

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

Java

used to add "null" values.

Bug

Regex lookahead assertions should not be contradictory

Bug

Back references in regular expressions should only refer to capturing groups that are matched before the reference

Bug

Regex boundaries should not be used in a way that can never be matched

Bug

Regex patterns following a possessive quantifier should not always fail

Bug

Regular expressions should be syntactically valid

Bug

Assertions comparing incompatible types should not be made

Bug

JUnit5 inner test classes should be annotated with @Nested

Bug

Only one method invocation is expected when testing checked exceptions

Bug

Assertion methods should not be used within the try block of a try-catch catching an Error

Bug

Getters and setters should access the expected fields

Bug

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... 🔍

Methods and field names should not be the same or differ only by capitalization

Analyze your code

Code Smell

Blocker

confusing

Looking at the set of methods in a class, including superclass methods, and finding two methods or fields that differ only by capitalization is confusing to users of the class. It is similarly confusing to have a method and a field which differ only in capitalization or a method and a field with exactly the same name and visibility.

In the case of methods, it may have been a mistake on the part of the original developer, who intended to override a superclass method, but instead added a new method with nearly the same name.

Otherwise, this situation simply indicates poor naming. Method names should be action-oriented, and thus contain a verb, which is unlikely in the case where both a method and a member have the same name (with or without capitalization differences). However, renaming a public method could be disruptive to callers. Therefore renaming the member is the recommended action.

Noncompliant Code Example

```
public class Car{

    public DriveTrain drive;

    public void tearDown(){...}

    public void drive() {...} // Noncompliant; duplicates field
}

public class MyCar extends Car{
    public void teardown(){...} // Noncompliant; not an override

    public void drivefast(){...}

    public void driveFast(){...} //Huh?
}
```

Compliant Solution





```
public class Car{

    private DriveTrain drive;

    public void tearDown(){...}

    public void drive() {...} // field visibility reduced
}

public class MyCar extends Car{
    @Override
    public void tearDown(){...}
}
```

<div>Zero should not be a possible denominator</div> <div> Bug</div>
<div>Locks should be released</div> <div> Bug</div>
<div>"runFinalizersOnExit" should not be called</div> <div> Bug</div>
<div>"ScheduledThreadPoolExecutor" should not have 0 core threads</div> <div> Bug</div>

```
public void drivefast(){...}

public void driveReallyFast(){...}

}
```

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](#)