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...

Security Hotspot

Having a permissive Cross-Origin Resource Sharing policy is security-sensitive
Security Hotspot

Delivering code in production with debug features activated is security-sensitive
Security Hotspot

Creating cookies without the "HttpOnly" flag is security-sensitive
Security Hotspot

Creating cookies without the "secure" flag is security-sensitive
Security Hotspot

Using hardcoded IP addresses is security-sensitive
Security Hotspot

Regular expression quantifiers and character classes should be used concisely
Code Smell

Regular expression literals should be used when possible
Code Smell

"await" should not be used redundantly
Code Smell

Redundant casts and non-null assertions should be avoided
Code Smell

Type aliases should be used
Code Smell

Type guards should be used
Code Smell

Loops with at most one iteration should be refactored

Analyze your code

Bug Major

A loop with at most one iteration is equivalent to the use of an `if` statement to conditionally execute one piece of code. No developer expects to find such a use of a loop statement. If the initial intention of the author was really to conditionally execute one piece of code, an `if` statement should be used instead.

At worst that was not the initial intention of the author and so the body of the loop should be fixed to use the nested `return`, `break` or `throw` statements in a more appropriate way.

Noncompliant Code Example

```
for (let i = 0; i < 10; i++) { // noncompliant, loop only ex
  console.log("i is " + i);
  break;
}
...
for (let i = 0; i < 10; i++) { // noncompliant, loop only ex
  if (i == x) {
    break;
  } else {
    console.log("i is " + i);
    return;
  }
}
```

Compliant Solution

```
for (let i = 0; i < 10; i++) {
  console.log("i is " + i);
}
...
for (let i = 0; i < 10; i++) {
  if (i == x) {
    break;
  } else {
    console.log("i is " + i);
  }
}
```

Available In:
sonarlint sonarcloud sonarqube

"module" should not be used

Code Smell

"for of" should be used with Iterables

Code Smell

Imports from the same modules should be merged

Code Smell

Jump statements should not be redundant

Code Smell