

# Effective Java

---

*Grundlagen zu equals(), hashCode(), compareTo(), clone()-Methode und Immutable-Objekts*

## Einleitung

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

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

- 1) EQUALS
- 2) HASHCODE
- 3) COMPARETO
- 4) CLONE
- 5) 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/effective" abgelegt. Für jeden Block gibt es ein eigenes Unterverzeichnis.

Unit Tests sind die beste Dokumentation die ich kenne, darum: Lese und verstehe die Test's!

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)**" durchzulesen, welches sich im selben Verzeichnis befindet wie dieses Dokument.

Anschliessend implementiere die entsprechenden Methoden in den Domänenklassen. Beachte, der JUnit-Test darf nicht verändert werden und auch die Methodensignatur der bestehenden Methoden der Domänenklassen sollen 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](#)" 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](#)" verwenden möchtest. Siehe API zu [EqualsBuilder](#).

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.andereis.edu.effective.hashcode".

### Vorgehen

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

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

### Tipp

Auch für diesen Block bietet "[Apache Commons Lang](#)" eine [Hilfsklasse](#) 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 CompareTo

### Ziel

Für die Domänenklasse Project soll das Interface Comparable implementieren, damit die Projekte verglichen/sortiert werden können. Die Methode getProjects() der Klasse Employee soll eine sortierte Collection zurück liefern (Welche Collection musst Du dafür einsetzen?). Der Test "CompareToTest" soll durchlaufen und Grün werden.

### Vorbereitung

Kopiere den Test "CompareToTest" ins Package "org.andereis.edu.effective.compare".

### Vorgehen

Für diesen Block ist das Dokument "**Objekt-Vergleich mittels Comparatoren (Angelika Langer)**" durchzulesen, welches sich im selben Verzeichnis befindet wie dieses Dokument.

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

## Tipp

Auch für diesen Block bietet "[Apache Commons Lang](#)" eine [Hilfsklasse](#) an.

## 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.andereis.edu.effective.clone".

### Vorgehen

Für diesen Block ist das Dokument "***Das Kopieren von Objekten (Angelika Langer)***" durchzulesen, welches sich im selben Verzeichnis befindet 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.andereis.edu.effective.immutability".

## Vorgehen

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

Anschliessend implementiere die entsprechenden Klassen / Methoden. Im Package "org.andere.s.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.