first

1.0

*java:Sonar way*

*js:Sonar way*

*xml:Sonar way*

*2020-07-29*

# 目录

# 1. first

报告提供了项目指标的概要，显示了与项目质量相关的最重要的指标。如果需要获取更详细的信息，请登陆网站进一步查询。

报告的项目为first，生成时间为2020-07-29，使用的质量配置为 java:Sonar way js:Sonar way xml:Sonar way，共计 427条规则。

## 1.1. 概述

### 编码问题

| | |
|---|---|
| Bug<br>2 | 可靠性修复工作<br>25min |
| 漏洞<br>2 | 安全修复工作<br>20min |
| 坏味道<br>54 | 技术债务<br>8h32min |

58
问题

| | |
|---|---|
| 开启问题 | 58 |
| 重开问题 | 0 |
| 确认问题 | 0 |
| 误判问题 | 0 |
| 不修复的问题 | 0 |
| 已解决的问题 | 0 |
| 已删除的问题 | 0 |
| 阻断 | 0 |
| 严重 | 5 |
| 主要 | 23 |
| 次要 | 30 |
| 提示 | 0 |

### 静态分析

项目规模

| | | |
|---|---|---|
| 8511 代码行数 | 行数 | 12580 |
| | 方法 | 839 |
| | 类 | 20 |
| | 文件 | 19 |
| | 目录 | 9 |
| | 重复行(%) | 3.9 |

复杂度

| | | |
|---|---|---|
| 2769 复杂度 | 文件 | 184.6 |

注释(%)

| | | |
|---|---|---|
| 16.4 注释(%) | 注释行数 | 1674 |

## 1.2. 问题分析

| 违反最多的规则TOP10 | |
|---|---|
| Useless imports should be removed | 6 |
| Generic exceptions should never be thrown | 6 |
| String literals should not be duplicated | 5 |
| Collection.isEmpty() should be used to test for emptiness | 5 |
| The diamond operator ("<>") should be used | 4 |
| Composed "@RequestMapping" variants should be preferred | 4 |
| Local variables should not be declared and then immediately returned or thrown | 3 |
| Nested blocks of code should not be left empty | 2 |
| A "while" loop should be used instead of a "for" loop | 2 |
| Boolean expressions should not be gratuitous | 2 |

| 违规最多的文件TOP5 | |
|---|---|
| CustomerExample.java | 11 |
| jquery-3.3.1.js | 10 |
| UserController.java | 10 |
| UserExample.java | 10 |
| CustomerController.java | 8 |

| 复杂度最高的文件TOP5 | |
|---|---|
| jquery-3.3.1.js | 2537 |
| UserExample.java | 92 |
| CustomerExample.java | 90 |
| UserController.java | 11 |
| Customer.java | 8 |

| 重复行最多的文件TOP5 | |
|---|---|
| CustomerExample.java | 246 |
| UserExample.java | 246 |

## 1.3. 问题详情

| | |
|---|---|
| 规则 | Useless imports should be removed |

| 规则描述 | The imports part of a file should be handled by the Integrated Development Environment (IDE), not manually by the developer. Unused and useless imports should not occur if that is the case. Leaving them in reduces the code's readability, since their presence can be confusing.<br>Noncompliant Code Example<br><br>package my.company;<br><br>import java.lang.String;        // Noncompliant; java.lang classes are always implicitly imported<br>import my.company.SomeClass;    // Noncompliant; same-package files are always implicitly imported<br>import java.io.File;            // Noncompliant; File is not used<br><br>import my.company2.SomeType;<br>import my.company2.SomeType;    // Noncompliant; 'SomeType' is already imported<br><br>class ExampleClass {<br><br>  public String someString;<br>  public SomeType something;<br><br>}<br><br>Exceptions<br>Imports for types mentioned in comments, such as Javadocs, are ignored. |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| CustomerController.java | 8, 11, 15, 16 |
| HomeController.java | 9 |
| CustomerServiceImpl.java | 9 |

| 规则 | Generic exceptions should never be thrown |
|---|---|

| | |
|---|---|
| 规则描述 | Using such generic exceptions as Error , RuntimeException , Throwable , and Exception prevents<br>calling methods from handling true, system-generated exceptions differently than application-generated errors.<br>Noncompliant Code Example<br><br>public void foo(String bar) throws Throwable { // Noncompliant<br>  throw new RuntimeException("My Message"); // Noncompliant<br>}<br><br>Compliant Solution<br><br>public void foo(String bar) {<br>  throw new MyOwnRuntimeException("My Message");<br>}<br><br>Exceptions<br>Generic exceptions in the signatures of overriding methods are ignored, because overriding method has to follow signature of the throw declaration<br>in the superclass. The issue will be raised on superclass declaration of the method (or won't be raised at all if superclass is not part of the<br>analysis).<br><br>@Override<br>public void myMethod() throws Exception {...}<br><br>Generic exceptions are also ignored in the signatures of methods that make calls to methods that throw generic exceptions.<br><br>public void myOtherMethod throws Exception {<br>  doTheThing(); // this method throws Exception<br>}<br><br>See<br><br>MITRE, CWE-397 - Declaration of Throws for Generic Exception<br>CERT, ERR07-J. - Do not throw RuntimeException, Exception, or Throwable |

| 文件名称 | 违规行 |
|---|---|
| CustomerExample.java | 88, 95, 102 |
| UserExample.java | 88, 95, 102 |

| 规则 | Collection.isEmpty() should be used to test for emptiness |
|---|---|

| 规则描述 | Using Collection.size() to test for emptiness works, but using Collection.isEmpty() makes the code more readable and can be more performant. The time complexity of any isEmpty() method implementation should be O(1) whereas some implementations of size() can be O(n) .<br>Noncompliant Code Example<br><br>if (myCollection.size() == 0) { // Noncompliant<br>  /* ... */<br>}<br><br>Compliant Solution<br><br>if (myCollection.isEmpty()) {<br>  /* ... */<br>} |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| UserController.java | 40 |
| CustomerExample.java | 49, 75 |
| UserExample.java | 49, 75 |

| 规则 | String literals should not be duplicated |
|---|---|

| 规则描述 | Duplicated string literals make the process of refactoring error-prone, since you must be sure to update all occurrences.<br>On the other hand, constants can be referenced from many places, but only need to be updated in a single place.<br>Noncompliant Code Example<br>With the default threshold of 3:<br><br>public void run() {<br>  prepare("action1");                              // Noncompliant - "action1" is duplicated 3 times<br>  execute("action1");<br>  release("action1");<br>}<br><br>@SuppressWarning("all")                            // Compliant - annotations are excluded<br>private void method1() { /* ... */ }<br>@SuppressWarning("all")<br>private void method2() { /* ... */ }<br><br>public String method3(String a) {<br>  System.out.println("'" + a + "'");               // Compliant - literal "'" has less than 5 characters and is excluded<br>  return "";                                       // Compliant - literal "" has less than 5 characters and is excluded<br>}<br><br>Compliant Solution<br><br>private static final String ACTION_1 = "action1";  // Compliant<br><br>public void run() {<br>  prepare(ACTION_1);                               // Compliant<br>  execute(ACTION_1);<br>  release(ACTION_1);<br>}<br><br>Exceptions<br>To prevent generating some false-positives, literals having less than 5 characters are excluded. |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| UserController.java | 42, 68 |
| CustomerExample.java | 178, 308 |
| UserExample.java | 248 |

