




 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... 🔍

| | |
|--|------------------|
| Jump statements should not occur in "finally" blocks | Bug |
| Using slow regular expressions is security-sensitive | Security Hotspot |
| Using publicly writable directories is security-sensitive | Security Hotspot |
| Using clear-text protocols is security-sensitive | Security Hotspot |
| Expanding archive files without controlling resource consumption is security-sensitive | Security Hotspot |
| Using weak hashing algorithms is security-sensitive | Security Hotspot |
| Disabling CSRF protections is security-sensitive | Security Hotspot |
| Using pseudorandom number generators (PRNGs) is security-sensitive | Security Hotspot |
| Dynamically executing code is security-sensitive | Security Hotspot |
| Equality operators should not be used in "for" loop termination conditions | Code Smell |
| Tests should not execute any code after "done()" is called | Code Smell |

"switch" statements should not contain non-case labels

Analyze your code

Code Smell

Blocker

suspicious

Even if it is legal, mixing case and non-case labels in the body of a switch statement is very confusing and can even be the result of a typing error.

Noncompliant Code Example

Case 1, the code is syntactically correct but the behavior is not the expected one

```
switch (day) {
  case MONDAY:
  case TUESDAY:
  WEDNESDAY: // instead of "case WEDNESDAY"
    doSomething();
    break;
  ...
}
```

Case 2, the code is correct and behaves as expected but is hardly readable

```
switch (day) {
  case MONDAY:
    break;
  case TUESDAY:
    foo:for(i = 0 ; i < X ; i++) {
      /* ... */
      break foo; // this break statement doesn't relate to
      /* ... */
    }
    break;
  /* ... */
}
```

Compliant Solution

Case 1

```
switch (day) {
  case MONDAY:
  case TUESDAY:
  case WEDNESDAY:
    doSomething();
    break;
  ...
}
```

Case 2

```
switch (day) {
  case MONDAY:
```

Union and intersection types should not be defined with duplicated elements

 Code Smell

"default" clauses should be last

 Code Smell

"await" should only be used with promises

 Code Smell

A conditionally executed single line should be denoted by indentation

 Code Smell

```
break;
case TUESDAY:
  compute(args); // put the content of the labelled "for"
  break;

  /* ... */
}
```

Available In:
 |  | 

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.
[Privacy Policy](#)