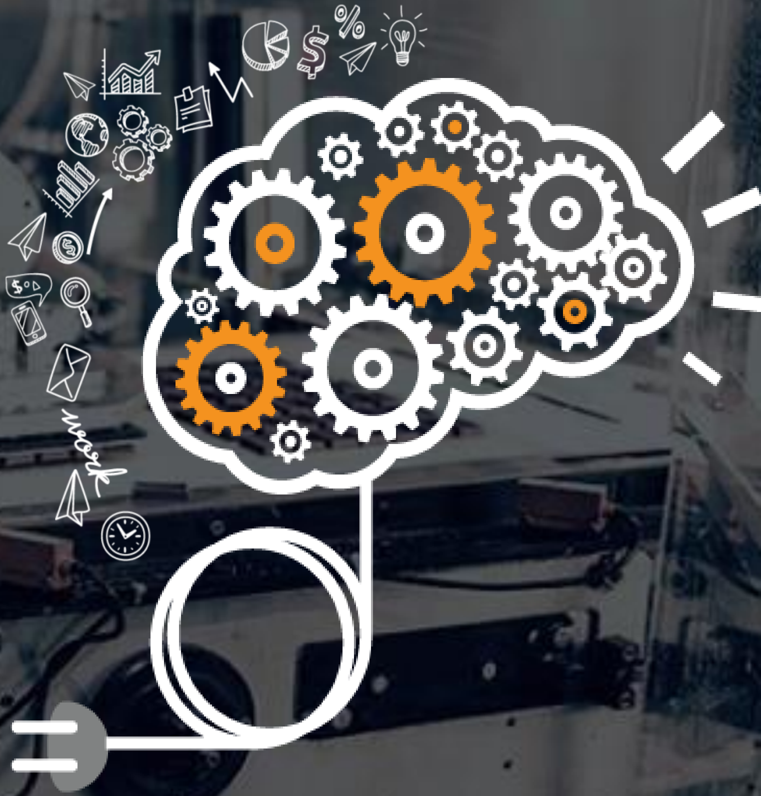
Static Code Analysis Using SonarQube



Java and Open Source Competency

Tech Mahindra's Automation Program

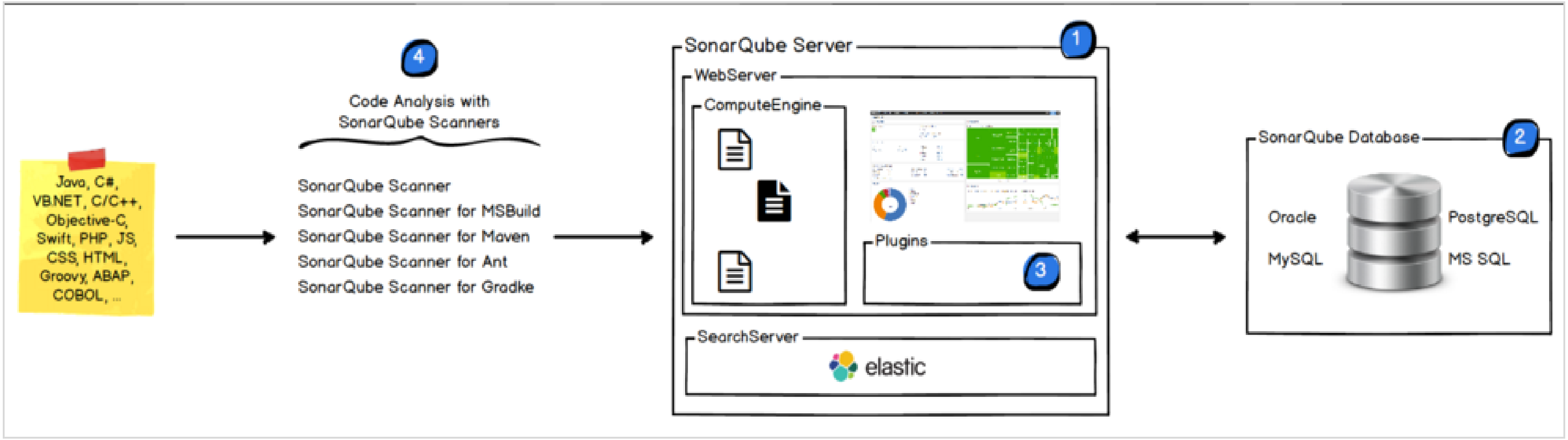
TABLE OF CONTENT (Day-1)

SonarQube- Overview and Features

- SonarQube (previously known as Sonar) is an open source platform for Continuous Inspection of code quality.
- SonarQube is written in java and supported for 25+ languages such as
    - Java, C/C++, C#, PHP, Flex, Groovy, JavaScript, Python, PL/SQL, COBOL etc., It is also supports Android Development projects.
- It helps for various tasks and provide reports on
    - duplicated code, coding standards, unit tests, code coverage, complex code, potential bugs , comments and, design and architecture.
- SonarQube is internally uses PMD, Find bugs, Check Style. Additionally plugins can be included according to project requirement.
- Records metrics history and provides evolution graphs ("time machine") and differential views
- Provides fully automated analyses using Maven, Ant, Gradle and Continuous Integration tools like Jenkins, Bamboo.
- Integrates with the Eclipse development environment
- Integrates with external tools like JIRA, Mantis, LDAP, Fortify
- Implements the **SQALE** (Software Quality Assessment based on Lifecycle Expectations) methodology to compute technical debt

SonarQube- Architecture

The SonarQube Platform is made of 4 components mainly SonarQube Server, Database, Plugins and Scanners. Below are the architecture for SonarQube:



Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JS, CSS, HTML, Groovy, ABAP, COBOL, ...
4
Code Analysis with SonarQube Scanners
SonarQube Scanner
SonarQube Scanner for MSBuild
SonarQube Scanner for Maven
SonarQube Scanner for Ant
SonarQube Scanner for Gradle
SonarQube Server
1
WebServer
ComputeEngine
Plugins
3
SearchServer
elastic
SonarQube Database
2
Oracle
MySQL
PostgreSQL
MS SQL

1) **SonarQube Server:** It contains mainly 2 process.
   - **Web Server:** For developers, managers to browse quality snapshot's
   - **Search Server: B**ased on Elastic search to back searches from the UI

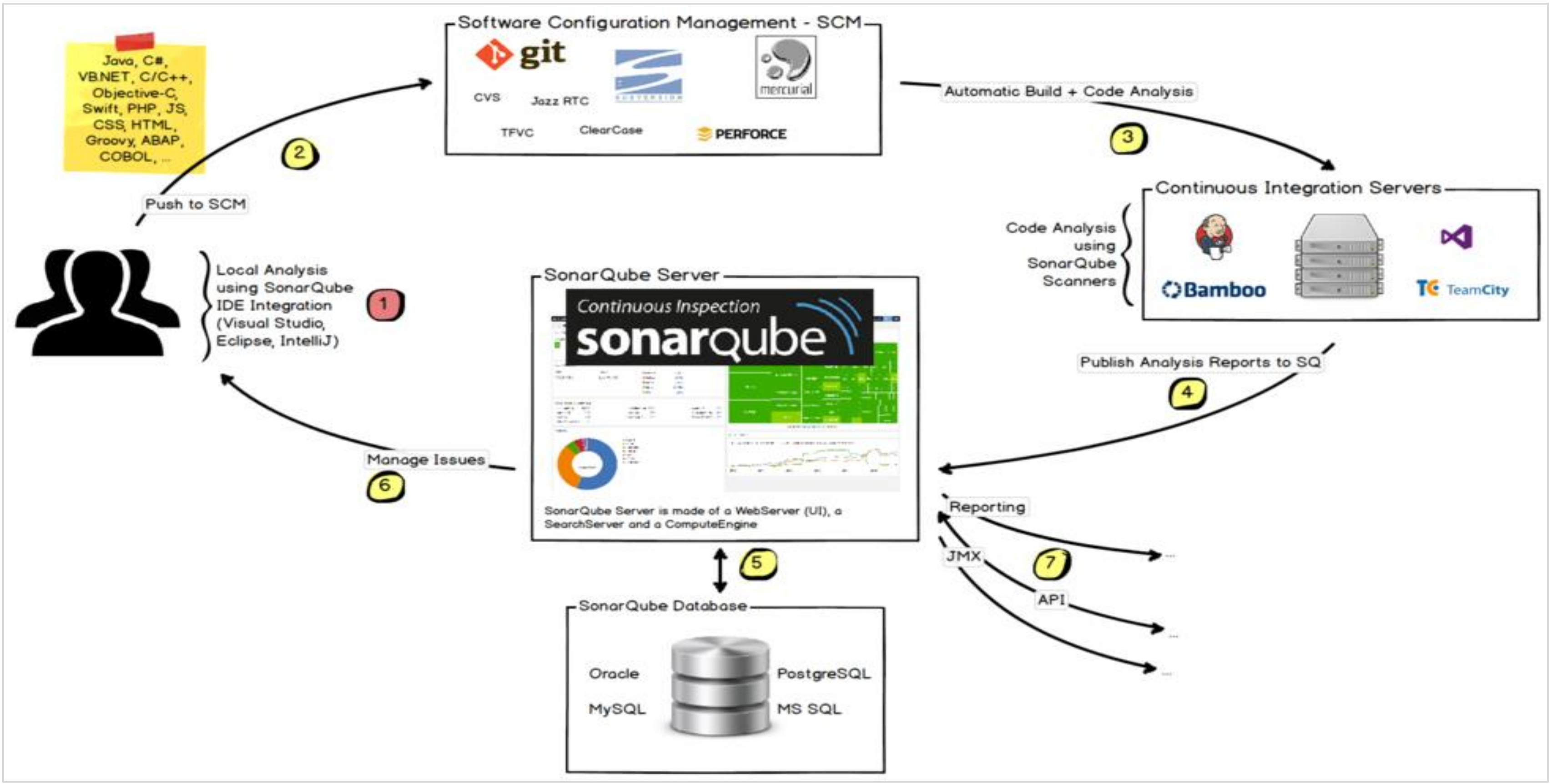
2) **SonarQube Database:** Configuration of the SonarQube instance (security, plugins settings, etc.) and the quality snapshots of projects, views etc.,

3) **SonarQube Plugin:** Installers on the server, possibly including language, SCM, integration, authentication, and governance plugins.

4) **SonarQube Scanners: R**unning on your Build / Continuous Integration Servers to analyse projects

SonarQube- Integration with ALM

The following schema shows how SonarQube integrates with other ALM tools and where the various components of SonarQube are used:


Java, C#, VB.NET, C/C++, Objective-C, Swift, PHP, JS, CSS, HTML, Groovy, ABAP, COBOL, ...
Push to SCM
2
Software Configuration Management - SCM
git
CVS
Jazz RTC
mercurial
TFVC
ClearCase
PERFORCE
Automatic Build + Code Analysis
3
Continuous Integration Servers
Code Analysis using SonarQube Scanners
Bamboo
TeamCity
Local Analysis using SonarQube IDE Integration (Visual Studio, Eclipse, IntelliJ)
1
SonarQube Server
Continuous Inspection
sonarqube
SonarQube Server is made of a WebServer (UI), a SearchServer and a ComputeEngine
Publish Analysis Reports to SQ
4
Manage Issues
6
Reporting
JMX
7
API
5
SonarQube Database
Oracle
PostgreSQL
MySQL
MS SQL

1. Developers code in their IDEs and use Sonar Lint to run local analysis.
2. Developers push their code into SCM: GIT, SVN, TFVC, ...
3. The Continuous Integration Server triggers an automatic build, and the execution of the SonarQube Scanner for analysis
4. The analysis report is sent to the SonarQube Server for processing.
5. SonarQube Server processes and stores the analysis report results in the SonarQube Database
6. Developers review, comment, challenge their Issues to manage and reduce their Technical Debt through the SonarQube UI.
7. Managers receive Reports from the analysis.
   - Ops use APIs to automate configuration and extract data and JMX to monitor SonarQube Server.

