




 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



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... 🔍

Abstract class names should comply with a naming convention
Code Smell
Strings literals should be placed on the left side when checking for equality
Code Smell
Files should contain an empty newline at the end
Code Smell
Source code should be indented consistently
Code Smell
A close curly brace should be located at the beginning of a line
Code Smell
Close curly brace and the next "else", "catch" and "finally" keywords should be on two different lines
Code Smell
Close curly brace and the next "else", "catch" and "finally" keywords should be located on the same line
Code Smell
An open curly brace should be located at the beginning of a line
Code Smell
An open curly brace should be located at the end of a line
Code Smell
Tabulation characters should not be used
Code Smell
Functions should not be defined with a variable number of arguments
Code Smell

Classes and enums with private members should have a constructor

Analyze your code

Code Smell

Major ?

pitfall

Non-abstract classes and enums with non-static, private members should explicitly initialize those members, either in a constructor or with a default value.

Noncompliant Code Example

```
class A { // Noncompliant
    private int field;
}
```

Compliant Solution

```
class A {
    private int field;

    A(int field) {
        this.field = field;
    }
}
```

Exceptions

- Class implementing a Builder Pattern (name ending with "Builder").
- Java EE class annotated with:
 - ManagedBean
 - MessageDriven
 - Singleton
 - Stateful
 - Stateless
 - WebService
 - WebFilter
 - WebServlet
- Class and field annotated with:
 - Plexus Component Annotations
 - Maven Mojo
- Field annotated with:
 - Resource
 - EJB
 - Inject
 - Autowired

Available In:

sonarlint | sonarcloud | sonarqube

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](#)

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