Portfolio-Workshop Refactoring with M. Fowler

Duale Hochschule Baden-Württemberg (DHBW) Karlsruhe
Studiengang Wirtschaftsinformatik, WWI17B2

Marc Schanne
marc@schanne.org

https://moodle.dhbw.de/course/view.php?id=3132

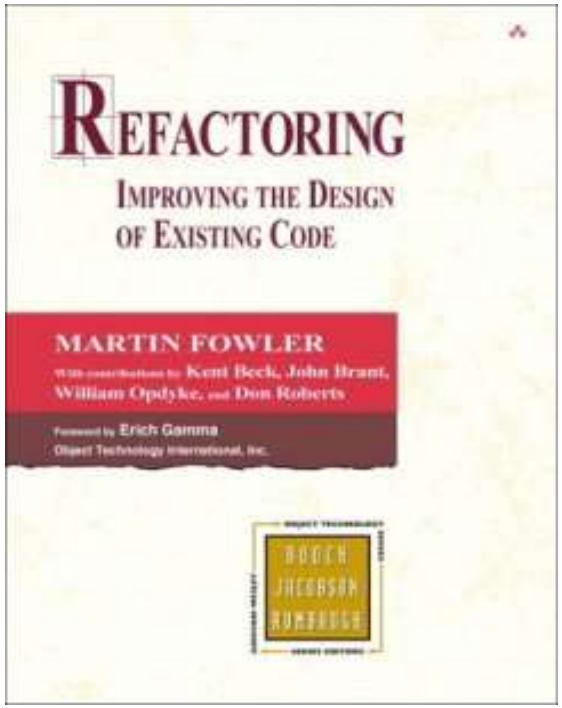
- Portfolio-Workshop
  - Praxis
  - Reflektion
  - Interview
- Refactoring
  - Code-Smells
  - Design Patterns
- Simple Example
  - Martin Fowler
  - Movie, Customer, Rental
  - Quellen und Tests
  - Geplante Änderungen

Portfolio Workshop I

- Refactoring am Beispiel
  - Quellen
    https://github.com/schanne-dhbw-wwi17/fowler
  - Reflektion
    - nach Workshop II, Abgabe bis 12.07./12h00
  - Interview
    - einzeln am 30.6.

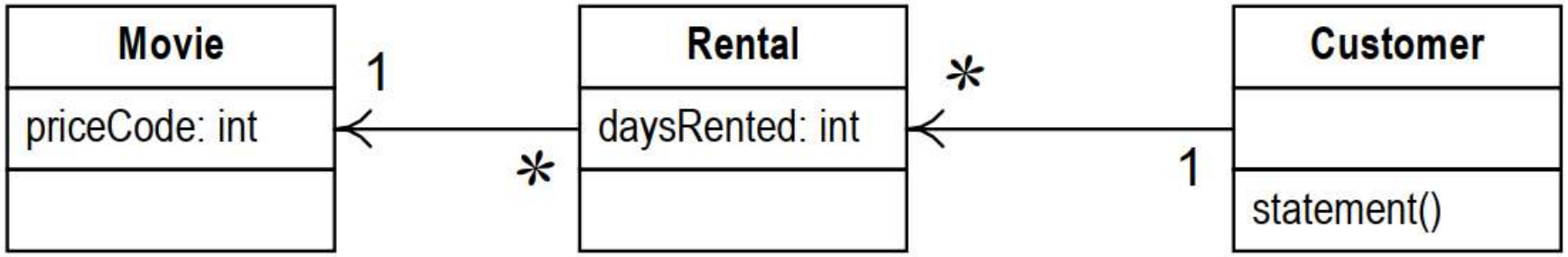
Refactoring mit M. Fowler

- Martin Fowler: Refactoring. Improving the Design of existing Code, Addison-Wesley, 1999



- „A series of small steps, each of which changes the program‘s internal structure without changing its external behaviour“
  - Es ist wichtig sicher zu stellen, dass sich das externe Verhalten des Programms beim Refactoring nicht verändert:
    - Existenz guter Tests ist wichtig!

Simple Example



- Beispiel-Ausgabe:

