




 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... 🔍


 Security Hotspot


Creating cookies without the "HttpOnly" flag is security-sensitive

 Security Hotspot


 Security Hotspot


Creating cookies without the "secure" flag is security-sensitive

 Security Hotspot


 Security Hotspot


Using hardcoded IP addresses is security-sensitive

 Security Hotspot


 Code Smell


Regular expression quantifiers and character classes should be used concisely

 Code Smell


 Code Smell


Regular expression literals should be used when possible

 Code Smell

 Code Smell

"await" should not be used redundantly

 Code Smell

 Code Smell

"for of" should be used with Iterables

 Code Smell

 Code Smell

Imports from the same modules should be merged

 Code Smell

 Code Smell

Jump statements should not be redundant

 Code Smell

 Code Smell

Default export names and file names should match

 Code Smell



 Code Smell

The global "this" object should not be used

 Code Smell

Function argument names should be unique

Analyze your code

 Bug  Major ?

Function arguments should all have different names to prevent any ambiguity. Indeed, if arguments have the same name, the last duplicated argument hides all the previous arguments with the same name (those previous arguments remain available through arguments[i], so they're not completely inaccessible).

This hiding makes no sense, reduces understandability and maintainability, and obviously can be error prone. Furthermore, in strict mode, declaring arguments with the same name produces an error.




Noncompliant Code Example

```
function compute(a, a, c) { // Noncompliant
}
```

Compliant Solution

```
function compute(a, b, c) { // Compliant
}
```

Available In:





 |  | 

© 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-1536

1/2

<div>"catch" clauses should do more than rethrow</div> <div> Code Smell</div>
<div>Boolean checks should not be inverted</div> <div> Code Smell</div>
<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>