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...

Security Hotspot

Mocking all non-private methods of a class should be avoided
Code Smell

Empty lines should not be tested with regex MULTILINE flag
Code Smell

Methods setUp() and tearDown() should be correctly annotated starting with JUnit4
Code Smell

Class members annotated with "@VisibleForTesting" should not be accessed from production code
Code Smell

"String#replace" should be preferred to "String#replaceAll"
Code Smell

Derived exceptions should not hide their parents' catch blocks
Code Smell

String offset-based methods should be preferred for finding substrings from offsets
Code Smell

"default" clauses should be last
Code Smell

"equals" method parameters should not be marked "@Nonnull"
Code Smell

A conditionally executed single line should be denoted by indentation
Code Smell

Conditionals should start on new lines
Code Smell

## Cryptographic keys should be robust

Analyze your code

Vulnerability | Critical | cwe privacy owasp rules

Most of cryptographic systems require a sufficient key size to be robust against brute-force attacks.

NIST recommendations will be checked for these use-cases:

**Digital Signature Generation** and **Verification:**

- p ≥ 2048 AND q ≥ 224 for DSA (p is key length and q the modulus length)
- n ≥ 2048 for RSA (n is the key length)

**Key Agreement**:

- p ≥ 2048 AND q ≥ 224 for DH and MQV
- n ≥ 224 for ECDH and ECMQV (Examples: `secp192r1` is a non-compliant curve (n < 224) but `secp224k1` is compliant (n >= 224))

**Symmetric keys**:

- key length ≥ 128 bits

This rule will not raise issues for ciphers that are considered weak (no matter the key size) like `DES`, `Blowfish`.

### Noncompliant Code Example

```
KeyPairGenerator keyPairGen1 = KeyPairGenerator.getInstance(
keyPairGen1.initialize(1024); // Noncompliant

KeyPairGenerator keyPairGen5 = KeyPairGenerator.getInstance(
ECGenParameterSpec ecSpec1 = new ECGenParameterSpec("secp112
keyPairGen5.initialize(ecSpec1);

KeyGenerator keyGen1 = KeyGenerator.getInstance("AES");
keyGen1.init(64); // Noncompliant
```

### Compliant Solution

```
KeyPairGenerator keyPairGen6 = KeyPairGenerator.getInstance(
keyPairGen6.initialize(2048); // Compliant

KeyPairGenerator keyPairGen5 = KeyPairGenerator.getInstance(
ECGenParameterSpec ecSpec10 = new ECGenParameterSpec("secp25
keyPairGen5.initialize(ecSpec10);

KeyGenerator keyGen2 = KeyGenerator.getInstance("AES");
keyGen2.init(128); // Compliant
```

### See

- OWASP Top 10 2021 Category A2 - Cryptographic Failures
- OWASP Top 10 2017 Category A3 - Sensitive Data Exposure

Cognitive Complexity of methods should not be too high

Code Smell

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

Code Smell

"static" base class members should not be accessed via derived types

Code Smell

Instance methods should not write to "static" fields

Code Smell

- OWASP Top 10 2017 Category A6 - Security Misconfiguration
- Mobile AppSec Verification Standard - Cryptography Requirements
- OWASP Mobile Top 10 2016 Category M5 - Insufficient Cryptography
- NIST 800-131A - Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths
- MITRE, CWE-326 - Inadequate Encryption Strength

Available In:

sonarlint | sonarcloud | sonarqube

Privacy Policy