10 January 2025
Prof. Dario Di Nucci
Software Dependability
University of Salerno

APACHE COMMONS-IMAGING LIBRARY

Dependability Analysis

Presented by

Johan Chicue Garcia
Martin Esteban Cardaci

Content

1. Objective

3. SonarCloud: Analysis and Results

4. Docker Implementation

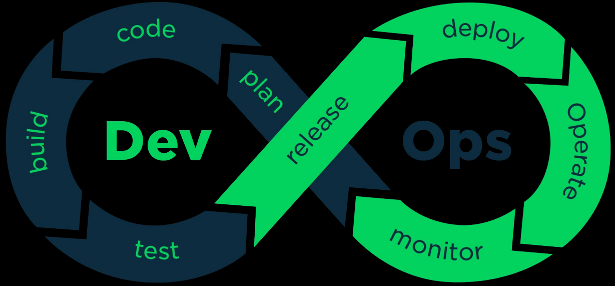
5. Software Testing Tools

6. Performance

7.Software Vulnerabilities

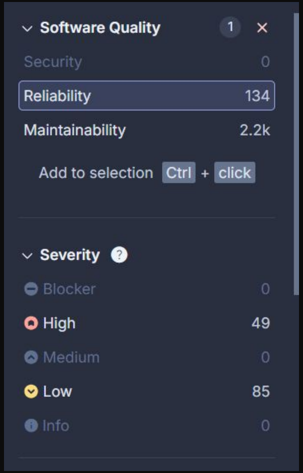
8. Conclusions

Objective



Explore the dependability of the open-source Java library Apache Commons Imaging by utilizing code review software analytics, conducting comprehensive testing, assessing software vulnerabilities, and implementing benchmarking strategies.

SonarCloud



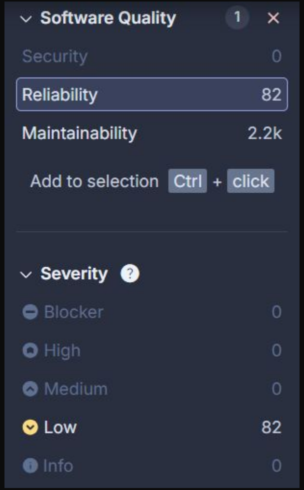
Software Quality 1 ×

Security	0
Reliability	134
Maintainability	2.2k

Add to selection Ctrl + click

Severity ?

Blocker	0
High	49
Medium	0
Low	85
Info	0



Software Quality 1 ×

Security	0
Reliability	82
Maintainability	2.2k

Add to selection Ctrl + click

Severity ?

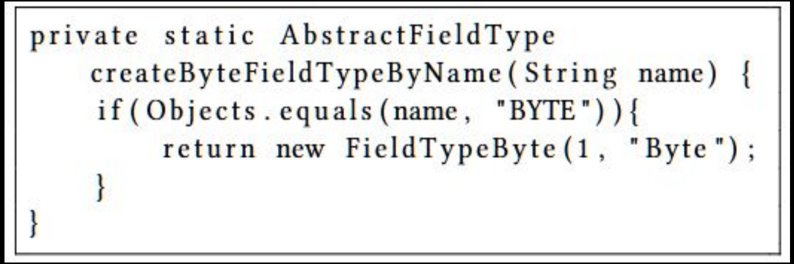
Blocker	0
High	0
Medium	0
Low	82
Info	0

Types of Errors

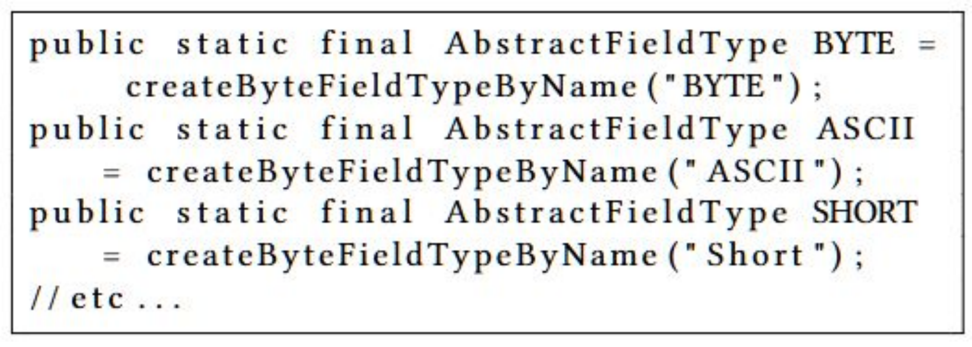
High = 49
These issues were addressed at their root through comprehensive refactoring, resulting in the elimination of all high-priority issues.

