




 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... 🔍

|   |
|---|
| 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  |
| Trailing commas should not be used  |
| Code Smell  |
| Array constructors should not be used   |
| Code Smell  |
| Quotes for string literals should be used consistently                        |
| Code Smell  |
| Statements should end with semicolons   |
| Code Smell  |
| Comments should not be located at the end of lines of code                    |
| Code Smell  |
| Loops should not contain more than a single "break" or "continue" statement   |
| Code Smell  |
| Variable, property and parameter names should comply with a naming convention |
| Code Smell  |
| Lines should not end with trailing whitespaces                                |
| Code Smell  |

Variables declared with "var" should be declared before they are used

Analyze your code

Code Smell

Blocker

pitfall

Variables declared with `var` have the special property that regardless of where they're declared in a function they "float" to the top of the function and are available for use even before they're declared. That makes scoping confusing, especially for new coders.

To keep confusion to a minimum, `var` declarations should happen before they are used for the first time.

Noncompliant Code Example

```
var x = 1;

function fun(){
  alert(x); // Noncompliant as x is declared later in the sa
  if(something) {
    var x = 42; // Declaration in function scope (not block
  }
}

fun(); // Unexpectedly alerts "undefined" instead of "1"
```

Compliant Solution

```
var x = 1;

function fun() {
  print(x);
  if (something) {
    x = 42;
  }
}

fun(); // Print "1"
```

Available In:

sonarlint | sonarcloud | sonarqube

|   |
|---|
| <div>Files should contain an empty newline at the end</div> <div> Code Smell</div>                 |
| <div>An open curly brace should be located at the end of a line</div> <div> Code Smell</div>       |
| <div>Tabulation characters should not be used</div> <div> Code Smell</div>                         |
| <div>Function and method names should comply with a naming convention</div> <div> Code Smell</div> |