




 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... 🔍

Structure elements that have already been implemented

Code Smell

Unused assignments should be removed

Code Smell

Function parameters with default values should be last

Code Smell

Functions should not be defined inside loops

Code Smell

"switch" statements should not have too many "case" clauses

Code Smell

Only "while", "do", "for" and "switch" statements should be labelled

Code Smell

Sections of code should not be commented out

Code Smell

Unused function parameters should be removed

Code Smell

Track uses of "FIXME" tags

Code Smell

Assignments should not be made from within sub-expressions

Code Smell

Labels should not be used

Code Smell

Variables should not be shadowed

Code Smell

Redundant uses of "new" operator

Code Smell

### The output of functions that don't return anything should not be used

Analyze your code

Bug

Major

If a function does not return anything, it makes no sense to use its output. Specifically, passing it to another function, or assigning its "result" to a variable is probably a bug because such functions return undefined, which is probably not what was intended.

#### Noncompliant Code Example

```
function foo() {
  console.log("Hello, World!");
}

a = foo();
```

#### Compliant Solution

```
function foo() {
  console.log("Hello, World!");
}

foo();
```

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-3699

1/2

JavaScript static code analysis: The output of functions that don't return anything should not be used	
redundant pairs of parentheses should be removed	 Code Smell
Nested blocks of code should not be left empty	 Code Smell
Functions should not have too many parameters	 Code Smell
OS commands should not be vulnerable to argument injection attacks	 Vulnerability