SonarQube- Requirements

Prerequisite:

- Java (Oracle JRE 7 onwards or OpenJDK 7 onwards)
- Requires at least 2 GB of RAM and 50 GB HDD based on project needs
- Java Script enabled browser(optional)

Supported Java Platforms:

SonarQube Java analyser is able to analyse any kind of Java source files regardless of the version of Java they comply to. But SonarQube analysis and the SonarQube Server require specific versions of the JVM.

JVM Version	Supported Version
Oracle JRE	7u75+, 8
Open JDK	7u55+, 8
IBM JRE / GCJ / Oracle JRocket	No Support

Supported DB Platforms

Data Base
MSSQL 2008 onwards
MySQL 5.1 onwards (only InnoDB)
Oracle 11G onwards
PostgreSQL 8.x onwards

Supported Browser Platforms

Browser Support
IE 11 onwards
Mozilla Firefox
Google Chrome
Safari

SonarQube- Installation

Sonar Server:

- Download and unzip the SonarQube distribution from below link
  http://www.sonarqube.org/downloads/

- Start the SonarQube server for executing below command :
  On Windows, execute:
  <<SonarQube path>>/bin/windows-x86-xx/StartSonar.bat
  On other operating system, execute:
  <<SonarQube path>>\/bin/[OS]/sonar.sh console

- Browse the results at http://localhost:9000 (default System administrator user id and password are admin / admin)

Sonar Scanner: SonarQube Scanner is recommended as the default launcher to analyse the project with SonarQube server.

- Download and unzip the SonarQube Scanner from the below link
  http://docs.sonarqube.org/display/SONAR/Analyzing+with+SonarQube+Scanner

- Update the global settings (server URL) updating *<install_directory>/conf/sonar-runner.properties* file in below lines:

```
#SonarQube server
#sonar.host.url=http://localhost:9000
```

- Create a new *SONAR_RUNNER_HOME* environment variable set to *<install_directory>/bin*

- Opening a new shell and executing the command sonar-runner -h

SonarQube- Configuration

SonarQube server will be configured by setting the values in [Sonar Home]/conf/sonar.properties file

Data base settings:

The default data base was H2. These values can be changed in *<install_directory>/conf/sonar.properties*:

```
sonar.jdbc.username=sonarqube
sonar.jdbc.password=mypassword
sonar.jdbc.url=jdbc:postgresql://localhost/sonarqube
```

Web Server Configuration:

The default port is "9000" and the context path is "/". These values can be changed in *<install_directory>/conf/sonar.properties*:

```
sonar.web.host=192.0.0.1
sonar.web.port=80
sonar.web.context=/sonar
```

SonarQube- Analyse Source Code

The following are the Code Analysis methods to analyze the code from Client machine.

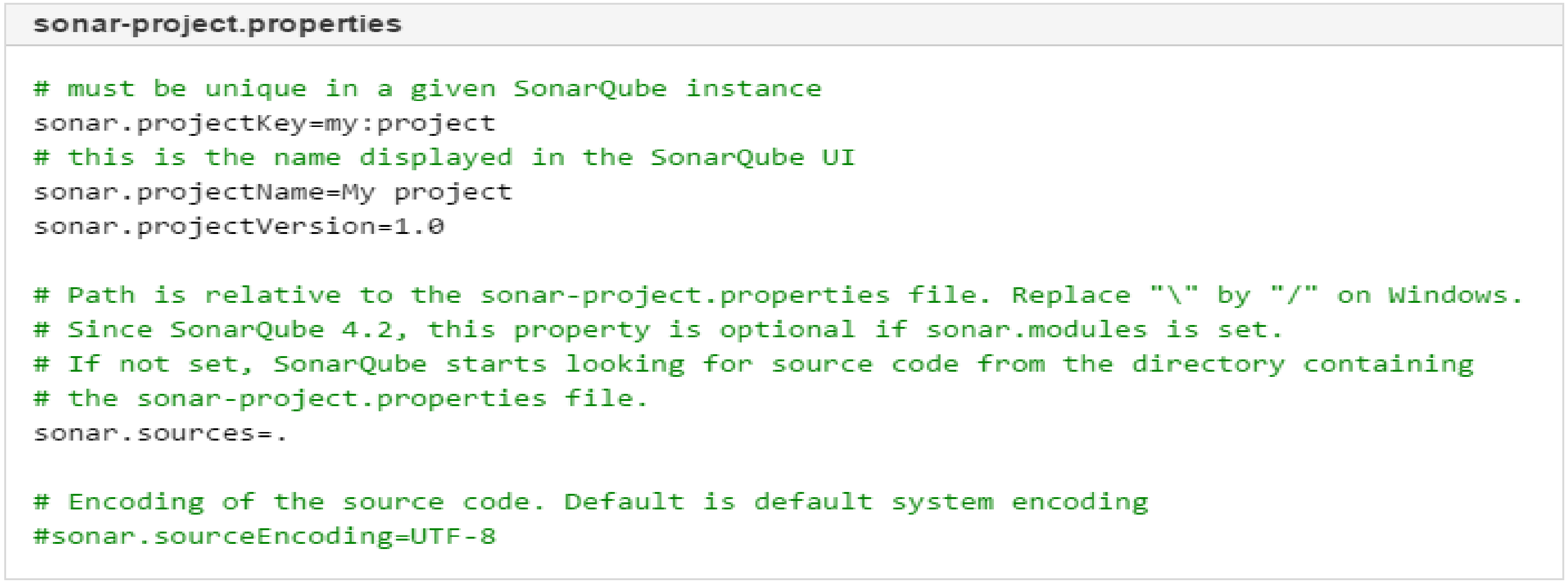
- **SonarQube Scanner -** Launch analysis from the command line
- **SonarQube Scanner for Ant-** Launch analysis from Ant tool
- **SonarQube Scanner for Maven -** Launch analysis from Maven with minimal configuration
- **SonarQube Scanner for Gradle -** Launch Gradle analysis
- **SonarQube Scanner for MSBuild -** Launch analysis of .NET projects
- **SonarQube Scanner For Jenkins -** Launch analysis from Jenkins

Analyse Source Code - Scanner

The SonarQube Scanner is recommended as the default launcher to analyse a project with SonarQube.

The following are the steps to execute “**Simple Project**”

Create a configuration file in the root directory for the project: sonar-*project.properties*



sonar-project.properties

```
# must be unique in a given SonarQube instance
sonar.projectKey=my:project
# this is the name displayed in the SonarQube UI
sonar.projectName=My project
sonar.projectVersion=1.0

# Path is relative to the sonar-project.properties file. Replace "\" by "/" on Windows.
# Since SonarQube 4.2, this property is optional if sonar.modules is set.
# If not set, SonarQube starts looking for source code from the directory containing
# the sonar-project.properties file.
sonar.sources=.

# Encoding of the source code. Default is default system encoding
#sonar.sourceEncoding=UTF-8
```

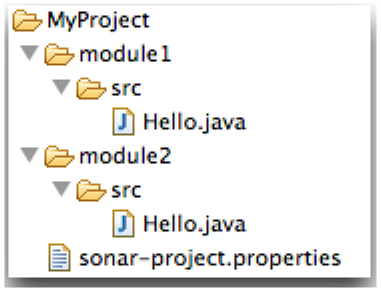
Run the following command from the project base directory to launch the analysis

- **sonar-runner** in command line interface.

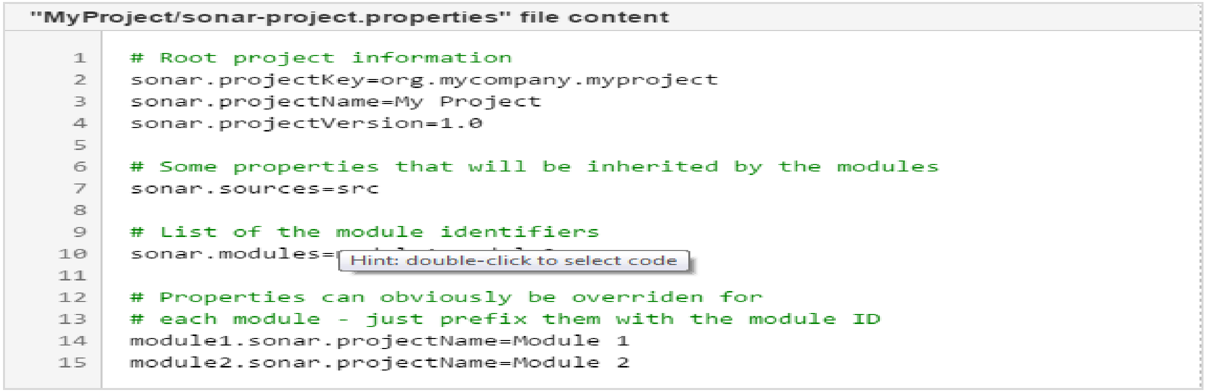
Analyse Source Code - Scanner

The following are the steps to execute "**Multi-Module Project**". There are two ways to define a multi-module structure in SonarQube

Way 1 - Set all the configuration in the properties file in the root folder



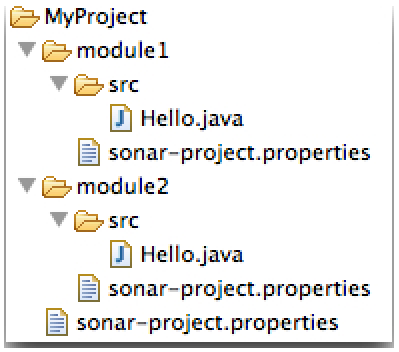
```
MyProject
  module1
    src
      Hello.java
  module2
    src
      Hello.java
  sonar-project.properties
```



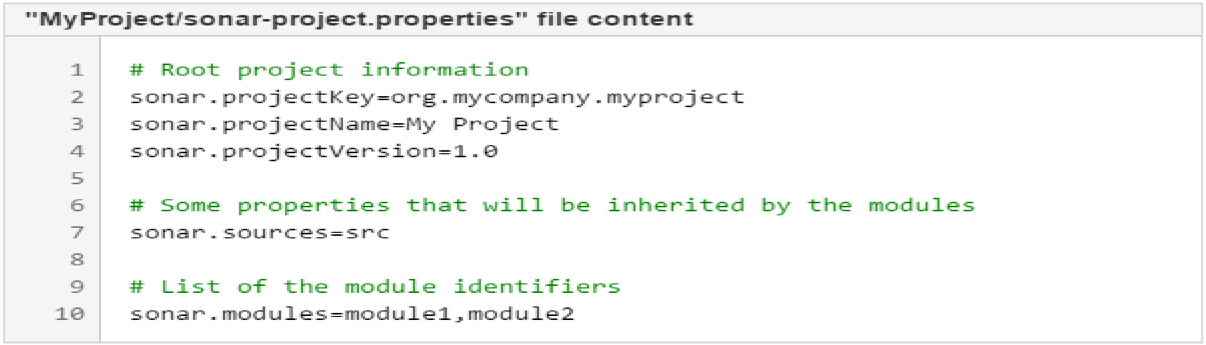
"MyProject/sonar-project.properties" file content

```
# Root project information
sonar.projectKey=org.mycompany.myproject
sonar.projectName=My Project
sonar.projectVersion=1.0

# Some properties that will be inherited by the modules
sonar.sources=src

# List of the module identifiers
sonar.modules=  Hint: double-click to select code

# Properties can obviously be overriden for
# each module - just prefix them with the module ID
module1.sonar.projectName=Module 1
module2.sonar.projectName=Module 2
```

