




 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... 🔍

| | |
|--|------------------|
| Functions should not have too many parameters | Code Smell |
| OS commands should not be vulnerable to argument injection attacks | Vulnerability |
| Repeated patterns in regular expressions should not match the empty string | Bug |
| Empty collections should not be accessed or iterated | Bug |
| "delete" should be used only with object properties | Bug |
| "with" statements should not be used | Bug |
| Function parameters, caught exceptions and foreach variables' initial values should not be ignored | Bug |
| Forwarding client IP address is security-sensitive | Security Hotspot |
| Allowing confidential information to be logged is security-sensitive | Security Hotspot |
| Allowing browsers to perform DNS prefetching is security-sensitive | Security Hotspot |
| Disabling Certificate Transparency monitoring is security-sensitive | Security Hotspot |

Setters should not return values

Analyze your code

 Bug

 Major



Functions declared with the `set` keyword will automatically return the values they were passed. Thus any value explicitly returned from a setter will be ignored, and explicitly returning a value is an error.

Noncompliant Code Example

```
var person = {
  // ...
  set name(name) {
    this.name = name;
    return 42; // Noncompliant
  }
}
```





Compliant Solution

```
var person = {
  // ...
  set name(name) {
    this.name = name;
  }
}
```

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](#)

| |
|---|
| <div>Disabling Strict-Transport-Security policy is security-sensitive</div> <div> Security Hotspot</div> |
| <div>Disabling strict HTTP no-referrer policy is security-sensitive</div> <div> Security Hotspot</div> |
| <div>Allowing browsers to sniff MIME types is security-sensitive</div> <div> Security Hotspot</div> |
| <div>Disabling content security policy frame-ancestors directive is security-sensitive</div> <div></div> |