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

Utility classes should not have public constructors

Analyze your code

Code Smell | Major | design

Utility classes, which are collections of `static` members, are not meant to be instantiated. Even abstract utility classes, which can be extended, should not have public constructors.

Java adds an implicit public constructor to every class which does not define at least one explicitly. Hence, at least one non-public constructor should be defined.

Noncompliant Code Example

```
class StringUtils { // Noncompliant

  public static String concatenate(String s1, String s2) {
    return s1 + s2;
  }

}
```

Compliant Solution

```
class StringUtils { // Compliant

  private StringUtils() {
    throw new IllegalStateException("Utility class");
  }

  public static String concatenate(String s1, String s2) {
    return s1 + s2;
  }

}
```

Exceptions

When class contains `public static void main(String[] args)` method it is not considered as utility class and will be ignored by this rule.

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