


-  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... 🔍

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

Unicode-aware versions of character classes should be preferred

Analyze your code

Code Smell

Minor ?

regex

When using POSIX classes like `\p{Alpha}` without the `UNICODE_CHARACTER_CLASS` flag or when using hard-coded character classes like `"[a-zA-Z]"`, letters outside of the ASCII range, such as umlauts, accented letters or letter from non-Latin languages, won't be matched. This may cause code to incorrectly handle input containing such letters.

To correctly handle non-ASCII input, it is recommended to use Unicode classes like `\p{IsAlphabetic}`. When using POSIX classes, Unicode support should be enabled by either passing `Pattern.UNICODE_CHARACTER_CLASS` as a flag to `Pattern.compile` or by using `(?U)` inside the regex.




Noncompliant Code Example

```
Pattern.compile("[a-zA-Z]");
Pattern.compile("\\p{Alpha}");
```

Compliant Solution

```
Pattern.compile("\\p{IsAlphabetic}"); // matches all letters
Pattern.compile("\\p{IsLatin}"); // matches latin letters, i
Pattern.compile("\\p{Alpha}", Pattern.UNICODE_CHARACTER_CLAS
Pattern.compile("(?U)\\p{Alpha}");
```

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 owners. All rights are expressly reserved.

Privacy Policy

https://rules.sonarsource.com/java/RSPEC-5867

1/2

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