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

Tags | Search by name...

- Literals should not be thrown — Code Smell
- Array indexes should be numeric — Code Smell
- Assertion arguments should be passed in the correct order — Code Smell
- Ternary operators should not be nested — Code Smell
- "delete" should not be used on arrays — Code Smell
- Variables and functions should not be redeclared — Code Smell
- "indexOf" checks should not be for positive numbers — Code Smell
- "arguments.caller" and "arguments.callee" should not be used — Code Smell
- Multiline blocks should be enclosed in curly braces — Code Smell
- Boolean expressions should not be gratuitous — Code Smell
- Variables should be used in the blocks where they are declared — Code Smell
- Parameters should be passed in the correct order — Code Smell

Unicode Grapheme Clusters should be avoided inside regex character classes

Analyze your code

Bug | Major | regex

When placing Unicode Grapheme Clusters (characters which require to be encoded in multiple Code Points) inside a character class of a regular expression, this will likely lead to unintended behavior.

For instance, the grapheme cluster c̈ requires two code points: one for `'c'`, followed by one for the *umlaut* modifier `'\u{0308}'`. If placed within a character class, such as `[c̈]`, the regex will consider the character class being the enumeration `[c\u{0308}]` instead. It will, therefore, match every `'c'` and every *umlaut* that isn't expressed as a single codepoint, which is extremely unlikely to be the intended behavior.

This rule raises an issue every time Unicode Grapheme Clusters are used within a character class of a regular expression.

Noncompliant Code Example

```
"cc̈d̈d".replace(/[c̈d̈]/g, "X"); // result is "XXXXXX" and not
```

Compliant Solution

```
"cc̈d̈d".replace(/c̈|d̈/g, "X"); // result is "cXXd"
```

Available In:

sonarlint | sonarcloud | sonarqube

Two branches in a conditional structure should not have exactly the same implementation

Code Smell

Unused assignments should be removed

Code Smell

Function parameters with default values should be last

Code Smell

Functions should not be defined inside loops