Automation Blueprint 5.0 recommends the installation of SonarQube 4.3.3. that can be downloaded from this URL: <http://dist.sonar.codehaus.org/sonarqube-4.3.3.zip>

The process starts by decompressing the sonarqube-4.3.3.zip file.

Assuming the that target folder is C:\jba\ci\sonarqube-4.3.3 the next step will be to configure the database and web server dashboard properties

## Database configuration

Edit the file **C:\jba\ci\sonarqube-4.3.3\conf\sonar.properties** to refer to the MySQL database created in previous section. The properties that should be updated are the following:

* **sonar.jdbc.url** = jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true
* **sonar.jdbc.username** = sonarqube
* **sonar.jdbc.password** = sonarsonar

Note that there are some provided templates for quick configuration of supported databases.

## Web server configuration

For the web server, the following configuration should be set:

Java runtime for the SonarQube wrapper. This should be configured in the file %SONARQUBE\_HOME%\conf\wrapper.conf

Assuming the JDK has been installed under the folder C:\java\jdk\ the configuration based on the version of the JDK recommended in the Java Automation Blueprint 5.0 should be:

wrapper.java.command=/java/jdk/jdk-8/bin/java

We can also configure the context root for the SonarQube server dashboard by editing the file %SONARQUBE\_HOME%\conf\sonar.properties and setting the value for the parameter sonar.web.context as following

sonar.web.context=/sonarqube

In the case you to configure the port where the web server will be listening once started, you should edit the sonar.properties configuration file and set the required value for sonar.web.port. By default the SonarQube server will be listening at port 9000

# TCP port for incoming HTTP connections. Disabled when value is -1.

#sonar.web.port=9000

## Installing SonarQube as a window service

The SonarQube distribution provides custom tools to install SonarQube server as a windows service. These tools are located at:

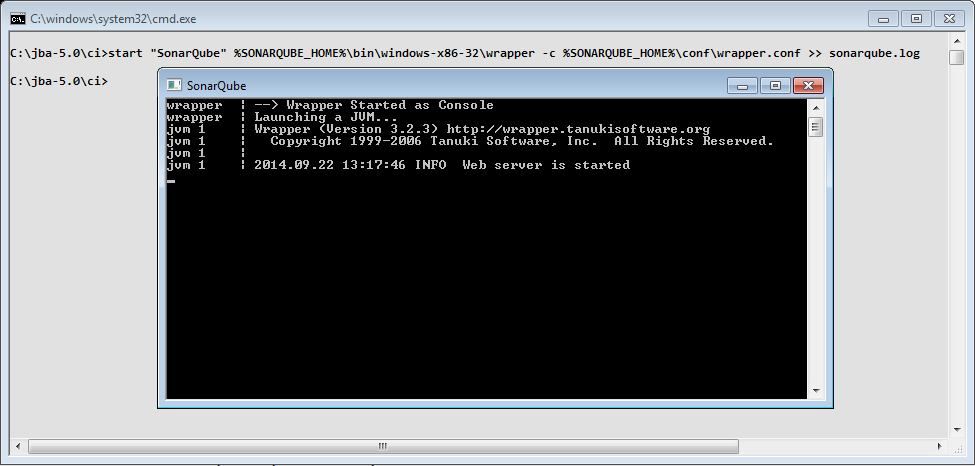
%SONARQUBE\_HOME%/bin/windows-x86-32/InstallNTService.bat

%SONARQUBE\_HOME%/bin/windows-x86-32/UninstallNTService.bat

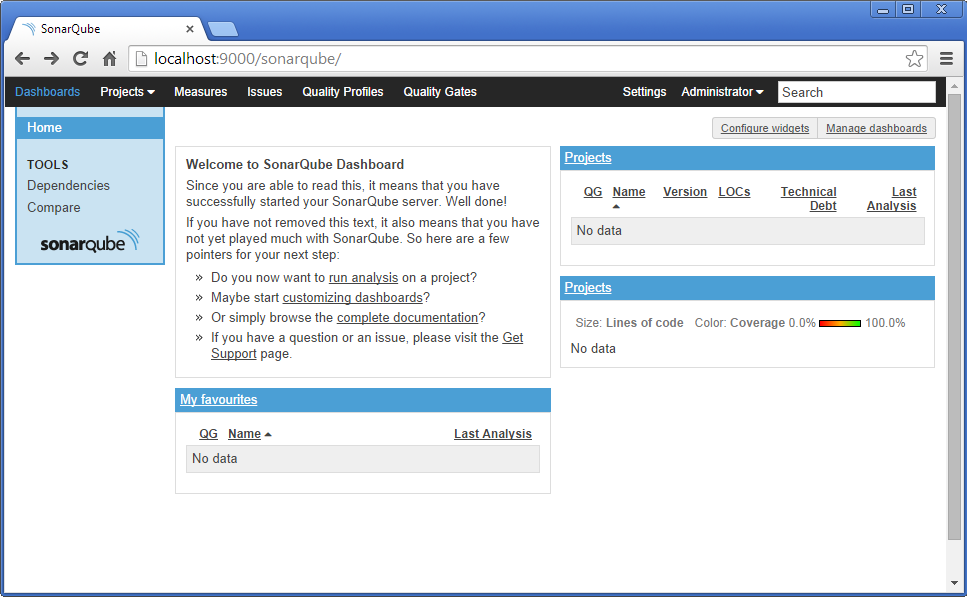
## Starting the server

Assuming that the MySQL service is up and running, we can now start the SonarQube server by launching the following command:

start "SonarQube" %SONARQUBE\_HOME%\bin\windows-x86-32\wrapper -c %SONARQUBE\_HOME%\conf\wrapper.conf