Way 2 - Set all the configuration in the properties file in multiple folder

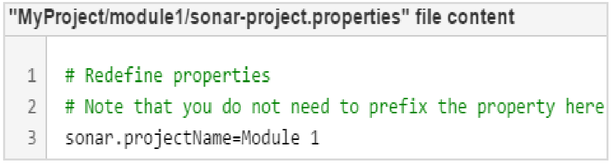


```
MyProject
  module1
    src
      Hello.java
    sonar-project.properties
  module2
    src
      Hello.java
    sonar-project.properties
  sonar-project.properties
```



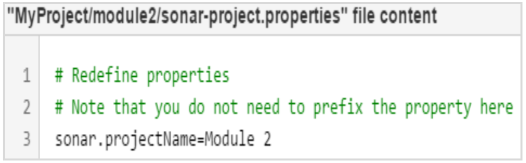
"MyProject/sonar-project.properties" file content

```
# Root project information
sonar.projectKey=org.mycompany.myproject
sonar.projectName=My Project
sonar.projectVersion=1.0

# Some properties that will be inherited by the modules
sonar.sources=src

# List of the module identifiers
sonar.modules=module1,module2
```



"MyProject/module1/sonar-project.properties" file content

```
# Redefine properties
# Note that you do not need to prefix the property here
sonar.projectName=Module 1
```

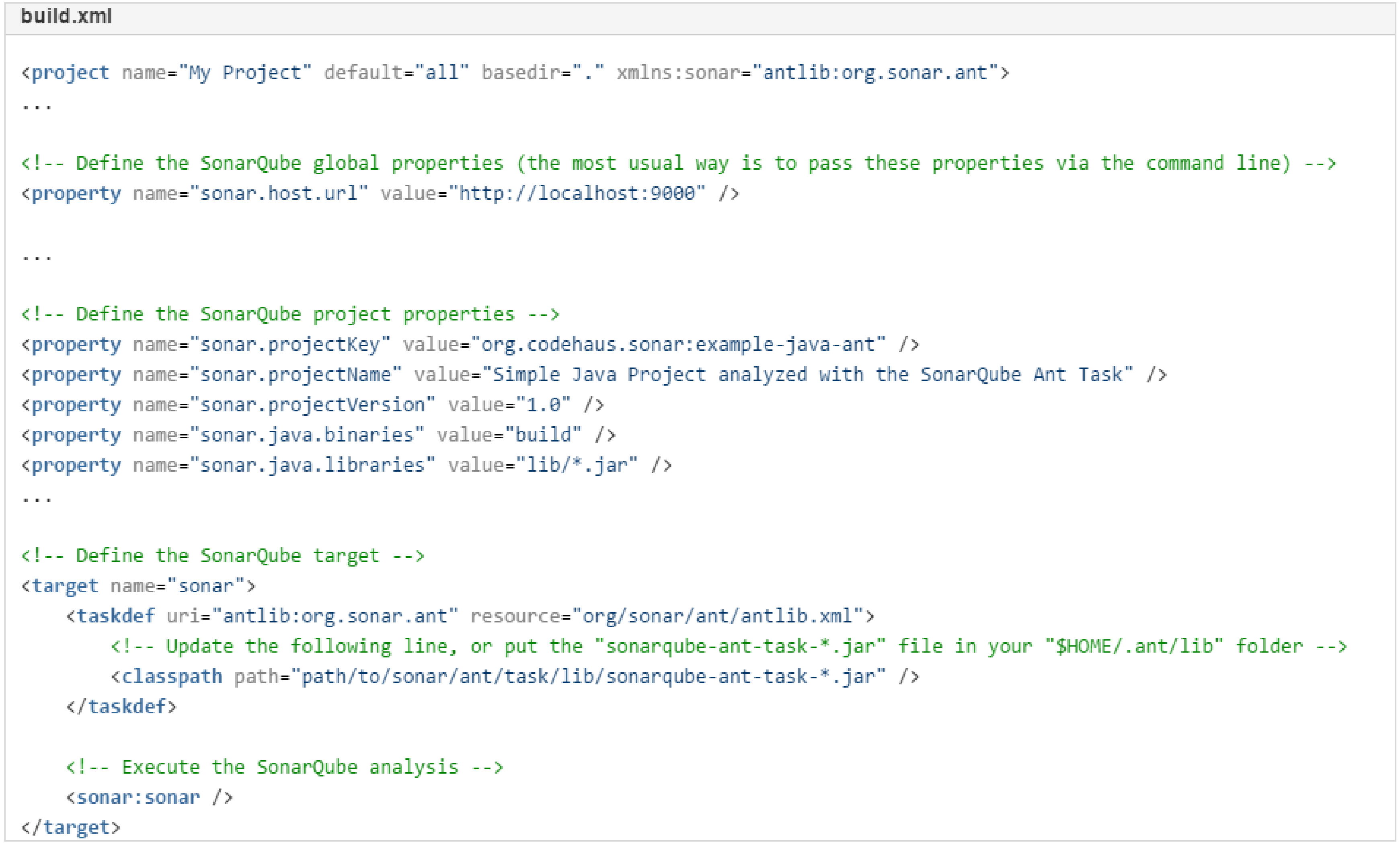


"MyProject/module2/sonar-project.properties" file content

```
# Redefine properties
# Note that you do not need to prefix the property here
sonar.projectName=Module 2
```

Analyse Source Code - ANT

The SonarQube Ant version 2.4 is compatible with SonarQube platform versions 4.5 and higher.
The following are the steps to execute "**Simple Project**"

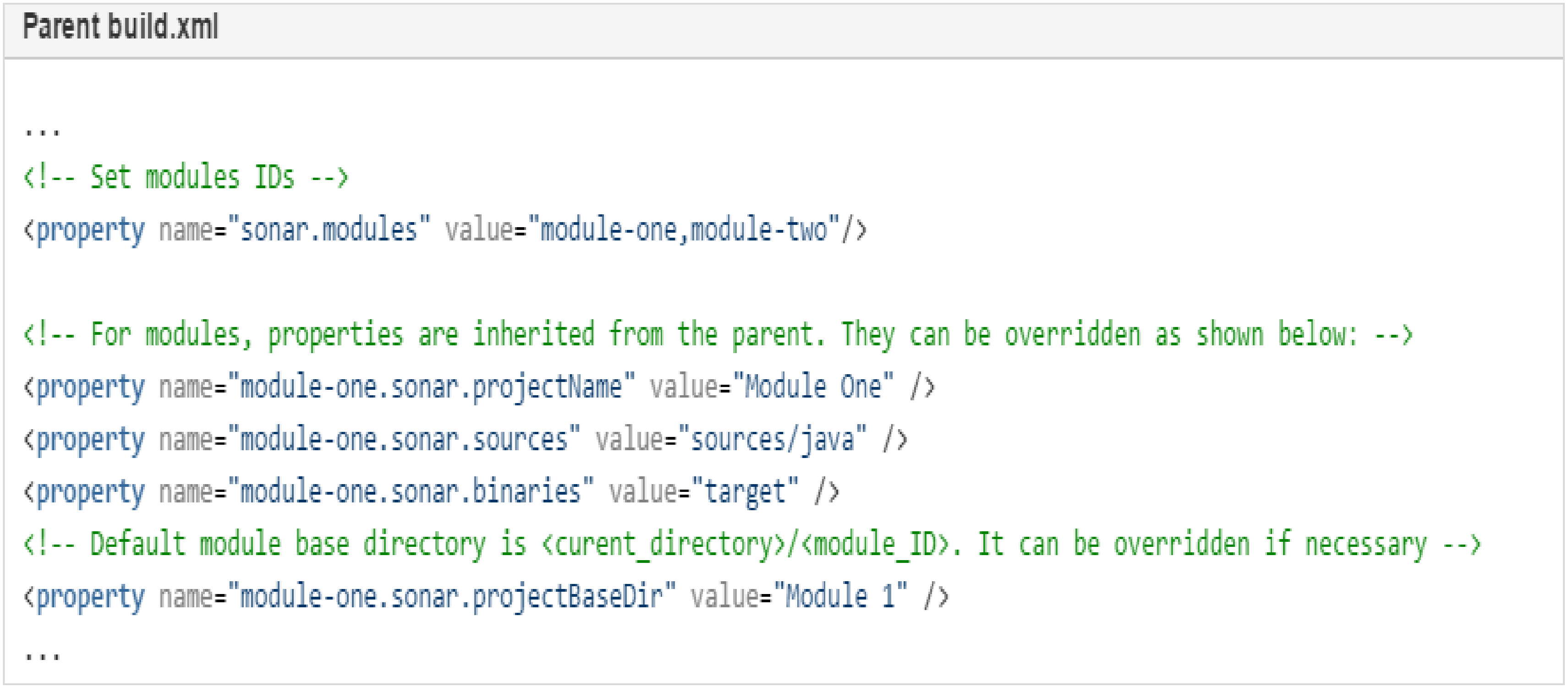


build.xml

```
<project name="My Project" default="all" basedir="." xmlns:sonar="antlib:org.sonar.ant">
...

<!-- Define the SonarQube global properties (the most usual way is to pass these properties via the command line) -->
<property name="sonar.host.url" value="http://localhost:9000" />


...

<!-- Define the SonarQube project properties -->
<property name="sonar.projectKey" value="org.codehaus.sonar:example-java-ant" />
<property name="sonar.projectName" value="Simple Java Project analyzed with the SonarQube Ant Task" />
<property name="sonar.projectVersion" value="1.0" />
<property name="sonar.java.binaries" value="build" />
<property name="sonar.java.libraries" value="lib/*.jar" />
...

<!-- Define the SonarQube target -->
<target name="sonar">
    <taskdef uri="antlib:org.sonar.ant" resource="org/sonar/ant/antlib.xml">
        <!-- Update the following line, or put the "sonarqube-ant-task-*.jar" file in your "$HOME/.ant/lib" folder -->
        <classpath path="path/to/sonar/ant/task/lib/sonarqube-ant-task-*.jar" />
    </taskdef>

    <!-- Execute the SonarQube analysis -->
    <sonar:sonar />
</target>
```

Analyse Source Code - ANT

The configuration for **parent and modules** must be done in the parent *build.xml* file:



