




 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... 


Using shell interpreter when executing OS commands is security-sensitive

 Security Hotspot


Setting loose POSIX file permissions is security-sensitive

 Security Hotspot


Formatting SQL queries is security-sensitive

 Security Hotspot


Comma operator should not be used

 Code Smell


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

Loop counters should not be assigned to from within the loop body

Analyze your code

 Code Smell

 Critical 




Loop counters should not be modified in the body of the loop. However other loop control variables representing logical values may be modified in the loop, for example a flag to indicate that something has been completed, which is then tested in the for statement.

Noncompliant Code Example

```
var names = [ "Jack", "Jim", "", "John" ];
for (var i = 0; i < names.length; i++) {
  if (!names[i]) {
    i = names.length;
  } else {
    console.log(names[i]);
  }
}
```

Compliant Solution






```
var names = [ "Jack", "Jim", "", "John" ];
for (var name of names) {
  if (!name) {
    break;
  } else {
    console.log(name);
  }
}
```

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/javascript/RSPEC-2310

1/2

| |
|---|
|  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 |