

MET Sonarqube and SonarLint

Moderne Softwareentwicklung

Inggita Arundina

BHT Berlin

1. Installation

1.1 Sonarqube

For this task, I chose Sonarqube and Sonarlint. First of all, I downloaded Sonarqube and I followed this installation process from this site


<https://docs.sonarqube.org/latest/setup/install-server/>. I had to do some configuration and edited sonar.properties as it is written in the documentation.

```
0 # address will be used for listening on the specified port
1 # By default, ports will be used on all IP addresses asso
2 sonar.web.host=192.168.0.1
3 sonar.web.port=80
4 sonar.web.context=/sonarqube
5
6 # Web context. When set, it must start with forward slash
7 # The default value is root context (empty value).
8
9 # Paths to persistent data files (embedded database and search in
10 # Can be absolute or relative to installation directory.
11 # Defaults are respectively <installation home>/data and <install
12 sonar.path.data=/var/sonarqube/data
13 sonar.path.temp=/var/sonarqube/temp
14
15 # Telemetry - Share anonymous SonarQube statistics
16 # By sharing anonymous SonarQube statistics, you help us understa
17 # We don't collect source code or IP addresses. And we don't shar
```

Once I finished, I started the server by running this command on my terminal
“bin/macosex-universal-64/sonar.sh start”. Then, I could browse on the browser
<http://localhost:9000/> and login as default in admin/admin

```
inggitas-MBP: inggita.arundina$ docker pull sonarqube
-bash: docker: command not found
[Inggitas-MBP:~ inggita.arundina$ cd /Users/inggita.arundina/Downloads/sonarqube-]
9.1.0.47736
Inggitas-MBP:sonarqube-9.1.0.47736 inggita.arundina$ bin/macosx-universal-64/son
ar.sh start
Starting SonarQube...
Started SonarQube.
Inggitas-MBP:sonarqube-9.1.0.47736 inggita.arundina$
```

Log In to SonarQube





Login 

Password


[Log in](#) [Cancel](#)

How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)? Create your project from your favorite DevOps platform.
First, you need to set up a DevOps platform configuration.

 From Azure DevOps Set up global configuration	 From Bitbucket Set up global configuration	 From GitHub Set up global configuration	 From GitLab Set up global configuration
--	---	--	--

Are you just testing or have an advanced use-case? Create a project manually.


Manually

I selected manually and could create a new project which I named ProjectTest. The installation run smoothly and it was working fine. I connected to a dummy project using Gradle that contains only a few classes. When I run it, it returns 0 bugs and 7 code smells. Because it is only an easy project, that's why it doesn't have a lot of reports in the dashboard.

ProjectTest ☆ master

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

Analyze your project

We initialized your project on SonarQube, now it's up to you to launch analyses!

- Provide a token ✓ d62f72150e41292e59f7aaf12000e863fae60d3
- Run analysis on your project

What option best describes your build?

Maven **Gradle** .NET Other (for JS, TS, Go, Python, PHP, ...)

Execute the Scanner for Gradle

Running an analysis with Gradle is straightforward. You just need to declare the `org.sonarqube` plugin in your `build.gradle` file:

```
plugins {
    id "org.sonarqube" version "3.3"
}
```

You can find the latest version of the Gradle plugin [here](#).

and run the following command:

```
./gradlew sonarqube \
    -Dsonar.projectKey=ProjectTest \
    -Dsonar.host.url=http://localhost:9000 \
    -Dsonar.login=d62f72150e41292e59f7aaf12000e863fae60d3
```

Please visit the [official documentation of the Scanner for Gradle](#) for more details.

Is my analysis done? If your analysis is successful, this page will automatically refresh in a few moments.

