




 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

Security Hotspot
Using hardcoded IP addresses is security-sensitive
Security Hotspot
Regular expression quantifiers and character classes should be used concisely
Code Smell
Regular expression literals should be used when possible
Code Smell
"await" should not be used redundantly
Code Smell
"for of" should be used with Iterables
Code Smell
Imports from the same modules should be merged
Code Smell
Jump statements should not be redundant
Code Smell
Default export names and file names should match
Code Smell
The global "this" object should not be used
Code Smell
"catch" clauses should do more than rethrow
Code Smell
Boolean checks should not be inverted
Code Smell

Property names should not be duplicated within a class or object literal

Analyze your code

Bug

Major

Quick Fix

pitfall

JavaScript allows duplicate property names in classes and object literals, but only the last instance of a duplicated name determines the actual value that will be used for it. Therefore, changing values of other occurrences of a duplicated name will have no effect and may cause misunderstandings and bugs.

Defining a class with a duplicated constructor will generate an error.

Before ECMAScript 2015, using duplicate names will generate an error in JavaScript strict mode code.

Noncompliant Code Example

```
var data = {
  "key": "value",
  "1": "value",
  "key": "value", // Noncompliant - duplicate of "key"
  'key': "value", // Noncompliant - duplicate of "key"
  key: "value", // Noncompliant - duplicate of "key"
  \u006bey: "value", // Noncompliant - duplicate of "key"
  "\u006bey": "value", // Noncompliant - duplicate of "key"
  "\x6bey": "value", // Noncompliant - duplicate of "key"
  1: "value" // Noncompliant - duplicate of "1"
}
```

Compliant Solution





```
var data = {
  "key": "value",
  "1": "value",
  "key2": "value",
  'key3': "value",
  key4: "value",
  \u006bey5: "value",
  "\u006bey6": "value",
  "\x6bey7": "value",
  1b: "value"
}
```

Available In:

sonarlint

sonarcloud

sonarqube

<div>Deprecated APIs should not be used</div> <div> Code Smell</div>
<div>Wrapper objects should not be used for primitive types</div> <div> Code Smell</div>
<div>Multiline string literals should not be used</div> <div> Code Smell</div>
<div>Local variables should not be declared and then immediately returned or thrown</div> <div> Code Smell</div>

SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.
[Privacy Policy](#)