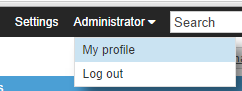
Verify that SonarQube is properly installed by opening in your browser the SonarQube dashboard URL: <http://localhost:9000/sonarqube>

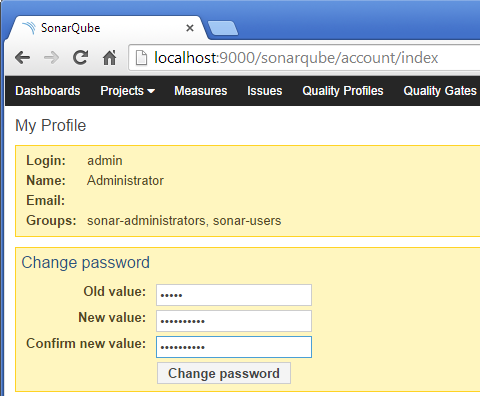


# SonarQube Initial Setup

Now that SonarQube is installed and we have access to SonarQube dashboard, we will prepare it for first use.

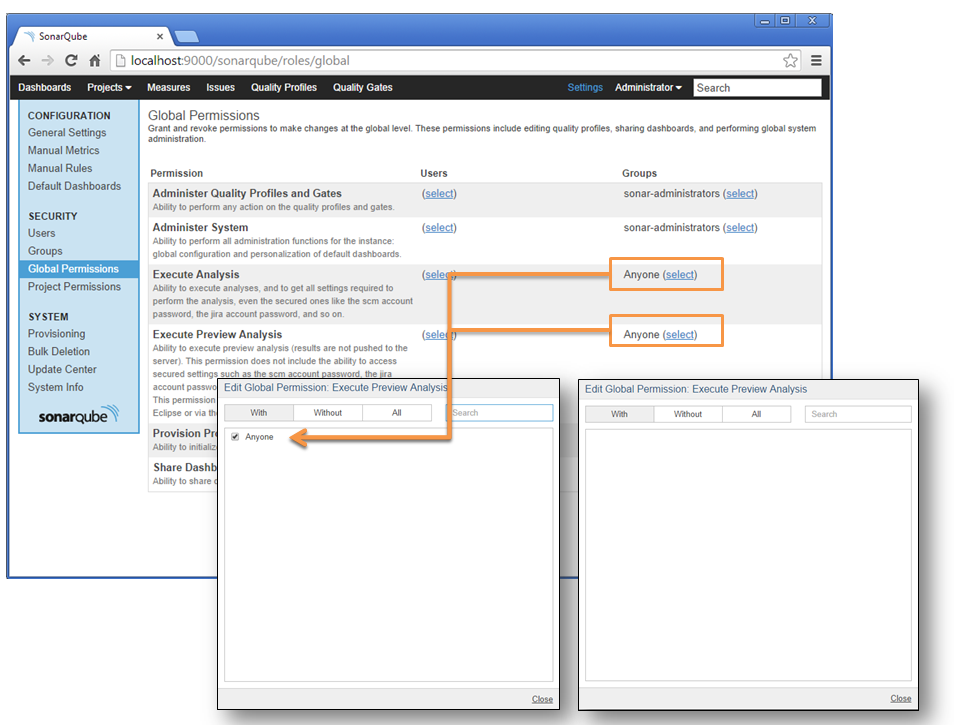
In the top menu, select **Log In**. Enter default administrator credentials, user **admin** and password **admin**. In the top menu again, select **Administrator-> My profile**. Use the change password form to update the administrator new password using for example **adminadmin**:





It is recommended to remove access to anonymous users. This will prevent unauthorized access to the SonarQube dashboard information, which includes not only metrics and historic data, but sources and Javadoc documentation. Note that sources and documentation can be hidden in the dashboard if security policies or client agreement requires that.

To remove anonymous access you should the Anonymous group from the groups that can execute and preview analysis that comes by default. To do that, access to the Security section within the global configuration by clicking on “Settings” and select “Global Permissions”. Once there, from the “Execute Analysis” and “Execute Preview Analysis” permissions, deselect the “Anonymous” group.



