sonar RULES

Products ⌄

🚫	Secrets
SAP	ABAP
APEX	Apex
C	C
C++	C++
	CloudFormation
COBOL	COBOL
C#	C#
CSS	CSS
✳	Flex
GO	Go
	HTML
☕	**Java**
JS	JavaScript
	Kotlin
	Objective C
php	PHP
PL/I	PL/I
PL/SQL	PL/SQL
	Python
RPG	RPG
	Ruby
	Scala
	Swift
	Terraform
	Text
TS	TypeScript
	T-SQL
VB	VB.NET
VB6	VB6
XML	XML

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 variables
🔴 Code Smell

"public static" fields should be constant
🔴 Code Smell

Loops should not contain more than a single "break" or "continue" statement
🔴 Code Smell

Declarations should use Java collection interfaces such as "List" rather than specific implementation classes such as "LinkedList"
🔴 Code Smell

"switch" statements should have at least 3 "case" clauses
🔴 Code Smell

A "while" loop should be used instead of a "for" loop
🔴 Code Smell

The default unnamed package should not be used
🔴 Code Smell

"equals(Object obj)" should be overridden along with the "compareTo(T obj)" method
🔴 Code Smell

Package names should comply with a naming convention
🔴 Code Smell

Nested code blocks should not be used
🔴 Code Smell

Array designators "[]" should be on the type, not the variable
🔴 Code Smell

Redundant constructors/methods should be avoided in records

Analyze your code

🔴 Code Smell   🔺 Major ⑦   🏷 java16

In Java 16 records represent a brief notation for immutable data structures. Records have autogenerated implementations for constructors with all parameters, `getters`, `equals`, `hashcode` and `toString`. Although these methods can still be overridden inside records, there is no use to do so if no special logic is required.

This rule reports an issue on empty compact constructors, trivial canonical constructors and simple getter methods with no additional logic.

Noncompliant Code Example

```
record Person(String name, int age) {
  Person(String name, int age) { // Noncompliant, already au
    this.name = name;
    this.age = age;
  }
}

record Person(String name, int age) {
  Person { // Noncompliant, no need for empty compact constr
  }
  public String name() { // Noncompliant, already autogenera
    return name;
  }
}
```

Compliant Solution

```
record Person(String name, int age) { } // Compliant

record Person(String name, int age) {
  Person(String name, int age) { // Compliant
    this.name = name.toLowerCase(Locale.ROOT);
    this.age = age;
  }
}

record Person(String name, int age) {
  Person { // Compliant
    if (age < 0) {
      throw new IllegalArgumentException("Negative age");
    }
  }
  public String name() { // Compliant
    return name.toUpperCase(Locale.ROOT);
  }
}
```

See

Array designators "[]" should be located after the type in method signatures

⊗ Code Smell

Type parameter names should comply with a naming convention

⊗ Code Smell

Overriding methods should do more than simply call the same method in the super class

⊗ Code Smell

Classes that override "clone" should be "Cloneable" and call "super.clone()"

⊗ Code Smell

- Records specification

Available In:

sonarlint ⊝ | sonarcloud ⬡ | sonarqube ⑉