




 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 rules632

Vulnerability53

Bug154

Security Hotspot36

Code Smell389

Quick Fix42

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

OS commands should not be vulnerable to argument injection attacks

Analyze your code

VulnerabilityMinor🔍injection cwe owasp sans-top25

Applications that allow execution of operating system commands from user-controlled data should control the arguments passed to the command, otherwise an attacker can inject additional arbitrary arguments which can change the behavior of the command.

User-controlled arguments should be sanitized by neutralizing argument delimiters (eg: ' , space, -) and thus preventing injection of unwanted additional arguments. A single user-controlled argument may still lead to vulnerabilities if it corresponds to a dangerous option supported by the command, such as -exec available with **find**, in that case, mark end of option processing on the command line using -- (double-dash) or restrict the options to only trusted values.

Noncompliant Code Example

```
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;

public void runUnsafe(HttpServletRequest request) throws IOException {
    String folder = request.getParameter("folder");

    String cmd = "mkdir " + folder;

    Runtime.getRuntime().exec(cmd); // Noncompliant
}
```

Compliant Solution

```
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;

public void runSafe(HttpServletRequest request) throws IOException {
    String folderarg1 = request.getParameter("folder");

    String cmd[] = new String[] { "mkdir", folderarg1 };

    Runtime.getRuntime().exec(cmd); // Compliant
}
```

See

- OWASP Top 10 2021 Category A3 - Injection
- OWASP OS Command Injection Defense [Cheat Sheet](#)
- OWASP Top 10 2017 Category A1 - Injection
- MITRE, CWE-20 - Improper Input Validation
- MITRE, CWE-88 - Argument Injection or Modification
- SANS Top 25 - Insecure Interaction Between Components

https://rules.sonarsource.com/java/RSPEC-5883

1/2

<b>Local-Variable Type Inference should be used</b>  Code Smell
<b>Migrate your tests from JUnit4 to the new JUnit5 annotations</b>  Code Smell
<b>Track uses of disallowed classes</b>  Code Smell
<b>Track uses of "@SuppressWarnings" annotations</b>  Code Smell

Available In:  
**sonarcloud**  | **sonarqube**  Developer Edition

© 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](#)