




 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... 🔍

Literal suffixes should be upper case

Code Smell

Unicode-aware versions of character classes should be preferred

Code Smell

Use Java 12 "switch" expression

Code Smell

"serialVersionUID" should not be declared blindly

Code Smell

"Stream.collect()" calls should not be redundant

Code Smell

Local constants should follow naming conventions for constants

Code Smell

Unit tests should throw exceptions

Code Smell

Test methods should comply with a naming convention

Code Smell

Value-based objects should not be serialized

Code Smell

Default annotation parameter values should not be passed as arguments

Code Smell

Method parameters should be declared with base types

Code Smell

Fields should not be initialized to default values

Code Smell

### "URL.hashCode" and "URL.equals" should be avoided

Analyze your code

Code Smell

Major

performance

The equals and hashCode methods of java.net.URL both may trigger a name service (usually DNS) lookup to resolve the host name or IP address. Depending on the configuration, and network status, that can take a long time. URI on the other hand makes no such calls and should be used instead unless the specific URL functionality is required.

In general it is better to use the URI class until access to the resource is actually needed, at which point you can just convert the URI to a URL using URI.toURL().

This rule checks for uses of URL's in Map and Set, and for explicit calls to the equals and hashCode methods.

#### Noncompliant Code Example

```
public void checkUrl(URL url) {
    Set<URL> sites = new HashSet<URL>(); // Noncompliant

    URL homepage = new URL("http://sonarsource.com"); // Comp
    if (homepage.equals(url)) { // Noncompliant
        // ...
    }
}
```

#### Compliant Solution

```
public void checkUrl(URL url) {
    Set<URI> sites = new HashSet<URI>(); // Compliant

    URI homepage = new URI("http://sonarsource.com"); // Comp
    URI uri = url.toURI();
    if (homepage.equals(uri)) { // Compliant
        // ...
    }
}
```

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](#)

https://rules.sonarsource.com/java/RSPEC-2112

1/2

<div>Multiple loops over the same set should be combined</div> <div> Code Smell</div>
<div>Classes without "public" constructors should be "final"</div> <div> Code Smell</div>
<div>Unnecessary semicolons should be omitted</div> <div> Code Smell</div>
<div>Literal boolean values and nulls should not be used in assertions</div> <div> Code Smell</div>