


-  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

Vulnerability

Classes should not be loaded dynamically

Vulnerability

Equality operators should not be used in "for" loop termination conditions

Code Smell

"Bean Validation" (JSR 380) should be properly configured

Code Smell

Spring beans should be considered by "@ComponentScan"

Code Smell

Number patterns should be regular

Code Smell

Lazy initialization of "static" fields should be "synchronized"

Code Smell

Wildcard imports should not be used

Code Smell

Modulus results should not be checked for direct equality

Code Smell

Comparators should be "Serializable"

Code Smell

"Serializable" classes should have a "serialVersionUID"

Code Smell

"switch" statements and expressions should not be nested

Code Smell

Constructors should only call non-overrideable methods

"Class.forName()" should not load JDBC 4.0+ drivers

Analyze your code

Code Smell

Major

obsolete

In the past, it was required to load a JDBC driver before creating a `java.sql.Connection`. Nowadays, when using JDBC 4.0 drivers, this is no longer required and `Class.forName()` can be safely removed because JDBC 4.0 (JDK 6) drivers available in the classpath are automatically loaded.

This rule raises an issue when `Class.forName()` is used with one of the following values:

- `com.mysql.jdbc.Driver`
- `oracle.jdbc.driver.OracleDriver`
- `com.ibm.db2.jdbc.app.DB2Driver`
- `com.ibm.db2.jdbc.net.DB2Driver`
- `com.sybase.jdbc.SybDriver`
- `com.sybase.jdbc2.jdbc.SybDriver`
- `com.teradata.jdbc.TeraDriver`
- `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- `org.postgresql.Driver`
- `sun.jdbc.odbc.JdbcOdbcDriver`
- `org.hsqldb.jdbc.JDBCDriver`
- `org.h2.Driver`
- `org.firebirdsql.jdbc.FBDriver`
- `net.sourceforge.jtds.jdbc.Driver`
- `com.ibm.db2.jcc.DB2Driver`

Noncompliant Code Example

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Demo {
    private static final String DRIVER_CLASS_NAME = "org.postg
    private final Connection connection;

    public Demo(String serverURI) throws SQLException, ClassNo
        Class.forName(DRIVER_CLASS_NAME); // Noncompliant; no lo
        connection = DriverManager.getConnection(serverURI);
    }
}
```

Compliant Solution


```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class Demo {
    private final Connection connection;


    public Demo(String serverURI) throws SQLException {
```

https://rules.sonarsource.com/java/RSPEC-4925


1/2

 Code Smell


**Methods should not be too complex**

 Code Smell

**Control flow statements "if", "for", "while", "switch" and "try" should not be nested too deeply**

 Code Smell




**"if ... else if" constructs should end with "else" clauses**

 Code Smell

**Control structures should use curly braces**

```
        connection = DriverManager.getConnection(serverURI);
    }
}
```

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](#)