




 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 rules279

Vulnerability27

Bug51

Security Hotspot43

Code Smell158

Quick Fix50

Tags ▾

Search by name... 🔍

commented out

Code Smell

Unused function parameters should be removed

Code Smell

Track uses of "FIXME" tags

Code Smell

Assignments should not be made from within sub-expressions

Code Smell

Labels should not be used

Code Smell

Variables should not be shadowed

Code Smell

Redundant pairs of parentheses should be removed

Code Smell

Nested blocks of code should not be left empty

Code Smell

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

The output of functions that don't return anything should not be used

Analyze your code

Bug

Major

If a function does not return anything, it makes no sense to use its output. Specifically, passing it to another function, or assigning its "result" to a variable is probably a bug because such functions return undefined, which is probably not what was intended.

Noncompliant Code Example

```
function foo() {
  console.log("Hello, World!");
}

a = foo();
```

Compliant Solution

```
function foo() {
  console.log("Hello, World!");
}

foo();
```





Available In:

sonarlint | sonarcloud | sonarqube

© 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/typescript/RSPEC-3699

1/2

<div>"delete" should be used only with object properties</div> <div> Bug</div>
<div>Function parameters, caught exceptions and foreach variables' initial values should not be ignored</div> <div> Bug</div>
<div>Forwarding client IP address is security-sensitive</div> <div> Security Hotspot</div>
<div>Allowing confidential information to be logged is security-sensitive</div> <div></div>