




 Secrets


 ABAP


 Apex


 C


 C++


 CloudFormation


 COBOL


 C#


 CSS


 Flex


 Go


 HTML


 **Java**


 JavaScript


 Kotlin


 Objective C


 PHP


 PL/I


 PL/SQL


 Python


 RPG


 Ruby


 Scala


 Swift


 Terraform


 Text


 TypeScript

 T-SQL

 VB.NET

 VB6

 XML



Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

All rules632

Vulnerability53

Bug154

Security Hotspot36


Code Smell389

Quick Fix42


Tags ▾

Search by name... 🔍


Regular expressions should not be vulnerable to Denial of Service attacks

 Vulnerability


"HttpServletRequest.getRequestSession" should not be used

 Vulnerability


Hashes should include an unpredictable salt

 Vulnerability


Calls to methods should not trigger an IllegalArgumentException

 Bug


Unsupported methods should not be called on some collection implementations

 Bug


Cast operations should not trigger a ClassCastException

 Bug


Members ignored during record serialization should not be used

 Bug


Map "computeIfAbsent()" and "computeIfPresent()" should not be used to add "null" values.

 Bug


Regex lookahead assertions should not be contradictory

 Bug

Back references in regular expressions should only refer to capturing groups that are matched before the reference


 Bug


Regex boundaries should not be used in a way that can never be matched


 Bug

Silly bit operations should not be performed

Analyze your code

 Code Smell




 Blocker

 suspicious

Certain bit operations are just silly and should not be performed because their results are predictable.

Specifically, using `& -1` with any value will always result in the original value, as will `anyValue ^ 0` and `anyValue | 0`.





Available In:

 |  | 

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.
[Privacy Policy](#)

https://rules.sonarsource.com/java/RSPEC-2437

1/2

 Bug
<p>Regex patterns following a possessive quantifier should not always fail</p>  Bug
<p>Regular expressions should be syntactically valid</p>  Bug
<p>Assertions comparing incompatible types should not be made</p>  Bug
<p>JUnit5 inner test classes should be annotated with @Nested</p>