## Configuring the Default Quality Profile

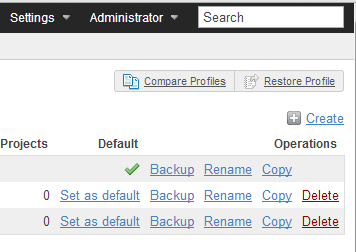
A quality profile in SonarQube is a set of quality rules (sometimes referred also as checkers, detectors or profilers) and their associated severity and configuration, plus definition of thresholds and alerts for multiple quality indicators. Quality rules come from the tools that SonarQube uses to collect quality data: Checkstyle, PMD, FindBugs and SonarQube Squid (based on JavaNCSS, JDepend and Architecture Rules).

An Accenture standards quality profile has been created as a recommendation for common Java developments in Accenture. Although some projects may need to customise it to match client requirements or standards, this profile should be the starting point as it includes not only coding standards, but best practices and known bug patterns.

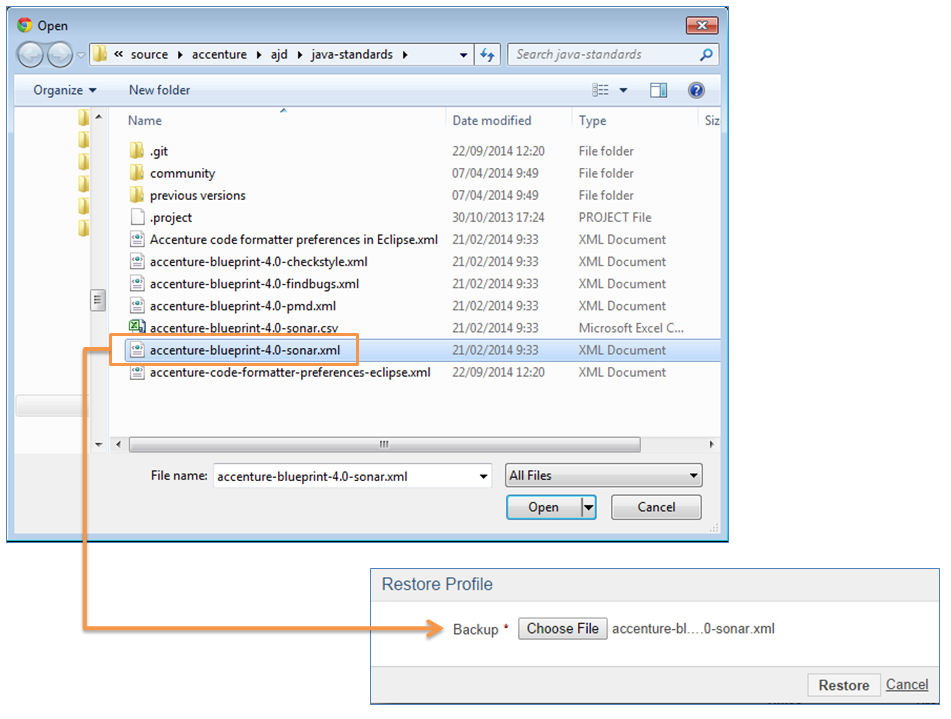
The Accenture standards quality profile comes already configured with the Java Build Accelerator installer, but also exists as a collaborative repo at Accenture’s Inner Source so you can always can download standalone version for use in other environments or projects can contribute with their custom rules.

The repo is available at <https://go.accenture.com/javastandards>

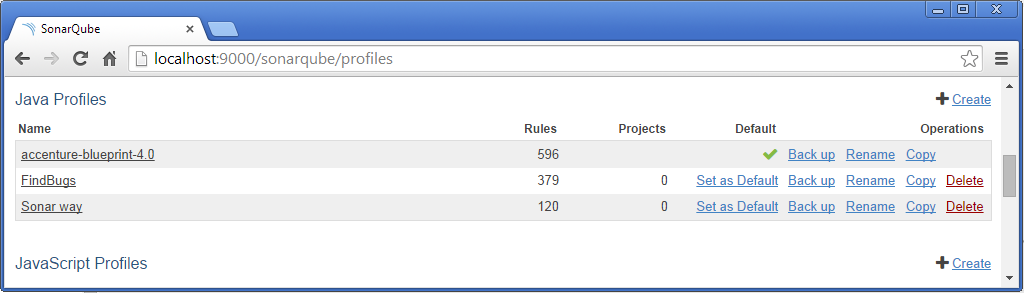
To import the profile into SonarQube, on the top menu select **Settings ->** **Quality Profiles** and below the top menu select **Restore profile**:



On the next screen, browse for the downloaded quality profile file and click the **Restore profile** button:



It is recommended that the Accenture profile is set as the default for new projects:

