Übungsblatt 3 (27.10.2015)

Weboberflächen-Programmierung mit GWT, Black-Box-Tests, Code-Inspektion, Komponententests

In dieser Übung:

- ✓ Programmieren Sie eine Weboberfäche mit GWT.
- ✓ Üben Sie Black-Box-Tests durch Betrachtung von Äquivalenzklassen.
- ✓ Spezifizieren Sie Testfälle.
- ✓ Führen Sie eine Code-Inspektion durch.
- ✓ Üben Sie Komponententests (Unit Test) mit JUnit zu implementieren, auszuführen sowie zu protokollieren.

Vorbereitung für Aufgabe 3.1: Weboberflächen-Programmierung mit GWT
In der nachfolgenden Aufgabe 3.1 programmieren Sie eine Weboberfläche mit GWT. Um den Einstieg für Sie zu erleichtern, stellen wir ein Eclipse-Projekt bereit, das Sie für diese Aufgabe verwenden sollen. Laden Sie sich das Projekt **03-movies-web.zip** aus Moodle herunter und importieren Sie das Projekt in Ihr Eclipse.

Schauen Sie sich die Klassen `Movie`, `MovieUI` und `MovieManager` im Package `de.unihd.movies.client` im Ordner `src` an. Machen Sie sich mit den Klassen und Operationen der Klassen vertraut. Beginnen Sie erst danach mit der Bearbeitung der Aufgabe 3.1.

Aufgabe 3.1: Weboberflächen-Programmierung mit GWT

Präsenz: Ja	**Punkte:** 10	**Team:** Nein	**Projekt:** Nein	**Testat**

Programmieren Sie mit Hilfe von GWT eine einfache tabellarische Oberfläche zum Anzeigen von Filmdaten in einer Klasse `MovieUI`. Orientieren Sie sich dabei an der Darstellung in Abbildung 1.

Verwenden Sie die Elemente `CellTable` und `Column` aus dem Paket `com.google.gwt.user.cellview.client` sowie das Element `ListDataProvider` aus dem Paket `com.google.gwt.view.client` der GWT-Widgetbibliothek.

Beim Klicken auf die Tabellenspaltenüberschrift werden die Einträge anhand des Attributes der Spalte sortiert. Verwenden Sie dafür den `ListHandler` aus dem Paket `com.google.gwt.user.cellview.client.ColumnSortEvent`.

Die Speicherung der Filmdaten soll mit einer `java.util.ArrayList` in der Klasse `MovieManager` umgesetzt werden. Die Anwendung wird durch das Erzeugen einer `MovieUI` Instanz in der `onModuleLoad()` Operation der Klasse `MovieManager` gestartet.

▲ ID	Name	Time	Language	Description	Place
2	Another Movie	120	Deutsch	really good	behind ...
3	Star Wars	123	Englisch	the one and only	tunesia
4	Terminator	107	Deutsch	Arnold is back	new york

Abbildung 1: Einfache Movie Manager UI mit GWT

Hinweis: *Es müssen keine Interaktionsmöglichkeiten zur Eingabe von neuen Filmen und Filmdaten programmiert werden. Testen Sie die Anzeige, indem Sie* `Movie` *Instanzen in der Klasse* `MovieManager` *erzeugen und diese in der* `ArrayList` *des* `MovieManager`*s abspeichern.*

Ergebnis:

Bitte speichern Sie das Projekt in Ihrem Git-Repository (mit der „Share Project“ Funktion und anschließendem „Add-to-Index“ und „Commit“)! Speichern Sie bitte Ihr exportiertes Eclipse-Projekt in gepackter Form als .zip bis **Montag 02.11.2015 um 10.00 Uhr** in Moodle.

Aufgabe 3.2: Black-Box-Test: Äquivalenzklassen und Testfallspezifikation

Präsenz: Nein	**Punkte:** 12	**Team:** Ja(2)	**Projekt:** Nein	**Testat**

Betrachten Sie die folgende Beschreibung eines Algorithmus zur Berechnung der Gesamtbewertung (overallRating) für einen Film (class `Movie`) mit seinen darin mitwirkenden SchauspielerInnen (class `Performer`):

public int overallRating ()

Die **Gesamtbewertung** eines Films berechnet sich aus der **Einzelbewertung** eines Films (int ratingMovie), zu welchem das **Maximum der Bewertungen der SchauspielerInnen** (ArrayList<Integer> ratingPerformer) des Films **addiert** wird.

