




 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... 🔍


 Code Smell


Track uses of "NOSONAR" comments

 Code Smell


 Code Smell


Classes and enums with private members should have a constructor

 Code Smell


 Code Smell


Track comments matching a regular expression

 Code Smell


 Code Smell


Statements should be on separate lines

 Code Smell


 Code Smell


Classes should not be coupled to too many other classes (Single Responsibility Principle)

 Code Smell


 Code Smell


"java.lang.Error" should not be extended

 Code Smell

 Code Smell

Anonymous classes should not have too many lines

 Code Smell

 Code Smell

Public types, methods and fields (API) should be documented with Javadoc

 Code Smell

 Code Smell

Exception handlers should preserve the original exceptions

 Code Smell

 Code Smell

Checked exceptions should not be thrown

 Code Smell


 Code Smell


Public methods should throw at most one checked exception


 Code Smell

### Unused type parameters should be removed

Analyze your code

 Code Smell

 Major ?

 cert unused




Type parameters that aren't used are dead code, which can only distract and possibly confuse developers during maintenance. Therefore, unused type parameters should be removed.

#### Noncompliant Code Example

```
int <T> Add(int a, int b) // Noncompliant; <T> is ignored
{
    return a + b;
}
```

#### Compliant Solution





```
int Add(int a, int b)
{
    return a + b;
}
```

Available In:  
 |  | 

© 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-2326

1/2

<div>"switch case" clauses should not have too many lines of code</div> <div> Code Smell</div>
<div>Methods should not have too many return statements</div> <div> Code Smell</div>
<div>Magic numbers should not be used</div> <div> Code Smell</div>
<div>Files should not have too many lines of code</div> <div> Code Smell</div>