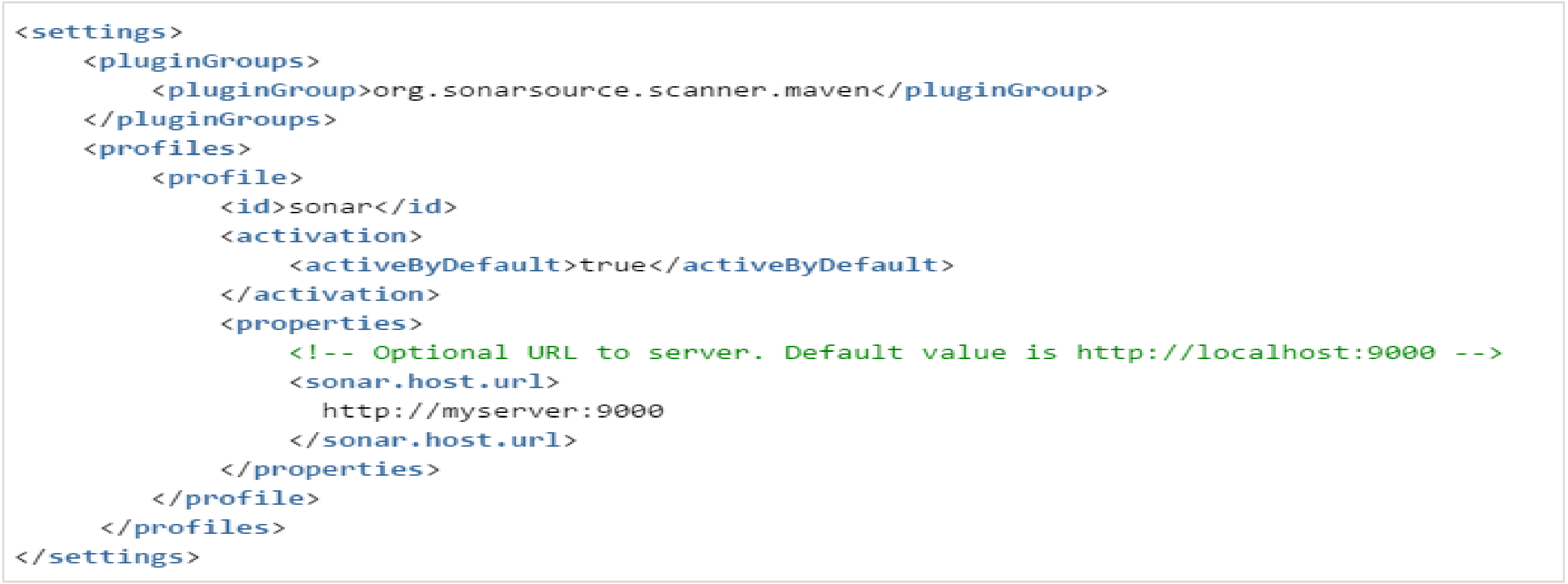
Parent build.xml

```
...
<!-- Set modules IDs -->
<property name="sonar.modules" value="module-one,module-two"/>

<!-- For modules, properties are inherited from the parent. They can be overridden as shown below: -->
<property name="module-one.sonar.projectName" value="Module One" />
<property name="module-one.sonar.sources" value="sources/java" />
<property name="module-one.sonar.binaries" value="target" />
<!-- Default module base directory is <curent_directory>/<module_ID>. It can be overridden if necessary -->
<property name="module-one.sonar.projectBaseDir" value="Module 1" />
...
```

Analyse Source Code - Maven

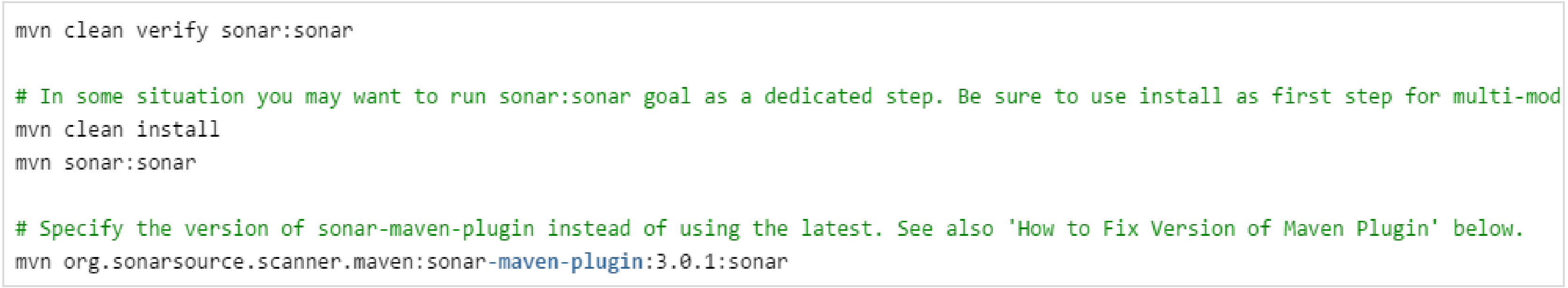
The SonarQube requires Maven 3 to launch analysis of the project:

Global Setting (Optional): Configure global Maven Setting to support SonarQube



```
<settings>
    <pluginGroups>
        <pluginGroup>org.sonarsource.scanner.maven</pluginGroup>
    </pluginGroups>
    <profiles>
        <profile>
            <id>sonar</id>
            <activation>
                <activeByDefault>true</activeByDefault>
            </activation>
            <properties>
                <!-- Optional URL to server. Default value is http://localhost:9000 -->
                <sonar.host.url>
                  http://myserver:9000
                </sonar.host.url>
            </properties>
        </profile>
     </profiles>
</settings>
```

Analysing Maven Project: Analysing a Maven project consists of running a Maven goal - sonar:sonar in the directory where the pom.xml file resides

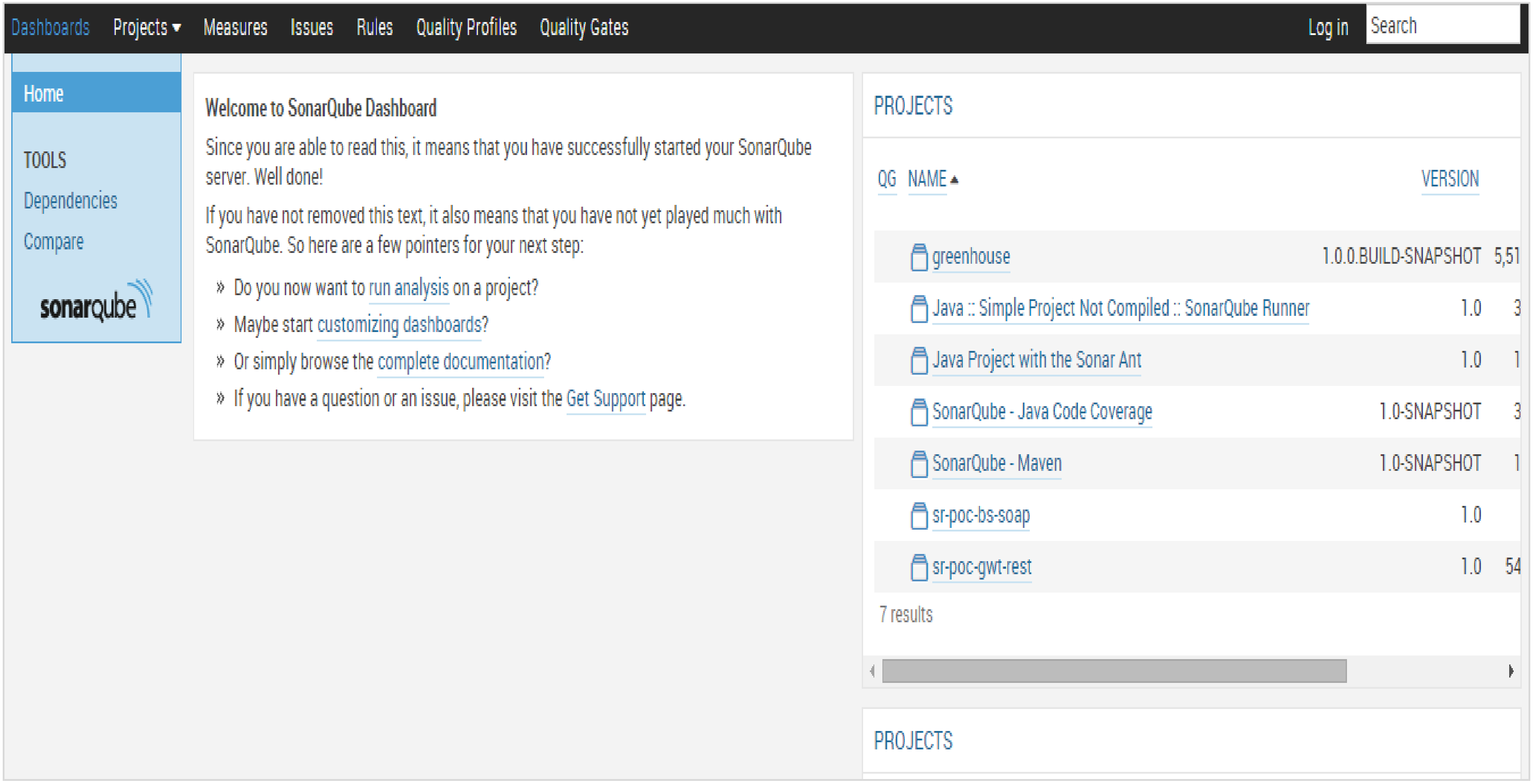


```
mvn clean verify sonar:sonar

# In some situation you may want to run sonar:sonar goal as a dedicated step. Be sure to use install as first step for multi-mod
mvn clean install
mvn sonar:sonar

# Specify the version of sonar-maven-plugin instead of using the latest. See also 'How to Fix Version of Maven Plugin' below.
mvn org.sonarsource.scanner.maven:sonar-maven-plugin:3.0.1:sonar
```

TABLE OF CONTENT (Day-2)

Dashboard

The following is the default view of the SonarQube. All user will view the below dashboard.



Welcome to SonarQube Dashboard

Since you are able to read this, it means that you have successfully started your SonarQube server. Well done!

If you have not removed this text, it also means that you have not yet played much with SonarQube. So here are a few pointers for your next step:

- Do you now want to run analysis on a project?
- Maybe start customizing dashboards?
- Or simply browse the complete documentation?
- If you have a question or an issue, please visit the Get Support page.

