Java static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your JAVA code

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

"private" and "final" methods that don't access instance data should be "static"

Analyze your code

Code Smell | Minor | Quick Fix | pitfall

Non-overridable methods (`private` or `final`) that don't access instance data can be `static` to prevent any misunderstanding about the contract of the method.

Noncompliant Code Example

```
class Utilities {
  private static String magicWord = "magic";

  private String getMagicWord() { // Noncompliant
    return magicWord;
  }

  private void setMagicWord(String value) { // Noncompliant
    magicWord = value;
  }

}
```

Compliant Solution

```
class Utilities {
  private static String magicWord = "magic";

  private static String getMagicWord() {
    return magicWord;
  }

  private static void setMagicWord(String value) {
    magicWord = value;
  }

}
```

Exceptions

When `java.io.Serializable` is implemented the following three methods are excluded by the rule:

- `private void writeObject(java.io.ObjectOutputStream out) throws IOException;`
- `private void readObject(java.io.ObjectInputStream in) throws IOException, ClassNotFoundException;`
- `private void readObjectNoData() throws ObjectStreamException;`

Available In:

sonarlint | sonarcloud | sonarqube

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