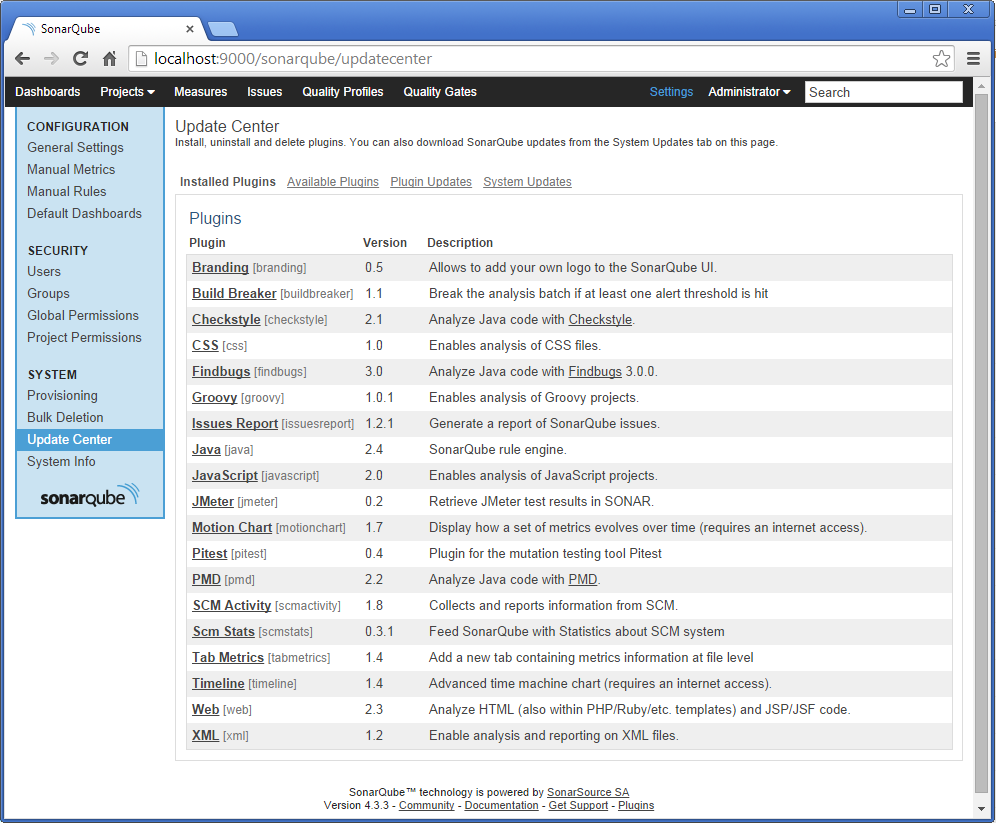


Once the quality profile is configured, SonarQube is ready to analyse new Java projects.

# SonarQube Update Centre

SonarQube comes with an update centre that allows easy installation and upgrade of SonarQube plugins. Although SonarQube is bundled with some core plugins that enable typical use cases you can install new plugins any time that will extend SonarQube functionality and match your needs. Note that there are free and commercial plugins, so you may need to review individual plugin licenses before installing them.

To access the update centre, in the top menu select **Settings** and then in the left menu select **Update Center**:



In the update centre screen there are four tabs: **Installed Plugins** shows installed plugins, **Available** **Plugins** shows the catalogue of available plugins, **Plugin Updates** shows the list of plugins that can be upgraded and allows upgrading them and **System Updates** shows whether SonarQube can be upgraded and instructions to do the upgrade.

The plugin catalogue includes additional languages (C, Cobol, Flex, PL/SQL, Visual Basic…), additional analysers and metrics (JMeter, Jira, Trac, C-rules), governance models (Quality Index, SIG MM, SQALE), integration with other tools and extended visualization and reporting (PDF Report).

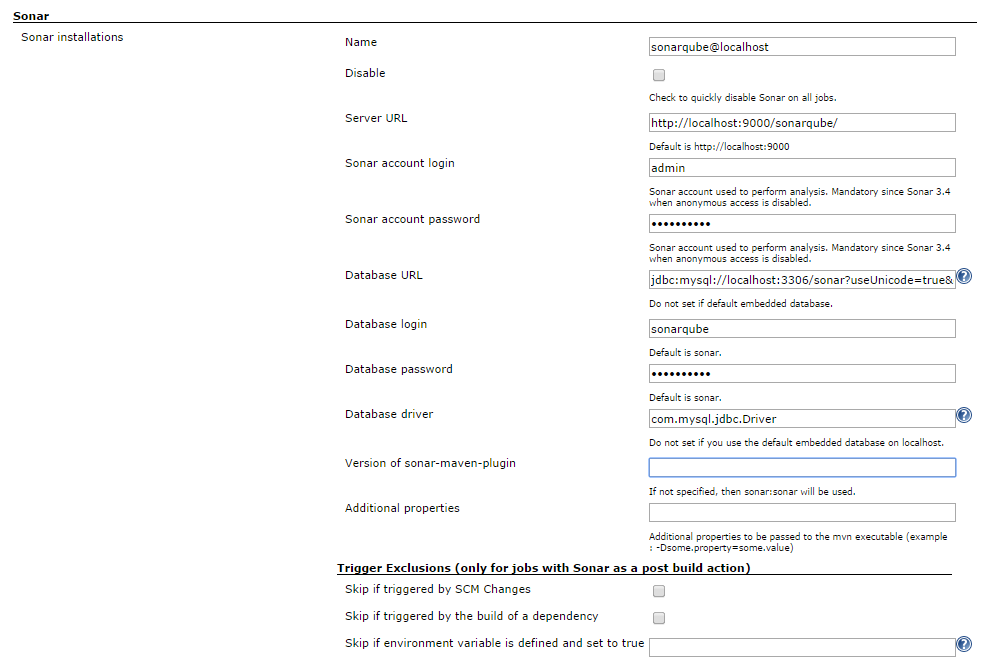
# Installing and Configuring the Jenkins SonarQube Plugin