PROJECTS

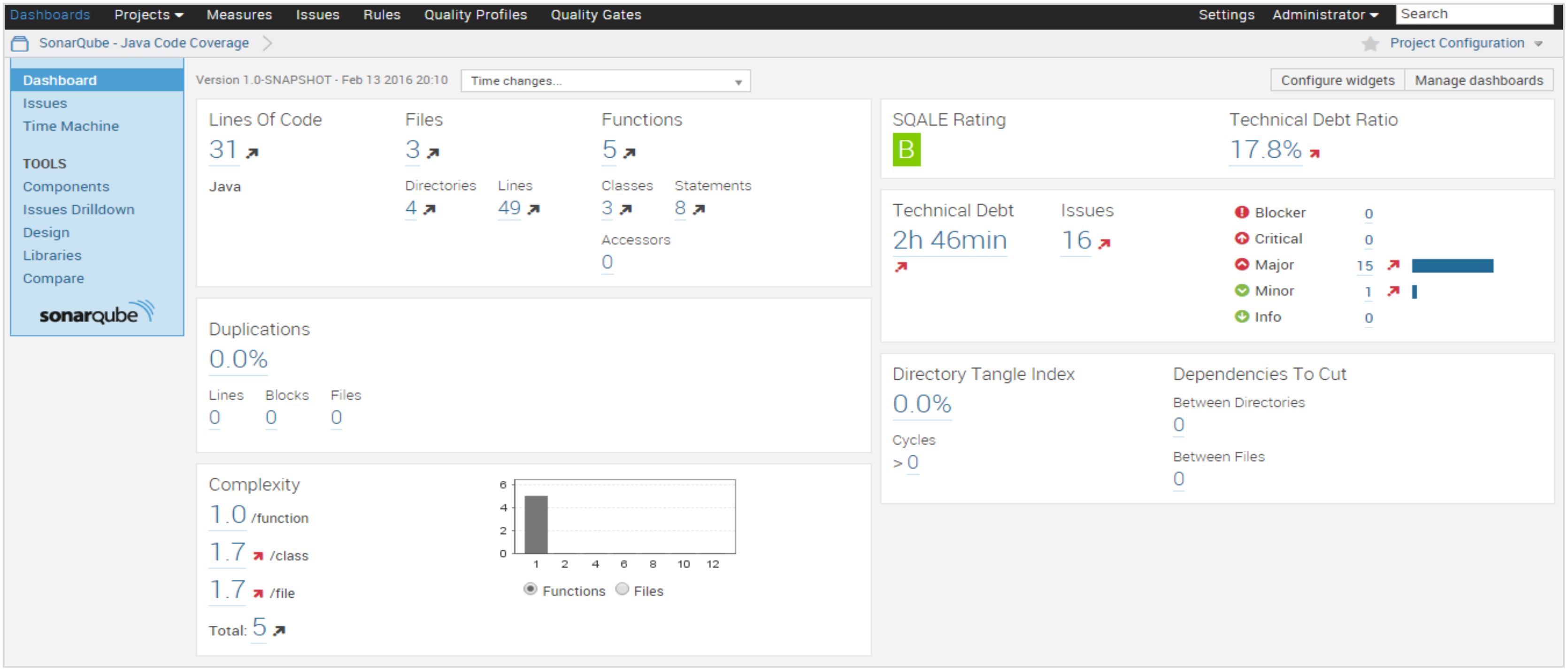
QG	NAME ▲	VERSION	
	greenhouse	1.0.0.BUILD-SNAPSHOT	5,51
	Java :: Simple Project Not Compiled :: SonarQube Runner	1.0	3
	Java Project with the Sonar Ant	1.0	1
	SonarQube - Java Code Coverage	1.0-SNAPSHOT	3
	SonarQube - Maven	1.0-SNAPSHOT	1
	sr-poc-bs-soap	1.0	
	sr-poc-gwt-rest	1.0	54

7 results

PROJECTS

Project View

The following is the default view of the project. All user / admin will view the below dashboard.



Dashboards | Projects ▾ | Measures | Issues | Rules | Quality Profiles | Quality Gates | Settings | Administrator ▾ | Search

SonarQube - Java Code Coverage > | Project Configuration ▾

Dashboard
Issues
Time Machine

TOOLS
Components
Issues Drilldown
Design
Libraries
Compare

sonarqube

Version 1.0-SNAPSHOT - Feb 13 2016 20:10 | Time changes... | Configure widgets | Manage dashboards

Lines Of Code: 31 ↗ — Java
Files: 3 ↗ — Directories 4 ↗, Lines 49 ↗
Functions: 5 ↗ — Classes 3 ↗, Statements 8 ↗, Accessors 0

SQALE Rating: B
Technical Debt Ratio: 17.8% ↗

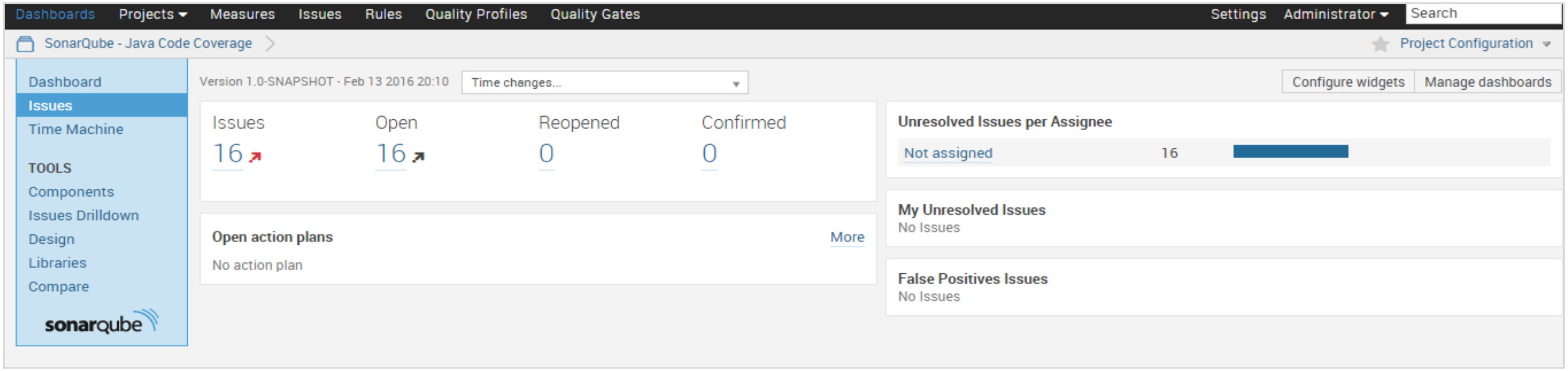
Technical Debt: 2h 46min ↗
Issues: 16 ↗

Blocker	0	
Critical	0	
Major	15	↗
Minor	1	↗
Info	0	

Duplications: 0.0% — Lines 0, Blocks 0, Files 0

Directory Tangle Index: 0.0% — Cycles > 0
Dependencies To Cut — Between Directories 0, Between Files 0

Complexity
1.0 /function
1.7 ↗ /class
1.7 ↗ /file
Total: 5 ↗



Dashboards | Projects ▾ | Measures | Issues | Rules | Quality Profiles | Quality Gates | Settings | Administrator ▾ | Search

SonarQube - Java Code Coverage > | Project Configuration ▾

Dashboard
Issues
Time Machine

TOOLS
Components
Issues Drilldown
Design
Libraries
Compare

sonarqube

Version 1.0-SNAPSHOT - Feb 13 2016 20:10 | Time changes... | Configure widgets | Manage dashboards

Issues	Open	Reopened	Confirmed
16 ↗	16 ↗	0	0

Open action plans — More
No action plan

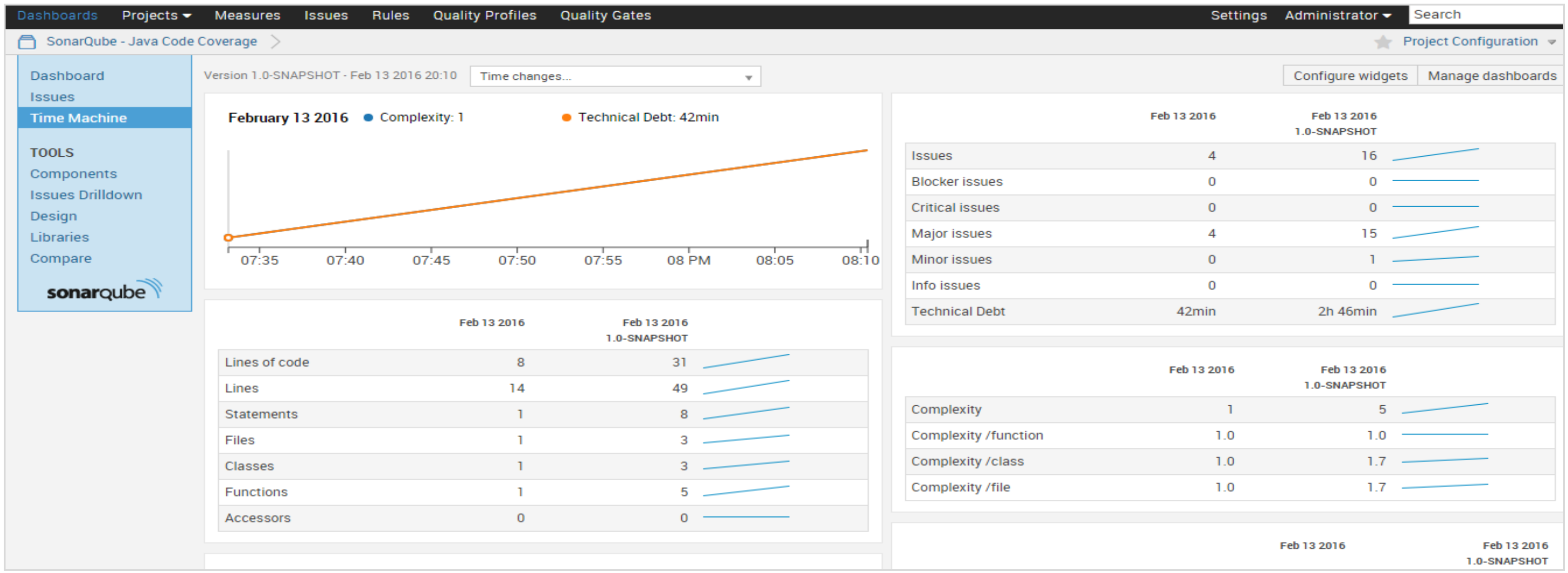
Unresolved Issues per Assignee
Not assigned 16

My Unresolved Issues
No Issues

False Positives Issues
No Issues

Project View

The following is the default view of the project. All user / admin will view the below dashboard.



Version 1.0-SNAPSHOT - Feb 13 2016 20:10

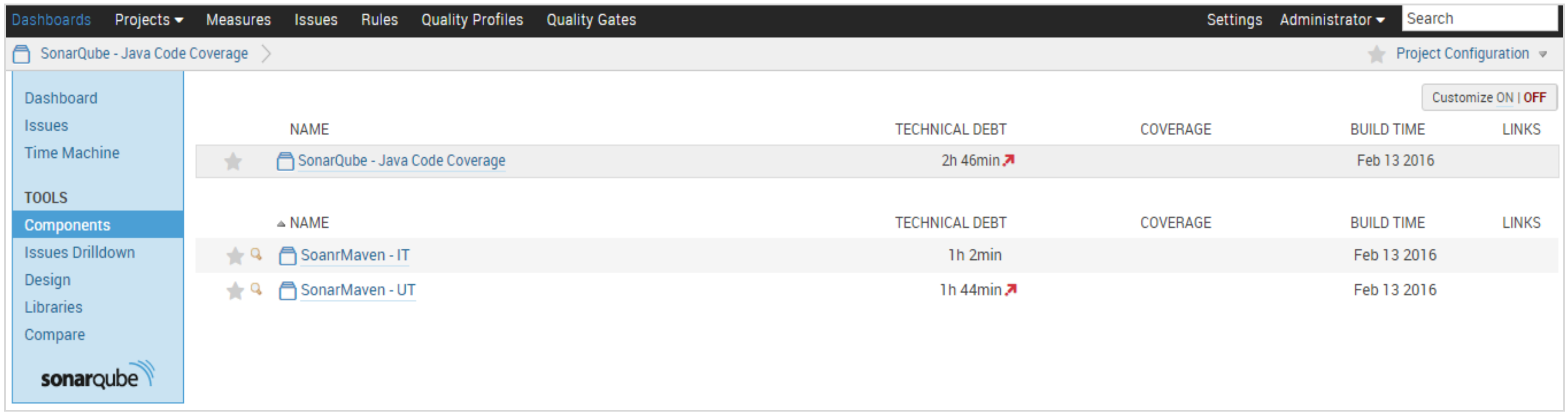
February 13 2016 ● Complexity: 1 ● Technical Debt: 42min

