



 **sonar** RULES


Products ▾


 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... 🔍

Unnecessary imports should be removed

Code Smell

Boolean literals should not be used in comparisons

Code Smell

Extra semicolons should be removed

Code Smell

Class names should comply with a naming convention

Code Smell

Track uses of "TODO" tags

Code Smell

Web SQL databases should not be used

Vulnerability

Variables declared with "var" should be declared before they are used

Code Smell

Track lack of copyright and license headers

Code Smell

Reading the Standard Input is security-sensitive

Security Hotspot

Using command line arguments is security-sensitive

Security Hotspot

Using Sockets is security-sensitive

Security Hotspot

Executing XPath expressions is security-sensitive

Security Hotspot

Single-character alternations in regular expressions should be replaced with character classes

Analyze your code

Code Smell

Major

regex

When an alternation contains multiple alternatives that consist of a single character, it can be rewritten as a character class. This should be preferred because it is more efficient and can even help prevent stack overflows when used inside a repetition (see rule {rule:javascript:S5998}).

Noncompliant Code Example

```
/a|b|c/; // Noncompliant
```

Compliant Solution

```
/[abc]/;  
// or  
/[a-c]/;
```

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-6035

1/2

<div>Encrypting data is security-sensitive</div> <div> Security Hotspot</div>
<div>Using regular expressions is security-sensitive</div> <div> Security Hotspot</div>
<div>Class methods should be used instead of "prototype" assignments</div> <div> Code Smell</div>
<div>Variables should be declared with "let" or "const"</div> <div> Code Smell</div>