Low = 85
Issues at this level have a minimal impact on the reliability of the software.

Refactoring



```
private static AbstractFieldType
    createByteFieldTypeByName(String name) {
    if(Objects.equals(name, "BYTE")){
        return new FieldTypeByte(1, "Byte");
    }
}
```



```
public static final AbstractFieldType BYTE =
    createByteFieldTypeByName("BYTE");
public static final AbstractFieldType ASCII
    = createByteFieldTypeByName("ASCII");
public static final AbstractFieldType SHORT
    = createByteFieldTypeByName("Short");
//etc...
```

Refactoring Highlights

CENTRALIZED INITIALIZATION LOGIC.

DEFERRED STATIC FIELD INITIALIZATION.

REDUCED DEPENDENCIES OUTCOME.

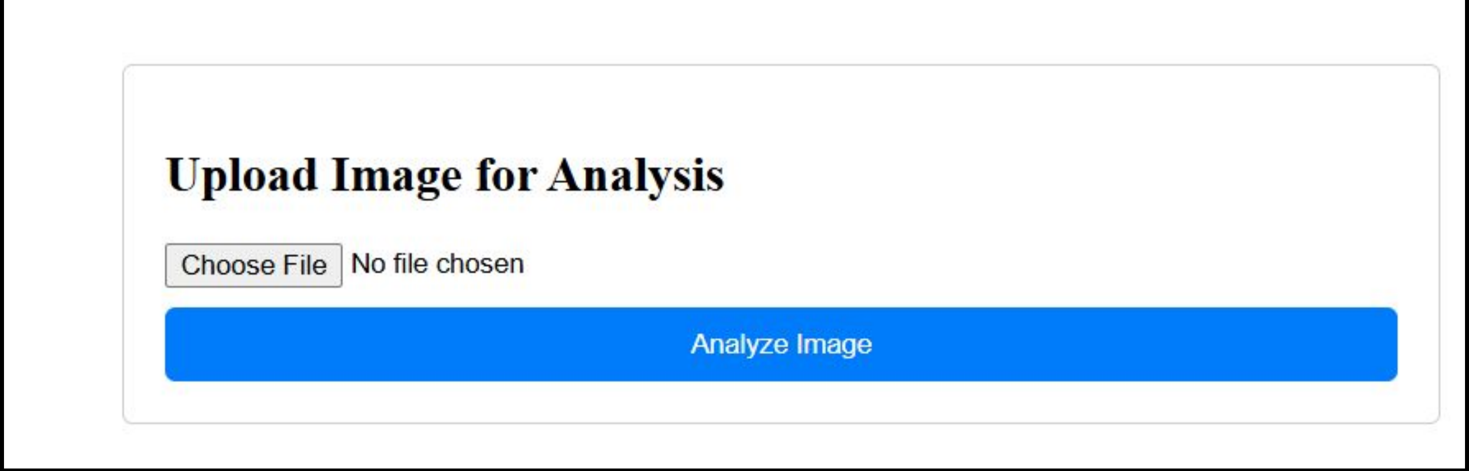
Refactoring Benefits

Reduced Coupling: Between static fields and concrete classes (direct dependencies).

Improved Modularity: Cleaner code, easier to maintain and extend.

Risk Mitigation: Prevention of critical issues such as initialization loops, deadlocks, and race conditions.