	Feb 13 2016	Feb 13 2016 1.0-SNAPSHOT
Issues	4	16
Blocker issues	0	0
Critical issues	0	0
Major issues	4	15
Minor issues	0	1
Info issues	0	0
Technical Debt	42min	2h 46min

	Feb 13 2016	Feb 13 2016 1.0-SNAPSHOT
Lines of code	8	31
Lines	14	49
Statements	1	8
Files	1	3
Classes	1	3
Functions	1	5
Accessors	0	0

	Feb 13 2016	Feb 13 2016 1.0-SNAPSHOT
Complexity	1	5
Complexity /function	1.0	1.0
Complexity /class	1.0	1.7
Complexity /file	1.0	1.7

Feb 13 2016 Feb 13 2016 1.0-SNAPSHOT

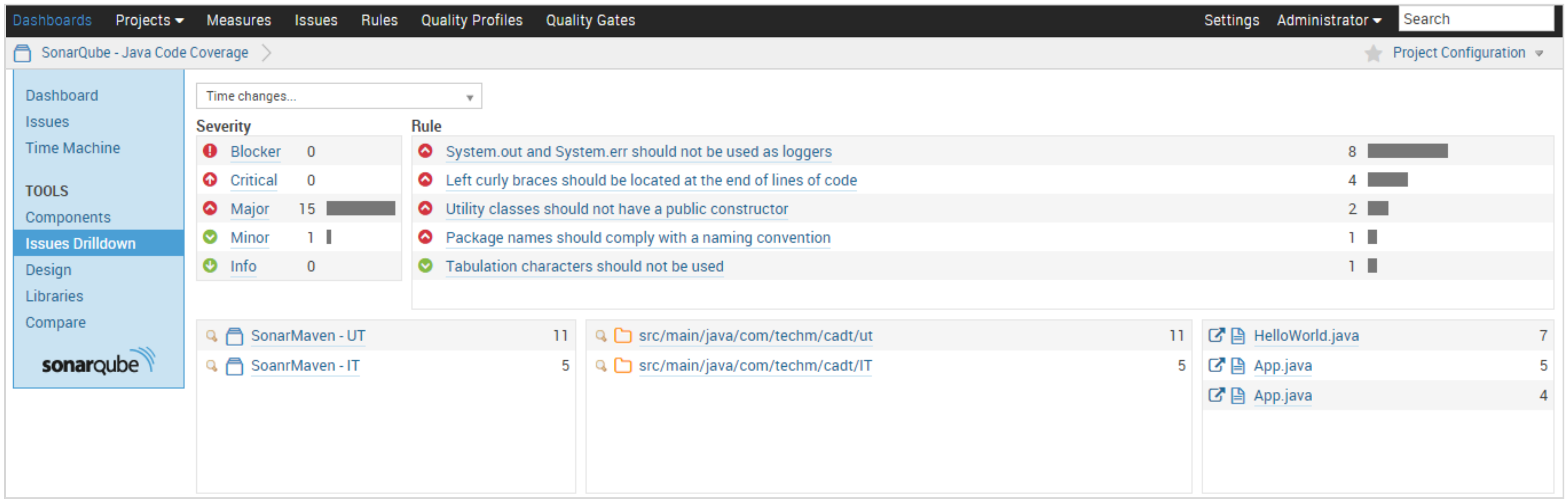


NAME	TECHNICAL DEBT	COVERAGE	BUILD TIME	LINKS
SonarQube - Java Code Coverage	2h 46min		Feb 13 2016	

NAME	TECHNICAL DEBT	COVERAGE	BUILD TIME	LINKS
SoanrMaven - IT	1h 2min		Feb 13 2016	
SonarMaven - UT	1h 44min		Feb 13 2016	

Project View

The following is the default view of the project. All user / admin will view the below dashboard.


Dashboards | Projects | Measures | Issues | Rules | Quality Profiles | Quality Gates | Settings | Administrator | Search
SonarQube - Java Code Coverage
Project Configuration
Dashboard
Issues
Time Machine
TOOLS
Components
Issues Drilldown
Design
Libraries
Compare
sonarqube
Time changes...

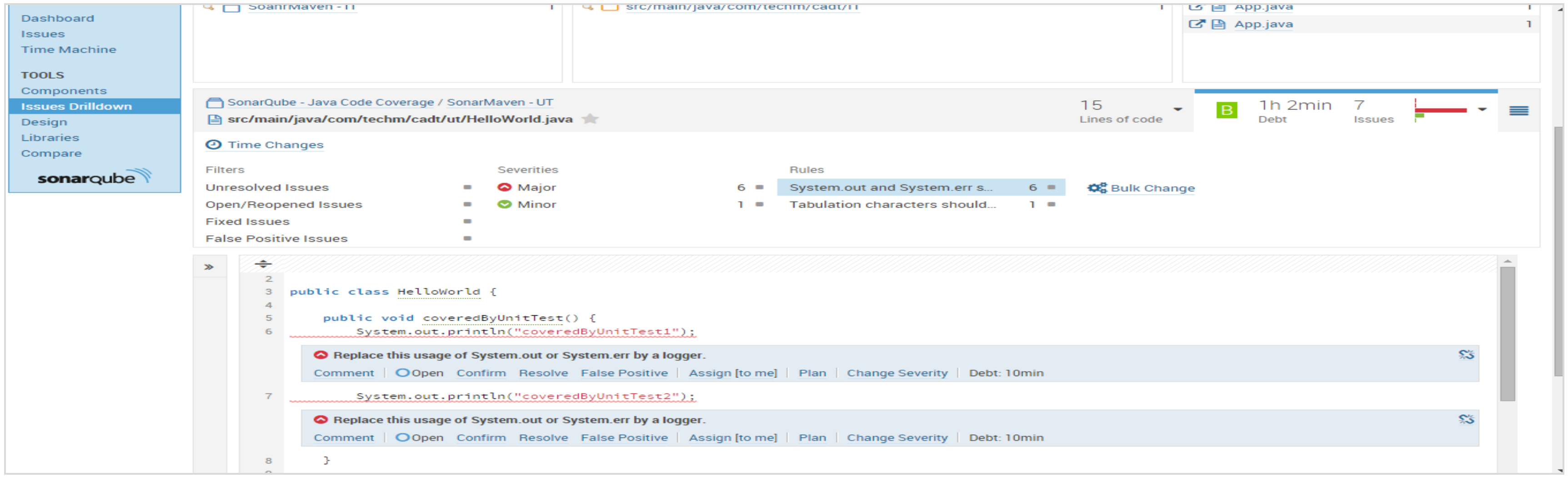
Severity	
Blocker	0
Critical	0
Major	15
Minor	1
Info	0

Rule	
System.out and System.err should not be used as loggers	8
Left curly braces should be located at the end of lines of code	4
Utility classes should not have a public constructor	2
Package names should comply with a naming convention	1
Tabulation characters should not be used	1

SonarMaven - UT	11
SoanrMaven - IT	5

src/main/java/com/techm/cadt/ut	11
src/main/java/com/techm/cadt/IT	5

HelloWorld.java	7
App.java	5
App.java	4

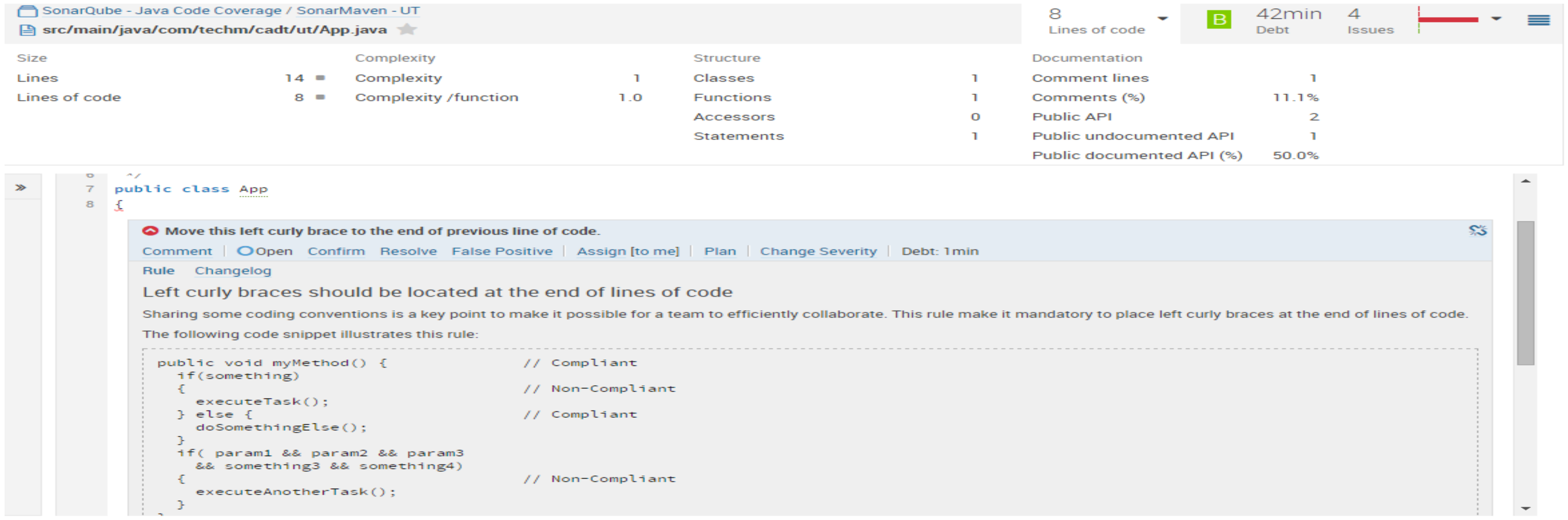


Filters		Severities		Rules	
Unresolved Issues		Major	6	System.out and System.err s...	6
Open/Reopened Issues		Minor	1	Tabulation characters should...	1
Fixed Issues					
False Positive Issues					

```
public class HelloWorld {

    public void coveredByUnitTest() {
        System.out.println("coveredByUnitTest1");
```
Replace this usage of System.out or System.err by a logger.
Comment | Open | Confirm | Resolve | False Positive | Assign [to me] | Plan | Change Severity | Debt: 10min
```
        System.out.println("coveredByUnitTest2");
```
Replace this usage of System.out or System.err by a logger.
Comment | Open | Confirm | Resolve | False Positive | Assign [to me] | Plan | Change Severity | Debt: 10min
```
    }
```

Project View

The following is the default view of the project. All user / admin will view the below dashboard.



SonarQube - Java Code Coverage / SonarMaven - UT
src/main/java/com/techm/cadt/ut/App.java

8 Lines of code | B 42min Debt | 4 Issues

Size		Complexity		Structure		Documentation	
Lines	14	Complexity	1	Classes	1	Comment lines	1
Lines of code	8	Complexity /function	1.0	Functions	1	Comments (%)	11.1%
				Accessors	0	Public API	2
				Statements	1	Public undocumented API	1
						Public documented API (%)	50.0%

```
*/
public class App
{
```

Move this left curly brace to the end of previous line of code.