Eine Einzelbewertung eines Films bzw. eines/einer SchauspielerIn liegt zwischen 0 und 5.
Die Gesamtbewertung liegt zwischen 0 und 10.

1. Geben Sie die **gültigen und ungültigen Äquivalenzklassen für ratingMovie und ratingPerformer** an. Beachten Sie, dass Ihre Äquivalenzklassen insbesondere die Grenzwerte und die wichtigsten Fälle bei der Maximumsuche abdecken. Beschreiben Sie alle **Testfälle**, die durch **sinnvolle Kombinationen der Äquivalenzklassen** entstehen, in einer Tabelle.
2. Wählen Sie aus der Tabelle Testfälle aus, die zusammen eine **minimale Testfallmenge** („jede gültige Äquivalenzlasse in mindestens einer Kombination“) bilden.
3. overallRating sei eine Operation der Klasse Filme, die als Attribute die Einzelbwertung des Filmes, die Liste der SchauspielerInnenbewertungen und die Gesamtbewertung hat. Die Tabelle `03-Testcases.xlsx` (Datei zu finden in Moodle) beinhaltet bereits die logische und konkrete Spezifikation eines Testfalls zur Operation `overallRating`. Wählen Sie mindestens 2 Testfälle aus 3.2.1. aus (verschieden von dem vorgegebenen) und spezifizieren Sie diese als l**ogische Testfälle** entsprechend dem vorgegebenen Schema.
4. Spezifizieren Sie danach die **konkreten Testfälle**, indem Sie für die logischen Testfälle konkrete Werte angeben.

Ergebnis:
Speichern Sie bitte eine .zip-Datei bis **Montag 02.11.2015 um 10.00 Uhr** in Moodle bestehend aus:

- PDF mit der Angabe der Äquivalenzklassen
- Ausgefüllte Testcase-Tabelle **03-test.xlsx** mit spezifizierten Testfällen

Vorbereitung für Aufgabe 3.3: Code-Inspektion
In der nachfolgenden Aufgabe 3.3 prüfen Sie die Anwendung „MovieManager“ durch Inspektion auf Fehler. Dazu benötigen Sie Quelltext, der als Inspektionsobjekt dient. Laden Sie sich das Projekt **03-moviemanager.zip** aus Moodle herunter und importieren Sie das Projekt in Eclipse.

Schauen Sie sich die Klassen `Movie`, `Performer` und `Gender` im Package `org.movie` im

Aufgabe 3.3: Code-Inspektion

Präsenz: Nein	**Punkte:** 5	**Team:** Ja(2)	**Projekt:** Nein	

1. Inspizieren Sie (ad-hoc) den Code der Klassen `Gender`, `Movie` und `Performer` (siehe Vorbereitung) auf Fehler. Dokumentieren Sie **alle gefundenen Fehler** in einer Textdatei. Verwenden Sie zur Fehlerdokumentation die folgende Tabelle (vgl. Foliensatz 3, Folie 41) und orientieren Sie sich bei Ihrer Lösung an dem gegebenen Beispiel.

EntdeckerIn	Datum	betroffenes Produkt	ausgeführte QS-Aktivität	Beschreibung	Fehler klasse	Status
Marcus Seiler	26.10.2015	MovieManager	Code-Inspektion	Movies werden in der Klasse MovieManager nicht gespeichert. Dies führt zum Verlust aller neu erstellten Movies. Alle neu erstellten Movies müssen gespeichert werden.	2	behoben

2. Beheben Sie die gefunden Fehler.

Hinweis: *Insgesamt sind 4 Fehler im Quellcode versteckt.*

Ergebnis:
Speichern Sie bitte das Ergebnis als .zip-Datei bis **Montag 02.11.2015 um 10.00 Uhr** in Moodle bestehend aus:

- 1x PDF-Datei mit den gefundenen Fehlern
- 1x Eclipse-Projekt **03-moviemanager.zip** mit der entsprechenden Fehlerbehebung

Testumgebung:
Für die nachfolgende **Aufgabe 3.4 „Komponententests“** benötigen Sie eine Testumgebung. Diese Testumgebung liefert Ihnen das Testobjekt sowie einen Testrahmen. Ein Testrahmen besteht aus Testtreibern (Testfällen) sowie einer Laufzeitumgebung (Stubs) (vgl. 3. Foliensatz).

