Portfolio-Workshop
Refactoring 2 with M. Fowler

Duale Hochschule Baden-Württemberg (DHBW) Karlsruhe
Studiengang Wirtschaftsinformatik

Marc Schanne
marc@schanne.org

dhbw.schanne.org

- Workshop
  - Praxis
  - Klausur
- Refactoring
  - Code-Smells
  - Design Patterns

- Next Example
  - Martin Fowler
  - Theater-Company
  - Quellen und Tests
  - Geplante Änderungen

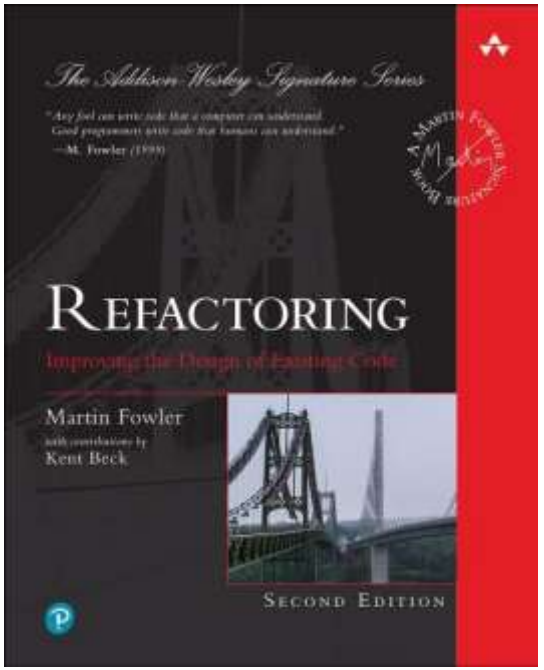
Praxis und Klausur

- Workshop 2, gleiche Teams, anderes Prozess-modell, wieder Refactoring:
  - https://moodle.dhbw.de/course/view.php?id=5670
- Klausur am 25.04.2022 oder 09.05.2022
- Aufgaben zu
  - Softwareentwicklung agil vs. Klassisch
  - Refactoring & CCD
  - Entwurfsmuster

  möglichst mit Bezug zu den Workshops.

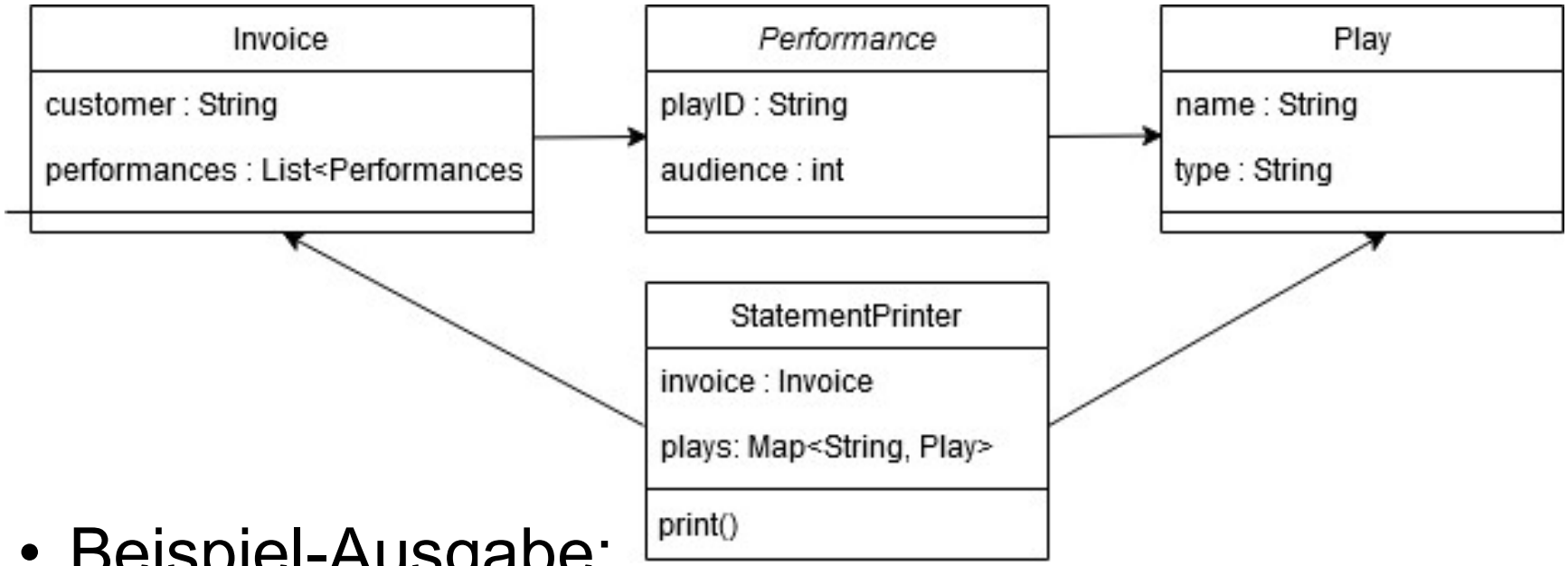
Refactoring mit M. Fowler

- Martin Fowler: Refactoring. Improving the Design of existing Code, 2nd Ed, Add.-Wesley, 2019



- „Any fool can write code that a computer can understand. Good programmers write code that human can understand. “
  - Ziel von Refactoring ist von Menschen lesbarer Code, keine Veränderung des äußeren Verhaltens:
    - Existenz guter Tests ist wichtig!

Next Example



- Beispiel-Ausgabe:

```
Statement for BigCo
   Hamlet: $650.00 (55 seats)
   As You Like It: $580.00 (35 seats)
   Othello: $500.00 (40 seats)
Amount owed is $1,730.00
You earned 47 credits
```

Bestandscode

- Der Code zu den vier Klassen Invoice, Play, Performance und StatementPrinter finden sich im Github-Repo:
  - https://github.com/schanne-dhbw-se2/wwi20b2-workshop2

- Um Refactoring zu erlauben, gibt es auch einen Unit-Test. Das sollte man sicher erweitern …

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 Next Example ein Refactoring nötig machen sind:
  - Abrechnung und Rabatt-Optionen sollen zusätzlich als HTML-Version ausgegeben werden.
  - Die von der Theater Company gespielten Stücke werden jetzt bzw. zukünftig erweitert.
    - Mögliche neue Typen sind: history, pastoral, pastoralcomical, historicalpastoral, tragicalhistorical, tragicalcomicalhistorical-pastoral, scene individable und poem unlimited

Obligatorische Refactorings

- Notwendige Refactorings zielen auf den für Menschen besser verständlichen Code.
  Im Wesentlichen entspricht dies den aus der letzten Wochen (Vortrag Fowler) bekannten Refactorings, siehe auch LH bzw. User Stories.
  - 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 Softwareent-wicklungsprozesses: **Wasserfall** bzw. **SCRUM**
  - Details finden sich im Git-Repo

Fragen ?

- Gibt es noch allgemeine Fragen?
  - Kommunikation wie letzte Woche: marc@schanne.org
  - Antworten zum Prozess gibt es jetzt im entsprechenden Separee …
- Viel Erfolg!