Comment | Open Confirm Resolve False Positive | Assign [to me] | Plan | Change Severity | Debt: 1min

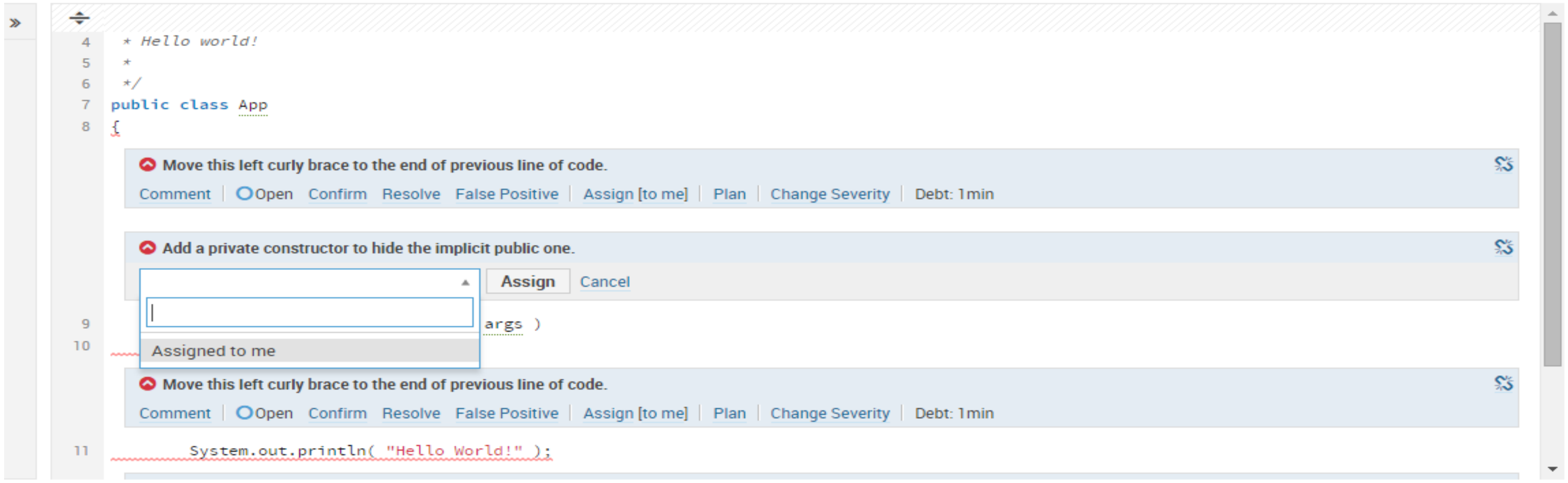
Rule Changelog

Left curly braces should be located at the end of lines of code

Sharing some coding conventions is a key point to make it possible for a team to efficiently collaborate. This rule make it mandatory to place left curly braces at the end of lines of code.

The following code snippet illustrates this rule:

```
public void myMethod() {                    // Compliant
  if(something)
  {                                         // Non-Compliant
    executeTask();
  } else {                                  // Compliant
    doSomethingElse();
  }
  if( param1 && param2 && param3
    && something3 && something4)
  {                                         // Non-Compliant
    executeAnotherTask();
  }
```



```
 * Hello world!
 *
 */
public class App
{
```

Move this left curly brace to the end of previous line of code.

Comment | Open Confirm Resolve False Positive | Assign [to me] | Plan | Change Severity | Debt: 1min

Add a private constructor to hide the implicit public one.

Assign Cancel

Assigned to me

```
args )
```

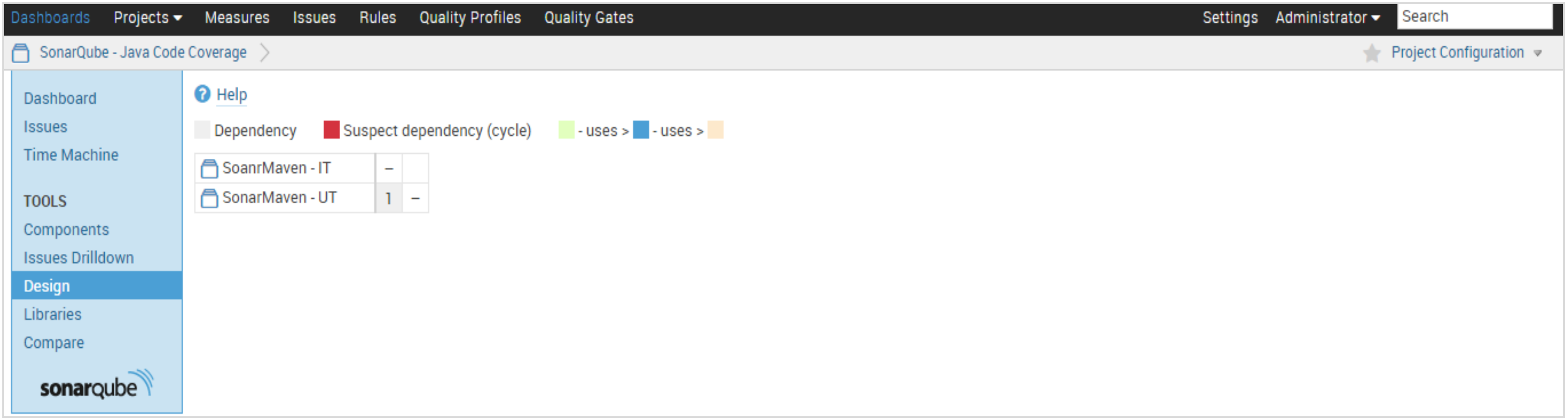
Move this left curly brace to the end of previous line of code.

Comment | Open Confirm Resolve False Positive | Assign [to me] | Plan | Change Severity | Debt: 1min

```
        System.out.println( "Hello World!" );
```

Project View

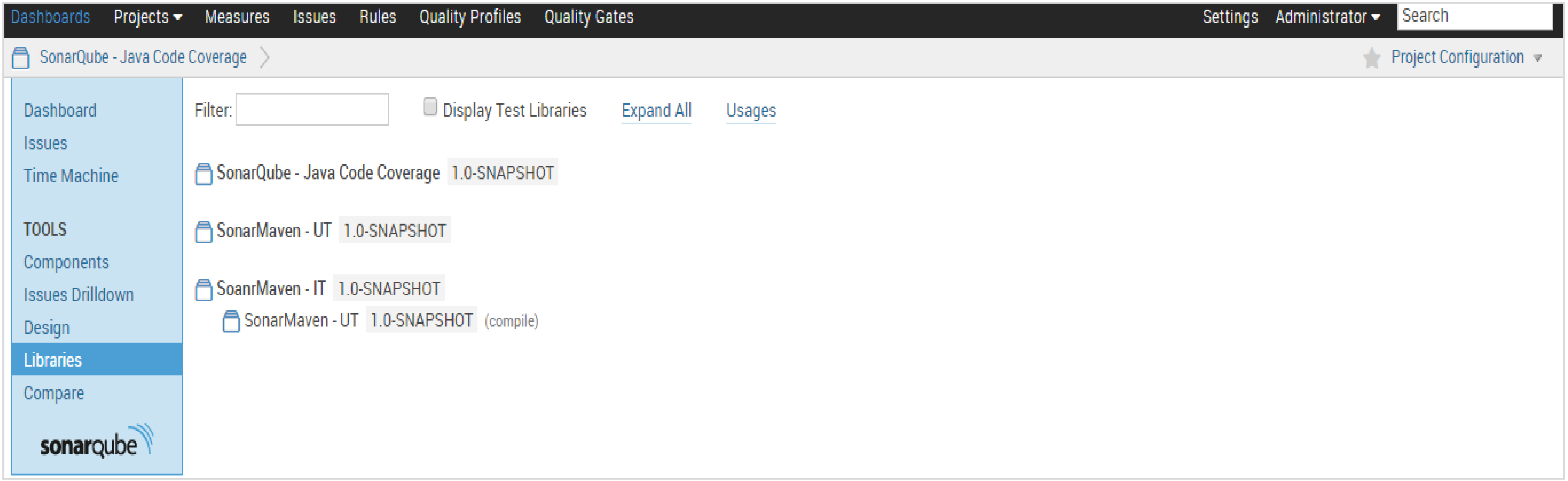
The following is the default view of the project. All user / admin will view the below dashboard.



Dashboards Projects Measures Issues Rules Quality Profiles Quality Gates Settings Administrator Search
SonarQube - Java Code Coverage
Project Configuration
Dashboard
Issues
Time Machine
TOOLS
Components
Issues Drilldown
Design
Libraries
Compare
sonarqube
Help

Dependency 🟥 Suspect dependency (cycle) 🟩 - uses > 🟦 - uses > 🟧

SoanrMaven - IT	–	
SonarMaven - UT	1	–



SonarQube - Java Code Coverage 1.0-SNAPSHOT

SonarMaven - UT 1.0-SNAPSHOT

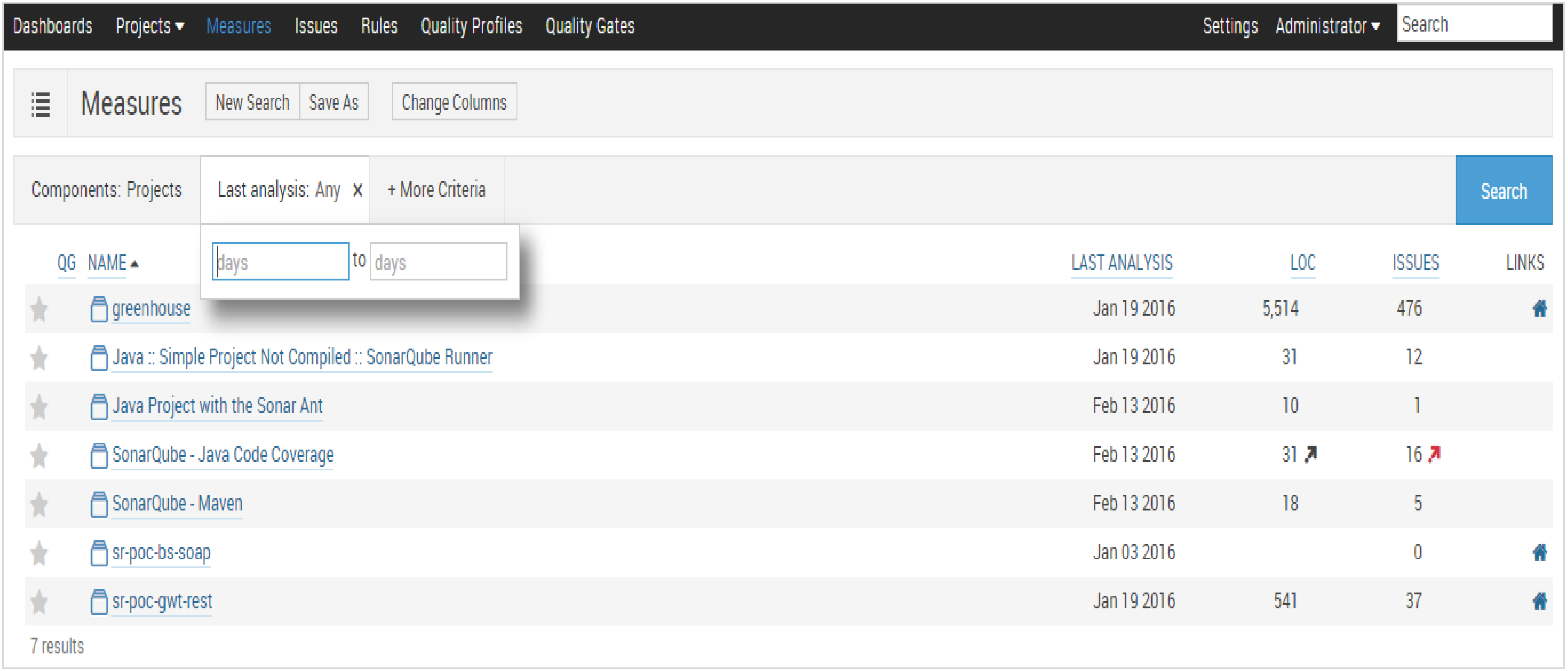
SoanrMaven - IT 1.0-SNAPSHOT
  SonarMaven - UT 1.0-SNAPSHOT (compile)

Project Measure

The Measures service provides a way to quickly execute all kinds of queries on project measures.

