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...

Encrypting data is security-sensitive
Security Hotspot

Using regular expressions is security-sensitive
Security Hotspot

Class methods should be used instead of "prototype" assignments
Code Smell

Variables should be declared with "let" or "const"
Code Smell

Unchanged variables should be marked "const"
Code Smell

Wildcard imports should not be used
Code Smell

"switch" statements should not be nested
Code Smell

Cyclomatic Complexity of functions should not be too high
Code Smell

"strict" mode should be used with caution
Code Smell

Control flow statements "if", "for", "while", "switch" and "try" should not be nested too deeply
Code Smell

"switch" statements should have "default" clauses
Code Smell

"if ... else if" constructs should end with "else" clauses

Optional property declarations should not use both '?' and 'undefined' syntax

Analyze your code

Code Smell   Major   Quick Fix   redundant

In TypeScript there are several ways to declare a property with `undefined` value: adding `| undefined` in the property type or using optional property syntax (`?` after its name). Use `| undefined` syntax when you want to be explicit that an object has that property, in that case TypeScript compiler will not allow omitting it:

```
interface Person {
  name: string;
  address: string | undefined;
}

let John = { name: "John" }; // will not compile
let John = { name: "John", address: undefined }; // will com
```

Use optional property syntax for properties holding some additional information.

```
interface Person {
  name: string;
  pet?: string;
}

let John = { name: "John" }; // will compile
let John = { name: "John", pet: undefined }; // will compile
let John = { name: "John", pet: "Benji" }; // will compile
```

Using `| undefined` for optional property is redundant, it can be omitted without change to the actual type. Still if you want to force the property in the object consider using only `| undefined` without `?`.

Noncompliant Code Example

```
interface Person {
  name: string;
  address? : string | undefined;   // Noncompliant, "?" shou
  pet?: Animal | undefined; // Noncompliant, "undefined" sho
}
```

Compliant Solution

```
interface Person {
  name: string;
  address: string | undefined;
  pet?: Animal;
}
```

Available In:

Code Smell

Control structures should use curly braces

Code Smell

String literals should not be duplicated

Code Smell

Expressions should not be too complex

Code Smell

Template literal placeholder syntax should not be used in regular strings

sonarlint sonarcloud sonarqube