QUALITY GATE STATUS

MEASURES

New Code Overall Code

0	Bugs	Reliability	A
0	Vulnerabilities	Security	A
0	Security Hotspots	Security Review	A
31min	Code Smells	Maintainability	A
0.0%	Coverage on 34 Lines to cover	Unit Tests	4
0.0%	Duplications on 87 Lines	Duplicated Blocks	0

ACTIVITY

Passed

Reliability (Bugs)

Security (Vulnerabilities)

Security Review (Security Hotspots)

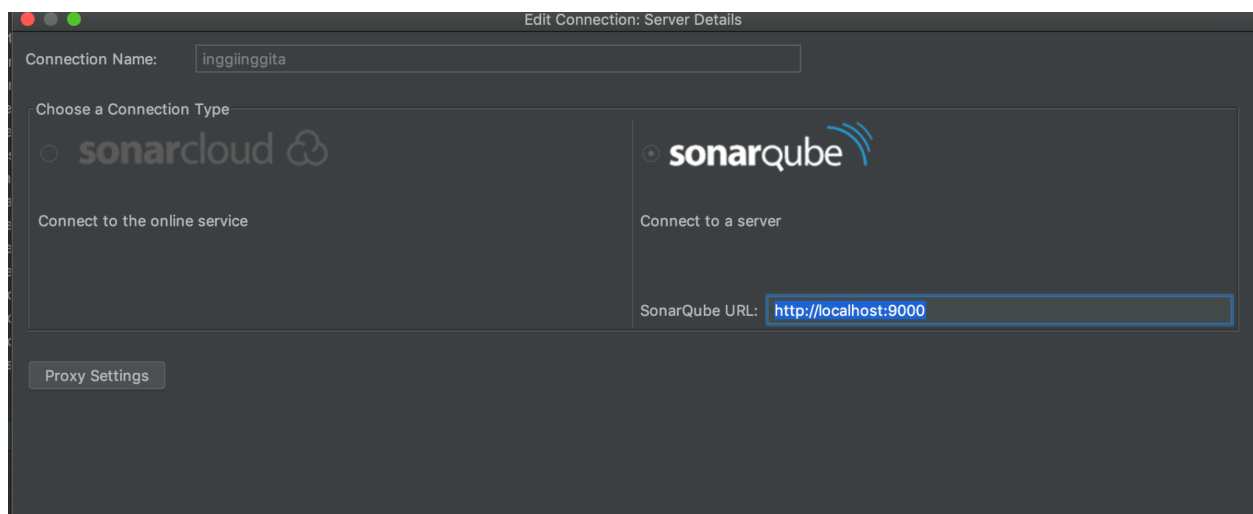
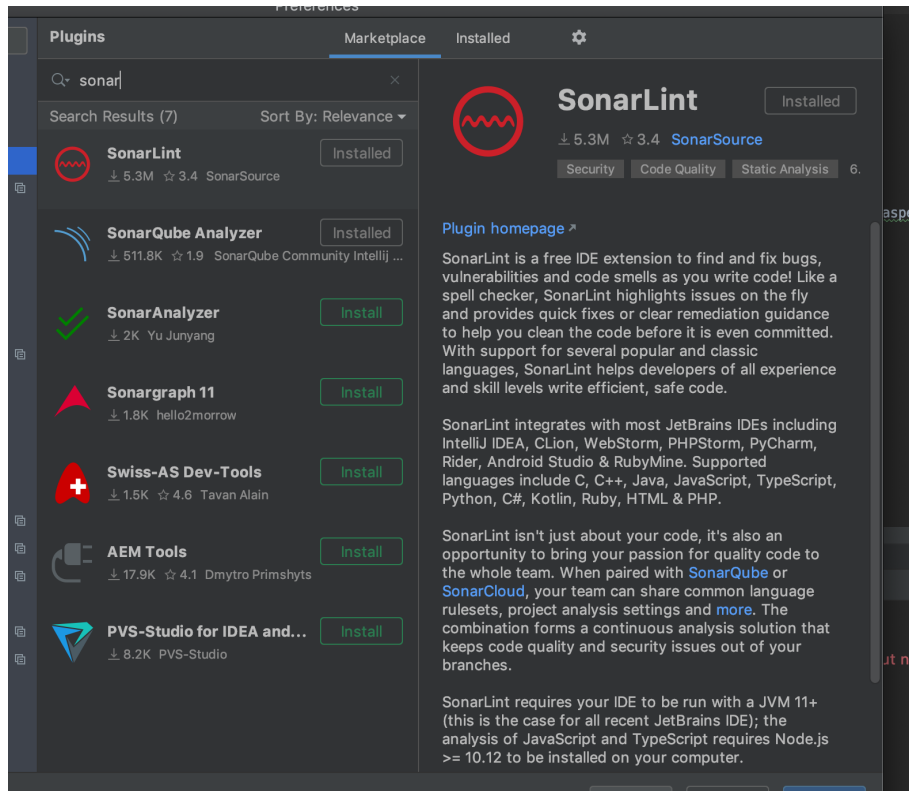
Maintainability (Code Smells)