Eine bereits implementierte und lauffähige Testumgebung für die Aufgabe 3.4 finden Sie im dem Projekt aus Aufgabe 3.3 (03-moviemanager.zip), d.h. Sie sollen Ihr Projekt aus Aufgabe 3.3 weiterverwenden.

JUnit ist in dieser Testumgebung **bereits aktiviert**. Die Testumgebung besteht aus:

- **Testobjekt:** `org.movie.Movie` (im Ordner `src`).
- **Testtreiber:** `org.movie.test.MovieOverallRatingTest` (im Ordner `test`).
- **Laufzeitumgebung:** Alle anderen notwendigen Klassen im Ordner `src`.

Aufgabe 3.4: Komponententest (Unit Test)

Präsenz: Nein	**Punkte:** 8	**Team:** Ja(2)	**Projekt:** Nein	

In dieser Aufgabe testen Sie die Operation `overallRating` der Klasse `Movie`. Die Operation kennen Sie bereits aus Aufgabe 3.2 sowie aus Aufgabe 3.3.

Für die Operation `overallRating` ist bereits ein Testfall mit JUnit implementiert worden (siehe Klasse `MovieOverallRatingTest` im Package `org.movie.test`). Machen Sie sich mit dem Testfall vertraut und führen Sie ihn anschließend in JUnit (Run As → JUnit Test) aus.

1. Testen Sie die Operation `overallRating` der Klasse `Movie`.
   Implementieren Sie dafür die **2 von Ihnen definierten konkrete Testfälle aus Aufgabe 3.2** in der Klasse `MovieOverallRatingTest`. Geben Sie den Testfallklassen sprechende Namen, d.h. benennen Sie Ihre Operationen entsprechend, z.B. `testOverallRatingWithEmptyListOfPerformers()`.

2. Führen Sie die 2 Testfälle aus, d.h. Ausführen der JUnit Testfälle in Eclipse, und protokollieren Sie die Ergebnisse in der Excel-Arbeitsmappe `TestProtocols` der Tabelle `03-Testcases.xlsx`. aus Aufgabe 3.2.

3. Falls Sie die Aufgabe 3.3 nicht oder nur unvollständig bearbeitet haben, können bei der Ausführung Ihrer implementierten Testfälle JUnit-Tests fehlschlagen. In diesem Fall protokollieren Sie die Fehlermeldung.

Ergebnis:

Bitte speichern Sie das Projekt in Ihrem Git-Repository (mit der „Share Project“ Funktion und anschließendem „Add-to-Index“ und „Commit“)!
Speichern Sie bitte eine .zip-Datei bis **Montag 02.11.2015 um 10.00 Uhr** in Moodle bestehend aus:

- 1x Testcase-Tabelle **03-test.xlsx** mit spezifizierten und protokollieren Testfällen
- 1x Eclipse-Projekt **03-unit-moviemanager.zip** mit den implementierten Testfällen

Aufgabe 3.5: Vorbereitung Test

Präsenz: Nein	**Punkte:** 5	**Team:** Nein	**Projekt:** Nein	

Bereiten Sie sich auf die nächsten Inhalte der Vorlesung vor indem Sie die Abschnitte **19.5 „Der Black-Box-Test“** (außer 19.5.3. und 19.5.4) bis einschließlich **19.6 „Der Glass-Box-Test“** aus dem Kapitel **19 „Programmtest“** im Software Engineering Buch von Ludewig und Lichter lesen. Schauen Sie sich insbesondere die Technik des Glass-Box-Tests mit den unterschiedlichen Überdeckungen an, um sie dann in der Vorlesung anwenden zu können.

Tragen Sie anschließend die Inhalte zu den folgenden 3 Punkten in das Trello Board „Vorbereitung Test“ des Teams ISW ein:

- Wichtige Begriffe (und ihre Definitionen) im Umfeld des Black-Box- und des Glass-Box-Tests, insbesondere die unterschiedlichen Ziele
- Techniken zur Testfallauswahl beim Black-Box-Test
- Überdeckungskriterien für den Glass-Box-Test

Für jeden der 3 Punkte ist eine eigene Liste vorhanden, in die Sie die Inhalte eintragen können.
Die Inhalte des Boards „Test“ werden in der Vorlesung besprochen (wenn nichts drin ist, wird auch nichts besprochen).

Ergebnis:

Tragen Sie bitte Ihre Inhalte bis **Montag 02.11.2015 um 10.00 Uhr** in das Trello Board ein.