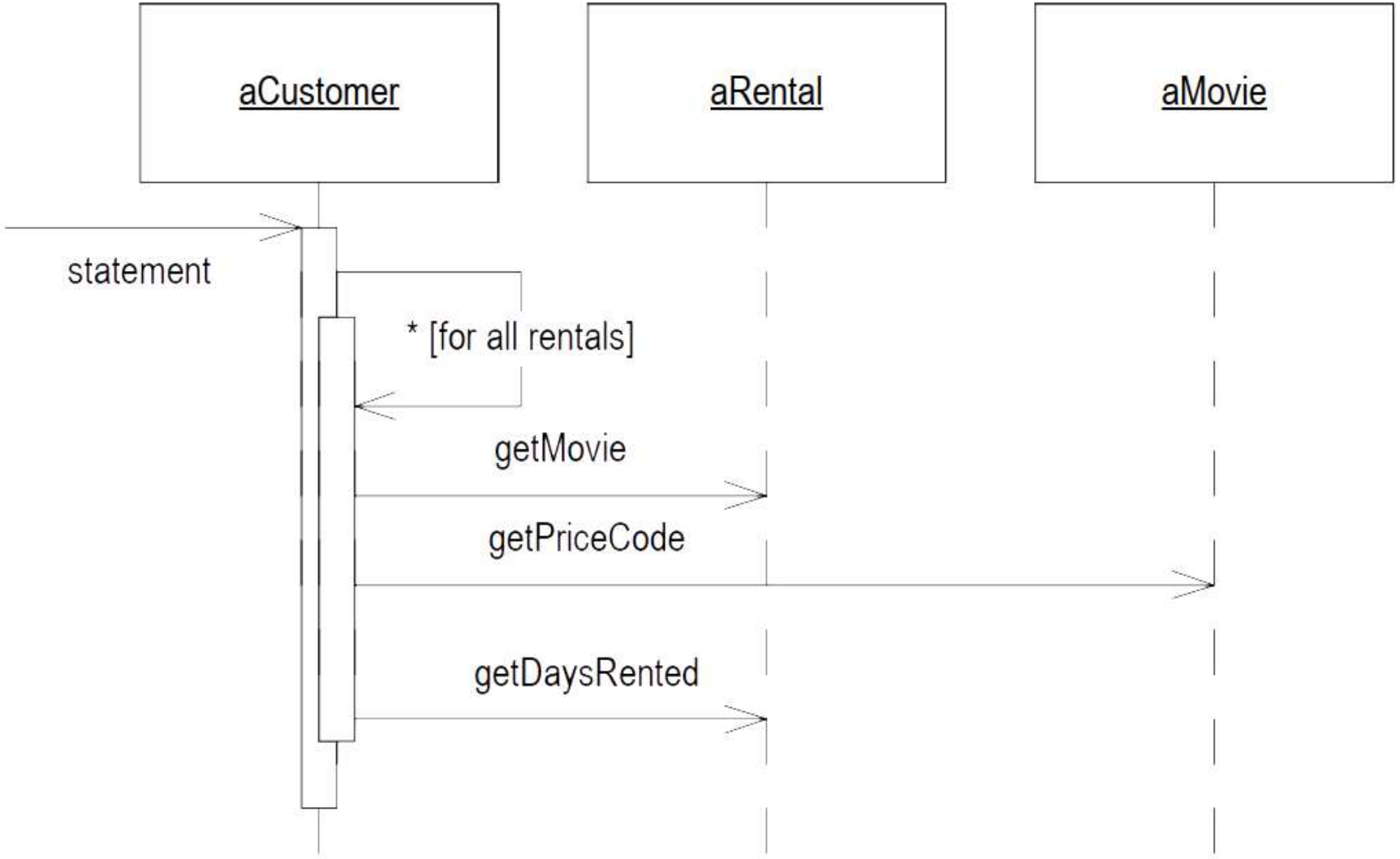
```
Rental Record for Dinsdale Pirhana
    Monty Python and the Holy Grail 3.5
    Ran 2
    Star Trek 27 6
    Star Wars 3.2 3
    Wallace and Gromit 6
Amount owed is 20.5
You earned 6 frequent renter points
```

Bestandscode

- Der Code zu den drei Klassen Movie, Customer und Rental finden sich im Github-Repo:
  - https://github.com/schanne-dhbw-wwi17/fowler

- Um Refactoring zu erlauben, gibt es auch ein paar Unit-Tests

Interaktion für Abrechnung (Statement)



Notwendige Änderungen/Erweiterungen

- Refactoring hat in der Praxis leider selten einen Selbstzweck, in der Regel kommt es nur zus. mit Änderungen oder Erweiterungen.
- Die Anforderungsänderungen die im Simple Example ein Refactoring nötig machen sind:
  - Abrechnung soll zusätzlich als HTML-Version ausgegeben werden.
  - Die Film-Klassifikation wird sich bald ändern
    - Zusammen mit Änderungen bei der Regelungen für die Bezahlung und Rabatten für Viel-Entleiher (frequent renter points)

Obligatorische Refactorings

- Notwendige Refactorings können dem Vortrag von Martin Fowler auf der JavaOne 2000 bzw. der Beschreibung im Lastenheft und den geplanten User Stories entnommen werden.
  - Extract Method
  - Move Method
  - Replace Temp with Query
  - Replace Type Code with State/Strategy
  - Replace Switch with Polymorphism
  - Form Template Method

Ziele des Workshops

- Durchführung der notwendigen Refactorings
  - und damit Einübung des Vorgehens

- Aber wichtiger ist die Praxis des Softwareentwicklungsprozesses: **Wasserfall** bzw. **SCRUM**
  - Details finden sich im Git-Repo

Fragen ?