

Property based Testing

Grundkonzept

Beim Property (im Sinne von Eigenschaft des zu prüfenden Algorithmus oder Moduls) basierten Testen wird der Testgegenstand (Modul, Klasse, Methode, Funktion, ...) mit zufälligen Eingangswerten oder Parametern aufgerufen und geprüft ob die resultierenden Ausgabewerte oder Rückgabewerte den Eigenschaften des Testgegenstandes entsprechen.

Beispiel.

Eine Methode zur Zerlegung einer Zahl in Primfaktoren wird mit zufälligen Werten aufgerufen. In der Assertion wird die Gegenoperation genutzt um die Korrektheit zu prüfen. Es werden also alle ermittelten Primfaktoren nacheinander multipliziert und dann geschaut ob die ursprüngliche Zahl wieder herauskommt.

Die meisten Tools bieten Möglichkeiten die Menge der Testwerte einzuschränken. Beispielsweise nur ganze Zahlen als Eingabewerte für die Primfaktorenzerlegung zu nutzen.

Die Wertebereiche welche bei der Einschränkung der Testwerte ausgegrenzt wurde muss durch andere Testverfahren abgesichert werden. Beispielsweise könnte ein Floating Parameter zu einer Exception führen, deren Werfen durch einen EBT (Example based Test) abgeprüft wird.

Literatur:

* <https://www.heise.de/developer/artikel/Property-based-Testing-mit-JUnit-QuickCheck-3935767.html>

Mutationstest - Wie gut sind meine Unittests?

Grundkonzept

Generell lassen sich viele Unittests schreiben. Die Erfahrung zeigt, dass einige davon immer grün sind, manche am Monatsende oder an speziellen Tagen im Jahr rot werden, manche bei Infrastrukturproblemen den build brechen.

Auch zeigt sich im Laufe der Zeit, dass die Anwendung erfolgreich betrieben wird und dennoch kleinere Fehler enthält und das obwohl die Kodecoverage bei über 95% lag.

Der Grund ist, dass der Kode trotz hoher Kodeabdeckung nicht vollständig vor Fehlern beim Refactoring oder bei der Einarbeitung neuer Fachlichkeit, geschützt ist.

Hier kommen die Mutationstests ins Spiel. Sie führen die normale Testsuite aus und merken sich deren Ergebnisse. Dann modifizieren sie den produktiven Kode und schauen ob ein Test rot geworden ist und diese Mutation entdeckt hat. Falls nicht ist eine Lücke in der Testabdeckung gefunden wurden.

Vorteil:

* Bestimmung der Testqualität möglich

Nachteil:

* Die Laufzeit der Tests verdoppelt sich mindestens, da jetzt die Suite mindestens zwei mal, wenn nicht noch mehrmals durchlaufen werden lassen muss.



By **Huluvu424242**
(FunThomas424242)

Published 29th June, 2019.
Last updated 29th June, 2019.
Page 1 of 1.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

cheatography.com/funthomas424242/
github.com/Huluvu424242