Example:

- the recently inspected projects
- the projects with blocker and critical issues
- the projects with bad coverage on added/changed code within the 10 past days
- The search query can then be saved as a filter to be displayed on dashboards.



QG	NAME	LAST ANALYSIS	LOC	ISSUES	LINKS
	greenhouse	Jan 19 2016	5,514	476	
	Java :: Simple Project Not Compiled :: SonarQube Runner	Jan 19 2016	31	12	
	Java Project with the Sonar Ant	Feb 13 2016	10	1	
	SonarQube - Java Code Coverage	Feb 13 2016	31	16	
	SonarQube - Maven	Feb 13 2016	18	5	
	sr-poc-bs-soap	Jan 03 2016		0	
	sr-poc-gwt-rest	Jan 19 2016	541	37	

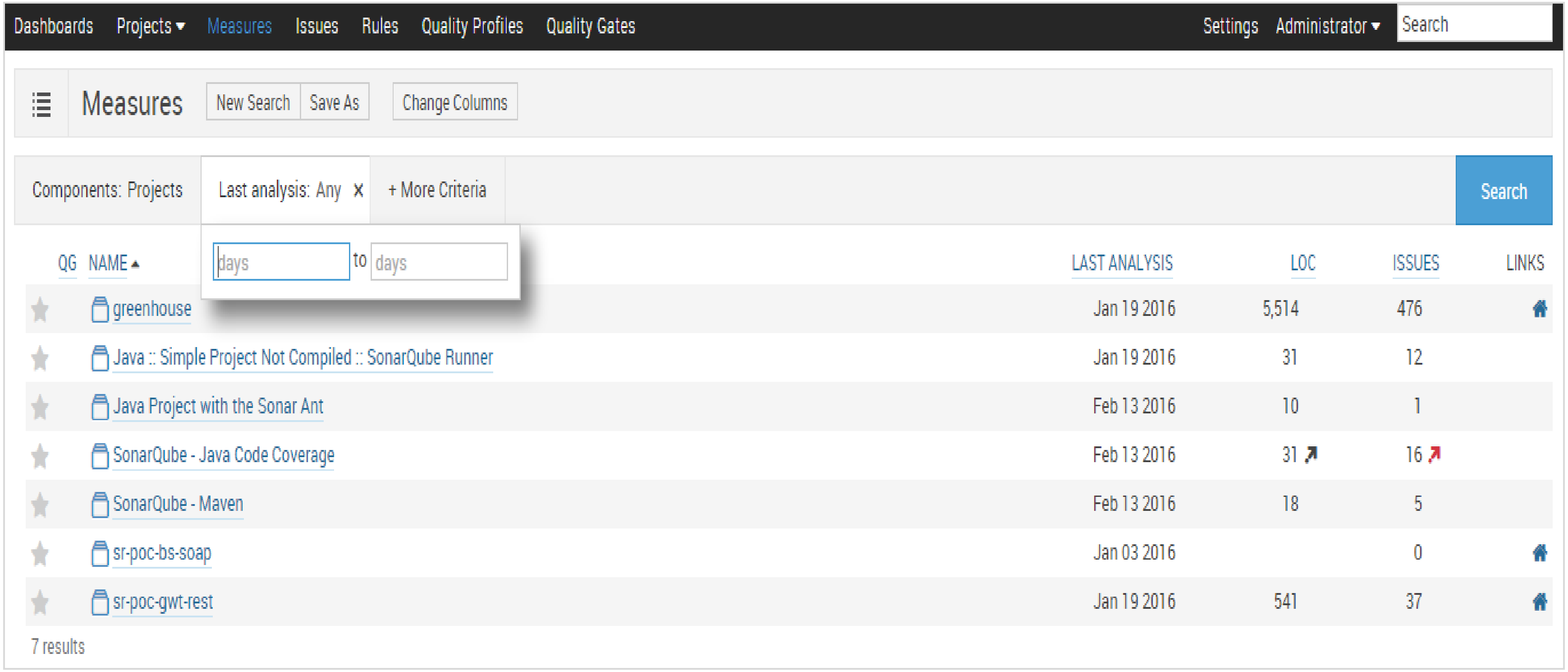
7 results

Project Measure

The Measures service provides a way to quickly execute all kinds of queries on project measures.

Example:

- the recently inspected projects
- the projects with blocker and critical issues
- the projects with bad coverage on added/changed code within the 10 past days
- The search query can then be saved as a filter to be displayed on dashboards.



QG	NAME	LAST ANALYSIS	LOC	ISSUES	LINKS
	greenhouse	Jan 19 2016	5,514	476	
	Java :: Simple Project Not Compiled :: SonarQube Runner	Jan 19 2016	31	12	
	Java Project with the Sonar Ant	Feb 13 2016	10	1	
	SonarQube - Java Code Coverage	Feb 13 2016	31	16	
	SonarQube - Maven	Feb 13 2016	18	5	
	sr-poc-bs-soap	Jan 03 2016		0	
	sr-poc-gwt-rest	Jan 19 2016	541	37	

7 results

Jenkins - SonarQube Process Flow

Sonar

- Install Sonar and Sonar Runner
- Create Users and Groups in Sonar server using administration login
- Assign projects to the sonar by using Sonar runner
- Assign Rules for the project
- Assign Users and Groups for the project to access

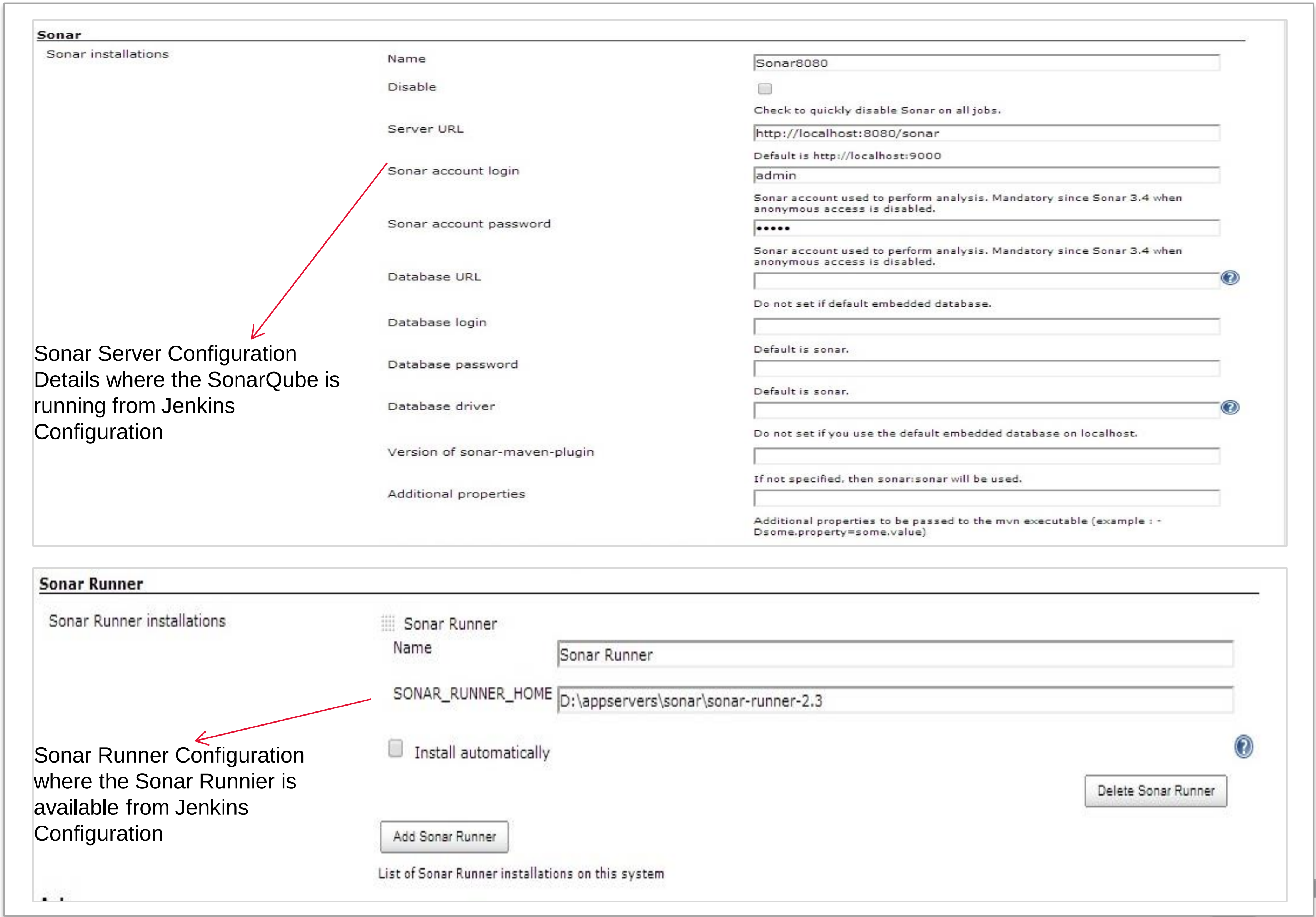
Jenkins

- Install Jenkins
- Add Sonar plugin in Jenkins plugin
- Create people OR assign LDAP to the Jenkins
- Configure Sonar and Sonar Runner in Jenkins
- Assign jobs to the people for execution
- Create Job and assign Sonar runner / ANT / Maven task to execute sonar rules

Sonar Reports

- Login to Sonar server with user details
- View the project code review report in Sonar Dashboard

Jenkins and Sonar Configuration



Sonar

Sonar installations

Name	Sonar8080
Disable	☐ Check to quickly disable Sonar on all jobs.
Server URL	http://localhost:8080/sonar Default is http://localhost:9000
Sonar account login	admin Sonar account used to perform analysis. Mandatory since Sonar 3.4 when anonymous access is disabled.
Sonar account password	••••• Sonar account used to perform analysis. Mandatory since Sonar 3.4 when anonymous access is disabled.
Database URL	Do not set if default embedded database.
Database login	Default is sonar.
Database password	Default is sonar.
Database driver	Do not set if you use the default embedded database on localhost.
Version of sonar-maven-plugin	If not specified, then sonar:sonar will be used.
Additional properties	Additional properties to be passed to the mvn executable (example : -Dsome.property=some.value)

Sonar Server Configuration Details where the SonarQube is running from Jenkins Configuration

Sonar Runner

Sonar Runner installations

Sonar Runner

Name	Sonar Runner
SONAR_RUNNER_HOME	D:\appservers\sonar\sonar-runner-2.3

☐ Install automatically

List of Sonar Runner installations on this system

Sonar Runner Configuration where the Sonar Runnier is available from Jenkins Configuration

Thank You