

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

Java static code analysis

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

All rules632

Vulnerability53

Bug154

Security Hotspot36

Code Smell389

Quick Fix42

wait , notify and notifyAll should only be called when a lock is obviously held on an object

Bug

Null pointers should not be dereferenced

Bug

Loop conditions should be true at least once

Bug

A "for" loop update clause should move the counter in the right direction

Bug

Non-public methods should not be "@Transactional"

Bug

Servlets should not have mutable instance fields

Bug

"toString()" and "clone()" methods should not return null

Bug

".equals()" should not be used to test the values of "Atomic" classes

Bug

Return values from functions without side effects should not be ignored

Bug

Child class methods named for parent class methods should be overrides

Bug

Inappropriate "Collection" calls should not be made

Bug

Silly equality checks should not be made

Bug

Factory method injection should be used in "@Configuration" classes

Analyze your code

Code SmellCriticalspring performance

When @Autowired is used, dependencies need to be resolved when the class is instantiated, which may cause early initialization of beans or lead the context to look in places it shouldn't to find the bean. To avoid this tricky issue and optimize the way the context loads, dependencies should be requested as late as possible. That means using parameter injection instead of field injection for dependencies that are only used in a single @Bean method.

Noncompliant Code Example

```
@Configuration
public class FooConfiguration {

    @Autowired private DataSource dataSource; // Noncompliant

    @Bean
    public MyService myService() {
        return new MyService(this.dataSource);
    }
}
```

Compliant Solution

```
@Configuration
public class FooConfiguration {

    @Bean
    public MyService myService(DataSource dataSource) {
        return new MyService(dataSource);
    }
}
```

Exceptions

Fields used in methods that are called directly by other methods in the application (as opposed to being invoked automatically by the Spring framework) are ignored by this rule so that direct callers don't have to provide the dependencies themselves.

Available In:

sonarlint





sonarcloud

sonarqube

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective

https://rules.sonarsource.com/java/RSPEC-3305

1/2

 Bug
Dissimilar primitive wrappers should not be used with the ternary operator without explicit casting
 Bug
"InterruptedException" should not be ignored
 Bug
Classes extending java.lang.Thread should override the "run" method
 Bug