




 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... 


in boolean contexts

 Bug


Constructing arguments of system commands from user input is security-sensitive

 Security Hotspot


Allowing requests with excessive content length is security-sensitive

 Security Hotspot


Statically serving hidden files is security-sensitive

 Security Hotspot


Using intrusive permissions is security-sensitive

 Security Hotspot


Disabling auto-escaping in template engines is security-sensitive

 Security Hotspot


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


A conditionally executed single line should be denoted by indentation

Analyze your code

 Code Smell

 Critical



 confusing suspicious

In the absence of enclosing curly braces, the line immediately after a conditional is the one that is conditionally executed. By both convention and good practice, such lines are indented. In the absence of both curly braces and indentation the intent of the original programmer is entirely unclear and perhaps not actually what is executed. Additionally, such code is highly likely to be confusing to maintainers.

Noncompliant Code Example

```
if (condition) // Noncompliant
doTheThing();

doTheOtherThing();
somethingElseEntirely();

foo();
```




Compliant Solution

```
if (condition)
doTheThing();

doTheOtherThing();
somethingElseEntirely();

foo();
```





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-3973

1/2

 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