Jenkins and SonarQube work closely together to provide the best of the Continuous Integration and Continuous Quality Assurance practices. Through the use of the Jenkins SonarQube plugin, we can launch a SonarQube analysis on selected jobs and link the Jenkins dashboard with SonarQube dashboard to review analysis results.

To install the Jenkins SonarQube plugin, just go to the Jenkins update centre as described in previous section and search for the SonarQube plugin to install it. Restart Jenkins to finish off with the plugin installation.

Although in the rest of the section we are going to describe how to configure the plugin parameters, note that the Java Build Accelerator installer already includes Jenkins and SonarQube configured to work together.

Once installed, log in to the Jenkins dashboard as administrator and in the left menu select **Manage Jenkins**. Then select **Configure System** in the list of management tasks. Now that the plugin is installed, near the end of the screen you will find a **SonarQube** section. In this section we can add as many SonarQube instances as needed, selecting for each one the connectivity parameters and details on how to trigger the SonarQube analysis.



The connectivity parameters, and default values when using recommended setup and settings, are:

* **sonar.jdbc.url** = jdbc:mysql://localhost:3306/sonar?useUnicode=true&characterEncoding=utf8&rewriteBatchedStatements=true
* **sonar.jdbc.driver** = com.mysql.jdbc.Driver
* **sonar.jdbc.username** = sonarqube
* **sonar.jdbc.password** = sonarsonar
* **sonar.host.url** = [http://localhost:<port>/<sonarqub](http://localhost:%3cport%3e/%3csonarqub)e context root>

Don’t forget to update previous settings in the case that you are not following the recommended setup and settings, e.g. you are using a different database engine or have Jenkins and SonarQube in different boxes.

The **Triggers** subsection can be used to globally enable or disable SonarQube analysis from selected types of builds. For example, we may enable SonarQube analysis in periodic builds but not in those triggered by an SCM poll. It is recommended, unless needed, to globally activate SonarQube in all builds except when triggered by a dependency.

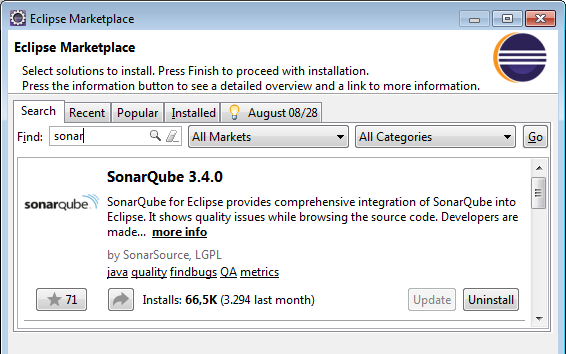
# Installing and configuring the Eclipse SonarQube plugin

Eclipse can be configured to work with a remote SonarQube instance. It is recommended that each developer need not install a copy of SonarQube instead they can connect to the remotely setup instance as part of continuous integration using the Eclipse SonarQube plugin. This way the developers can view the violations in SonarQube within Eclipse without having to leave the IDE.

