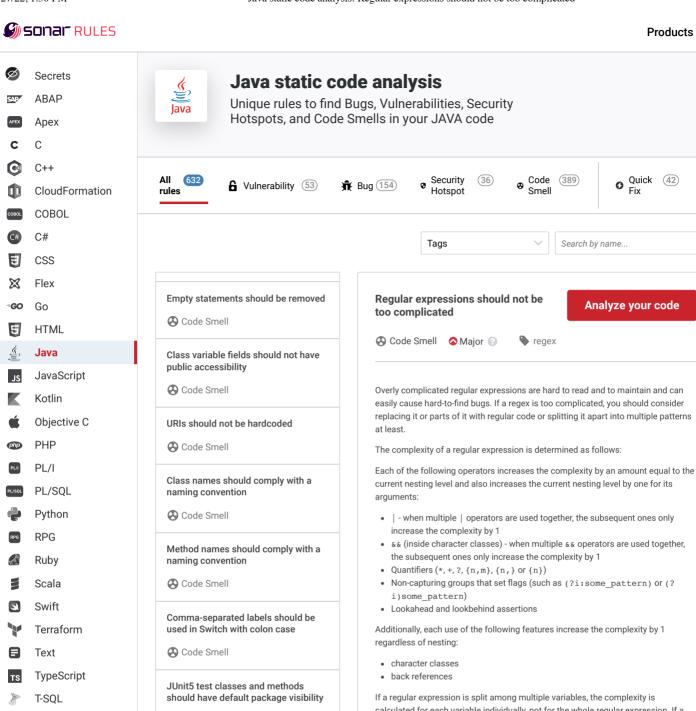


Products ✓

O Quick 42



Additionally, each use of the following features increase the complexity by 1

calculated for each variable individually, not for the whole regular expression. If a regular expression is split over multiple lines, each line is treated individually if it is accompanied by a comment (either a Java comment or a comment within the regular expression), otherwise the regular expression is analyzed as a whole.

## **Noncompliant Code Example**

```
if (dateString.matches("^(?:(?:31(\\/|-|\\.)(?:0?[13578]|1[0
   handleDate(dateString);
```

## **Compliant Solution**

```
if (dateString.matches("^\d{1,2}([-/.])\d{1,2}\)\d{1,4}$
   String dateParts[] = dateString.split("[-/.]");
   int day = Integer.parseInt(dateParts[0]);
   int month = Integer.parseInt(dateParts[1]);
   int year = Integer.parseInt(dateParts[2]);
    // Put logic to validate and process the date based on {\rm i}
```

Exceptions

"main" should not "throw" anything Code Smell Track lack of copyright and license headers Code Smell Octal values should not be used Code Smell Exit methods should not be called Code Smell HTTP response headers should not be vulnerable to injection attacks

Regular expressions are only analyzed if all parts of the regular expression are either string literals, effectively final local variables or static final fields, all of which can be combined using the '+' operator.

When a regular expression is split among multiple variables or commented lines, each part is only analyzed if it is syntactically valid by itself.

Available In:

sonarlint ⊕ | sonarcloud ↔ | sonarqube

© 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