




 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... 🔍

Literals should not be thrown

Code Smell

Functions should not be called both with and without "new"

Code Smell

Array indexes should be numeric

Code Smell

Assertion arguments should be passed in the correct order

Code Smell

Ternary operators should not be nested

Code Smell

"delete" should not be used on arrays

Code Smell

Variables and functions should not be redeclared

Code Smell

"indexOf" checks should not be for positive numbers

Code Smell

"arguments.caller" and "arguments.callee" should not be used

Code Smell

Multiline blocks should be enclosed in curly braces

Code Smell

Boolean expressions should not be gratuitous

Code Smell

Variables should be used in the blocks where they are declared

Code Smell

Assertions should not be given twice the same argument

Analyze your code

Bug Major tests

Many assertions compare two objects or properties of these objects. Passing twice the same argument is likely to be a bug due to developer's carelessness.

This rule raises an issue when a Chai assertion is given twice the same argument.

Noncompliant Code Example

```
const assert = require('chai').assert;

describe("test the same object", function() {
  it("uses chai 'assert'", function() {
    const expected = '1'
    const actual = (1).toString()
    assert.equal(actual, actual); // Noncompliant
  });
});
```

Compliant Solution

```
const assert = require('chai').assert;

describe("test the same object", function() {
  it("uses chai 'assert'", function() {
    const expected = '1'
    const actual = (1).toString()
    assert.equal(actual, expected);
  });
});
```





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](#)

https://rules.sonarsource.com/javascript/RSPEC-5863

1/2

 Code Smell
<p>Parameters should be passed in the correct order</p>  Code Smell
<p>Two branches in a conditional structure should not have exactly the same implementation</p>  Code Smell
<p>Unused assignments should be removed</p>  Code Smell
<p>Function parameters with default values should be last</p>