| 规则 | Composed "@RequestMapping" variants should be preferred |
|---|---|

| 规则描述 | Spring framework 4.3 introduced variants of the @RequestMapping annotation to better represent the semantics of the annotated methods.<br>The use of @GetMapping , @PostMapping , @PutMapping , @PatchMapping and @DeleteMapping should be preferred to the use of the raw @RequestMapping(method = RequestMethod.XYZ) .<br>Noncompliant Code Example<br><br>@RequestMapping(path = "/greeting", method = RequestMethod.GET) // Noncompliant<br>public Greeting greeting(@RequestParam(value = "name", defaultValue = "World") String name) {<br>...<br>}<br><br>Compliant Solution<br><br>@GetMapping(path = "/greeting") // Compliant<br>public Greeting greeting(@RequestParam(value = "name", defaultValue = "World") String name) {<br>...<br>} |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| UserController.java | 32, 57, 85, 91 |

| 规则 | The diamond operator ("<>") should be used |
|---|---|
| 规则描述 | Java 7 introduced the diamond operator ( <> ) to reduce the verbosity of generics code. For instance, instead of having to declare<br>a List 's type in both its declaration and its constructor, you can now simplify the constructor declaration with <> ,<br>and the compiler will infer the type.<br>Note that this rule is automatically disabled when the project's sonar.java.source is lower than 7 .<br>Noncompliant Code Example<br><br>List<String> strings = new ArrayList<String>(); // Noncompliant<br>Map<String,List<Integer>> map = new HashMap<String,List<Integer>>(); // Noncompliant<br><br>Compliant Solution<br><br>List<String> strings = new ArrayList<>();<br>Map<String,List<Integer>> map = new HashMap<>(); |

| 文件名称 | 违规行 |
|---|---|
| CustomerExample.java | 14, 71 |
| UserExample.java | 14, 71 |

| 规则 | Local variables should not be declared and then immediately returned or thrown |
|---|---|
| 规则描述 | Declaring a variable only to immediately return or throw it is a bad practice.<br>Some developers argue that the practice improves code readability, because it enables them to explicitly name what is being returned. However, this<br>variable is an internal implementation detail that is not exposed to the callers of the method. The method name should be sufficient for callers to<br>know exactly what will be returned.<br>Noncompliant Code Example<br><br>public long computeDurationInMilliseconds() {<br>  long duration = (((hours * 60) + minutes) * 60 + seconds ) * 1000 ;<br>  return duration;<br>}<br><br>public void doSomething() {<br>  RuntimeException myException = new RuntimeException();<br>  throw myException;<br>}<br><br>Compliant Solution<br><br>public long computeDurationInMilliseconds() {<br>  return (((hours * 60) + minutes) * 60 + seconds ) * 1000 ;<br>}<br><br>public void doSomething() {<br>  throw new RuntimeException();<br>} |

| 文件名称 | 违规行 |
|---|---|
| CustomerExample.java | 56 |
| UserExample.java | 56 |
| UserServiceImpl.java | 23 |

| 规则 | Sections of code should not be commented out |
|---|---|
| 规则描述 | Programmers should not comment out code as it bloats programs and reduces readability.<br>Unused code should be deleted and can be retrieved from source control history if required.<br>See<br><br>MISRA C:2004, 2.4 - Sections of code should not be "commented out".<br>MISRA C++:2008, 2-7-2 - Sections of code shall not be "commented out" using C-style comments.<br>MISRA C++:2008, 2-7-3 - Sections of code should not be "commented out" using C++ comments.<br>MISRA C:2012, Dir. 4.4 - Sections of code should not be "commented out" |

| 文件名称 | 违规行 |
|---|---|
| CustomerController.java | 33 |
| UserController.java | 62 |

| 规则 | A "while" loop should be used instead of a "for" loop |
|---|---|
| 规则描述 | When only the condition expression is defined in a for loop, and the initialization and increment expressions are missing, a while loop should be used instead to increase readability. Note that this rule requires Node.js to be available during analysis. Noncompliant Code Example<br><br>for (;condition;) { /*...*/ }<br><br>Compliant Solution<br><br>while (condition) { /*...*/ } |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 2100, 2108 |

| 规则 | Standard outputs should not be used directly to log anything |
|---|---|
| 规则描述 | When logging a message there are several important requirements which must be fulfilled:<br><br>The user must be able to easily retrieve the logs<br>The format of all logged message must be uniform to allow the user to easily read the log<br>Logged data must actually be recorded<br>Sensitive data must only be logged securely<br><br>If a program directly writes to the standard outputs, there is absolutely no way to comply with those requirements. That's why defining and using a dedicated logger is highly recommended.<br>Noncompliant Code Example<br><br>System.out.println("My Message");  // Noncompliant<br><br>Compliant Solution<br><br>logger.log("My Message");<br><br>See<br><br>CERT, ERR02-J.  - Prevent exceptions while logging data |

| 文件名称 | 违规行 |
|---|---|
| CustomerController.java | 47 |
| UserController.java | 98 |

| 规则 | Dead stores should be removed |
|---|---|
| 规则描述 | A dead store happens when a local variable is assigned a value that is not read by any subsequent instruction. Calculating or retrieving a value<br>only to then overwrite it or throw it away, could indicate a serious error in the code. Even if it's not an error, it is at best a waste of resources.<br>Therefore all calculated values should be used.<br>Noncompliant Code Example<br><br>i = a + b; // Noncompliant; calculation result not used before value is overwritten<br>i = compute();<br><br>Compliant Solution<br><br>i = a + b;<br>i += compute();<br><br>Exceptions<br>This rule ignores initializations to -1, 0, 1, null , true , false and "" .<br>See<br><br>MITRE, CWE-563 - Assignment to Variable without Use ('Unused Variable')<br>CERT, MSC13-C. - Detect and remove unused values<br>CERT, MSC56-J. - Detect and remove superfluous code and values |

| 文件名称 | 违规行 |
|---|---|
| CustomerServiceImpl.java | 32, 38 |

| 规则 | Boolean expressions should not be gratuitous |
|---|---|

| | |
|---|---|
| 规则描述 | If a boolean expression doesn't change the evaluation of the condition, then it is entirely unnecessary, and can be removed. If it is gratuitous<br>because it does not match the programmer's intent, then it's a bug and the expression should be fixed.<br>Noncompliant Code Example<br><br>a = true;<br>if (a) { // Noncompliant<br>  doSomething();<br>}<br><br>if (b &amp;amp;&amp;amp; a) { // Noncompliant; "a" is always "true"<br>  doSomething();<br>}<br><br>if (c \|\| !a) { // Noncompliant; "!a" is always "false"<br>  doSomething();<br>}<br><br>Compliant Solution<br><br>a = true;<br>if (foo(a)) {<br>  doSomething();<br>}<br><br>if (b) {<br>  doSomething();<br>}<br><br>if (c) {<br>  doSomething();<br>}<br><br>See<br><br>MISRA C:2004, 13.7 - Boolean operations whose results are invariant shall not be permitted.<br>MISRA C:2012, 14.3 - Controlling expressions shall not be invariant<br>MITRE, CWE-571  - Expression is Always True<br>CERT, MSC12-C.  - Detect and remove code that has no effect or is never<br>executed |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 7236, 7318 |

| | |
|---|---|
| 规则 | Throwable.printStackTrace(...) should not be called |

| 规则描述 | Throwable.printStackTrace(...) prints a Throwable and its stack trace to some stream. By default that stream System.Err , which could inadvertently expose sensitive information.<br>Loggers should be used instead to print Throwable s, as they have many advantages:<br><br>Users are able to easily retrieve the logs.<br>The format of log messages is uniform and allow users to browse the logs easily.<br><br>This rule raises an issue when printStackTrace is used without arguments, i.e. when the stack trace is printed to the default stream.<br>Noncompliant Code Example<br><br>try {<br>  /* ... */<br>} catch(Exception e) {<br>  e.printStackTrace();        // Noncompliant<br>}<br><br>Compliant Solution<br><br>try {<br>  /* ... */<br>} catch(Exception e) {<br>  LOGGER.log("context", e);<br>}<br><br>See<br><br>MITRE, CWE-489 - Leftover Debug Code<br>OWASP Top 10 2017 Category A3 - Sensitive Data Exposure |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| JBCrypt.java | 16 |
| CustomerController.java | 51 |

| 规则 | Source files should not have any duplicated blocks |
|---|---|
| 规则描述 | An issue is created on a file as soon as there is at least one block of duplicated code on this file |

| 文件名称 | 违规行 |
|---|---|
| CustomerExample.java | N/A |
| UserExample.java | N/A |

| 规则 | Unused local variables should be removed |
|---|---|

| | |
|---|---|
| 规则描述 | If a local variable is declared but not used, it is dead code and should be removed. Doing so will improve maintainability because developers will not wonder what the variable is used for. Noncompliant Code Example `public int numberOfMinutes(int hours) { int seconds = 0; // seconds is never used return hours * 60; }` Compliant Solution `public int numberOfMinutes(int hours) { return hours * 60; }` |

| 文件名称 | 违规行 |
|---|---|
| CustomerServiceImpl.java | 32, 38 |

| | |
|---|---|
| 规则 | Nested blocks of code should not be left empty |
| 规则描述 | Most of the time a block of code is empty when a piece of code is really missing. So such empty block must be either filled or removed. Noncompliant Code Example `for (var i = 0; i < length; i++) {} // Empty on purpose or missing piece of code ?` Exceptions When a block contains a comment, this block is not considered to be empty. Moreover catch blocks are ignored. |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 709, 3109 |

| | |
|---|---|
| 规则 | Functions should not be defined inside loops |

| | |
|---|---|
| 规则描述 | Defining a function inside of a loop can yield unexpected results. Such a function keeps references to the variables which are defined in outer scopes. All function instances created inside the loop therefore see the same values for these variables, which is probably not expected.<br>Noncompliant Code Example<br><br>var funs = [];<br>for (var i = 0; i < 13; i++) {<br>  funs[i] = function() { // Non-Compliant<br>    return i;<br>  };<br>}<br>console.log(funs[0]()); // 13 instead of 0<br>console.log(funs[1]()); // 13 instead of 1<br>console.log(funs[2]()); // 13 instead of 2<br>console.log(funs[3]()); // 13 instead of 3<br>... |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 7005 |

| | |
|---|---|
| 规则 | Redundant casts should not be used |

| 规则描述 | Unnecessary casting expressions make the code harder to read and understand.<br>Noncompliant Code Example<br><br>`public void example() {`<br>`  for (Foo obj : (List<Foo>) getFoos()) {  // Noncompliant; cast unnecessary because List<Foo> is what's returned`<br>`    //...`<br>`  }`<br>`}`<br><br>`public List<Foo> getFoos() {`<br>`  return this.foos;`<br>`}`<br><br>Compliant Solution<br><br>`public void example() {`<br>`  for (Foo obj : getFoos()) {`<br>`    //...`<br>`  }`<br>`}`<br><br>`public List<Foo> getFoos() {`<br>`  return this.foos;`<br>`}`<br><br>Exceptions<br>Casting may be required to distinguish the method to call in the case of overloading:<br><br>`class A {}`<br>`class B extends A{}`<br>`class C {`<br>`  void fun(A a){}`<br>`  void fun(B b){}`<br><br>`  void foo() {`<br>`    B b = new B();`<br>`    fun(b);`<br>`    fun((A) b); //call the first method so cast is not redundant.`<br>`  }`<br><br>`}` |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| CustomerController.java | 69 |

| 规则 | Utility classes should not have public constructors |
|---|---|

| 规则描述 | 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.<br>Java adds an implicit public constructor to every class which does not define at least one explicitly. Hence, at least one non-public constructor<br>should be defined.<br>Noncompliant Code Example<br><br>class StringUtils { // Noncompliant<br><br>  public static String concatenate(String s1, String s2) {<br>    return s1 + s2;<br>  }<br><br>}<br><br>Compliant Solution<br><br>class StringUtils { // Compliant<br><br>  private StringUtils() {<br>    throw new IllegalStateException("Utility class");<br>  }<br><br>  public static String concatenate(String s1, String s2) {<br>    return s1 + s2;<br>  }<br><br>}<br><br>Exceptions<br>When class contains public static void main(String[] args) method it is not considered as utility class and will be ignored by this<br>rule. |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| JBCrypt.java | 5 |

| 规则 | Dead stores should be removed |
|---|---|

| 规则描述 | A dead store happens when a local variable is assigned a value that is not read by any subsequent instruction. Calculating or retrieving a value<br>only to then overwrite it or throw it away, could indicate a serious error in the code. Even if it's not an error, it is at best a waste of resources.<br>Therefore all calculated values should be used.<br>Noncompliant Code Example<br><br>i = a + b; // Noncompliant; calculation result not used before value is overwritten<br>i = compute();<br><br>Compliant Solution<br><br>i = a + b;<br>i += compute();<br><br>Exceptions<br>This rule ignores initializations to -1, 0, 1, null , undefined , true , false , "" ,<br>[] and {} .<br>This rule also ignores variables declared with object destructuring using rest syntax (used to exclude some properties from object):<br><br>let {a, b, ...rest} = obj; // 'a' and 'b' are ok<br>doSomething(rest);<br><br>let [x1, x2, x3] = arr; // but 'x1' is noncompliant, as omitting syntax can be used: "let [, x2, x3] = arr;"<br>doSomething(x2, x3);<br><br>See<br><br>MITRE, CWE-563 - Assignment to Variable without Use ('Unused Variable')<br>CERT, MSC13-C. - Detect and remove unused values<br>CERT, MSC56-J. - Detect and remove superfluous code and values |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 337 |

| 规则 | Strict equality operators should not be used with dissimilar types |
|---|---|

| | |
|---|---|
| 规则描述 | Comparing dissimilar types using the strict equality operators `===` and `!==` will always return the same value, respectively `false` and `true`, because no type conversion is done before the comparison. Thus, such comparisons can only be bugs.<br>Noncompliant Code Example<br><br>`var a = 8;`<br>`var b = "8";`<br><br>`if (a === b) {  // Noncompliant; always false`<br>`  // ...`<br>`}`<br><br>Compliant Solution<br><br>`var a = 8;`<br>`var b = "8";`<br><br>`if (a == b) {`<br>`  // ...`<br>`}`<br><br>or<br><br>`var a = 8;`<br>`var b = "8";`<br><br>`if (a === Number(b)) {`<br>`  // ...`<br>`}` |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 7318 |

| | |
|---|---|
| 规则 | Extra semicolons should be removed |

| | |
|---|---|
| 规则描述 | Extra semicolons ( ; ) are usually introduced by mistake, for example because:<br><br>It was meant to be replaced by an actual statement, but this was forgotten.<br>There was a typo which lead the semicolon to be doubled, i.e. ;; .<br>There was a misunderstanding about where semicolons are required or useful.<br><br>Noncompliant Code Example<br><br>var x = 1;; // Noncompliant<br><br>function foo() {<br>}; // Noncompliant<br><br>Compliant Solution<br><br>var x = 1;<br><br>function foo() {<br>}<br><br>See<br><br>MISRA C:2004, 14.3 - Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment provided that<br>the first character following the null statement is a white-space character.<br>MISRA C++:2008, 6-2-3 - Before preprocessing, a null statement shall only occur on a line by itself; it may be followed by a comment, provided<br>that the first character following the null statement is a white-space character.<br>CERT, MSC12-C. - Detect and remove code that has no effect or is never<br>executed<br>CERT, MSC51-J. - Do not place a semicolon immediately following an if, for,<br>or while condition<br>CERT, EXP15-C. - Do not place a semicolon on the same line as an if, for,<br>or while statement |

| 文件名称 | 违规行 |
|---|---|
| jquery-3.3.1.js | 2807 |

| | |
|---|---|
| 规则 | Non-serializable objects should not be stored in "HttpSession" objects |

| 规则描述 | If you have no intention of writting an HttpSession object to file, then storing non- serializable objects in it may not seem like a big deal. But whether or not you explicitly serialize the session, it may be written to disk anyway, as the server manages its memory use in a process called "passivation". Further, some servers automatically write their active sessions out to file at shutdown &amp; deserialize any such sessions at startup.<br>The point is, that even though HttpSession does not extend Serializable , you must nonetheless assume that it will be serialized, and understand that if you've stored non-serializable objects in the session, errors will result.<br>Noncompliant Code Example<br><br>public class Address {<br>  //...<br>}<br><br>//...<br>HttpSession session = request.getSession();<br>session.setAttribute("address", new Address());  // Noncompliant; Address isn't serializable<br><br>See<br><br>MITRE, CWE-579 - J2EE Bad Practices: Non-serializable Object Stored in Session |
|---|---|

| 文件名称 | 违规行 |
|---|---|
| UserController.java | 66 |

## 1.4. 质量配置

| 质量配置 | java:Sonar way   Bug:101   漏洞:28   坏味道:195 |
|---|---|

| 规则 | 类型 | 违规级别 |
|---|---|---|
| Methods should not call same-class methods with incompatible "@Transactional" values | Bug | 阻断 |
| Methods "wait(...)", "notify()" and "notifyAll()" should not be called on Thread instances | Bug | 阻断 |
| "PreparedStatement" and "ResultSet" methods should be called with valid indices | Bug | 阻断 |
| "wait(...)" should be used instead of "Thread.sleep(...)" when a lock is held | Bug | 阻断 |
| Printf-style format strings should not lead to unexpected behavior at runtime | Bug | 阻断 |
| "@SpringBootApplication" and "@ComponentScan" should not be used in the default package | Bug | 阻断 |
| "@Controller" classes that use "@SessionAttributes" must call "setComplete" on their "SessionStatus" objects | Bug | 阻断 |

| | | |
|---|---|---|
| Loops should not be infinite | Bug | 阻断 |
| "wait" should not be called when multiple locks are held | Bug | 阻断 |
| Double-checked locking should not be used | Bug | 阻断 |
| Resources should be closed | Bug | 阻断 |
| Locks should be released | Bug | 严重 |
| "Random" objects should be reused | Bug | 严重 |
| Dependencies should not have "system" scope | Bug | 严重 |
| The signature of "finalize()" should match that of "Object.finalize()" | Bug | 严重 |
| "runFinalizersOnExit" should not be called | Bug | 严重 |
| "ScheduledThreadPoolExecutor" should not have 0 core threads | Bug | 严重 |
| "super.finalize()" should be called at the end of "Object.finalize()" implementations | Bug | 严重 |
| Zero should not be a possible denominator | Bug | 严重 |
| Getters and setters should access the expected fields | Bug | 严重 |
| "toString()" and "clone()" methods should not return null | Bug | 主要 |
| Servlets should not have mutable instance fields | Bug | 主要 |
| Value-based classes should not be used for locking | Bug | 主要 |
| Conditionally executed blocks should be reachable | Bug | 主要 |
| "DefaultMessageListenerContainer" instances should not drop messages during restarts | Bug | 主要 |
| Reflection should not be used to check non-runtime annotations | Bug | 主要 |
| "SingleConnectionFactory" instances should be set to "reconnectOnException" | Bug | 主要 |
| "hashCode" and "toString" should not be called on array instances | Bug | 主要 |
| Collections should not be passed as arguments to their own methods | Bug | 主要 |
| "BigDecimal(double)" should not be used | Bug | 主要 |
| Jump statements should not occur in "finally" blocks | Bug | 主要 |
| Non-public methods should not be "@Transactional" | Bug | 主要 |
| Invalid "Date" values should not be used | Bug | 主要 |
| Non-serializable classes should not be written | Bug | 主要 |
| Optional value should only be accessed after calling isPresent() | Bug | 主要 |
| Blocks should be synchronized on "private final" fields | Bug | 主要 |
| "notifyAll" should be used | Bug | 主要 |
| ".equals()" should not be used to test the values of "Atomic" classes | Bug | 主要 |

| | | |
|---|---|---|
| Return values from functions without side effects should not be ignored | Bug | 主要 |
| Non-serializable objects should not be stored in "HttpSession" objects | Bug | 主要 |
| "InterruptedException" should not be ignored | Bug | 主要 |
| Silly equality checks should not be made | Bug | 主要 |
| Dissimilar primitive wrappers should not be used with the ternary operator without explicit casting | Bug | 主要 |
| "wait", "notify" and "notifyAll" should only be called when a lock is obviously held on an object | Bug | 主要 |
| "Double.longBitsToDouble" should not be used for "int" | Bug | 主要 |
| Values should not be uselessly incremented | Bug | 主要 |
| Null pointers should not be dereferenced | Bug | 主要 |
| Expressions used in "assert" should not produce side effects | Bug | 主要 |
| Classes extending java.lang.Thread should override the "run" method | Bug | 主要 |
| Loop conditions should be true at least once | Bug | 主要 |
| A "for" loop update clause should move the counter in the right direction | Bug | 主要 |
| The Object.finalize() method should not be called | Bug | 主要 |
| Intermediate Stream methods should not be left unused | Bug | 主要 |
| Consumed Stream pipelines should not be reused | Bug | 主要 |
| Variables should not be self-assigned | Bug | 主要 |
| Inappropriate regular expressions should not be used | Bug | 主要 |
| "=+" should not be used instead of "+=" | Bug | 主要 |
| Loops with at most one iteration should be refactored | Bug | 主要 |
| Classes should not be compared by name | Bug | 主要 |
| Identical expressions should not be used on both sides of a binary operator | Bug | 主要 |
| Thread.run() should not be called directly | Bug | 主要 |
| "null" should not be used with "Optional" | Bug | 主要 |
| "read" and "readLine" return values should be used | Bug | 主要 |
| Methods should not be named "tostring", "hashcode" or "equal" | Bug | 主要 |
| Non-thread-safe fields should not be static | Bug | 主要 |
| Getters and setters should be synchronized in pairs | Bug | 主要 |
| "StringBuilder" and "StringBuffer" should not be instantiated with a character | Bug | 主要 |
| Unary prefix operators should not be repeated | Bug | 主要 |
| Week Year ("YYYY") should not be used for date formatting | Bug | 主要 |
| "equals" method overrides should accept "Object" parameters | Bug | 主要 |

| | | |
|---|---|---|
| Exception should not be created without being thrown | Bug | 主要 |
| Collection sizes and array length comparisons should make sense | Bug | 主要 |
| Synchronization should not be based on Strings or boxed primitives | Bug | 主要 |
| Related "if/else if" statements should not have the same condition | Bug | 主要 |
| All branches in a conditional structure should not have exactly the same implementation | Bug | 主要 |
| "Iterator.hasNext()" should not call "Iterator.next()" | Bug | 主要 |
| Raw byte values should not be used in bitwise operations in combination with shifts | Bug | 主要 |
| Custom serialization method signatures should meet requirements | Bug | 主要 |
| "Externalizable" classes should have no-arguments constructors | Bug | 主要 |
| "iterator" should not return "this" | Bug | 主要 |
| Child class methods named for parent class methods should be overrides | Bug | 主要 |
| Inappropriate "Collection" calls should not be made | Bug | 主要 |
| "compareTo" should not be overloaded | Bug | 主要 |
| Map values should not be replaced unconditionally | Bug | 主要 |
| "getClass" should not be used for synchronization | Bug | 主要 |
| "compareTo" results should not be checked for specific values | Bug | 次要 |
| Double Brace Initialization should not be used | Bug | 次要 |
| Boxing and unboxing should not be immediately reversed | Bug | 次要 |
| "Iterator.next()" methods should throw "NoSuchElementException" | Bug | 次要 |
| "@NonNull" values should not be set to null | Bug | 次要 |
| Neither "Math.abs" nor negation should be used on numbers that could be "MIN_VALUE" | Bug | 次要 |
| The value returned from a stream read should be checked | Bug | 次要 |
| Method parameters, caught exceptions and foreach variables' initial values should not be ignored | Bug | 次要 |
| "equals(Object obj)" and "hashCode()" should be overridden in pairs | Bug | 次要 |
| "Serializable" inner classes of non-serializable classes should be "static" | Bug | 次要 |
| Math operands should be cast before assignment | Bug | 次要 |
| Ints and longs should not be shifted by zero or more than their number of bits-1 | Bug | 次要 |
| "compareTo" should not return "Integer.MIN_VALUE" | Bug | 次要 |

| | | |
|---|---|---|
| The non-serializable super class of a "Serializable" class should have a no-argument constructor | Bug | 次要 |
| "toArray" should be passed an array of the proper type | Bug | 次要 |
| "equals(Object obj)" should test argument type | Bug | 次要 |
| Databases should be password-protected | 漏洞 | 阻断 |
| Neither DES (Data Encryption Standard) nor DESede (3DES) should be used | 漏洞 | 阻断 |
| Cryptographic keys should not be too short | 漏洞 | 阻断 |
| LDAP deserialization should be disabled | 漏洞 | 阻断 |
| "HostnameVerifier.verify" should not always return true | 漏洞 | 阻断 |
| Credentials should not be hard-coded | 漏洞 | 阻断 |
| Default EJB interceptors should be declared in "ejb-jar.xml" | 漏洞 | 阻断 |
| Persistent entities should not be used as arguments of "@RequestMapping" methods | 漏洞 | 严重 |
| Defined filters should be used | 漏洞 | 严重 |
| Cryptographic RSA algorithms should always incorporate OAEP (Optimal Asymmetric Encryption Padding) | 漏洞 | 严重 |
| XML transformers should be secured | 漏洞 | 严重 |
| "HttpServletRequest.getRequestedSessionId()" should not be used | 漏洞 | 严重 |
| AES encryption algorithm should be used with secured mode | 漏洞 | 严重 |
| LDAP connections should be authenticated | 漏洞 | 严重 |
| "File.createTempFile" should not be used to create a directory | 漏洞 | 严重 |
| Web applications should not have a "main" method | 漏洞 | 严重 |
| SMTP SSL connection should check server identity | 漏洞 | 严重 |
| SQL binding mechanisms should be used | 漏洞 | 严重 |
| "SecureRandom" seeds should not be predictable | 漏洞 | 严重 |
| TrustManagers should not blindly accept any certificates | 漏洞 | 主要 |
| Weak SSL protocols should not be used | 漏洞 | 主要 |
| Throwable.printStackTrace(...) should not be called | 漏洞 | 次要 |
| Mutable fields should not be "public static" | 漏洞 | 次要 |
| "public static" fields should be constant | 漏洞 | 次要 |
| Exceptions should not be thrown from servlet methods | 漏洞 | 次要 |
| Class variable fields should not have public accessibility | 漏洞 | 次要 |
| "enum" fields should not be publicly mutable | 漏洞 | 次要 |
| Return values should not be ignored when they contain the operation status code | 漏洞 | 次要 |

| | | |
|---|---|---|
| Child class fields should not shadow parent class fields | 坏味道 | 阻断 |
| JUnit framework methods should be declared properly | 坏味道 | 阻断 |
| Assertions should be complete | 坏味道 | 阻断 |
| "clone" should not be overridden | 坏味道 | 阻断 |
| "switch" statements should not contain non-case labels | 坏味道 | 阻断 |
| Methods returns should not be invariant | 坏味道 | 阻断 |
| Silly bit operations should not be performed | 坏味道 | 阻断 |
| Switch cases should end with an unconditional "break" statement | 坏味道 | 阻断 |
| Methods and field names should not be the same or differ only by capitalization | 坏味道 | 阻断 |
| JUnit test cases should call super methods | 坏味道 | 阻断 |
| TestCases should contain tests | 坏味道 | 阻断 |
| Future keywords should not be used as names | 坏味道 | 阻断 |
| Short-circuit logic should be used in boolean contexts | 坏味道 | 阻断 |
| Constant names should comply with a naming convention | 坏味道 | 严重 |
| "default" clauses should be last | 坏味道 | 严重 |
| IllegalMonitorStateException should not be caught | 坏味道 | 严重 |
| Cognitive Complexity of methods should not be too high | 坏味道 | 严重 |
| Package declaration should match source file directory | 坏味道 | 严重 |
| Null should not be returned from a "Boolean" method | 坏味道 | 严重 |
| Instance methods should not write to "static" fields | 坏味道 | 严重 |
| String offset-based methods should be preferred for finding substrings from offsets | 坏味道 | 严重 |
| "indexOf" checks should not be for positive numbers | 坏味道 | 严重 |
| Factory method injection should be used in "@Configuration" classes | 坏味道 | 严重 |
| "Object.finalize()" should remain protected (versus public) when overriding | 坏味道 | 严重 |
| "Cloneables" should implement "clone" | 坏味道 | 严重 |
| Methods should not be empty | 坏味道 | 严重 |
| "Object.wait(...)" and "Condition.await(...)" should be called inside a "while" loop | 坏味道 | 严重 |
| "equals" method parameters should not be marked "@Nonnull" | 坏味道 | 严重 |
| Classes should not access their own subclasses during initialization | 坏味道 | 严重 |
| Exceptions should not be thrown in finally blocks | 坏味道 | 严重 |
| Method overrides should not change contracts | 坏味道 | 严重 |

| | | |
|---|---|---|
| "for" loop increment clauses should modify the loops' counters | 坏味道 | 严重 |
| Constants should not be defined in interfaces | 坏味道 | 严重 |
| Generic wildcard types should not be used in return parameters | 坏味道 | 严重 |
| Execution of the Garbage Collector should be triggered only by the JVM | 坏味道 | 严重 |
| The Object.finalize() method should not be overriden | 坏味道 | 严重 |
| Conditionals should start on new lines | 坏味道 | 严重 |
| A conditionally executed single line should be denoted by indentation | 坏味道 | 严重 |
| Fields in a "Serializable" class should either be transient or serializable | 坏味道 | 严重 |
| "switch" statements should have "default" clauses | 坏味道 | 严重 |
| JUnit assertions should not be used in "run" methods | 坏味道 | 严重 |
| "readResolve" methods should be inheritable | 坏味道 | 严重 |
| String literals should not be duplicated | 坏味道 | 严重 |
| Class names should not shadow interfaces or superclasses | 坏味道 | 严重 |
| Try-with-resources should be used | 坏味道 | 严重 |
| Boolean expressions should not be gratuitous | 坏味道 | 主要 |
| Track uses of "FIXME" tags | 坏味道 | 主要 |
| Parameters should be passed in the correct order | 坏味道 | 主要 |
| "ResultSet.isLast()" should not be used | 坏味道 | 主要 |
| Nested blocks of code should not be left empty | 坏味道 | 主要 |
| "URL.hashCode" and "URL.equals" should be avoided | 坏味道 | 主要 |
| Try-catch blocks should not be nested | 坏味道 | 主要 |
| Methods should not have too many parameters | 坏味道 | 主要 |
| Synchronized classes Vector, Hashtable, Stack and StringBuffer should not be used | 坏味道 | 主要 |
| Generic exceptions should never be thrown | 坏味道 | 主要 |
| "Lock" objects should not be "synchronized" | 坏味道 | 主要 |
| Classes with only "static" methods should not be instantiated | 坏味道 | 主要 |
| Multiline blocks should be enclosed in curly braces | 坏味道 | 主要 |
| "static" members should be accessed statically | 坏味道 | 主要 |
| Utility classes should not have public constructors | 坏味道 | 主要 |
| Assertion arguments should be passed in the correct order | 坏味道 | 主要 |
| Unused type parameters should be removed | 坏味道 | 主要 |
| "switch" statements should not have too many "case" clauses | 坏味道 | 主要 |
| Unused "private" methods should be removed | 坏味道 | 主要 |
| Redundant pairs of parentheses should be removed | 坏味道 | 主要 |

| | | |
|---|---|---|
| Ternary operators should not be nested | 坏味道 | 主要 |
| Inner class calls to super class methods should be unambiguous | 坏味道 | 主要 |
| Nullness of parameters should be guaranteed | 坏味道 | 主要 |
| Only static class initializers should be used | 坏味道 | 主要 |
| Unused method parameters should be removed | 坏味道 | 主要 |
| Unused "private" fields should be removed | 坏味道 | 主要 |
| Collapsible "if" statements should be merged | 坏味道 | 主要 |
| Unused labels should be removed | 坏味道 | 主要 |
| Throwable and Error should not be caught | 坏味道 | 主要 |
| Printf-style format strings should be used correctly | 坏味道 | 主要 |
| "Integer.toHexString" should not be used to build hexadecimal strings | 坏味道 | 主要 |
| Labels should not be used | 坏味道 | 主要 |
| Constructors should not be used to instantiate "String", "BigInteger", "BigDecimal" and primitive-wrapper classes | 坏味道 | 主要 |
| Enumeration should not be implemented | 坏味道 | 主要 |
| Empty arrays and collections should be returned instead of null | 坏味道 | 主要 |
| Objects should not be created only to "getClass" | 坏味道 | 主要 |
| Primitives should not be boxed just for "String" conversion | 坏味道 | 主要 |
| "@Override" should be used on overriding and implementing methods | 坏味道 | 主要 |
| "entrySet()" should be iterated when both the key and value are needed | 坏味道 | 主要 |
| Assignments should not be made from within sub-expressions | 坏味道 | 主要 |
| "Preconditions" and logging arguments should not require evaluation | 坏味道 | 主要 |
| Java 8's "Files.exists" should not be used | 坏味道 | 主要 |
| Two branches in a conditional structure should not have exactly the same implementation | 坏味道 | 主要 |
| Sections of code should not be commented out | 坏味道 | 主要 |
| "Map.get" and value test should be replaced with single method call | 坏味道 | 主要 |
| "Arrays.stream" should be used for primitive arrays | 坏味道 | 主要 |
| Non-constructor methods should not have the same name as the enclosing class | 坏味道 | 主要 |
| "Threads" should not be used where "Runnables" are expected | 坏味道 | 主要 |
| "readObject" should not be "synchronized" | 坏味道 | 主要 |
| "for" loop stop conditions should be invariant | 坏味道 | 主要 |
| Inheritance tree of classes should not be too deep | 坏味道 | 主要 |
| Unused "private" classes should be removed | 坏味道 | 主要 |

| | | |
|---|---|---|
| A field should not duplicate the name of its containing class | 坏味道 | 主要 |
| Dead stores should be removed | 坏味道 | 主要 |
| "DateUtils.truncate" from Apache Commons Lang library should not be used | 坏味道 | 主要 |
| Local variables should not shadow class fields | 坏味道 | 主要 |
| "Thread.sleep" should not be used in tests | 坏味道 | 主要 |
| Anonymous inner classes containing only one method should become lambdas | 坏味道 | 主要 |
| Tests should not be ignored | 坏味道 | 主要 |
| Deprecated elements should have both the annotation and the Javadoc tag | 坏味道 | 主要 |
| "Object.wait(...)" should never be called on objects that implement "java.util.concurrent.locks.Condition" | 坏味道 | 主要 |
| Silly math should not be performed | 坏味道 | 主要 |
| Standard outputs should not be used directly to log anything | 坏味道 | 主要 |
| "writeObject" should not be the only "synchronized" code in a class | 坏味道 | 主要 |
| Classes named like "Exception" should extend "Exception" or a subclass | 坏味道 | 主要 |
| Static fields should not be updated in constructors | 坏味道 | 主要 |
| Exception types should not be tested using "instanceof" in catch blocks | 坏味道 | 主要 |
| Classes from "sun.*" packages should not be used | 坏味道 | 主要 |
| String function use should be optimized for single characters | 坏味道 | 主要 |
| Assignments should not be redundant | 坏味道 | 主要 |
| "java.nio.Files#delete" should be preferred | 坏味道 | 主要 |
| Methods should not have identical implementations | 坏味道 | 主要 |
| Asserts should not be used to check the parameters of a public method | 坏味道 | 主要 |
| Source files should not have any duplicated blocks | 坏味道 | 主要 |
| Field names should comply with a naming convention | 坏味道 | 次要 |
| Interface names should comply with a naming convention | 坏味道 | 次要 |
| Type parameter names should comply with a naming convention | 坏味道 | 次要 |
| Local variable and method parameter names should comply with a naming convention | 坏味道 | 次要 |
| Package names should comply with a naming convention | 坏味道 | 次要 |
| A "while" loop should be used instead of a "for" loop | 坏味道 | 次要 |
| "Collections.EMPTY_LIST", "EMPTY_MAP", and "EMPTY_SET" should not be used | 坏味道 | 次要 |

| | | |
|---|---|---|
| Useless imports should be removed | 坏味道 | 次要 |
| Return of boolean expressions should not be wrapped into an "if-then-else" statement | 坏味道 | 次要 |
| Boolean literals should not be redundant | 坏味道 | 次要 |
| Local variables should not be declared and then immediately returned or thrown | 坏味道 | 次要 |
| Deprecated "${pom}" properties should not be used | 坏味道 | 次要 |
| Unused local variables should be removed | 坏味道 | 次要 |
| Catches should be combined | 坏味道 | 次要 |
| Null checks should not be used with "instanceof" | 坏味道 | 次要 |
| Methods of "Random" that return floating point values should not be used in random integer generation | 坏味道 | 次要 |
| Public constants and fields initialized at declaration should be "static final" rather than merely "final" | 坏味道 | 次要 |
| Overriding methods should do more than simply call the same method in the super class | 坏味道 | 次要 |
| Static non-final field names should comply with a naming convention | 坏味道 | 次要 |
| Classes that override "clone" should be "Cloneable" and call "super.clone()" | 坏味道 | 次要 |
| Case insensitive string comparisons should be made without intermediate upper or lower casing | 坏味道 | 次要 |
| Primitive wrappers should not be instantiated only for "toString" or "compareTo" calls | 坏味道 | 次要 |
| Collection.isEmpty() should be used to test for emptiness | 坏味道 | 次要 |
| String.valueOf() should not be appended to a String | 坏味道 | 次要 |
| Method names should comply with a naming convention | 坏味道 | 次要 |
| Class names should comply with a naming convention | 坏味道 | 次要 |
| Exception classes should be immutable | 坏味道 | 次要 |
| Parsing should be used to convert "Strings" to primitives | 坏味道 | 次要 |
| Multiple variables should not be declared on the same line | 坏味道 | 次要 |
| "switch" statements should have at least 3 "case" clauses | 坏味道 | 次要 |
| Strings should not be concatenated using '+' in a loop | 坏味道 | 次要 |
| Maps with keys that are enum values should be replaced with EnumMap | 坏味道 | 次要 |
| "catch" clauses should do more than rethrow | 坏味道 | 次要 |
| Nested "enum"s should not be declared static | 坏味道 | 次要 |
| "equals(Object obj)" should be overridden along with the "compareTo(T obj)" method | 坏味道 | 次要 |

| | | |
|---|---|---|
| Private fields only used as local variables in methods should become local variables | 坏味道 | 次要 |
| Arrays should not be created for varargs parameters | 坏味道 | 次要 |
| Methods should not return constants | 坏味道 | 次要 |
| The default unnamed package should not be used | 坏味道 | 次要 |
| Declarations should use Java collection interfaces such as "List" rather than specific implementation classes such as "LinkedList" | 坏味道 | 次要 |
| Jump statements should not be redundant | 坏味道 | 次要 |
| Boolean checks should not be inverted | 坏味道 | 次要 |
| "close()" calls should not be redundant | 坏味道 | 次要 |
| "indexOf" checks should use a start position | 坏味道 | 次要 |
| Redundant casts should not be used | 坏味道 | 次要 |
| "ThreadLocal.withInitial" should be preferred | 坏味道 | 次要 |
| "@Deprecated" code should not be used | 坏味道 | 次要 |
| Abstract classes without fields should be converted to interfaces | 坏味道 | 次要 |
| Lambdas should be replaced with method references | 坏味道 | 次要 |
| "toString()" should never be called on a String object | 坏味道 | 次要 |
| Parentheses should be removed from a single lambda input parameter when its type is inferred | 坏味道 | 次要 |
| Annotation repetitions should not be wrapped | 坏味道 | 次要 |
| JUnit rules should be used | 坏味道 | 次要 |
| Lamdbas containing only one statement should not nest this statement in a block | 坏味道 | 次要 |
| Loops should not contain more than a single "break" or "continue" statement | 坏味道 | 次要 |
| Abstract methods should not be redundant | 坏味道 | 次要 |
| "private" methods called only by inner classes should be moved to those classes | 坏味道 | 次要 |
| Fields in non-serializable classes should not be "transient" | 坏味道 | 次要 |
| Composed "@RequestMapping" variants should be preferred | 坏味道 | 次要 |
| Empty statements should be removed | 坏味道 | 次要 |
| "write(byte[],int,int)" should be overridden | 坏味道 | 次要 |
| Nested code blocks should not be used | 坏味道 | 次要 |
| Array designators "[]" should be on the type, not the variable | 坏味道 | 次要 |
| URIs should not be hardcoded | 坏味道 | 次要 |
| "finalize" should not set fields to "null" | 坏味道 | 次要 |
| Array designators "[]" should be located after the type in method signatures | 坏味道 | 次要 |
| Subclasses that add fields should override "equals" | 坏味道 | 次要 |
| "throws" declarations should not be superfluous | 坏味道 | 次要 |

| | | |
|---|---|---|
| The diamond operator ("<>") should be used | 坏味道 | 次要 |
| Modifiers should be declared in the correct order | 坏味道 | 次要 |
| "Stream" call chains should be simplified when possible | 坏味道 | 次要 |
| Functional Interfaces should be as specialised as possible | 坏味道 | 次要 |
| Packages containing only "package-info.java" should be removed | 坏味道 | 次要 |
| Classes should not be empty | 坏味道 | 次要 |
| Track uses of "TODO" tags | 坏味道 | 提示 |
| Deprecated code should be removed | 坏味道 | 提示 |

| 质量配置 | js:Sonar way Bug:41 漏洞:5 坏味道:43 | |
|---|---|---|
| 规则 | 类型 | 违规级别 |
| Callbacks of array methods should have return statements | Bug | 阻断 |
| Loops should not be infinite | Bug | 阻断 |
| "yield" expressions should not be used outside generators | Bug | 阻断 |
| "in" should not be used with primitive types | Bug | 严重 |
| Function calls should not pass extra arguments | Bug | 严重 |
| "Symbol" should not be used as a constructor | Bug | 严重 |
| Results of "in" and "instanceof" should be negated rather than operands | Bug | 严重 |
| "super()" should be invoked appropriately | Bug | 严重 |
| Destructuring patterns should not be empty | Bug | 主要 |
| Conditionally executed blocks should be reachable | Bug | 主要 |
| Jump statements should not occur in "finally" blocks | Bug | 主要 |
| Property names should not be duplicated within a class or object literal | Bug | 主要 |
| Return values from functions without side effects should not be ignored | Bug | 主要 |
| "NaN" should not be used in comparisons | Bug | 主要 |
| Generators should "yield" something | Bug | 主要 |
| Function argument names should be unique | Bug | 主要 |
| Related "if/else if" statements and "cases" in a "switch" should not have the same condition | Bug | 主要 |
| All branches in a conditional structure should not have exactly the same implementation | Bug | 主要 |
| The output of functions that don't return anything should not be used | Bug | 主要 |
| Values should not be uselessly incremented | Bug | 主要 |
| Jump statements should not be followed by dead code | Bug | 主要 |

| | | |
|---|---|---|
| Special identifiers should not be bound or assigned | Bug | 主要 |
| Properties of variables with "null" or "undefined" values should not be accessed | Bug | 主要 |
| A "for" loop update clause should move the counter in the right direction | Bug | 主要 |
| Variables should not be self-assigned | Bug | 主要 |
| Non-empty statements should change control flow or have at least one side-effect | Bug | 主要 |
| Calls should not be made to non-callable values | Bug | 主要 |
| Non-existent operators '=+', '=-' and '=!' should not be used | Bug | 主要 |
| "new" operators should be used with functions | Bug | 主要 |
| Identical expressions should not be used on both sides of a binary operator | Bug | 主要 |
| Array-mutating methods should not be used misleadingly | Bug | 主要 |
| Strict equality operators should not be used with dissimilar types | Bug | 主要 |
| Setters should not return values | Bug | 主要 |
| Comma and logical OR operators should not be used in switch cases | Bug | 主要 |
| Collection elements should not be replaced unconditionally | Bug | 主要 |
| Bitwise operators should not be used in boolean contexts | Bug | 主要 |
| Attempts should not be made to update "const" variables | Bug | 主要 |
| Errors should not be created without being thrown | Bug | 主要 |
| Collection sizes and array length comparisons should make sense | Bug | 主要 |
| "delete" should be used only with object properties | Bug | 次要 |
| "with" statements should not be used | Bug | 次要 |
| Cross-document messaging domains should be carefully restricted | 漏洞 | 严重 |
| Code should not be dynamically injected and executed | 漏洞 | 严重 |
| Function constructors should not be used | 漏洞 | 严重 |
| Debugger statements should not be used | 漏洞 | 次要 |
| "alert(...)" should not be used | 漏洞 | 次要 |
| Octal values should not be used | 坏味道 | 阻断 |
| Variables should be declared explicitly | 坏味道 | 阻断 |
| "future reserved words" should not be used as identifiers | 坏味道 | 阻断 |
| "switch" statements should not contain non-case labels | 坏味道 | 阻断 |
| Function returns should not be invariant | 坏味道 | 阻断 |

| | | |
|---|---|---|
| Switch cases should end with an unconditional "break" statement | 坏味道 | 阻断 |
| Conditionals should start on new lines | 坏味道 | 严重 |
| A conditionally executed single line should be denoted by indentation | 坏味道 | 严重 |
| Equality operators should not be used in "for" loop termination conditions | 坏味道 | 严重 |
| Boolean expressions should not be gratuitous | 坏味道 | 主要 |
| Redundant pairs of parentheses should be removed | 坏味道 | 主要 |
| Functions should not be called both with and without "new" | 坏味道 | 主要 |
| Comma operator should not be used | 坏味道 | 主要 |
| Multiline blocks should be enclosed in curly braces | 坏味道 | 主要 |
| Labels should not be used | 坏味道 | 主要 |
| "switch" statements should not have too many "case" clauses | 坏味道 | 主要 |
| "indexOf" checks should not be for positive numbers | 坏味道 | 主要 |
| Arguments to built-in functions should match documented types | 坏味道 | 主要 |
| Nested blocks of code should not be left empty | 坏味道 | 主要 |
| Dead stores should be removed | 坏味道 | 主要 |
| Array indexes should be numeric | 坏味道 | 主要 |
| Variables and functions should not be redeclared | 坏味道 | 主要 |
| "delete" should not be used on arrays | 坏味道 | 主要 |
| Function parameters with default values should be last | 坏味道 | 主要 |
| Jump statements should not be used unconditionally | 坏味道 | 主要 |
| Two branches in a conditional structure should not have exactly the same implementation | 坏味道 | 主要 |
| Assignments should not be redundant | 坏味道 | 主要 |
| Functions should not be defined inside loops | 坏味道 | 主要 |
| Collection and array contents should be used | 坏味道 | 主要 |
| Default export names and file names should match | 坏味道 | 次要 |
| Boolean checks should not be inverted | 坏味道 | 次要 |
| A "while" loop should be used instead of a "for" loop | 坏味道 | 次要 |
| Function call arguments should not start on new lines | 坏味道 | 次要 |
| Extra semicolons should be removed | 坏味道 | 次要 |
| Return of boolean expressions should not be wrapped into an "if-then-else" statement | 坏味道 | 次要 |
| Unnecessary imports should be removed | 坏味道 | 次要 |
| Wrapper objects should not be used for primitive types | 坏味道 | 次要 |

| | | |
|---|---|---|
| Unary operators "+" and "-" should not be used with objects | 坏味道 | 次要 |
| Multiline string literals should not be used | 坏味道 | 次要 |
| "switch" statements should have at least 3 "case" clauses | 坏味道 | 次要 |
| The global "this" object should not be used | 坏味道 | 次要 |
| "catch" clauses should do more than rethrow | 坏味道 | 次要 |
| Unused local variables and functions should be removed | 坏味道 | 次要 |

| 质量配置 | xml:Sonar way　Bug:1 | |
|---|---|---|
| 规则 | 类型 | 违规级别 |
| XML files containing a prolog header should start with "<?xml" characters | Bug | 严重 |