Übungen zu „Einführung in Software Engineering“ (WS 19/20)
Prof. Dr. Barbara Paech, Astrid Rohmann, Anja Kleebaum
http://se.ifi.uni-heidelberg.de

Übungsblatt 6 (19.11.2019),
Klassendiagramm Movie Manager Modell, Movie Manager Programmierung, Vorbereitung Sequenzdiagramme

In dieser Übung:
- ✓ Erstellen Sie ein Klassendiagramm für den Modellcode des *Movie Managers.*
- ✓ Erkunden Sie den Code des *Movie Managers*
- ✓ Bereiten Sie Sequenzdiagramme vor.

Quellcode Movie Manager
Zur späteren Implementierung von neuen Funktionalitäten stellt Ihnen die Firma *MovieExperience* den Quellcode der *Movie Manager* App zur Verfügung. Die Implementierung der *Movie Manager* App basiert auf Java und Android. Die Implementierung **wird in der Technologievorlesung in dieser und der nächsten Woche** näher vorgestellt.

Auf diesem und den kommenden Übungsblättern werden Sie mit dem Quellcode der *Movie Manager* App arbeiten. Damit Sie die zugehörigen Übungsaufgaben möglichst gut bearbeiten können, ist es besonders wichtig, dass Sie sich ausführlich in die bestehende Implementierung einarbeiten. Die nachfolgenden Aufgaben unterstützen Sie bei der Einarbeitung.

Hinweis zur Aufgabe 6.1
Die nachfolgende Aufgabe 6.1 dient der Einarbeitung in den Quellcode des *Movie Managers*. Die Aufgabe vertieft die Grundlagen zu Klassendiagrammen und dient als Vorbereitung auf die Aufgaben der nächsten Übungsblätter.

Aufgabe 6.1: Analyse der bestehenden Movie Manager App (Teil 6): Klassendiagramm für die Datenklassen des Movie Managers

Präsenz: Nein	**Punkte:** 9	**Team:** Nein	**Testat**

In dieser Aufgabe erweitern Sie das Klassendiagramm des *Movie Managers* um die Datenklassen. Schauen Sie sich das **unvollständige Klassendiagramm** des *Movie Managers* im JIRA-Projekt an. Dabei sind die vorhandenen Klassen wie folgt eingefärbt: **grün** für Datenklassen, **rot** für Oberflächenklassen und **blau** für Klassen des Android-Frameworks und der Java Standardbibliothek, **gelb** für Hilfsklassen des Movie-Manager-Frameworks, die für den Speicher und reversible Operationen zuständig sind. Die Vorlage *06-MovieManager-ClassDiagram.png* (Datei zu finden in Moodle) enthält einen Ausschnitt des Klassendiagramms, der nur die Datenklassen und dazugehörige Klassen des Movie-Manager-Frameworks enthält.

Schauen Sie sich den Quelltext des *Movie Managers* an und ergänzen Sie die Datenklassen in der Vorlage passend zum Quelltext. Sie können das Klassendiagramm entweder per Hand ergänzen und dann einscannen oder Sie verwenden ein Zeichenwerkzeug ihrer Wahl, z.B. Paint, Visio, draw.io.

1) Zeichnen Sie alle Klassen aus dem Paket `moviemanager.data` (ohne die Klassen des Pakets `proxy`) in das Klassendiagramm ein und färben Sie die Klassen **grün** ein. Ergänzen Sie im Klassendiagramm für alle Klassen jeweils alle Attribute und Operationen. In Ihrem JIRA-Projekt finden Sie für die obigen Klassen jeweils Issues vom Typ „Class“. Als Arbeitserleichterung können Sie die Attribute und Operationen aus den „Class“-Issues kopieren und in Ihr Klassendiagramm einfügen.
2) Erkennen Sie aus dem Quellcode die Assoziationen und Vererbungsbeziehungen zwischen den Klassen sowie Implementierungsbeziehungen zwischen Klassen und Interfaces und zeichnen Sie diese in das Klassendiagramm ein. Benennen Sie die Assoziationen und ergänzen Sie die Multiplizitäten. Berücksichtigen Sie dabei insbesondere auch Einschränkungen, die in der Klasse `RuntimeStorage` in der Operation `setupAssociations()` sichergestellt werden. Beschreiben Sie in einem PDF-Dokument in 1-2 Sätzen pro Assoziation, Vererbung und Interface-Implementierung im Klassendiagramm wie die Klassen zusammenarbeiten und eine Begründung für Multiplizitäten.

Ergebnis:
Speichern Sie bitte das Ergebnis als .zip-Datei bis **Montag 25.11.2019 um 10.00 Uhr** in Moodle bestehend aus:

- 1x PNG-Datei des Klassendiagramms mit den ergänzten Klassen, Attributen, Operationen und Assoziationen.
- 1x PDF mit Beschreibung der Assoziationen und Vererbungsbeziehungen und Multiplizitäten.

Aufgabe 6.2: Analyse der bestehenden Movie Manager App (Teil 7): Erkundung des Codes

Präsenz: Ja	**Punkte:** 13	**Team:** Nein	**Testat**

In dieser Übung beschäftigen Sie sich näher mit dem Quellcode der Movie Manager App, indem Sie sich diesen genauer anschauen und Fragen dazu beantworten. Betrachten Sie dazu den Quellcode, den wir Ihnen über Confluence zur Verfügung gestellt haben. Halten Sie die Antworten zu den Fragen in einem PDF-Dokument fest.