## Installing the Eclipse SonarQube plugin

### Getting SonarQube using Eclipse Marketplace

By searching for “SonarQube” in the Eclipse Marketplace

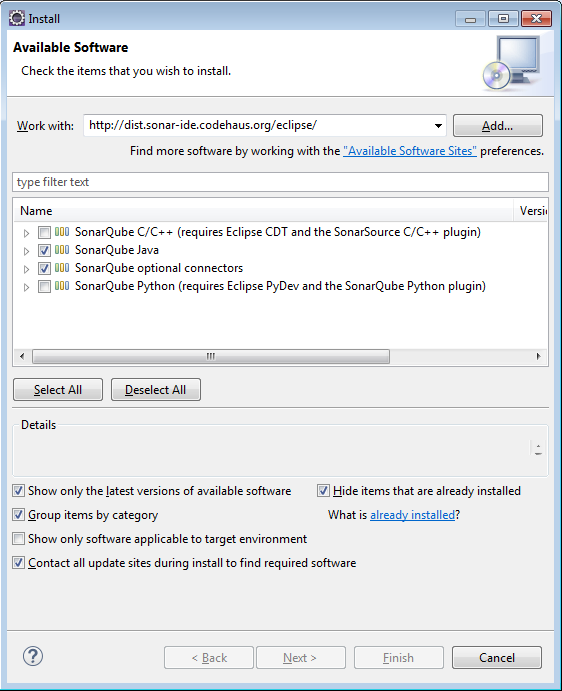


### Installing SonarQube using the Eclipse update center

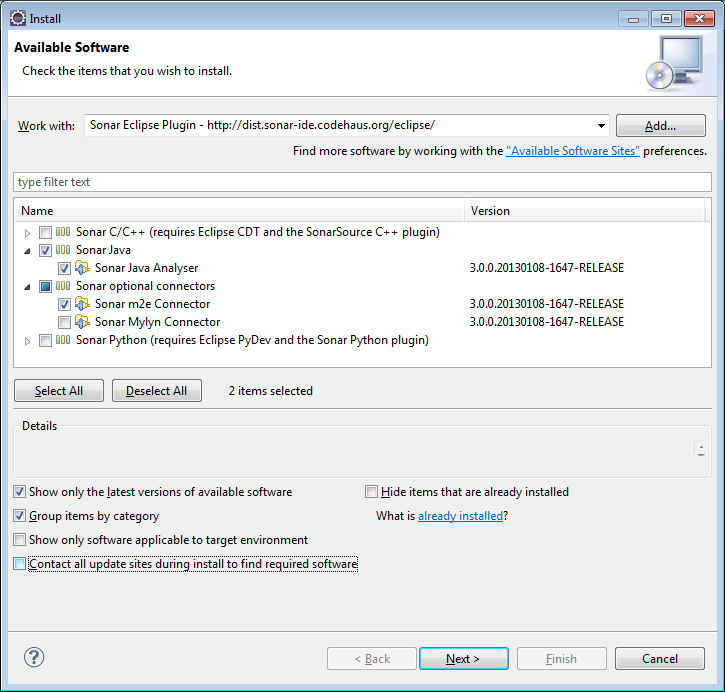
1. In Eclipse Go to **Help -> Install New Software.**
2. Add the following url as a update site: <http://dist.sonar-ide.codehaus.org/eclipse/>

### Installation

This will list available components to download and install:



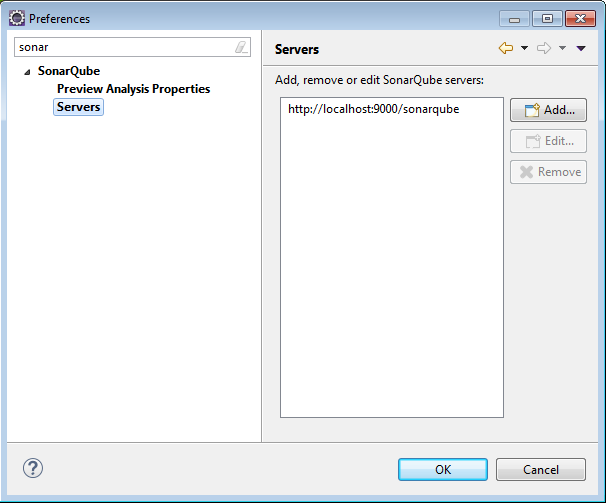
1. Select the SonarQube Java and SonarQube m2e Connector to install and complete the installation.



1. Once installation is complete restart Eclipse as required.

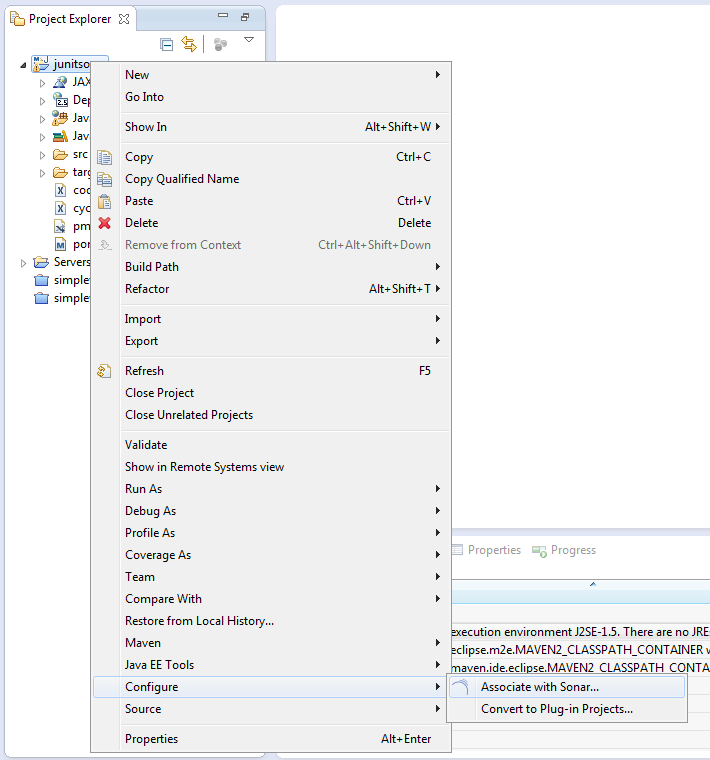
## Configuring the Eclipse SonarQube plugin

Once installation is complete Eclipse needs to be configured to connect to a SonarQube instance. Navigate to **Windows->Preferences** and under **SonarQube->Servers** add entry for your remote SonarQube server.

