



SKS-5, WS 2018
Bernhard Löwenstein

Softwarekomponentensysteme

Übungsaufgaben

Übungsablauf

- Teams zu jeweils max. 2 Studierenden
- 8 aufeinander aufbauende Übungsaufgaben
- gemeinsame Code Reviews ausgewählter Aufgabenlösungen während Übungseinheiten
- Abgabegespräch mit jedem Team am Semesterende
- alle Sourcen in englischer Sprache verfassen

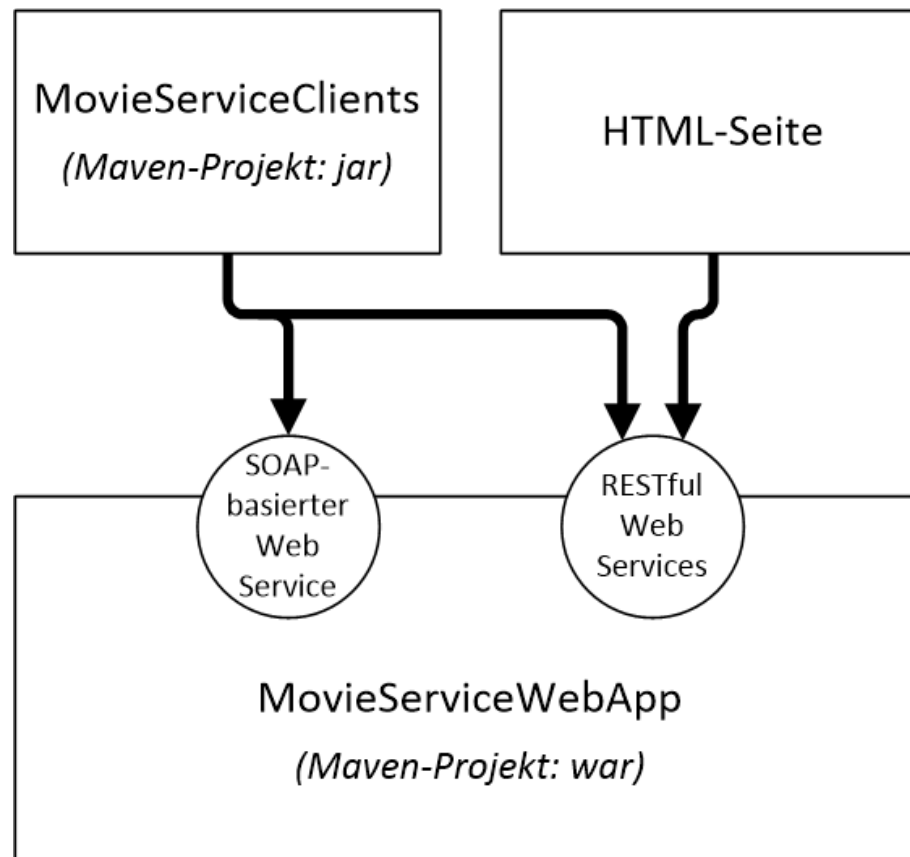
Technologiestack

- Betriebssystem
 - Windows 10
- Java
 - Java Standard Edition 8
 - Java Enterprise Edition 8
- Entwicklungswerkzeuge
 - Maven 3.5
 - Eclipse IDE for Java EE Developers Oxygen
- Applikationsserver
 - WildFly 14
- Datenbank
 - MySQL Community Server 5.7

Einführung (1)

- Es soll eine zentrale Filmverwaltung (Film ⇔ Schauspieler ⇔ Studio) in Form einer mehrschichtigen Java Enterprise-Applikation entwickelt werden.
- Über verschiedene Schnittstellen (siehe nächste Seite) lassen sich Daten importieren, manipulieren und abfragen.