Betrachten Sie zunächst die Struktur des Projekts im Ordner java. Die Pakete `de.associations`, `de.storage`, `de.util` und `de.wiki` sind nur für die Verwendung gedacht und benötigen im weiteren Verlauf der Übungen keinerlei Modifikationen. Für das Verständnis genügt es hier die `public` Methoden zu kennen.
Im Paket `de.moviemanager` finden Sie den Hauptteil der Movie Manager App.
Betrachten Sie das Paket `de.moviemanager.data`. Dort sehen Sie die Datenklassen für die App.
Beantworten Sie dazu folgende Fragen:

- Von Welcher Klasse erbt die Klasse `Movie`?
- Wofür ist diese abstrakte Klasse zuständig?
- Erben weitere Klassen von dieser abstrakten Klasse, wenn ja, welche?

Betrachten Sie das Manifest der App (`AndroidManifest.xml`).

- Wie wird im Manifest eine Aktivität markiert, die beim Start der App ausgeführt wird?
- Welche Aktivität ist auf diese Weise gekennzeichnet?
- Welche implementierten Lebenszyklus-Operationen dieser Aktivität werden beim Start der Movie-Manager App aufgerufen?
- Was geschieht in diesen Operationen?

Betrachten Sie das Paket `de.moviemanager.android`. Hier finden Sie Klassen für die Kommunikation zwischen Aktivitäten bzw. Fragmenten, bei der ein Ergebnis erwartet wird. Die

Klasse `ResultHandlingPlugin` übernimmt dabei die Verwaltung der Intents, der Requestcodes und der Ergebnisse, die nach dem Aufruf einer Aktivität von dieser zurückgegeben werden.

- Vergleichen Sie diesen ResultHandling-Mechanismus mit der von Android bereitgestellten Version von `startActivityForResult` ohne `ResultHandling`. Diskutieren Sie die Vor- und Nachteile in Form einer Diskussion von Rationale, wie in Vorlesung 6. Verwenden Sie als Kriterien die Einfachheit der Implementierung sowie die Erweiterbarkeit der Implementierung.

Betrachten Sie das Paket `de.moviemanager.core.storage`. Dort finden Sie Klassen, die für das Speichern der Daten des Movie Managers zuständig sind. Betrachten Sie insbesondere die Klasse `RuntimeStorage`, die in der App nur indirekt über die Klasse `RuntimeStorageAccess` verwendet wird. Betrachten Sie im Vergleich dazu auch den `Storage`, der im Pokemon Manager gegeben war.

- Welche Klassen der Movie Manager App werden in der Klasse `RuntimeStorage` verwendet?
- Vergleichen Sie die Verwendung von (reversiblen) Transaktionen mit der Verwendung des Storage im Pokemon Manager. Diskutieren Sie die Vor- und Nachteile in Form einer Diskussion von Rationale, wie in Vorlesung 6. Verwenden Sie als Kriterien die Einfachheit der Umsetzung sowie die Nutzerfreundlichkeit der damit möglichen UI.
- Was müsste man, unabhängig von der UI, tun, wenn man eine neue Datenklasse hinzufügt? (Hinweis: Im JavaDoc der Klasse `Movie` ist beschrieben, welche Klassen angepasst werden müssen, falls man die Klasse `Movie` ändern möchte)

Betrachten Sie das Paket `de.moviemanager.ui.detail`. Dort finden Sie die Klassen für die Detail-Ansichten und deren editierbaren Versionen. Vergleichen Sie die Aktivitäten `MovieDetailActivity` und `MovieDetailEditActivity`.

- Was fällt Ihnen bei den Attributen für die Darstellung der Attribute eines Films auf?
- Welchen Zweck haben diese in `MovieDetailEditActivity`?

Betrachten Sie die abstrakte Klasse `PortrayableRVAdapter` im Paket `de.moviemanager.ui.adapter`. Diese Klasse bildet den Adapter sowohl für den RecyclerView, der die Liste der Filme darstellt, als auch für den RecyclerView, der die Liste der SchauspielerInnen darstellt. Betrachten Sie nun die Klassen `MovieMasterFragment` und `PerformerMasterFragment` im Paket `de.moviemanager.ui.masterfragments`.

- Wie wird der `PortrayableRVAdapter` in diesen Klassen verwendet?

Ergebnis:

Bitte speichern Sie Ihr PDF-Dokument bis **Montag 25.11.2019 um 10.00 Uhr** in Moodle.

Aufgabe 6.3: Vorbereitung – Sequenzdiagramme

Präsenz: Nein	**Punkte:** 6	**Team:** Nein	

1) Bereiten Sie sich auf die nächsten Inhalte der Vorlesung vor, indem Sie sich die Konzepte und Notationen der Sequenzdiagramme erarbeiten. Verwenden Sie dazu die Materialien des Buches UML@Classroom. Die Materialien sind unter dem Link http://www.uml.ac.at/de/lernen abrufbar. Eine identische Version der Folien „Sequenzdiagramme“ mit gedrucktem Sprechertext finden Sie unter *Literatur – UML@Classroom* in **Dateien und Materialien** in Confluence.
   Lesen Sie alle Folien des Foliensatzes „Sequenzdiagramme“.

2) Halten Sie anschließend die Inhalte zu den folgenden 3 Punkten anhand der Informationen aus dem Foliensatz in einem PDF-Dokument fest:

- Was sind die wichtigsten Elemente der Sequenzdiagramme?
- Welche Arten der Kommunikation werden im Sequenzdiagramm unterschieden?
- Was ist der Unterschied zwischen einem Sequenzdiagramm und einem Kommunikationsdiagramm?

Ergebnis:

Speichern Sie bitte Ihr PDF-Dokument bis **Montag 25.11.2019 um 10.00 Uhr** in Moodle.