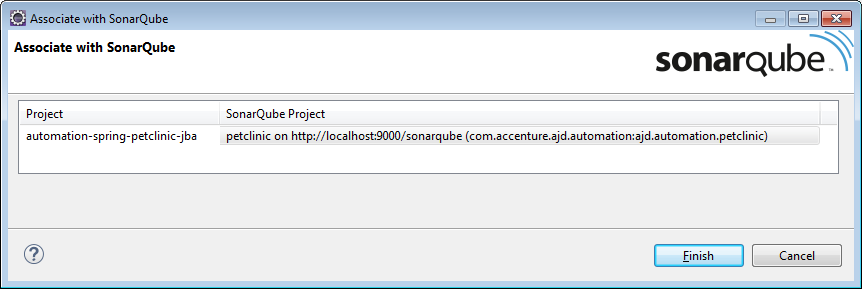


Test the connection to make sure Eclipse is able to connect to the remote SonarQube instance.

To link the project for the first time with the SonarQube instance right click on the project explorer and select **Configure->Associate with SonarQube.**

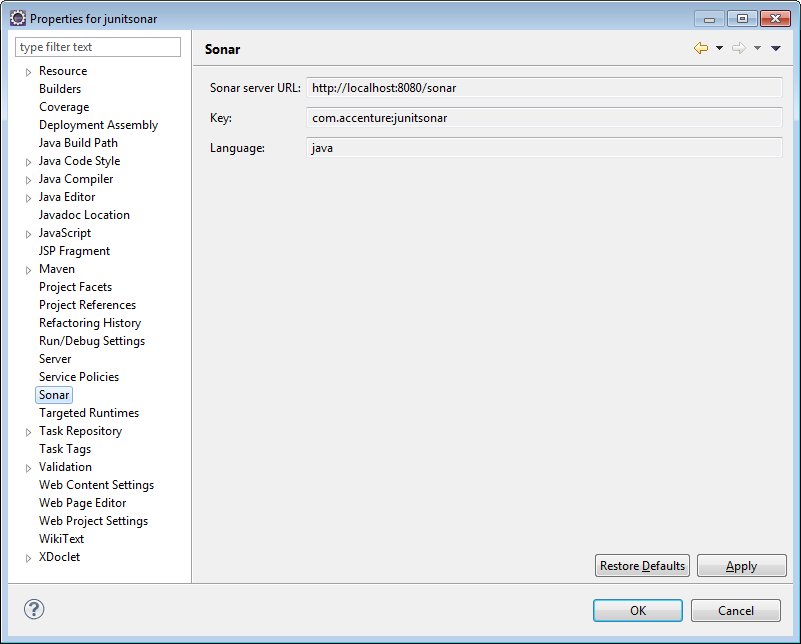


The SonarQube instance configured should show up for association.



Maven projects are automatically associated with SonarQube. By default the url used during this association is localhost. In case you have a remote instance configured you may want to remove the link to the default association and link to correct instance of SonarQube.

Right click on the Project and under properties navigate the SonarQube section to verify if the project has been associated to the right SonarQube instance.



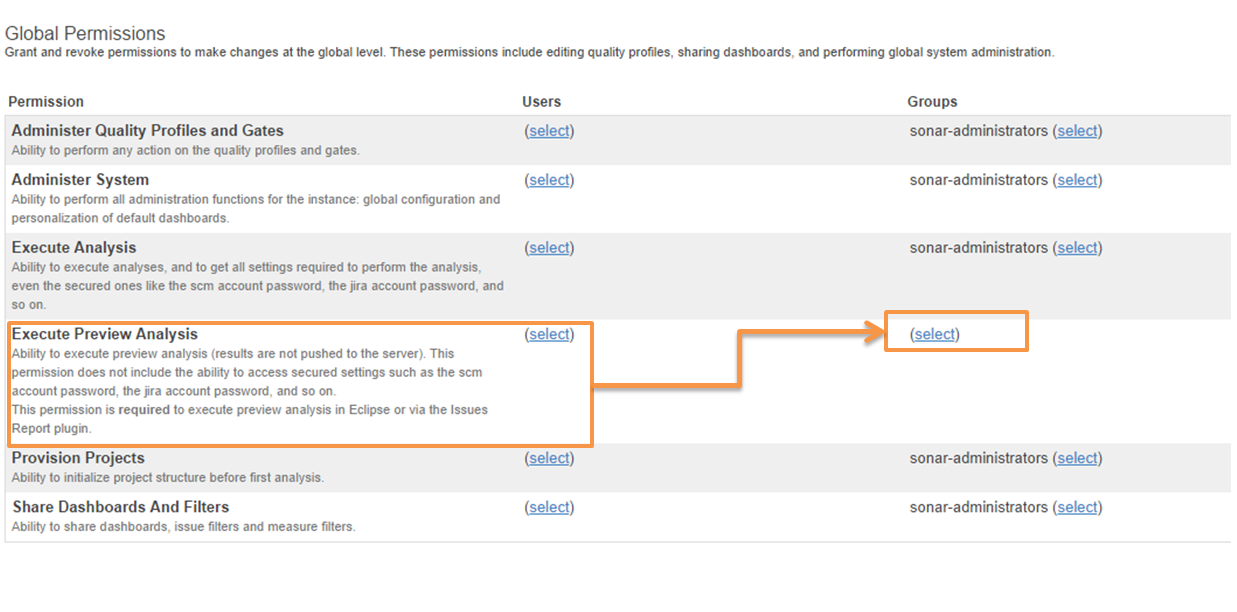
A project can be disassociated with SonarQube by selecting **SonarQube->Remove SonarQube nature** at the menu options available on the right click of the project.

