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...

Primitive return types should be used
Code Smell

Default type parameters should be omitted
Code Smell

Type assertions should use "as"
Code Smell

Method overloads should be grouped together
Code Smell

Interfaces should not be empty
Code Smell

Trailing commas should be used
Code Smell

"import" should be used to include external code
Code Smell

Braces and parentheses should be used consistently with arrow functions
Code Smell

Destructuring syntax should be used for assignments
Code Smell

Template strings should be used instead of concatenation
Code Smell

Shorthand object properties should be grouped at the beginning or end of an object declaration
Code Smell

Object literal shorthand syntax should be used
Code Smell

Unused function parameters should be removed

Analyze your code

Code Smell | Major | Quick Fix | unused

Unused parameters are misleading. Whatever the values passed to such parameters, the behavior will be the same.

Noncompliant Code Example

```
function doSomething(a, b) { // "a" is unused
  return compute(b);
}
```

Compliant Solution

```
function doSomething(b) {
  return compute(b);
}
```

or

```
function doSomething(_a, b) {
  return compute(b);
}
```

Exceptions

When `arguments` is used in the function body, no parameter is reported as unused.

```
function doSomething(a, b, c) {
  compute(arguments);
}
```

Also, the rule ignores all parameters whose name starts with an underscore (_). This is a common practice to acknowledge the fact that some parameter is unused (e.g. in TypeScript compiler).

```
function doSomething(_a, b) {
  return compute(b);
}
```

Available In:
sonarlint sonarcloud sonarqube

Strings and non-strings should not be added

Code Smell

Primitive types should be omitted from initialized or defaulted declarations

Code Smell

Non-null assertions should not be used

Code Smell

"undefined" should not be assigned

Code Smell