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...

Code Smell

Track uses of "TODO" tags

Code Smell

Web SQL databases should not be used

Vulnerability

Variables should be defined before being used

Bug

Variables declared with "var" should be declared before they are used

Code Smell

Track lack of copyright and license headers

Code Smell

Reading the Standard Input is security-sensitive

Security Hotspot

Using command line arguments is security-sensitive

Security Hotspot

Using Sockets is security-sensitive

Security Hotspot

Executing XPath expressions is security-sensitive

Security Hotspot

Encrypting data is security-sensitive

Security Hotspot

Using regular expressions is security-sensitive

Security Hotspot

Class methods should be used instead of "prototype" assignments

Regular expressions should not contain multiple spaces

Analyze your code

Code Smell Major regex

Multiple spaces in a regular expression can make it hard to tell how many spaces should be matched. It's more readable to use only one space and then indicate with a quantifier how many spaces are expected.

Noncompliant Code Example

```
const pattern = /Hello,   world!/;
```

Compliant Solution

```
const pattern = /Hello, {3}world!/;
```

Available In:

sonarlint sonarcloud sonarqube

Code Smell

Function constructors should not be used

Code Smell

Variables should be declared with "let" or "const"

Code Smell

Unchanged variables should be marked "const"

Code Smell

Wildcard imports should not be used