AAA-Regel

Um wirklich gute Unit Tests zu schreiben, gibt es ein paar Faustformeln, wie ein Unit Test aussehen sollte. Eine davon ist die AAA-Regel.

AAA steht dabei für

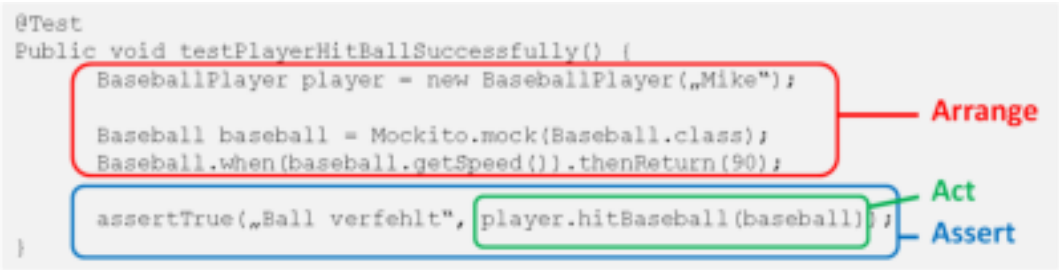
- Arrange
- Act
- Assert

und beschreibt den Aufbau eines Unit Tests.

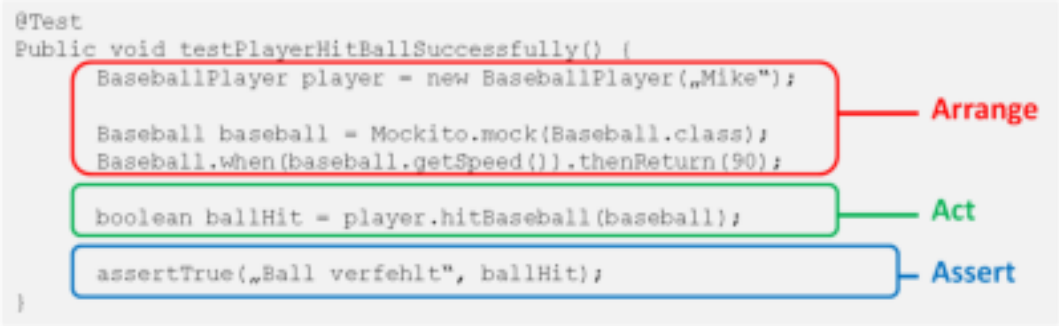
Im **Arrange** wird die Startsituation des Tests definiert. Die Komponente wird erzeugt und initialisiert, Mocks werden erzeugt, gesetzt und evtl. das Verhalten der Mocks definiert, Variablen werden gesetzt.

Im **Act** wird die eigentliche Aktion, die das zu verifizierende Ergebnis erzeugen soll, durchgeführt. In der Regel ist das der Aufruf der Methode, deren Verhalten getestet werden soll. Ein guter Unit-Test besteht hier nur aus einem Aufruf. Ist mehr als ein Aufruf notwendig, dann ist das ein Hinweis darauf, dass mehr als nur ein einfacher Aspekt getestet wird!

Und schlussendlich wird im **Assert** nun überprüft, ob auch das erwartete Ergebnis durch das Act erzielt wurde. Hier gilt, dass nur ein einzelnes fachliches Ergebnis geprüft werden soll. Dabei liegt die Betonung auf fachliches Ergebnis, nicht technisches. Das bedeutet also nicht zwangsläufig, dass dafür nur ein einzelnes Assert-Statement verwendet werden darf. Wenn über mehrere Vergleiche verschiedene Teilaspekte des fachlichen Ergebnisses überprüft werden müssen, dann ist das schon in Ordnung. Wenn allerdings z.B. ein Unit-Test TestInsertKunde im Assert prüft, ob der Kunde wirklich in der die Datenbank geschrieben UND ein entsprechender Logeintrag im Logfile erzeugt wurde, dann ist das ein Hinweis darauf, dass hier mehr als nur ein einzelner Aspekt getestet wurde. So simple und vielleicht auch nervig, wie die Einhaltung der AAA-Regel auch scheinen mag, sie hilft, echte Unit Tests zu identifizieren.



Wir sehen in dem Beispiel, dass hier Act und Assert in einer Zeile, quasi ineinander verschachtelt stehen. Hier könnte man nun argumentieren, dass für die bessere Lesbarkeit dieses zu trennen wäre, also den Schlag des Spielers in einer einzelnen Zeile und das Ergebnis daraus in einer lokalen Variable fangen und dann diese im Assert auswerten.



Ob das nun der besseren Lesbarkeit und Wartbarkeit wegen wirklich notwendig ist, ist schwer objektiv zu beurteilen. Deshalb überlassen wir diese Entscheidung denen, die mit dem Code und den Tests arbeiten müssen.