

SOFTWAREQUALITÄTSSICHERUNG

Fehlermanagement



Gruppenmitglieder:	Dennis Jongebloed Julian Hinxlage Johannes Theiner
Semester:	Sommersemester 2020
letzte Änderung:	27. Mai 2020

1 Untersuchung des Fehlermanagements

1.1 Welches Werkzeug wird zur Verwaltung von Fehlermeldungen verwendet ?

PMD verwendet die GitHub Issues Seite des github Repositorys für die Verwaltung von Issues. Bis 2017 wurde das Sourceforge Issues Tool benutzt. In Sourceforge gibt es auch noch offene Issues, aber dies werden nicht mehr aktualisiert.

1.2 Wozu wird allgemein ein Fehlerstatusmodell benötigt ?

Ohne ein Fehlerstatusmodell wäre es nicht einsehbar, ob an einem Fehler gearbeitet wird oder nicht. Wenn ein Fehler in Progress ist weiß der Entwickler, dass ein anderer Entwickler bereits an dem Problem arbeitet. Das verhindert, dass parallel (und ohne Absprache) in dem Problem gearbeitet wird. Auch kann sofort erkannt werden, wenn ein Problem bereits behoben wurde (closed). Ohne das Modell müssten sich Entwickler immer gegenseitig befragen, wer woran arbeitet.

1.3 Welche Status werden bei Fehlermeldungen in Ihrem Projekt verwendet? Wie bilden sie sich auf das in den Themenfolien beschriebene Fehlerstatusmodell (9 Fehlermanagement, Folie 187) ab?

Unter GitHub Issues kann ein Issue 'Open' oder 'Closed' sein. Ein Aktiver Pull request mit einem assoziiertem Issue kann als 'In Progress' angesehen werden.

- Ein 'Open' Issue ist analog zu 'Offen'.
- Ein 'Closed' Issue ist analog zu 'Erledigt'
- Ein aktiver Pull Request ist analog zu 'Analyse' oder 'Korrektur'

1.4 Was ist allgemein der Unterschied zwischen Fehlerklasse und Fehlerpriorität?

Fehlerklasse: Die aufgedeckten Fehler werden nach Schwere der Fehlerwirkung eingeteilt.

Fehlerpriorität: Die Dringlichkeit der Fehlerkorrektur werden festgelegt.

1.5 Welche Fehlerklassen und Fehlerprioritäten werden in Ihrem Projekt unterschieden?

Fehlerklassen: Für die Fehlerklassen verwendet PMD Labels in Github. Fehlerbeschreibungen "9 Fehlermanagement" Folie 190

Klasse 1: Systemabsturz mit ggf. Datenverlust des Testobjekt ist in dieser Form nicht einsetzbar.

- a:bug: PMD crashes or fails to analyse a file.

Klasse 2: Wesentliche Funktion ist fehlerhaft. Anforderungen nicht beachtet oder falsch umgesetzt. Das Testobjekt ist nur mit großen Einschränkungen einsetzbar.

- a:bug: PMD crashes or fails to analyse a file.

Klasse 3: Funktionale Abweichung bzw. Einschränkung ("normaler" Fehler). Anforderungen fehlerhaft oder nur teilweise umgesetzt. System kann mit Einschränkungen genutzt werden.

- a:false-negative: PMD doesn't flag a problematic piece of code.
- a:false-positive: PMD flags a piece of code that is not problematic.

Klasse 4: Geringfügige Abweichung. System kann ohne Einschränkungen genutzt werden.

- Keine passenden Labels vorhanden.

Klasse 5: Schönheitsfehler (z.B. Rechtschreibfehler oder Mangel im Maskenlayout). System kann ohne Einschränkungen genutzt werden.

- an:enhancement: An improvement on existing features/rules.
- for:performance: The goal of this change is to improve PMD's performance.

PMD verwendet keine Fehlerprioritäten.

1.6 Welche Vorgaben für die Elemente einer Fehlermeldung gibt es in Ihrem Projekt ?

Unter `.github/ISSUE_TEMPLATE` gibt es Vorlagen mit Vorgaben für die unterschiedlichen Typen von Issues. Alle Issues müssen ein Name, Beschreibung und Label enthalten.

Es gibt folgende Vorlagen mit weiteren Vorgaben:

- bug report
 - Affects PMD Version
 - Exception Stacktrace
 - Code Sample demonstrating the issue
 - Steps to reproduce
 - Running PMD through
- feature request
 - Is your feature request related to a problem
 - Describe the solution you'd like
 - Describe alternatives you've considered
 - Additional context
- new rule
 - Proposed Rule Name
 - Proposed Category

- Code Sample
 - Possible Properties
- question
- rule violation
 - Affects PMD Version
 - Rule
 - Code Sample demonstrating the issue
 - Expected outcome
 - Running PMD through

2 Fehlermeldung

Nummer	#2545
Testobjekt	Regel UseDiamondOperator
Version	6.23.0; 6.24.0
Plattform	pmd-java
Entdecker	Maikel Steneker
Entwickler	PMD Team
Erfassung	26.05.2020 12:21
Status	Neu
Klasse	3-major
Priorität	2-next-version
Anforderungen	Die Regel wird nicht ausgelöst innerhalb eines geschachtelten Typ Parameters, bei Verwendung eines ternary Operators oder innerhalb eines statischen Initialisierers
Fehlerquelle	Programmierung
Testfall	Anwendung der UseDiamondOperator Regel auf Auflistung 1
Problem	in Zeilen 6,9 und 12 soll die Verwendung des Diamond Operators vorgeschlagen werden.
Kommentar	
Korrektur	ausstehend
Verweis	Vermutlich ausgelöst durch Behebung von Issue #1723

```
1  import java.util.List;
2  import java.util.ArrayList;
3  import java.util.Map;
4  import java.util.HashMap;
5  public class File {
6      List<Map<String,Object>> l = new ArrayList<Map<String,Object>>();
7      public void test() {
8          final List<String> l2;
9          l2 = true ? new ArrayList<String>() : new ArrayList<String>();
10     }
11     static {
12         l = new ArrayList<Map<String,Object>>();
13     }
14 }
```

Auflistung 1: Testfall