Demo

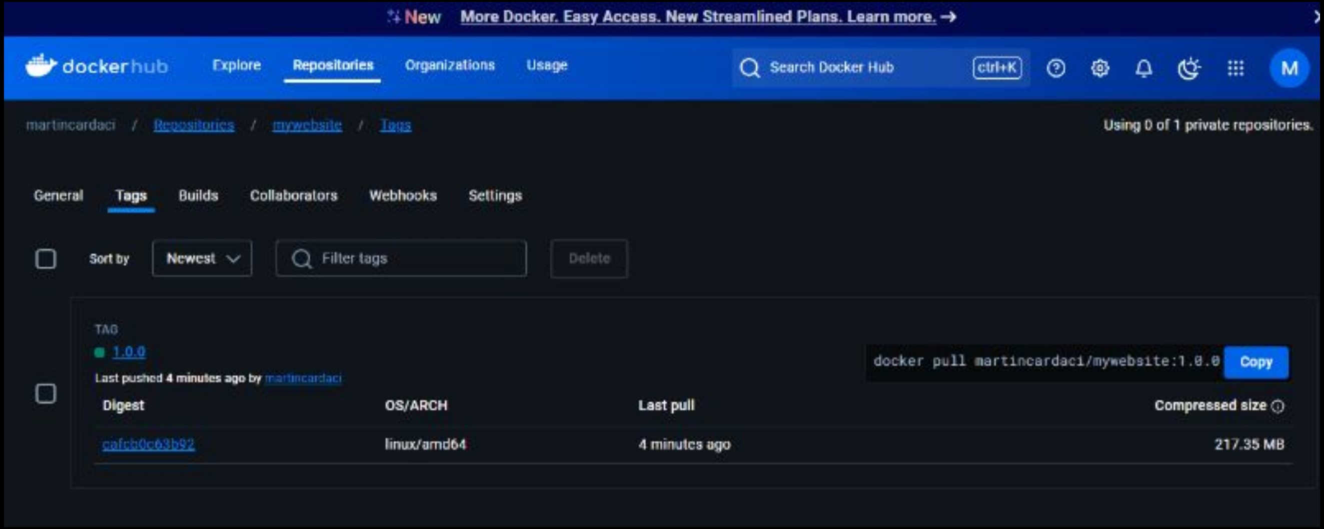


Upload Image for Analysis

Docker

Docker Desktop

docker hub



martincardaci / Repositories / mywebsite / Tags

Using 0 of 1 private repositories.

TAG

1.0.0

Last pushed 4 minutes ago by martincardaci

```
docker pull martincardaci/mywebsite:1.0.0
```

Digest	OS/ARCH	Last pull	Compressed size
cafcb0c63b92	linux/amd64	4 minutes ago	217.35 MB

