

Vorlesung "Spezielle Themen der Softwaretechnik"

Freie Universität Berlin, Institut für Informatik, Arbeitsgruppe Software Engineering
Prof. Dr. Lutz Prechelt, Stephan Salinger
Übungsblatt 1 WS 2007/2008 zum 2007-10-22

Ziel dieses Aufgabenzettels ist die Einarbeitung in JUnit sowie das Sammeln von ersten Erfahrungen mit dem Qualitätssicherungsverfahren Durchsicht. JUnit werden wir im Zusammenhang mit dem Vorlesungsabschnitt „Agile Softwareprozesse“ und Durchsichten im Zusammenhang mit dem Vorlesungsabschnitt „Cleanroom“ benötigen.

Aufgabe 1-1: (JUnit kennen lernen)

Ziel dieser Aufgabe ist die Einarbeitung in JUnit, ein Java-Framework zum Schreiben und Ausführen automatischer Unit-Tests. Da viele Projekte auch noch heutzutage die etwas betagte Version 3.8.1 einsetzen, sehen wir uns zuerst diese an (und nicht die aktuelle Version 4.4)

- a) **Lesen Sie über JUnit 3.8.x nach:** Lesen Sie die JUnit-Beschreibung *JUnit Test Infected: Programmers Love Writing Tests* unter <http://junit.sourceforge.net/doc/testinfected/testing.htm> vollständig und die Beschreibung *JUnit – A Cook's Tour* unter <http://junit.sourceforge.net/doc/cookstour/cookstour.htm> derart, dass Sie ein gutes Verständnis über das Frameworks erlangen und Teil b) dieser Aufgabe bearbeiten können. Die Autoren beider Beschreibungen sind die Entwickler von JUnit (Erich Gamma und Kent Beck). Recherchieren Sie notfalls nach weiteren Informationsquellen.
Zum besseren Verständnis können Sie sich auch JUnit installieren und es anhand der Beispiele in *JUnit Test Infected: Programmers Love Writing Tests* ausprobieren. Sie finden JUnit 3.8.2 unter http://sourceforge.net/project/showfiles.php?group_id=15278&package_id=12472
- b) **Beantworten Sie folgende Fragen derart, dass Sie diese Ihren Kommilitonen in der Übung verständlich beantworten können:**
- Welche beiden unterschiedlichen Verfahren kennt JUnit, um einen einzelnen Test laufen zu lassen? Wie funktionieren diese Verfahren im Detail?
 - Welche Verfahren kennt JUnit, um eine Gruppe von Testfällen ausführen zu können?
 - Was passiert, wenn ein TestRunner mit einer Testklasse (Erweiterung von `TestCase`) aufgerufen wird, in der *nicht* die Methode `suite()` überschrieben wurde?
 - Was ist der Unterschied zw. Errors und Failures in JUnit?
 - An welcher Stelle der Implementierung von JUnit wird das Entwurfsmuster Kompositum (Composite) verwendet? Warum wurde dies so gemacht?

Aufgabe 1-2: (Durchsichten anhand JUnit 3.8 durchführen)

Durchsichten sind ein manuelles statisches Verfahren der analytischen Qualitätssicherung. In dieser Aufgabe sollen Durchsichten im Code von JUnit 3.8 durchgeführt werden. Beachten Sie hierbei: JUnit ist ein extrem gut getestetes Framework. Es ist nicht zu erwarten, dass in ihm Fehler gefunden werden können. Aus diesem Grund sollen Sie für diese Übung eine spezielle Version von JUnit 3.8 verwenden, in die extra Fehler *hineingepflanzt* wurden. Verwenden Sie diese Version nur für diese Aufgabe und auf keinen Fall für Ihre weitere Arbeit mit JUnit. Sie finden diese Version auf der Webseite der Veranstaltung unter `JUnit_Error.zip`.

- a) **Einarbeiten in das statische Verfahren Durchsichten:** In der Vorlesung Softwaretechnik wurde eine Einführung in Durchsichten gegeben. Lesen Sie im entsprechenden Foliensatz (http://www.inf.fu-berlin.de/inst/ag-se/teaching/V-SWT-2005/42_Analytische-QS2.pdf ab Folie 29) noch einmal nach.

b) **Durchführung einer Codedurchsicht:** Besorgen Sie sich von der Webseite die Datei `JUnit_Error.zip` und entpacken Sie diese. Führen Sie nun eine Durchsicht der Klasse `junit.framework.TestSuite` durch und notieren Sie alle gefundenen Fehler. Beachten Sie dabei:

- Ihr Fokus liegt auf dem Entdecken von Defekten, die durch Programmierfehler entstanden sind.
- Versuchen Sie evtl., nicht gleich alle Arten von Defekten auf einmal zu finden, sondern konzentrieren Sie sich auf eine bestimmte Perspektive wie z.B. Exceptionhandling oder Nebenläufigkeit. Was wäre wohl eine geeignete Perspektive für die vorliegende Klasse?
- Falls Sie bestimmte Codeteile nicht verstehen, dann ziehen Sie ggf. den Code anderer Klassen oder die JavaDocs von JUnit (<http://junit.sourceforge.net/javadoc/>) zu Rate.
- Notieren Sie für jede Methode (und ggf. jede Eigenschaft), wie lange Sie für die ihre Durchsicht aufgewendet haben. Errechnen Sie den Gesamtaufwand, den durchschnittlichen Aufwand pro Codezeile und den durchschnittlichen Aufwand pro gefundenem Defekt.

Senden Sie diese Werte bis **Montag, den 22.10.2007 09:00Uhr** an `salingerXX*at*XXinf.fu-berlin.de`. Verwenden Sie folgendes Topic:

[SWT2 UE1][<nachname>,<vorname>]
(also z.B. [SWT2 UE2][Mustermann,Peter])

c) **Weitere Defektsuche:** Betrachten Sie nun die Klasse `junit.swingui.TestRunner`. Soviel sei verraten: Diese Klasse enthält zumindest einen Defekt, den Sie finden sollen.

Allerdings ist diese Klasse ziemlich umfangreich. Entscheiden Sie nun selbst, ob Sie eine Durchsicht oder eine andere Methode zur Lokalisierung des Defektes verwenden wollen.

Eine andere Möglichkeit wäre z.B. die Durchführung von Anwendungstests auf der Klasse. Stoßen Sie hierbei auf ein Versagen, können Sie versuchen, den zugehörigen Defekt z.B. unter Zuhilfenahme eines Debuggers zu lokalisieren. Die Schwierigkeit liegt hier in der Konstruktion geeigneter Testfälle. Allerdings sollten Sie mit hier nahe liegenden Testfällen schon ein Versagen erzeugen können. Sollte Ihnen dies nicht gelingen, lesen Sie den nachstehenden Tipp. Eines sollten Sie in diesem Fall auf jeden Fall zuerst tun: Beheben Sie die unter Aufgabenteil b) gefundenen Fehler.

TIPP: Schreiben Sie sich eine einfache Klasse und zugehörig eine Testklasse mit zwei Testmethoden, die sicher fehlschlagen und zwei Testmethoden, die sicher nicht fehlschlagen. Fügen Sie die Testfälle zu einer Suite zusammen und führen Sie diese mit dem Swingui-Testrunner aus. Sehen Sie sich das Ergebnis genau an.