ToiletPaper #106

jambit
WHERE INNOVATION WORKS

Bevor ich hier den Depp mach', nehm' ich lieber MapStruct

Autor: Magnus S. / Software Architect / Business Division Automotive Bavaria

✘ Problem

Eine ungeliebte und fehleranfällige, aber (vor allem in Mehrschichtenarchitekturen) wiederkehrende Aufgabe ist das Schreiben von Mapperklassen für ähnliche Objekte, z.B. zwischen Datenbankentitäten und Request/Response-DTOs. Auch das Schreiben von Unit Tests für solche Mapper ist mühsam.

✓ Lösung

Die Java-Library **MapStruct** (http://mapstruct.org) nimmt einem diese Aufgabe ab, indem sie Annotations-basiert automatisch Mapping-Code generiert.

Vorteile:

- Generierung zur Compile-Zeit: kein Einfluss auf Performance
- Generierter Code nutzt Getter/Setter-Methoden, ist daher typensicher, leicht zu debuggen und einfach zu verstehen/ erweitern
- Sinnvolle Default-Konventionen, Verhalten kann leicht überschrieben werden

➔ Beispiel

Zwei komplementäre Klassen (Getter/Setter-Methoden nicht gezeigt):

```
public class CupOfCoffee {
    private Enums.Brand brand;
    private double temperature;
    private int volume;
    private double milk;
    private double sugar;
    private boolean isCorretto;
}
```

```
public class CupOfCoffeeDTO {
    private String brand;
    private String temperature;
    private int volumeOfCup;
    private double milk;
    private double sugar;
    private boolean isCorretto;
}
```

MapStruct-Basisklasse:

```
@Mapper(componentModel = "spring")
public abstract class CoffeeMapper {
    @Mapping(source = "volume", target = "volumeOfCup")
    @Mapping(source = "temperature", target="temperature", numberFormat = "$#.0")
    public abstract CupOfCoffeeDTO toCupOfCoffeeDTO(CupOfCoffee entity);

    @InheritInverseConfiguration
    public abstract CupOfCoffee toCupOfCoffee(CupOfCoffeeDTO dto);
}
```

Beim Kompilieren generiert MapStruct den Mapping-Code entsprechend den Annotationen. Die expliziten @Mapping-Annotationen sind nur für nicht-triviale Mappings (z.B. bei unterschiedlichen Variablennamen, Zahlenformaten etc.) nötig. Die abstrakte Klasse kann noch um eigene Mapping-Funktionen erweitert werden.

Obacht:

- nach jeder Änderung der abstrakten Mapper-Klasse muss neu kompiliert werden
- funktioniert in Verbindung mit Lombok (ein weiterer annotationsbasierter Codegenerator) manchmal nicht ganz reibungslos

✚ Weiterführende Aspekte

- https://github.com/mapstruct/mapstruct
- http://mapstruct.org/documentation/dev/reference/html/