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...

Vulnerability

Non-empty statements should change control flow or have at least one side-effect
Bug

Regular expressions with the global flag should be used with caution
Bug

Replacement strings should reference existing regular expression groups
Bug

Regular expressions should not contain control characters
Bug

Alternation in regular expressions should not contain empty alternatives
Bug

Mocha timeout should be disabled by setting it to "0".
Bug

Unicode Grapheme Clusters should be avoided inside regex character classes
Bug

Assertions should not be given twice the same argument
Bug

Alternatives in regular expressions should be grouped when used with anchors
Bug

Promise rejections should not be caught by 'try' block
Bug

Collection elements should not be replaced unconditionally

Types without members, 'any' and 'never' should not be used in type intersections

Analyze your code

Bug Critical pitfall

An intersection type combines multiple types into one. This allows you to add together existing types to get a single type that has all the features you need. However an intersection with a type without members doesn't change the resulting type. In the opposite the usage of `any` or `never` as part of an intersection will always results in `any` or `never` respectively. This is almost certainly an error.

Noncompliant Code Example

```
function foo(p: MyType & null) { // Noncompliant
 // ...
}

function bar(p: MyType & any) { // Noncompliant
 // ...
}
```

Compliant Solution

```
function foo(p: MyType | null) {
 // ...
}
// or
function foo(p: MyType & AnotherType) {
 // ...
}

function bar(p: any) {
 // ...
}
```

Available In:
sonarlint sonarcloud sonarqube

Bug

Constructors should not be declared inside interfaces

Bug

Errors should not be created without being thrown

Bug

Collection sizes and array length comparisons should make sense

Bug

All branches in a conditional structure should not have exactly the same