sonar RULES Products

Java static code analysis

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

All rules 632 | Vulnerability 53 | Bug 154 | Security Hotspot 36 | Code Smell 389 | Quick Fix 42

Tags | Search by name...

the parameters of a public method
Code Smell

Assignments should not be redundant
Code Smell

Methods should not have identical implementations
Code Smell

"java.nio.Files#delete" should be preferred
Code Smell

Unused "private" classes should be removed
Code Smell

"Stream.peek" should be used with caution
Code Smell

"Map.get" and value test should be replaced with single method call
Code Smell

"@RequestMapping" methods should not be "private"
Code Smell

Raw types should not be used
Code Smell

"Arrays.stream" should be used for primitive arrays
Code Smell

Printf-style format strings should be used correctly
Code Smell

Assertion arguments should be passed in the correct order
Code Smell

Unicode Grapheme Clusters should be avoided inside regex character classes

Analyze your code

Bug Major regex

When placing Unicode Grapheme Clusters (characters which require to be encoded in multiple Code Points) inside a character class of a regular expression, this will likely lead to unintended behavior.

For instance, the grapheme cluster c̈ requires two code points: one for `'c'`, followed by one for the *umlaut* modifier `'\u{0308}'`. If placed within a character class, such as `[c̈]`, the regex will consider the character class being the enumeration `[c\u{0308}]` instead. It will, therefore, match every `'c'` and every *umlaut* that isn't expressed as a single codepoint, which is extremely unlikely to be the intended behavior.

This rule raises an issue every time Unicode Grapheme Clusters are used within a character class of a regular expression.

Noncompliant Code Example

```
"cc̈d̈d".replaceAll("[c̈d̈]", "X"); // Noncompliant, print "XXXX
```

Compliant Solution

```
"cc̈d̈d".replaceAll("c̈|d̈", "X"); // print "cXXd"
```

Available In:

sonarlint | sonarcloud | sonarqube

Ternary operators should not be nested

Code Smell

"writeObject" should not be the only "synchronized" code in a class

Code Smell

Reflection should not be used to increase accessibility of classes, methods, or fields

Code Smell

Static fields should not be updated in constructors

Code Smell