




 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



JavaScript static code analysis

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

- All rules 285
- Vulnerability 29
- Bug 62
- Security Hotspot 43
- Code Smell 151
- Quick Fix 41

Tags ▾

Search by name... 🔍

Regular expressions should not contain empty groups

Code Smell

Regular expressions should not contain multiple spaces

Code Smell

Chai assertions should have only one reason to succeed

Code Smell

Single-character alternations in regular expressions should be replaced with character classes

Code Smell

Reluctant quantifiers in regular expressions should be followed by an expression that can't match the empty string

Code Smell

Tests should check which exception is thrown

Code Smell

Character classes in regular expressions should not contain the same character twice

Code Smell

Names of regular expressions named groups should be used

Code Smell

Regular expressions should not be too complicated

Code Smell

Shorthand promises should be used

Code Smell

Template literals should not be nested

Code Smell

"for" loop increment clauses should modify the loops' counters

Analyze your code

Code Smell Critical ? confusing

It can be extremely confusing when a `for` loop's counter is incremented outside of its increment clause. In such cases, the increment should be moved to the loop's increment clause if at all possible.

Noncompliant Code Example

```
for (i = 0; i < 10; j++) { // Noncompliant
  // ...
  i++;
}
```

Compliant Solution

```
for (i = 0; i < 10; i++, j++) {
  // ...
}
```





Or

```
for (i = 0; i < 10; i++) {
  // ...
  j++;
}
```

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](#)

<div><div>"in" should not be used on arrays</div><div> Code Smell</div></div>
<div><div>Assignments should not be redundant</div><div> Code Smell</div></div>
<div><div>Functions should not have identical implementations</div><div> Code Smell</div></div>
<div><div>Sparse arrays should not be declared</div><div> Code Smell</div></div>
<div><div>Array-mutating methods should not be used misleadingly</div></div>