sonar RULES

Products

- 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

JS

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...

Control flow statements "if", "for", "while", "switch" and "try" should not be nested too deeply

Code Smell

"switch" statements should have "default" clauses

Code Smell

"if ... else if" constructs should end with "else" clauses

Code Smell

Control structures should use curly braces

Code Smell

String literals should not be duplicated

Code Smell

Expressions should not be too complex

Code Smell

Local storage should not be used

Vulnerability

Template literal placeholder syntax should not be used in regular strings

Bug

Built-in objects should not be overridden

Bug

"for...in" loops should filter properties before acting on them

Bug

Results of operations on strings should not be ignored

Bug

Increment (++) and decrement (--) operators should not be used in a method call or mixed with other operators in an expression

Assignments should not be redundant

Analyze your code

Code Smell | Major | redundant

The transitive property says that if `a == b` and `b == c`, then `a == c`. In such cases, there's no point in assigning `a` to `c` or vice versa because they're already equivalent.

This rule raises an issue when an assignment is useless because the assigned-to variable already holds the value on all execution paths.

Noncompliant Code Example

```
a = b;
c = a;
b = c; // Noncompliant: c and b are already the same
```

Compliant Solution

```
a = b;
c = a;
```

Available In:

sonarlint sonarcloud sonarqube

operators in an expression

Code Smell

"for in" should not be used with iterables

Code Smell

Functions should use "return" consistently

Code Smell

Variables and functions should not be declared in the global scope

Code Smell

Arithmetic operators should only have numbers as operands