Einführung (2)



Aufgabe 1

- Modelliere die notwendigen Entitäten mittels eines passenden UML-Diagrammtyps.
- **Artefakte:**
 - UML-Diagramm
- **Codereview am 23.11.2018**

Aufgabe 2

- Entwickle mit JPA die zuvor modellierten Entitäten, wobei die erforderlichen Datenbankobjekte automatisch angelegt werden sollen.
- Richte eine passende Data Source im Applikationsserver ein.
- **Artefakte:**
 - **Maven-Projekt "MovieServiceWebApp"**
 - **Konfigurationsdatei "standalone.xml"**
- **Codereview am 23.11.2018**

Aufgabe 3

- Entwickle mit JAX-WS und JAXB einen SOAP-basierten Web Service, der den Import von Filmen sowie die Suche nach Filmen über deren Titel (auch nur Teilen davon) ermöglicht.
- Ein Film darf nur dann importiert werden, wenn alle zugehörigen Schauspieler und das zugehörige Studio bereits in der Datenbank vorhanden sind.
- Die eigentliche Verarbeitung der zu importierenden Filme hat transaktional (entweder alle oder keines) in der EJB-basierten Businesslogik zu erfolgen.
- **Artefakte:**
 - **Maven-Projekt "MovieServiceWebApp"**
- **Codereview am 14.12.2018**

Aufgabe 4 (1)

- Entwickle mit Java ein Kommandozeilen-Werkzeug, das den Import von Filmen aus einer XML-Datei festgelegten Formats (siehe nächste Seite) ermöglicht.
- Rufe dazu den zuvor entwickelten SOAP-basierten Web Service auf.
- Generiere die erforderlichen Zugriffsklassen für den Web Service-Client mittels des JDK-Werkzeugs wsimport.
- **Artefakte:**
 - **Maven-Projekt "MovieServiceClients"**
- **Codereview am 18.12.2018**

Aufgabe 4 (2)

```
<movies>
  <movie title="..." description="..." genre="..." length="..."
    releaseyear="...">
    <studio name="..." countrycode="..." postcode="..." />
    <actors>
      <actor firstname="..." lastname="..." sex="..." birthdate="..." />
      <actor firstname="..." lastname="..." sex="..." birthdate="..." />
    </actors>
  </movie>
  ...
</movies>
```

Aufgabe 5

- Entwickle zum Verwalten der Schauspieler und der Studios mit JAX-RS und JSON-B jeweils einen RESTful Web Service, der die klassischen CRUD-Methoden bereitstellt und über den man sich auch jeweils alle Ressourcen zurückliefern lassen kann.
- **Artefakte:**
 - **Maven-Projekt "MovieServiceWebApp"**
- **Codereview am 18.12.2018**

Aufgabe 6

- Entwickle mit Java zum Anlegen von Schauspielern und Studios jeweils ein Kommandozeilen-Werkzeug.
- Rufe dazu die zuvor entwickelten RESTful Web Services auf.
- **Artefakte:**
 - **Maven-Projekt "MovieServiceClients"**
- **Codereview am 19.12.2018**

Aufgabe 7

- Erstelle eine HTML-Seite, die beim Laden alle Studios ausgibt und beim Anklicken eines Studios die zugehörigen Studiodelails in Form einer Message-Box ausgibt.
- Rufe dazu den zuvor entwickelten RESTful Web Service durch entsprechende AJAX-Requests auf.
- **Artefakte:**
 - HTML-Seite
- **Codereview am 19.12.2018**

Aufgabe 8

- Sichere die Filmverwaltung mittels Java EE Security ab, so dass man nur nach erfolgreicher Authentifizierung (mit Benutzername und Passwort) Zugriff auf die Web Services erhält.
- Je nachdem, ob eine Methode einen schreibenden oder lesenden Zugriff durchführt, soll sie mit der Rolle "MSWrite" oder "MSRead" ausgezeichnet werden.
- Alle Benutzer mit Schreibrechten sollen automatisch auch Leserechte haben.
- **Artefakte:**
 - **HTML-Seite**
 - **Maven-Projekt "MovieServiceClients"**
 - **Maven-Projekt "MovieServiceWebApp"**
 - **Konfigurationsdatei "standalone.xml"**
- **kein Codereview**

Leistungskriterien

- Vorlesung (50 %)
 - schriftliche 90-minütige Prüfung
- Übung (50 %)
 - Code Review von Aufgabenlösung während Übungseinheit (20 %)
 - mündliches Abgabegespräch um Qualität und Korrektheit von Gesamtlösung zu überprüfen (80 %)

Leistungsbeurteilung

1 = Sehr gut (≥ 88 %)
2 = Gut (≥ 75 %)
3 = Befriedigend (≥ 63 %)
4 = Genügend (≥ 50 %)
5 = Nicht genügend