




 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



TypeScript static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your TYPESCRIPT code

- All rules 279
- Vulnerability 27
- Bug 51
- Security Hotspot 43
- Code Smell 158
- Quick Fix 50

Chai assertions should have only one reason to succeed
Code Smell
Single-character alternations in regular expressions should be replaced with character classes
Code Smell
Reluctant quantifiers in regular expressions should be followed by an expression that can't match the empty string
Code Smell
Tests should check which exception is thrown
Code Smell
Character classes in regular expressions should not contain the same character twice
Code Smell
Names of regular expressions named groups should be used
Code Smell
Regular expressions should not be too complicated
Code Smell
Optional property declarations should not use both '?' and 'undefined' syntax
Code Smell
Shorthand promises should be used
Code Smell
Template literals should not be nested
Code Smell
"undefined" should not be passed as the value of optional parameters
Code Smell

Tags ▾

Search by name... 🔍

"for" loop increment clauses should modify the loops' counters

Analyze your code

- Code Smell
- Critical
- confusing

It can be extremely confusing when a `for` loop's counter is incremented outside of its increment clause. In such cases, the increment should be moved to the loop's increment clause if at all possible.

Noncompliant Code Example

```
for (i = 0; i < 10; j++) { // Noncompliant
  // ...
  i++;
}
```

Compliant Solution





```
for (i = 0; i < 10; i++, j++) {
  // ...
}
```

Or

```
for (i = 0; i < 10; i++) {
  // ...
  j++;
}
```

Available In:

sonarlint | sonarcloud | sonarqube

"in" should not be used on arrays  Code Smell
Assignments should not be redundant  Code Smell
Functions should not have identical implementations  Code Smell
Sparse arrays should not be declared  Code Smell
Array-mutating methods should not be used with readonly arrays