 sonar RULES

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

| |
|---|
| Hard-coded credentials are security-sensitive |
| Security Hotspot |
| Methods returns should not be invariant |
| Code Smell |
| "ThreadGroup" should not be used |
| Code Smell |
| "clone" should not be overridden |
| Code Smell |
| Assertions should be complete |
| Code Smell |
| Tests should include assertions |
| Code Smell |
| Silly bit operations should not be performed |
| Code Smell |
| Child class fields should not shadow parent class fields |
| Code Smell |
| JUnit test cases should call super methods |
| Code Smell |
| TestCases should contain tests |
| Code Smell |
| Short-circuit logic should be used in boolean contexts |
| Code Smell |
| Methods and field names should not be the same or differ only by capitalization |
| Code Smell |

Tags ▾

Search by name... 🔍

"@SpringBootApplication" and "@ComponentScan" should not be used in the default package

Analyze your code

Bug Blocker ? spring

@ComponentScan is used to determine which Spring Beans are available in the application context. The packages to scan can be configured thanks to the basePackageClasses or basePackages (or its alias value) parameters. If neither parameter is configured, @ComponentScan will consider only the package of the class annotated with it. When @ComponentScan is used on a class belonging to the default package, the entire classpath will be scanned.

This will slow-down the start-up of the application and it is likely the application will fail to start with an BeanDefinitionStoreException because you ended up scanning the Spring Framework package itself.

- This rule raises an issue when:
- @ComponentScan, @SpringBootApplication and @ServletComponentScan are used on a class belonging to the default package
 - @ComponentScan is explicitly configured with the default package

Noncompliant Code Example

```
import org.springframework.boot.SpringApplication;

@SpringBootApplication // Noncompliant; RootBootApplication is declared
public class RootBootApplication {
    ...
}
```

```
@ComponentScan("")
public class Application {
    ...
}
```





Compliant Solution

```
package hello;

import org.springframework.boot.SpringApplication;

@SpringBootApplication // Compliant; RootBootApplication belongs to
public class RootBootApplication {
    ...
}
```

Available In:
sonarlint | sonarcloud | sonarqube

| |
|---|
| Switch cases should end with an unconditional "break" statement |
|  Code Smell |
| "switch" statements should not contain non-case labels |
|  Code Smell |
| Future keywords should not be used as names |
|  Code Smell |
| Thread suspensions should not be vulnerable to Denial of Service attacks |
|  Vulnerability |

© 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 owners. All rights are expressly reserved.
[Privacy Policy](#)