




 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


 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


Tags ▾


Search by name... 🔍


Redundant casts and non-null assertions should be avoided

Analyze your code

 Code Smell

 Minor

 Quick Fix

 redundant

The TypeScript compiler automatically casts a variable to the relevant type inside conditionals where it is possible to infer the type (because `typeof`, `instanceof`, etc was used). This compiler feature makes casts and not-null assertions unnecessary.

Noncompliant Code Example

```
function getName(x?: string | UserName) {
  if (x) {
    console.log("Getting name for " + x!); // Noncompliant

    if (typeof x === "string")
      return (x as string); // Noncompliant
    else
      return (x as UserName).name; // Noncompliant
  }
  return "NoName";
}
```

Compliant Solution

```
function getName(x?: string | UserName) {
  if (x) {
    console.log("Getting name for " + x);

    if (typeof x === "string")
      return x;
    else
      return x.name;
  }
  return "NoName";
}
```

Available In:

<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>