Software Testing Tools

- JaCoCo
- CodeCov
- PITEST
- RANDOOP

Instructions Coverage: 77.6%

Branches Coverage: 64.3%

Missed Cyclomatic Complexities: 2445/6277

Missed Lines: 3760/16901

Missed Methods: 504/2547

Missed Classes: 15/432

JaCoCo

CodeCov

Instructions Coverage: 71.56%

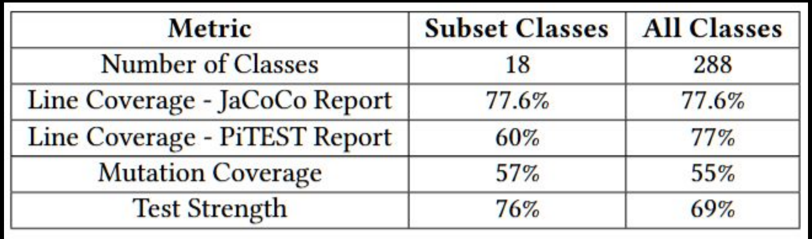
Tracked Lines: 16901

Tracked Lines Covered: 12100

Tracked Lines Partial: 1040

Tracked Lines Missed: 3761

PiTEST

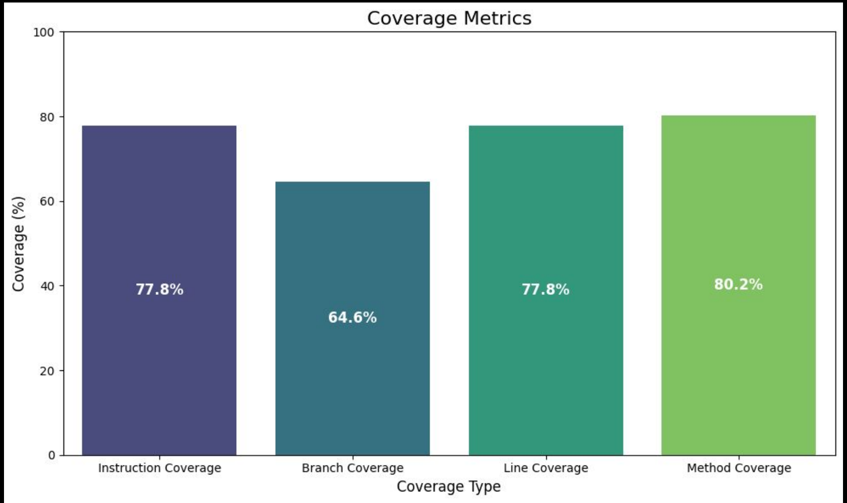


Metric	Subset Classes	All Classes
Number of Classes	18	288
Line Coverage - JaCoCo Report	77.6%	77.6%
Line Coverage - PiTEST Report	60%	77%
Mutation Coverage	57%	55%
Test Strength	76%	69%

Instructions Coverage: 77.6%

Branches Coverage: 64.3%

Randoop



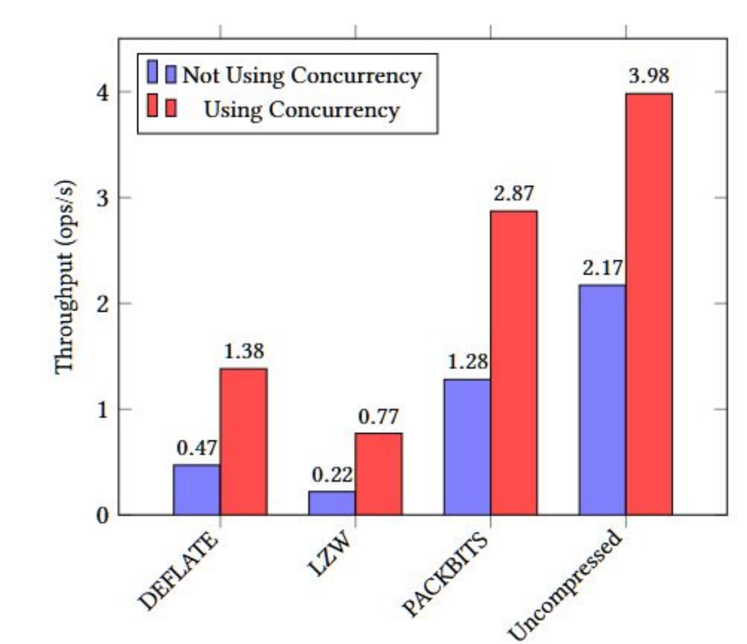
Instructions Coverage: 77.8%

Branches Coverage: 64.6%

Performance (JMH)

Optimized TIFF compression algorithms (*TiffRoundtrip test class*):

- 3x performance improvement through concurrency and memory handling.
- **Benefit**: Higher throughput and lower variability in operations.
- **Key lesson**: Benchmarking helps identify bottlenecks and validate improvements in resource-intensive code.

