Effective Java

Grundlagen zu

* equals()-Methode
* hashCode()-Methode
* clone()-Methode
* Immutable-Objekts

# Einleitung

Mittels eines Beispielprojekts sollen die Aufgaben und Möglichkeiten von Equals, Hash-Code Clone und Immutability aufgezeigt und geübt werden.

Es gibt vier Übungsblöcke, für jedes Thema einen. Die Blöcke bauen auf einander auf:

1. **Equals**
2. **HashCode**
3. **Clone**
4. **Immutable**

Jeder Block besteht aus einem **Theorie**- und einem **Praxis-Teil** bei dem das gelernte umgesetzt werden kann.

Für jeden Block gibt es einen entsprechenden JUnit-Test der nach Abschluss der Übung "Grün" sein soll, ohne dass dieser verändert wird. Die Klassen sind im Verzeichnis / Package "src/**test**/java/org/anderes/edu/effecitive" abgelegt. Für jeden Block gibt es ein eigens Unterverzeichnis.

Die Domänenklassen welche mit jedem Block erweitert werden, sind im Verzeichnis "src/**main**/java/org/anderes/edu/effective/domain" zu finden:

* org.anderes.edu.effective.domain.***Employee***
* org.anderes.edu.effective.domain.***Address***
* org.anderes.edu.effective.domain.***Project***

# Vorbereitung

Das Projekt ist mit Maven aufgebaut und benötigt mind. Java 8. Mittels Import kann das Projekt ins Eclipse übernommen werden. Beachte: Die JUnit-Test kompilieren nicht alle. Im Verlauf der Übung werden die bestehenden Klassen erweitert, so dass alle Test-Kompilieren und auch OK d.h. Grün sind.

# Block Equals

## Ziel

Für alle Domänenklassen soll eine entsprechende Equals-Methode implementiert sein, so dass der JUnit-Test "EmployeeEqualsTest" durchläuft, d.h. Grün wird.

## Vorbereitung

Damit es nicht ständig zu Compiler-Fehler kommt, löschen wir alle JUnit-Tests ausser "EmployeeEqualsTest" aus dem Projekt. Wir werden Sie später zu einem angemessenen Zeitpunkt wieder ins Projekt kopieren.

## Vorgehen

Wie bereits erwähnt ist bei jedem Block ein bisschen Theorie angesagt um in die Materie einzusteigen und sie besser zu verstehen. Für diesen Block ist das Dokument "***Objektvergleich (Angelika Langer).docx***" durchzulesen, welches sich im selben Verzeichnis befinden wie dieses Dokument.

Anschliessend implementiere die entsprechenden Methoden in den Domänenklassen. Beachte, der JUnit-Test darf nicht verändert werden.

## Tipp

Es gibt auch 3th Party-Library die Dir ev. Arbeit abnehmen können. Natürlich sollte hier auf Qualität geachtet werden. Denn Fehler kann man -wie Du im Dokument gelesen hast- viele machen. Die "[Apache Commons Lang](http://commons.apache.org/proper/commons-lang/)" Komponente ist sicher eine dieser vertrauensvolle Quelle. Wäge ab, ob Du die Equals-Methode selber implementierst oder ob Du eine Klasse / Funktion der "[Apache Commons Lang](http://commons.apache.org/proper/commons-lang/)" verwenden möchtest. Siehe API zu [EqualsBuilder](http://commons.apache.org/proper/commons-lang/javadocs/api-3.4/org/apache/commons/lang3/builder/EqualsBuilder.html).

Eclipse bietet die Möglichkeit mittels "Source" --> "Generate hashCode and equals…" die Equals-Methode generieren zu lassen. Betrachte das Ergebnis und vergleiche es mit der eigenen Lösung und/oder der Lösung mittels "Apache Commons Lang".

# Block HashCode

## Ziel

Für alle Domänenklassen soll eine entsprechende hashCode-Methode implementiert sein, so dass der JUnit-Test "EmployeeHashCodeTest" durchläuft, d.h. Grün wird.

## Vorbereitung

Kopiere den Test "EmployeeHashCodeTest" ins Package "org.anderes.edu.effective.hashcode".

## Vorgehen

Für diesen Block ist das Dokument "***Hash-Code-Berechnung (Angelika Langer).docx***" durchzulesen, welches sich im selben Verzeichnis befinden wie dieses Dokument.

Anschliessend implementiere die entsprechenden Methoden in den Domänenklassen. Beachte, der JUnit-Test darf nicht verändert werden.

## Tipp

Auch für diesen Block bietet "[Apache Commons Lang](http://commons.apache.org/proper/commons-lang/)" eine [Hilfsklasse](http://commons.apache.org/proper/commons-lang/javadocs/api-3.4/org/apache/commons/lang3/builder/HashCodeBuilder.html) an.

Eclipse bietet die Möglichkeit mittels "Source" --> "Generate hashCode and equals…" die HashCode-Methode generieren zu lassen. Betrachte das Ergebnis und vergleiche es mit der eigenen Lösung und/oder der Lösung mittels "Apache Commons Lang".

# Block Clone

## Ziel

Für alle Domänenklassen soll eine entsprechende clone-Methode implementiert sein, so dass der JUnit-Test "EmployeeCloneTest" durchläuft, d.h. Grün wird.

## Vorbereitung

Kopiere den Test "EmployeeCloneTest" ins Package "org.anderes.edu.effective.clone".

## Vorgehen

Für diesen Block ist das Dokument "***Das Kopieren von Objekten (Angelika Langer).docx***" durchzulesen, welches sich im selben Verzeichnis befinden wie dieses Dokument.

Anschliessend implementiere die entsprechenden Methoden in den Domänenklassen. Beachte, auch hier darf der JUnit-Test nicht verändert werden.

# Block Immutable

## Ziel

Es soll die Lösung "*Duale Klassen mit gemeinsamen Interface*" so implementiert werden, dass der JUnit-Test "EmployeeImmutableTest" durchläuft, d.h. Grün wird.

## Vorbereitung

Kopiere den Test "EmployeeImmutableTest" ins Package "org.anderes.edu.effective.immutability".

## Vorgehen

Für diesen Block ist das Dokument "***Unveränderliche Typen (Angelika Langer).docx***" durchzulesen, welches sich im selben Verzeichnis befinden wie dieses Dokument.

Anschliessend implementiere die entsprechenden Klassen / Methoden. Im Package "org.anderes.edu.effective.immutability" gibt es bereits Skeleton's der Klassen und die Interfaces die Du für diese Übung benötigst. Beachte, auch hier darf der JUnit-Test nicht verändert werden.