

Aufgabenblatt 1

Ziel dieses Aufgabenblatts ist es, Sie mit SWA-Prinzipien und -Muster vertraut zu machen. Zudem dient dieses Aufgabenblatt dazu, die Java-Kenntnisse wieder aufzufrischen.

Abgabe: Gr. 1+2: 29.04.2021; Max. Punktzahl: 20; Min. Punktzahl: 12 Punkte

Aufgabe 1.1: Einarbeiten in das taktische Design mit dem Domain Driven Design

(3 Punkte, Gruppenarbeit; 2er oder 3er Gruppe)

Arbeiten Sie sich, in das „Domain Driven Design“ ein. Alle Gruppenteilnehmer*innen sollten folgende Patterns des taktischen Designs kennen und erklären können:

- Layered Architecture + Modules + Anti-Corruption-Layer
- Services
- Entities
- Object-Values
- Repositories
- Aggregates und
- Factories

Verwenden Sie bspw. folgende Unterlagen:

- a) Foliensatz „02_SWA_Grundlagen.pdf“ (Osca) und vorhandene Online-Vorlesung im OpenCast-Portal (Zugang: siehe SWA_Infos.pdf)
- b) InnoQ, Domain Driven Design Quickly, Online: verfügbar:
<https://www.infoq.com/minibooks/domain-driven-design-quickly/>
- c) Plöd M., Implementierung der DDD-Patterns mit Spring, Vortrag auf der W-JAX 2020, Online verfügbar: <https://www.youtube.com/watch?v=BFXuFb40P8k>

Aufgabe 1.2: Verstehen eines vorhandenen Software-Designentwurfs und identifizieren der zugrundeliegenden Prinzipien und Muster

(3 Punkte, Gruppenarbeit, 2er oder 3er Gruppe)

In der Anlage A finden Sie einen ersten Entwurf einer Shopping-App, namens Simple-Shopping-App (kurz: SSA). Letztendlich soll diese Software genau das machen, was viele andere Software-Produkte jetzt auch schon machen: Waren suchen und finden, dem Warenkorb hinzufügen und letztendlich die Ware bezahlen.

Bei dieser Anwendung soll die Mensch-Maschine Schnittstelle als Konsolen-Anwendung programmiert werden. Dies natürlich nur zu Testzwecken. Die GUI soll später ersetzt werden.

Lediglich das Modul „Suchen“ wird in Anlage A detaillierter beschrieben. Die anderen Module „Warenkorb“ und „bezahlen“ könnten ähnlich aufgebaut sein. Allerdings soll das „Suchen“-Team, unabhängig von den anderen Modulen an der Umsetzung arbeiten können.

Notieren Sie wesentliche im Entwurf vorhandene und in der Vorlesung besprochene SWA-Prinzipien und -Muster. Erläutern Sie diese kurz und beschreiben die Ziele, die mit deren Einsatz verfolgt werden. Sie sollten den Entwurf verstanden haben, so dass Sie diesen selber umsetzen und anderen Entwickler*innen (oder den Dozenten) erklären können.

Sollten Sie Verbesserungsvorschläge haben, notieren Sie diese. Sollten Fragen offen bleiben, notieren Sie diese, überlegen sich eine mögliche Umsetzung und besprechen diese mit Ihren Kommiliton*innen und / oder Dozenten.

Prof. Dr.-Ing. R. Roosmann, M. Teuber

Aufgabe 1.3: Umsetzung des Software-Entwurfs aus Aufgabe 1.2

(8 Punkte, Gruppenarbeit, 2er oder 3er Gruppe)

Nachdem Sie sich in Aufgabe 1.2 Gedanken über die wesentlichen Prinzipien, Muster und Ziele des Software-Entwurfs zur Simple-Shopping-App gemacht und diese verstanden haben, geht es jetzt an die Umsetzung. Verwenden Sie dazu Java SE 11.0.

Die UI ist als Konsolenanwendung zu programmieren, bei der Nutzer*innen über eine Menüführung durch das Programm geführt werden. Bei jedem Programmstart wird der Nutzerin / dem Nutzer ein Warenkorb zugewiesen. Ist ein Warenkorb leer und der Nutzer meldet sich ab, soll der Warenkorb entfernt werden.

Die Daten der Anwendungen sollen in einer Datenbank persistiert werden. Nehmen Sie bspw. Apache Derby und greifen per JDBC auf Ihre Datenbank zu. Machen Sie sich initial Gedanken über das Datenbankschema, setzen dieses dann um und füllen die Datenbank mit sinnvollen Daten.

Die Interfaces im Package `de.hsos.swa.ssa.suche.acl` sollen gemockt werden. Dazu können Sie Mock-Implementierungen programmieren, die später durch Adapter ausgetauscht werden können, um die definierten Interfaces auf aktuell noch nicht vorhandene Interfaces des Moduls Warenkorb abzubilden.

Sollten Fragen zur Umsetzung bestehen, suchen Sie eine sinnvolle pragmatische Lösung, die im Idealfall einfach austauschbar ist.

Wichtig: der Software-Entwurf ist kein Dogma!! Der Entwurf ist eine theoretische Arbeit und Sie kennen den Spruch: in der Theorie sind Theorie und Praxis gleich, in der Praxis sind sie es nicht. Dokumentieren Sie Anpassungsvorschläge, die sich aus der Implementierung ergeben haben (evtl. direkt im Source-Code).

Aufgabe 1.4: Erweiterung des Entwurfs

(6 Punkte, Gruppenarbeit, 2er oder 3er Gruppe)

In dieser Aufgabe ist die erstellte Lösung aus Aufgabe 1.3 zu erweitern. Um Artikel suchen und finden zu können, soll eine Katalogverwaltung bereitgestellt werden, so dass Artikel gesucht, hinzugefügt, geändert oder entfernt werden können. Auf eine Nutzerverwaltung soll verzichtet werden.

Erstellen Sie einen Software-Designentwurf für dieses zusätzliche Modul namens „katalogVerwalten“. Sie können sich am Entwurf aus Aufgabe 1.2 orientieren. Wichtig: Sie definieren selber, was gemacht wird!! Der Designentwurf soll auf in der Vorlesung behandelten Prinzipien basieren und sinnvolle Muster verwenden. Verwenden Sie bspw. Visual Paradigm in der Community Edition als UML-Tool. Das Ergebnis ist zu dokumentieren.

Der eigene Design-Entwurf kann eigenständig oder als Erweiterung der Lösung zu Aufgabe 1.3 umgesetzt werden.

Als Vorgabe ist zu berücksichtigen, dass auf Ebene der JDBC Datenbank-Connection `AutoCommit` auf `false` gesetzt werden soll (`connection.setAutoCommit(false)`). Standardmäßig ist `AutoCommit=true` und entsprechend wird jedes einzelne SQL-Statement als eigene Transaktion aufgefasst, die direkt nach der Ausführung automatisch committed wird. Mehrere SQL-Statements in einer Transaktion sind nicht möglich. Da dies gewünscht ist, werden Sie die Transaktionssteuerung selber übernehmen.

Viel Erfolg!!