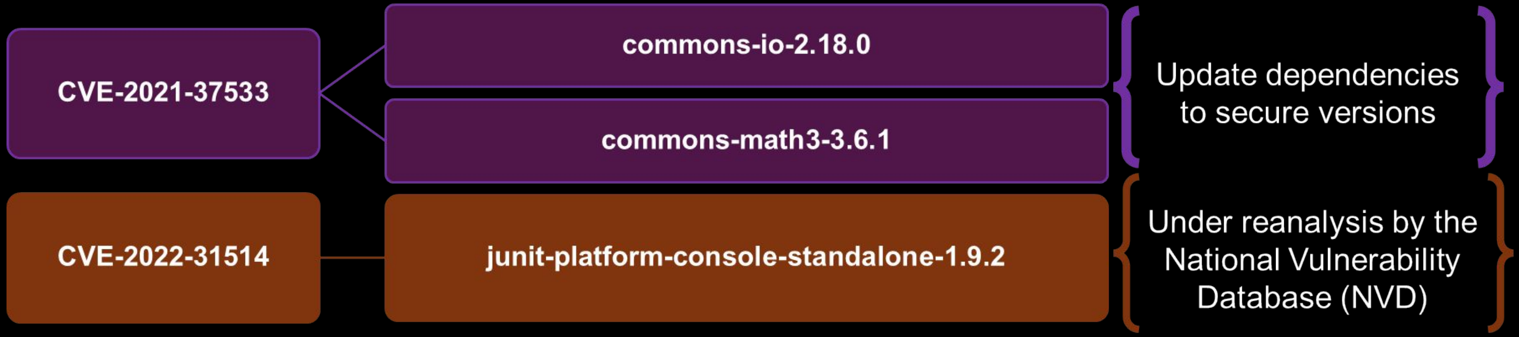


Software Vulnerabilities

OWASP

SpotBugs

OWASP



CVE-2021-37533

commons-io-2.18.0

commons-math3-3.6.1

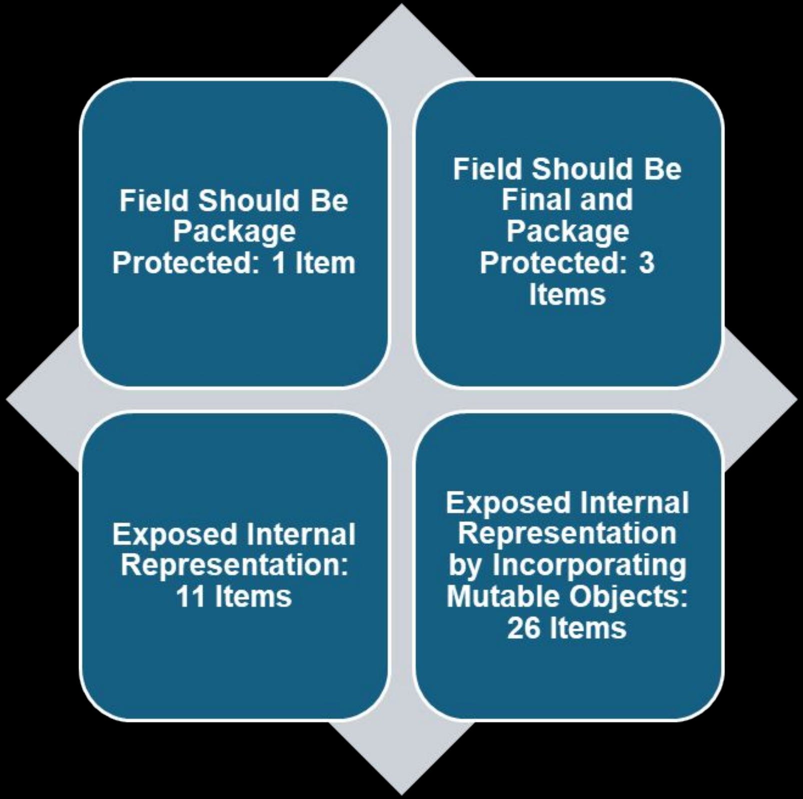
Update dependencies to secure versions

CVE-2022-31514

junit-platform-console-standalone-1.9.2

Under reanalysis by the National Vulnerability Database (NVD)