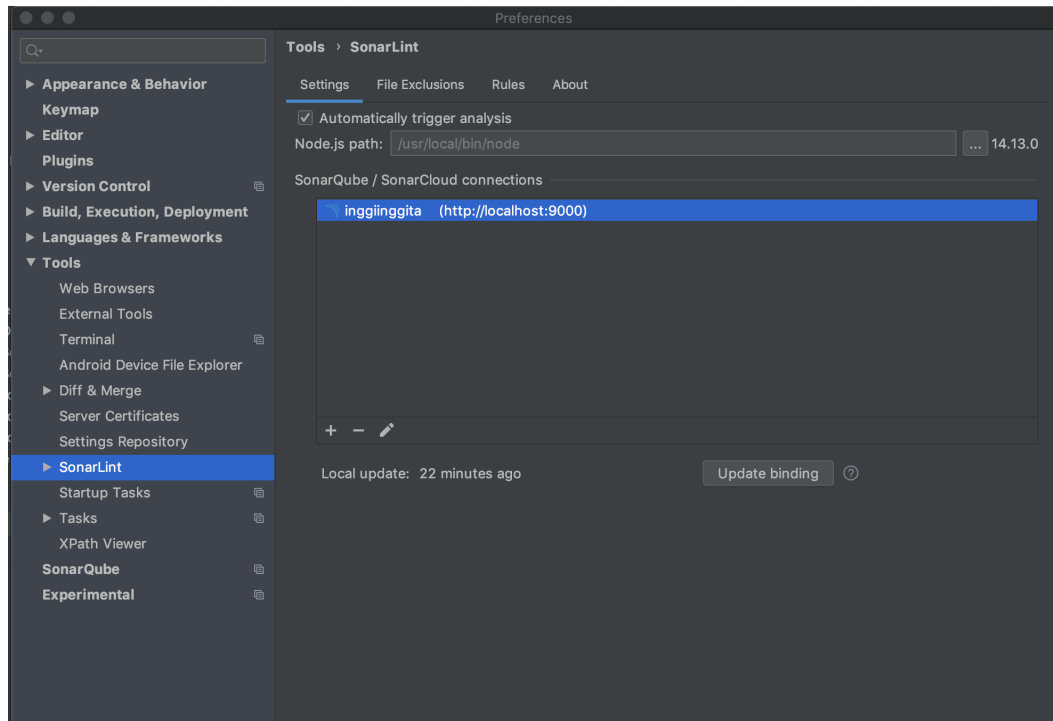
1 of 1 shown

1.2. Sonarlint

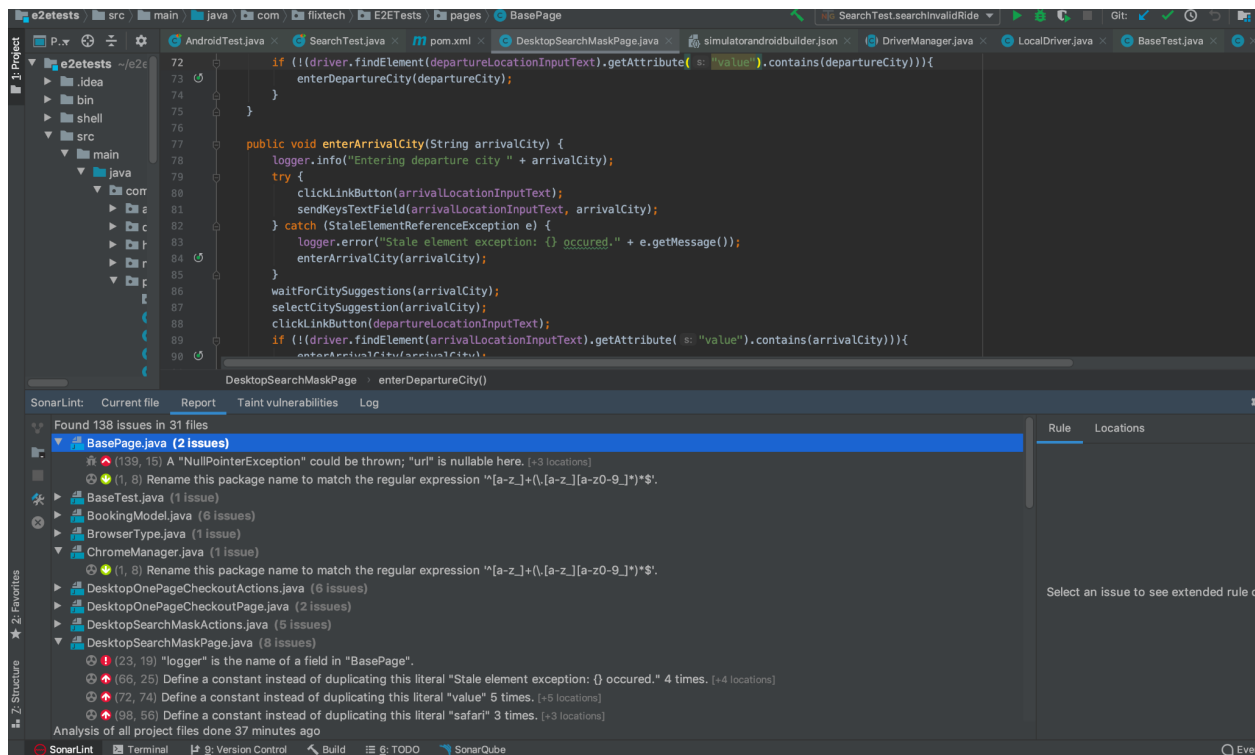
During integration, I found out there is a plugin that can be connected to IntelliJ <https://www.sonarlint.org/intellij>. I already have an automation project(Maven Project by using Selenium/Appium for building end-to-end testing for Web and an Android app) in my

IntelliJ file. So I tried it out for this tool too. First I needed to go to project preferences and go to settings. There I added plugins SonarLint. After that, I selected SonarLint under tools and chose Sonarqube to my localhost <http://localhost:9000>. Then I copy-pasted the token that I got on the Sonarqube dashboard as I explained in chapter 1.1.





After my project is integrated with SonarLint, then I ran the test by tapping Analyze > Analyze All Files with SonarLint. And I was surprised as I got a lot of reports from SonarLint as follows:



2. Conclusion

I personally prefer Sonarlint that is integrated by plugin directly to IntelliJ. The reason because I would prefer to see the report directly on my code. I don't want to have another dashboard to see my code analysis. As I am doing my automation project, I need an emulator, selenium/appium dashboard, and terminal to connect to my project. If I had another dashboard it would give more complexity to work on my project :) So, I prefer to use Sonarlint for my code analysis. And this is something that I want to integrate into all the projects that I have. I am glad that I found this tool. Very useful and easy to integrate.