Apache Commons-email

Luigi Allocca, Simone Della Porta, Rocco Iuliano

ACM Reference Format:
Luigi Allocca, Simone Della Porta, Rocco Iuliano. 2023. Apache Commons-email. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 CONTEXT OF THE PROJECT

Apache Commons-Email aims to provide a API for sending email. It is built on top of the Java Mail API, which it aims to simplify. This project is structured in 3 package:

(1) **org.apache.commons.mail** that aims to provide a API for sending email;
(2) **org.apache.commons.mail.resolver** that contains implementation classes to resolve data sources from the following locations: class path file system URL (??);
(3) **org.apache.commons.mail.util** that contains some utility classes.

2 PRELIMINAR ANALISYS

We choose a project with the following characteristics in order to improve the dependability of the software with CI/CD paradigm using the tools introduced in the course:

- Git Actions to check code quality, coverage, Java CI and security.
- The project should use Maven to manage the project build, so it should have the pom.xml file.

After selecting the project Commons-email, we have created a fork of the repository, cloned the repository and built the project in order to run all test cases which result passed successfully. Then we have conducted a preliminar analisys of the project by using *sonarcloud* [1] and *codecov* [2] and we obtained this results:

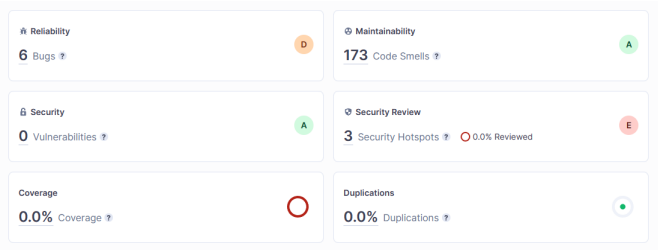

Figure 1: Sonarcloud analysis

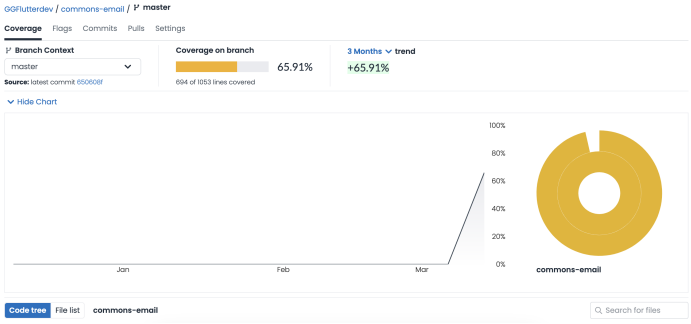

Figure 2: Codecov analysis

The result of the analysis are:

(1) The coverage is 65,91%;
(2) The project has 6 bugs, of which 2 critic and 4 major;
(3) 3 security hotspot;
(4) 173 code smells.

So for having a continuous analisys of the project we decided to integrate these tools in the our project in order to analyze the code after every push or pull request.

3 GOALS OF THE PROJECT

After the review of the analysis results, we will focus on the following aspects in order to improve the software dependability:

- fix as many project bugs as possible, prioritizing the crucial bugs;
- improve the coverage of project testing by developing new test cases and improving the existing ones;
- minimize the number of code smells;
- reduce security hotspot issues;
- verify the project performance.

4 METHODOLOGICAL STEPS CONDUCTED TO ADDRESS THE GOALS

5 RESULTS AND FINDINGS

6 CONCLUSIONS

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn