|  |  |  |  |
| --- | --- | --- | --- |
|  | Zweck | False Positives | False Negatives |
| FindBugs | Durchsucht Java-Bytecode mittels statischer Analyse nach bekannten Fehlermustern. | Das Programm berücksichtigt nicht, wenn man Code nur zu testzwecken schreibt, und der Code damit für richtige Zwecke als inkorrekt oder unnütz erkannt wird.  In anderen Fällen findet die statische Analyse nicht abgedeckte Fälle, die aber von der Programmlogik niemals auftreten können. | Da nur Anhand von statischen Konventionen und Fehlermustern analysiert wird, können keinerlei Logikfehler entdeckt werden, die programmspezifisch sind. |
| PMD | Durchsucht Java-Quellcode mittels statischer Analyse nach Fehlern und Ineffizienzen. | PMD verlangt, dass der primitive Datentyp short nie verwendet wird, was bei bestimmten mobilen Anwendungen, denen wenig Speicherplatz zur Verfügung steht nötig ist. Große short[] können dann durchaus Sinn machen, wenn sie final sind. |  |
| CheckStyle | Überprüft Java-Quellcode mittels statischer Analyse nach schlechtem Programmierstil und nicht wiederverwendbarem Code. | Da die Style-conventions alle konfigurierbar sind gibt es hier keine false positives im engen Sinne. | Die meisten Konventionen die sich auf das Gliedern von Klassen beziehen oder Ähnliches werden vom Programm nicht erkannt. |
| JDepend | Visualisiert und analysiert Abhängigkeiten im Java-Bytecode. | Abhängigkeiten von packages innerhalb eines Projektes sind nur dann ein Problem wenn die intention besteht einzelne Packages zu exportieren.  Es liegt also ein false positive vor, wenn das Projekt immer nur „als ganzes“ exportiert wird. |  |
| Dependency Finder | Analysiert und visualisiert Abhängigkeiten im Java-Bytecode. Gibt Verbesserungsvorschläge. |  |  |
| Checker Framework | Findet Fehler oder bestätigt ihre Abwesenheit. Basiert auf Compiler-Plugins. | @NonNull Typen kann, je nach Kontext, sinvoll ein @Nullable Typ zugewiesen werden. | Der erste Fehler, den wir zu FindBugs angegeben haben, wurde vom CheckerFramework nicht gefunden. |

|  |  |  |
| --- | --- | --- |
| FindBugs | Commons-collections4/src/test/java/org/apache/commons/collections4/CollectionUtilsTest.java:assertCollectResult(Collection)  Zeile 1200  “Long is incompatible with expected argument type Integer [...]”  True Positive: Die Variable collectionA ist vom generischen Typ Integer. An dieser Stelle wird abgefragt, ob sich darin ein Long-Wert befindet. | Ersetzte durch:  assertTrue(collectionA.contains(1) && !collectionA.contains(2));  Damit in collectionA nicht nach einem long gesucht wird. |
| Commons-lang3/src/test/java/org/apache/commons/lang3/concurrent/ConcurrenceUtilsTest.java:testConcurrentExceptionCauseError()  Zeile 54  “This code creates an exception (or error) object, but doesn't do anything with it.”  False Positive: In diesem Test soll der Konstruktor von ConcurrentException auf das Werfen von IllegalArgumentException geprüft werden. BugFinder meldet, dass „new ConcurrentException([…]“ ohne „throw“ aufgerufen wird. | Da es ein False Positive ist, gibt es hier nichts zu tun. |
| Commons-lang3/src/test/java/org/apache/commons/lang3/ArrayUtilsTest.java:testNullToEmptyBooleanEmptyArray()  Zeile 367  “**Bug**: Using .equals to compare two boolean[]'s, (equivalent to ==) in […]”  True Positive: An dieser Stelle hat das Programm erkannt, dass nicht die Elemente im boolean[] verglichen werden sondern nur auf Gleichheit der Arrays selbst geprüft wird, was vermutlich nicht die Intention des Tests war.  Sollte der Test doch den Zweck gehabt haben auf Objektgleichheit zu prüfen, so liegt schlechter Stil vor. | Hierfür eignet sich org.JUnit.assetArrayEquals(boolean[], boolean[]). |
| PMD | /commons-beanutils/src/main/java/org/apache/commons/beanutils/converters/AbstractConverter.java  Zeile 165  “AvoidThrowingNullPointerException”  True Positive: An dieser Stelle wird manuell eine NullPointerException geworfen, obwohl Zeile 169 genau das Gleiche tut. | Hier kann man den if-Block einfach weglassen. |
| /commons-beanutils/src/main/java/org/apache/commons/beanutils/BeanMap.java  Zeile 67  “VariableNamingConventions”  True Positive: Variablen die static und final sind, sollten in Großbuchstaben geschrieben warden. | Umbenennung der Variable in Großbuchstaben. |
| /commons-beanutils/src/main/java/org/apache/commons/beanutils/ResultSetIterator.java  Zeile 120  “AvoidThrowingRawExceptions”  True Positive: Es wird eine RuntimeException geworfen (und kein Subtyp). | Hier sollte man die SQLException entsprechend behandeln und nicht als RuntimeException weitergeben. |
| CheckStyle | /commons-cli/src/main/java/org/apache/commons/cli/PatternOptionBuilder.java  Zeile 96-121  “Switch ohne default”  True Positive: Der Code funktioniert zwar einwandfrei, nach Konvention sollte „return null“ aber in den „default“-case gehören. | Einen default case mit “return null” hinzufügen. |
| /commons-beanutils/src/test/java/org/apache/commons/beanutils/bugs/other/Jira87BeanFactory.java  Zeile 19  “Nicht benutztes import”  True Positive: Dieser Import wird nicht benutzt. | Import löschen. |
| /commons-beanutils/src/test/java/org/apache/commons/beanutils/DynaPropertyTestCase.java  Zeile 82  “Innere Zuweisungen sollten vermieden warden”  True Positive: An dieser Stelle ist es zwar noch überschaubar, trotzdem ist es schlechter Stil. | Die Zuweisung in zwei Zeilen aufteilen. |
| JDepend | Zwischen org.apache.commons.beanutils.locale und org.apache.commons.beanutils.locale.conversters besteht ein Zyklus. | Da die Zykel in allen Fällen zwischen einem Package und seinem subpackage bestehen, ist eine Möglichkeit die packages zu einem zu vereinigen. |
| Zwischen org.apache.commons.chain und org.apache.commons.chain.impl besteht ein Zyklus. |
| Zwischen org.apache.commons.jexl2 und org.apache.commons.jexl2.internal besteht ein Zyklus. |
| c) |  |  |
| Checker Framework | /commons-chain/src/java/org/apache/commons/chain/web/faces/FacesSetLocaleCommand.java  Zeile: 47  “dereference of possibly-null reference fcontext”  Context.get(String) kann null zurückgeben, also kann der Aufruf von fcontext.getViewRoot() eine NullPointerException werfen. |  |
| /commons-chain/src/java/org/apache/commons/chain/web/faces/FacesWebContext.java  Zeile 108  “incompatible types in assignment.  context = null;  found : null  required: @Initialized @NonNull FacesContext FacesWebContext.java /commons-chain/src/java/org/apache/commons/chain/web/faces line 108  Checker Framework Problem”  Die Variable context ist vom Typ @NonNull FacesContext. Ihr sollte deswegen nicht null zugewiesen werden. |  |
| /commons-chain/src/java/org/apache/commons/chain/web/ChainListener.java  Zeile 226  “Description Resource Path Location Type  incompatible types in argument.  context.setAttribute(attr, catalog);  found : @Initialized @Nullable Catalog  required: @Initialized @NonNull Object ChainListener.java”  Die Methode ServletContext.setAttribute(String, Object) erwartet als zweiten Parameter ein @NonNull Object, erhält jedoch ein @Nullable Catalog. |  |