SoSe 2025 – Softwaretechnologie I

U06 – Entwurfsmuster (I)

Kursverantwortlicher: Prof. Uwe Aßmann

Übungsleiter: Dr. Sebastian Götz

Inhalt der Übung

- Entwurfsmuster und deren Implementierung
- Datenstrukturen

Aufgabe 1 (Bauteil)

Gegeben ist das nachfolgende UML-Klassendiagramm für eine einfache Stücklistenverwaltung. In diesem Modell bestehen Bauteile entweder aus Einzelteilen oder Baugruppen. Der Preis einer Baugruppe ergibt sich aus der Summe der Preise seiner Bestandteile.

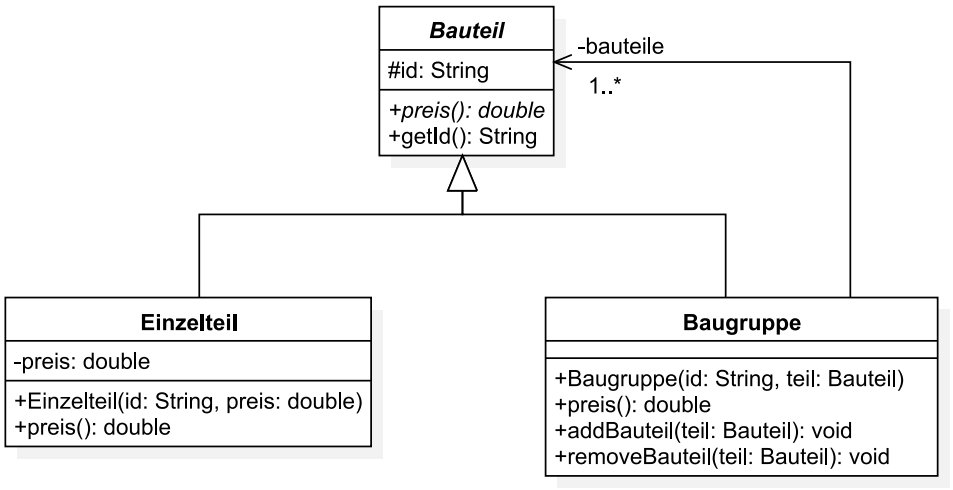

Abbildung 1: UML-Klassendiagramm für Stücklistenverwaltung

- Welches aus der Vorlesung bekannte Entwurfsmuster wurde in diesem Modell verwendet?
- Zeichnen Sie dieses Entwurfsmuster und die Rollen der daran beteiligten Klassen in das obige Klassendiagramm ein!
- Ergänzen Sie die gegebene Implementierung des Modells (`Bauteil.zip`), in dem Sie die Klasse `Baugruppe` erstellen!
  - Verwenden Sie zur Implementierung das Java-`Collection`-Framework.
  - Beachten Sie, dass die Aggregation zwischen `Baugruppe` und `Bauteil` (Attribut `bauteile`) standardmäßig als `java.util.Set` implementiert wird.

- Diskutieren Sie den Entwurf und refaktorisieren Sie die Klasse `Baugruppe`!
  - **Refaktorisierung 1:**
    a) Diskutieren Sie die `removeBauteil()`-Methode!
    b) Würden Sie weitere Methoden refaktorisieren?
  - **Refaktorisierung 2:**
    Überlegen Sie sich anhand des folgenden Beispiels, was passiert, wenn Sie die Aggregation zwischen `Baugruppe` und `Bauteil` als Liste implementieren!

```
Einzelteil t1 = new Einzelteil("E001", 2.30);
Einzelteil t2 = new Einzelteil("E002", 4.70);
Baugruppe t3 = new Baugruppe("B001", t1);
Baugruppe t4 = new Baugruppe("B002", t2);
t3.addBauteil(t4);
t3.addBauteil(t2);
t3.addBauteil(t4);
```

Aufgabe 2 (MyCollection)

Gegeben ist das Programm aus `Bestellung.zip`. Dieses Java-Programm arbeitet mit einer selbst definierten Implementierung (`MyCollection`) des Interfaces `java.util.Collection`. Die Klasse `MyCollection` implementiert die Methode `iterator()` entsprechend dem Iterator-Pattern.

- Veranschaulichen Sie sich das Programm mit Hilfe eines UML-Diagramms.
- Machen Sie sich die Arbeitsweise des implementierungsspezifischen Iterators klar und zeichnen Sie das *Iterator-Pattern* ein!
- Reimplementieren Sie `MyCollection` als generische Datenstruktur!

Empfehlungen zum Selbststudium

- **Bevor Sie mit der Lösung dieser Aufgaben beginnen**
  - Sie sollten die Vorlesungen von Prof. Aßmann zu den Entwurfsmustern (Folien (*22-st-design-patterns* und *23-st-connectors-iterators-channels*) / Videos (*[T2] 3. Einführung in Entwurfsmuster* und *[T2] 4. Konnektoren, Iteratoren und Kanäle*)) gehört und verstanden haben.
  - Informieren Sie sich in den **Learning Outcomes dieser Übung** über die wichtigsten Entwurfsmuster und forschen Sie nach ihren Zielen und Eigenschaften (z.Bsp. Vorlesungsfolien, Wikipedia, ...).
- **Ab Ende der Übungswoche U07:**
  - Versuchen Sie sich an der Implementierung der Artemis-*Exam*-Aufgaben `Pricing`, `Predicate Iterator`, `Renovation Project`, `Part Management`, `Desktop Search Engine`, `Project Management` und `Payroll`! Überlegen Sie, welche Entwurfsmuster in den Programmen enthalten sind!
  - Lösen Sie Aufgabe 3 der Klausur SS 2009 (Zugriff nur über TU VPN), die die *Aufgabe 1 (Bauteil)* dieser Übung erweitert!

- **Literaturempfehlung**: Softwaretechnologie für Einsteiger. PEARSON Studium, 3. erweiterte Auflage, 2019 (als E-Book im TU-Netz verfügbar, siehe Hinweis der OPAL Linksammlung).
  - Abschnitt 5.7 Rollenmodellierung, S. 111-113
  - Kapitel 9 Objektentwurf: Wiederverwendung von Mustern, S. 195 ff.