SpotBugs



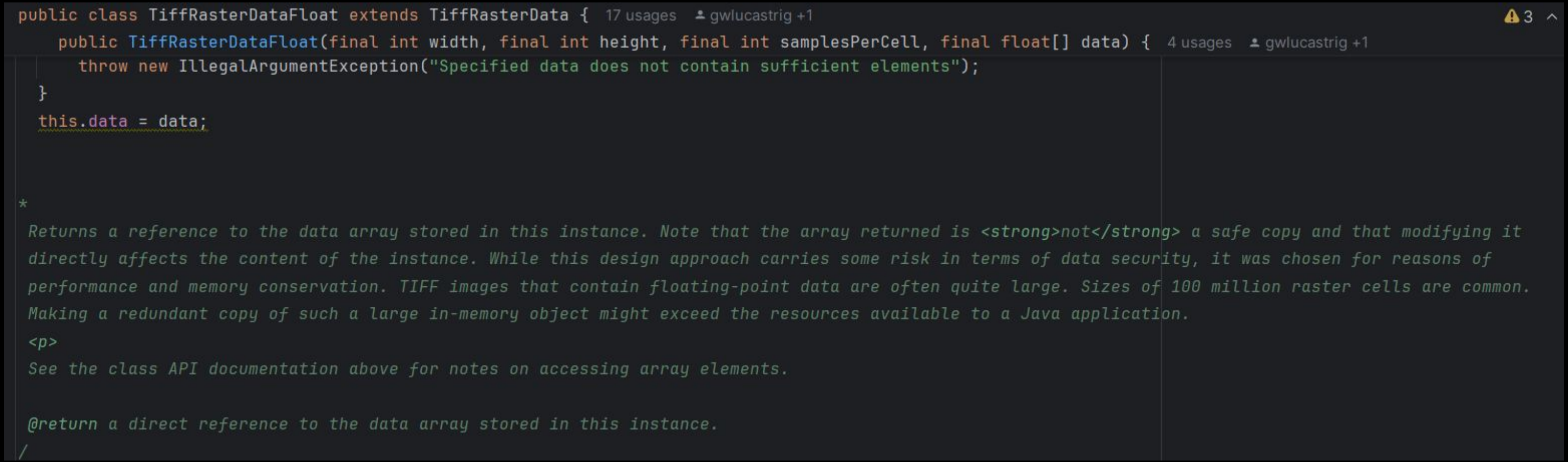
Field Should Be Package Protected: 1 Item

Field Should Be Final and Package Protected: 3 Items

Exposed Internal Representation: 11 Items

Exposed Internal Representation by Incorporating Mutable Objects: 26 Items

For “*Problem May expose internal representation by incorporating reference to mutable object*” (EI_EXPOSE_REP2)



```
public class TiffRasterDataFloat extends TiffRasterData {
    public TiffRasterDataFloat(final int width, final int height, final int samplesPerCell, final float[] data) {
        throw new IllegalArgumentException("Specified data does not contain sufficient elements");
    }
    this.data = data;



*
 Returns a reference to the data array stored in this instance. Note that the array returned is <strong>not</strong> a safe copy and that modifying it
 directly affects the content of the instance. While this design approach carries some risk in terms of data security, it was chosen for reasons of
 performance and memory conservation. TIFF images that contain floating-point data are often quite large. Sizes of 100 million raster cells are common.
 Making a redundant copy of such a large in-memory object might exceed the resources available to a Java application.
 <p>
 See the class API documentation above for notes on accessing array elements.

 @return a direct reference to the data array stored in this instance.
/
```

Conclusion

- **Code Quality and Optimization:** Refactoring efforts improved modularity, reduced static dependencies, and optimized initialization, making the library more maintainable and scalable.
- **Test Coverage:** Tools like JaCoCo, PIT Mutation Testing, and Randoop revealed gaps in specific areas but also demonstrated the effectiveness of automated testing in strengthening code robustness.
- **Performance Enhancements:** JMH benchmarks showed significant improvements in TIFF compression algorithms through concurrency and memory handling optimizations, resulting in better throughput and reduced variability.

- **Security:** Dependency and static code analysis identified vulnerabilities and encapsulation issues, emphasizing the need for continuous monitoring, proactive dependency management, and better data protection strategies.
- **Future Focus Areas:** Emphasizing efforts on tackling vulnerabilities and refining code coverage guarantees sustained dependability for future enhancements.

Thank

You

Original Repository

Project Repository

Docker Repository