sonar RULES Products

Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

All rules 632 | Vulnerability 53 | Bug 154 | Security Hotspot 36 | Code Smell 389 | Quick Fix 42

Tags Search by name...

Local-Variable Type Inference should be used

Analyze your code

Code Smell Info java10

In Java 10 Local-Variable Type Inference was introduced. It allows you to omit the expected type of a variable by declaring it with the `var` keyword.

While it is not always possible or cleaner to use this new way of declaring a variable, when the type on the left is the same as the one on the right in an assignment, using the `var` will result in a more concise code.

This rule reports an issue when the expected type of the variable is the same as the returned type of assigned expression and the type can be easily inferred by the reader, either when the tye is already mentioned in the name or the initializer, or when the expression is self-explained.

Noncompliant Code Example

```
MyClass myClass = new MyClass();

int i = 10; // Type is self explained

MyClass something = MyClass.getMyClass(); // Type is already

MyClass myClass = get(); // Type is already mentioned in th
```

Compliant Solution

```
var myClass = new MyClass();

var i = 10;

var something = MyClass.getMyClass();

var myClass = get();
```

See

- JEP 286: Local-Variable Type Inference

Available In:

sonarlint | sonarcloud | sonarqube

Local-Variable Type Inference should be used

Code Smell

Migrate your tests from JUnit4 to the new JUnit5 annotations

Code Smell

Track uses of disallowed classes

Code Smell

Track uses of "@SuppressWarnings" annotations

Code Smell