




 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 rules285

Vulnerability29

Bug62

Security Hotspot43

Code Smell151

Quick Fix41

Tags ▾

Search by name... 🔍

Trailing commas should be used

Code Smell

"import" should be used to include external code

Code Smell

Braces and parentheses should be used consistently with arrow functions

Code Smell

Destructuring syntax should be used for assignments

Code Smell

Template strings should be used instead of concatenation

Code Smell

Shorthand object properties should be grouped at the beginning or end of an object declaration

Code Smell

Object literal shorthand syntax should be used

Code Smell

Strings and non-strings should not be added

Code Smell

Object literal syntax should be used

Code Smell

"undefined" should not be assigned

Code Smell

Trailing commas should not be used

Code Smell

Array constructors should not be used

Code Smell

Function parameters with default values should be last

Analyze your code

Code Smell

Major

es2015

The ability to define default values for function parameters can make a function easier to use. Default parameter values allow callers to specify as many or as few arguments as they want while getting the same functionality and minimizing boilerplate, wrapper code.

But all function parameters with default values should be declared after the function parameters without default values. Otherwise, it makes it impossible for callers to take advantage of defaults; they must re-specify the defaulted values or pass undefined in order to "get to" the non-default parameters.

Noncompliant Code Example

```
function multiply(a = 1, b) { // Noncompliant
  return a*b;
}

var x = multiply(42); // returns NaN as b is undefined
```

Compliant Solution

```
function multiply(b, a = 1) {
  return a*b;
}

var x = multiply(42); // returns 42 as 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-1788

1/2

<div>Quotes for string literals should be used consistently</div> <div> Code Smell</div>
<div>Statements should end with semicolons</div> <div> Code Smell</div>
<div>Comments should not be located at the end of lines of code</div> <div> Code Smell</div>
<div>Loops should not contain more than a single "break" or "continue" statement</div> <div> Code Smell</div>