




 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

Mutable members should not be stored or returned directly

Analyze your code

Code Smell

Minor

cwe unpredictable cert

Mutable objects are those whose state can be changed. For instance, an array is mutable, but a String is not. Mutable class members should never be returned to a caller or accepted and stored directly. Doing so leaves you vulnerable to unexpected changes in your class state.

Instead use an unmodifiable Collection (via `Collections.unmodifiableCollection`, `Collections.unmodifiableList`, ...) or make a copy of the mutable object, and store or return the copy instead.

This rule checks that arrays, collections and Dates are not stored or returned directly.

Noncompliant Code Example

```
class A {
    private String [] strings;

    public A () {
        strings = new String[]{"first", "second"};
    }

    public String [] getStrings() {
        return strings; // Noncompliant
    }

    public void setStrings(String [] strings) {
        this.strings = strings; // Noncompliant
    }
}

public class B {

    private A a = new A(); // At this point a.strings = {"fir

    public void wreakHavoc() {
        a.getStrings()[0] = "yellow"; // a.strings = {"yellow",
    }
}
```

Compliant Solution

```
class A {
    private String [] strings;

    public A () {
        strings = new String[]{"first", "second"};
    }

    public String [] getStrings() {
```

https://rules.sonarsource.com/java/RSPEC-2384

1/2


Local-Variable Type Inference should be used

 Code Smell

Migrate your tests from JUnit4 to the new JUnit5 annotations

 Code Smell

Track uses of disallowed classes

 Code Smell

Track uses of "@SuppressWarnings" annotations

 Code Smell

```
return strings.clone();
}

public void setStrings(String [] strings) {
    this.strings = strings.clone();
}
}

public class B {

    private A a = new A(); // At this point a.strings = {"fir

    public void wreakHavoc() {
        a.getStrings()[0] = "yellow"; // a.strings = {"first",
    }
}
```

See

- [MITRE, CWE-374](#) - Passing Mutable Objects to an Untrusted Method
- [MITRE, CWE-375](#) - Returning a Mutable Object to an Untrusted Caller
- [CERT, OBJ05-J](#) - Do not return references to private mutable class members
- [CERT, OBJ06-J](#) - Defensively copy mutable inputs and mutable internal components
- [CERT, OBJ13-J](#) - Ensure that references to mutable objects are not exposed

Available In:

 |  | 