sonar RULES

Products ⌄

- 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...

An iteration on a Collection should be performed on the type handled by the Collection
Code Smell

"StandardCharsets" constants should be preferred
Code Smell

"@CheckForNull" or "@Nullable" should not be used on primitive types
Code Smell

Composed "@RequestMapping" variants should be preferred
Code Smell

"write(byte[],int,int)" should be overridden
Code Smell

Functional Interfaces should be as specialised as possible
Code Smell

Null checks should not be used with "instanceof"
Code Smell

"close()" calls should not be redundant
Code Smell

"ThreadLocal.withInitial" should be preferred
Code Smell

"Stream" call chains should be simplified when possible
Code Smell

Packages containing only "package-info.java" should be removed
Code Smell

Arrays should not be created for varargs parameters

Classes should not be compared by name

Analyze your code

Bug Major cwe cert

There is no requirement that class names be unique, only that they be unique within a package. Therefore trying to determine an object's type based on its class name is an exercise fraught with danger. One of those dangers is that a malicious user will send objects of the same name as the trusted class and thereby gain trusted access.

Instead, the `instanceof` operator or the `Class.isAssignableFrom()` method should be used to check the object's underlying type.

Noncompliant Code Example

```
package computer;
class Pear extends Laptop { ... }

package food;
class Pear extends Fruit { ... }

class Store {

  public boolean hasSellByDate(Object item) {
    if ("Pear".equals(item.getClass().getSimpleName())) {  /
      return true;  // Results in throwing away week-old com
    }
    return false;
  }

  public boolean isList(Class<T> valueClass) {
    if (List.class.getName().equals(valueClass.getName())) {
      return true;
    }
    return false;
  }
}
```

Compliant Solution

```
class Store {

  public boolean hasSellByDate(Object item) {
    if (item instanceof food.Pear) {
      return true;
    }
    return false;
  }

  public boolean isList(Class<T> valueClass) {
    if (valueClass.isAssignableFrom(List.class)) {
      return true;
    }
```

Code Smell

Jump statements should not be redundant

Code Smell

Test classes should comply with a naming convention

Code Smell

Loggers should be named for their enclosing classes

Code Smell

Methods should not return constants

```
    return false;
  }
}
```

See

- MITRE, CWE-486 - Comparison of Classes by Name
- CERT, OBJ09-J. - Compare classes and not class names

Available In:

sonarlint | sonarcloud | sonarqube