




 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

"ResultSet.isLast()" should not be used

Code Smell

Code Smell

"static" members should be accessed statically

Code Smell

Code Smell

Silly math should not be performed

Code Smell

Code Smell

Classes named like "Exception" should extend "Exception" or a subclass

Code Smell

Code Smell

Exceptions should be either logged or rethrown but not both

Code Smell

Code Smell

Objects should not be created only to "getClass"

Code Smell

Code Smell

Primitives should not be boxed just for "String" conversion

Code Smell

Code Smell

Constructors should not be used to instantiate "String", "BigInteger", "BigDecimal" and primitive-wrapper classes

Code Smell

Code Smell

"URL.hashCode" and "URL.equals" should be avoided

Code Smell

Code Smell

Two branches in a conditional structure should not have exactly the same implementation

Code Smell

Code Smell

Unused assignments should be removed

Code Smell

Week Year ("YYYY") should not be used for date formatting

Analyze your code

Bug

Major

Quick Fix

suspicious

Few developers are aware of the difference between `y` for "Week year" and `Y` for Year when formatting and parsing a date with `SimpleDateFormat` or `DateTimeFormatter`. That's likely because for most dates, Week year and Year are the same, so testing at any time other than the first or last week of the year will yield the same value for both `y` and `Y`. But in the last week of December and the first week of January, you may get unexpected results.

According to the [Javadoc](#):

A week year is in sync with a `WEEK_OF_YEAR` cycle. All weeks between the first and last weeks (inclusive) have the same week year value. Therefore, the first and last days of a week year may have different calendar year values. For example, January 1, 1998 is a Thursday. If `getFirstDayOfWeek()` is `MONDAY` and `getMinimalDaysInFirstWeek()` is 4 (ISO 8601 standard compatible setting), then week 1 of 1998 starts on December 29, 1997, and ends on January 4, 1998. The week year is 1998 for the last three days of calendar year 1997. If, however, `getFirstDayOfWeek()` is `SUNDAY`, then week 1 of 1998 starts on January 4, 1998, and ends on January 10, 1998; the first three days of 1998 then are part of week 53 of 1997 and their week year is 1997.

Noncompliant Code Example

```
Date date = new SimpleDateFormat("yyyy/MM/dd").parse("2015/1");
String result = new SimpleDateFormat("YYYY/MM/dd").format(date);
result = DateTimeFormatter.ofPattern("YYYY/MM/dd").format(date);
```




Compliant Solution

```
Date date = new SimpleDateFormat("yyyy/MM/dd").parse("2015/1");
String result = new SimpleDateFormat("yyyy/MM/dd").format(date);
result = DateTimeFormatter.ofPattern("yyyy/MM/dd").format(date);
```

Exceptions

```
Date date = new SimpleDateFormat("yyyy/MM/dd").parse("2015/1");
String result = new SimpleDateFormat("YYYY-ww").format(date);
DateTimeFormatter.ofPattern("YYYY-ww").format(date); //compliant
```

Available In:

 |  | 

https://rules.sonarsource.com/java/RSPEC-3986

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR,

1/2

"Object.wait(...)" should never be called on objects that implement "java.util.concurrent.locks.Condition"

A field should not duplicate the name of its containing class

JUnit4 @Ignored and JUnit5 @Disabled annotations should be used to disable tests and should provide a rationale

Anonymous inner classes containing