## Executing local analysis

### Configuring global security settings

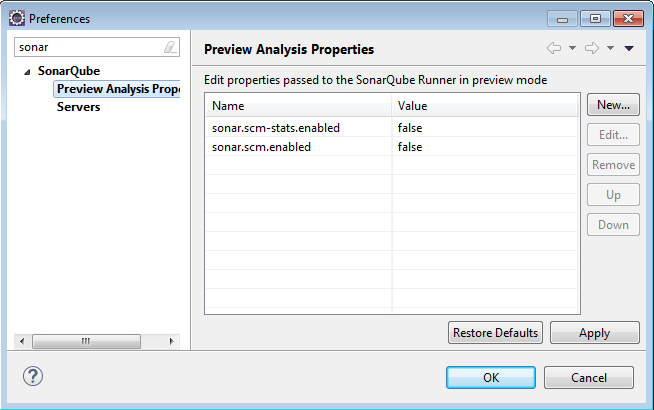
To enable the local analysis execution from the Eclipse SonarQube plugin you will need to configure global security settings at SonarQube server

To do that, go to the SonarQube dashboard acces Settings > Global Permissions and select the groups that can execute the “Execute Preview Analysis”



### Adding properties for the Preview Mode.

Sometimes you will need to add some extra configuration to the local analysis. In particular sometimes it is required to avoid the execution of some SonarQube plugins during the local analysis as they are not compatible in preview mode. To disable these plugins, access the SonarQube preferences in Eclipse and add the required parameters to be passed to the local analysis



# MAVEN SCRIPT UPDATES for SONAR ANALYSIS

Inorder to generate the sonar static analysis report add the following plugin to the maven pom files.

<plugin>

<groupId>org.codehaus.mojo</groupId>

<artifactId>sonar-maven-plugin</artifactId>

<version>2.7.1</version>

</plugin>

# References

Accenture Portal for Java: <https://ajd.accenture.com>

MySQL: <http://www.mysql.com/>

SonarQube: <http://www.sonarsource.org/>

How to install MySQL from Zipped distributions: <http://dev.mysql.com/doc/refman/5.5/en/windows-install-archive.html>

SonarQube on-line documentation: <http://docs.codehaus.org/display/SONAR/Documentation>

Video Demos:

Introduction to SonarQube: <https://mediaexchange.accenture.com/media/DIIP+Java+Automation+Blueprint+-+Introduction+to+Sonar/0_96zm4icp>

Monsoon - Quality and Testing: <https://mediaexchange.accenture.com/media/Monsoon+-+Quality+and+Testing/0_xcwicuov>

# Document Control

## Change History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Author | Approver | Comment |
| 0.1 | 05/19/2011 | Jorge Hidalgo |  | First draft version |
| 0.2 | 05/22/2011 | Jorge Hidalgo |  | Added Jenkins Sonar plugin installation |
| 1.0 | 05/25/2011 | Jorge Hidalgo |  | First published version |
| 1.1 | 03/11/2013 | Swaminathan Gopalakrishnan |  | Updated for blueprint refresh v3.2. Sonar version updated to 3.4.1 |
| 1.2 | 03/25/2013 | Swaminathan Gopalakrishnan |  | Added Eclipse Sonar plugin installation and configuration information |
| 1.3 | 04/15/2013 | Vicente Gonzalez |  | * Some minor changes. * Updated metadata. * Removed ACQT as recommended tool as this is discontinued since Java Blueprint 3.2. * Added links to JBA for CI download and documentation * Tomcat updated to version 7.0.22 that is the version used in the Java Build AcceleratorSonar. * Adding the eclipse market place for installing Sonar for Eclipse. * Added video demos from mediaexchange. |
| 2.0 | 05/29/2013 | Vicente Gonzalez |  | Publish version 2.0 for the Java Automation Blueprint 3.2 release |
| 3.0 | 03/09/2014 | Vicente Gonzalez |  | Publish version 3.0 for the Java Automation Blueprint 4.0 release |
| 4.0 |  | Vicente Gonzalez |  | * Removed tomcat setup as this is not more part of the SonarQube standard setup * Updated global security settings configuratio * Added SonarQube local analysis execution extra configuration. * Updated tool versions according to Java Automation Blueprint 5.0 recommendations |

## Open Issues

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Opening Version | Date | Author | Action | Owner | Status | Closing version |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |