sonar RULES

Products

- 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...

- Classes should not have too many methods — Code Smell
- Methods should not have too many lines — Code Smell
- Track uses of "NOSONAR" comments — Code Smell
- Classes and enums with private members should have a constructor — Code Smell
- Track comments matching a regular expression — Code Smell
- Statements should be on separate lines — Code Smell
- Classes should not be coupled to too many other classes (Single Responsibility Principle) — Code Smell
- "java.lang.Error" should not be extended — Code Smell
- Anonymous classes should not have too many lines — Code Smell
- Public types, methods and fields (API) should be documented with Javadoc — Code Smell
- Exception handlers should preserve the original exceptions — Code Smell
- Checked exceptions should not be

"Threads" should not be used where "Runnables" are expected

Analyze your code

Code Smell Major multi-threading pitfall

While it is technically correct to use a `Thread` where a `Runnable` is called for, the semantics of the two objects are different, and mixing them is a bad practice that will likely lead to headaches in the future.

The crux of the issue is that `Thread` is a larger concept than `Runnable`. A `Runnable` is an object whose running should be managed. A `Thread` expects to manage the running of itself or other `Runnables`.

Noncompliant Code Example

```
public static void main(String[] args) {
    Thread r =new Thread() {
        int p;
        @Override
        public void run() {
            while(true)
                System.out.println("
        }
    };
    new Thread(r).start();  // Noncompliant
```

Compliant Solution

```
public static void main(String[] args) {
    Runnable r =new Runnable() {
        int p;
        @Override
        public void run() {
            while(true)
                System.out.println("
        }
    };
    new Thread(r).start();
```

Available In:

sonarlint | sonarcloud | sonarqube

Checked exceptions should not be thrown

Code Smell

Public methods should throw at most one checked exception

Code Smell

"switch case" clauses should not have too many lines of code

Code Smell

Methods should not have too many return statements

Code Smell

Magic numbers should not be used