**Pflichtenheft**

**Version:** 1 .0

**Datum:** 06.03.2021

DOKUMENTVERSIONEN

|  |  |  |  |
| --- | --- | --- | --- |
| Versionsnr. | Datum | Autor | Änderungsgrund / Bemerkungen |
| 1.0 | 06.03.2021 | Gruppe G | Ersterstellung |
| 0.2 |  |  |  |
|  |  |  |  |
|  |  |  |  |

INHALT

[DOKUMENTVERSIONEN 1](#_Toc536202136)

[INHALT 2](#_Toc536202137)

[1. Einleitung 3](#_Toc536202138)

[1.1 Allgemeines 3](#_Toc536202139)

[1.1.1 Ziel und Zweck dieses Dokuments 3](#_Toc536202140)

[1.1.2 Projektbezug 3](#_Toc536202141)

[1.1.3 Abkürzungen 3](#_Toc536202142)

[1.1.4 Ablage, Gültigkeit und Bezüge zu anderen Dokumenten 3](#_Toc536202143)

[1.2 Verteiler und Freigabe 3](#_Toc536202144)

[1.2.1 Verteiler für dieses Lastenheft 3](#_Toc536202145)

[1.3 Reviewvermerke und Meeting-Protokolle 3](#_Toc536202146)

[1.3.1 Erstes bis n-tes Review 3](#_Toc536202147)

[2. Konzept und Rahmenbedingungen 4](#_Toc536202148)

[2.1 Benutzer / Zielgruppe 4](#_Toc536202149)

[2.2 Ziele des Anbieters 4](#_Toc536202150)

[2.3 Ziele und Nutzen des Anwenders 4](#_Toc536202151)

[2.4 Systemvoraussetzungen 4](#_Toc536202152)

[2.5 Ressourcen 4](#_Toc536202153)

[2.6 Übersicht der Meilensteine 4](#_Toc536202154)

[3. Anforderungsbeschreibung 5](#_Toc536202155)

[3.1 1. Anforderung 5](#_Toc536202156)

[3.1.1 Beschreibung 5](#_Toc536202157)

[3.1.2 Wechselwirkungen 5](#_Toc536202158)

[3.1.3 Risiken 5](#_Toc536202159)

[3.1.4 Testhinweise 5](#_Toc536202160)

[3.1.5 Vergleich mit bestehenden Lösungen 5](#_Toc536202161)

[3.1.6 Schätzung des Aufwands 5](#_Toc536202162)

[4. Genehmigung 6](#_Toc536202163)

[5. Anhang 7](#_Toc536202164)

# Einleitung

## Allgemeines

### Ziel und Zweck dieses Dokuments

Ziel dieses Pflichtenhefts ist es die technischen Umsetzungen an dem vorliegenden Refactoring-Projekt gemäß der im Lastenheft formulierten Anforderungen des AG zu dokumentieren.

Es ist Basis aller weiteren vertraglichen Vereinbarungen, insbesondere für die Definition der Abgabe-Artefakte mit entsprechenden Meilensteinen.

### Projektbezug

### Das Refactoring-Projekt entspricht dem Code-Beispiel aus dem Buch von Martin Fowler. Refactoring: Improving the Design of Existing Code, Addison-Wesley, 1999, 1. Auflage.

### Ablage, Gültigkeit und Bezüge zu anderen Dokumenten

Dieses Pflichtenheft beschreibt die vom AG genannten Anforderungen aus dem Lastenheft.

Weitere Dokumentationen finden sich auf dem Github-Repository. Es handelt sich hierbei um den überarbeiteten Quellcode inklusive Tests, das Benutzerhandbuch, sowie Lasten- und Plichtenheft.

## Verteiler und Freigabe

### Verteiler für dieses Pflichtenheft

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rolle | Name | Telefon | E-Mail | Bemerkungen |
| Projektleiter | Kevin Heid | 015733243807 | Kevin-neupotz@web.de |  |
| Kunde, AG | Dr. Marc Schanne | 015771750265 | marc@schanne.org |  |
| Projektteam, AN | Kevin Heid  Emelie Heß  Samuel Braun  Tim Schlegel | 015733243807  017647604866  017647604867  017647604865 | gruppeG@gmail.com |  |
|  |  |  |  |  |

## Reviewvermerke und Meeting-Protokolle

### Erstes bis n-tes Review

Ihr Text

# Konzept und Rahmenbedingungen

## Benutzer / Zielgruppe

Videothek

## Ziele des Anbieters

Ihr Text

## Ziele und Nutzen des Anwenders

Ihr Text

## Systemvoraussetzungen

Ihr Text

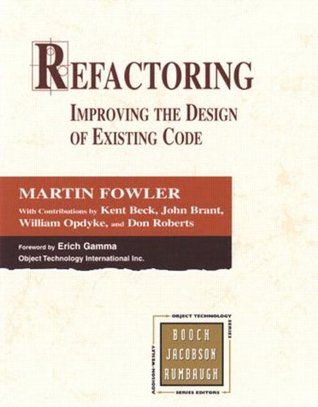
## Ressourcen

Ihr Text

## Übersicht der Meilensteine

|  |  |
| --- | --- |
| **Vorbereitungsphase abschließen 12:45** | |
| Abnahme Pflichtenheft |  |
| Revision Pflichtenheft |  |
| **Implementierung und Test abschließen 13:30** | |
| Produktentwurf |  |
| Implementierung/Refactoring |  |
| Test |  |
| **Einführung 14:30** | |
| Benutzerhandbuch |  |
| **Finales Release** | 06.03.2021 14:30 |

# Anforderungsbeschreibung

Die Anforderungen dieses Refactoring-Projekts betreffen ein Filmverleih-Projekt wie es Martin Fowler in seinem Buch Refactoring: Improving the Design of existing Code aus dem Jahr 1999 als Beispiel einführt.

Die Anforderungen sind aus Kundensicht und für das anstehende Refactoring muss der AN diese sinnvoll verfeinern und in einem Pflichtenheft die gesamte Entwicklung mit Meilensteinden planen.

## 1. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 101 | **Nichttechnischer Titel** | | Projektplanung mit Meilensteinen und Definition von Artefakten | | |
| **Quellen** | Lastenheft | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

Anforderung 101 ist das Verfassen des Pflichtenhefts, sowie die vollständige Planung des Projekts inklusive der Meilensteinplanung.

### Wechselwirkungen

Die entstehenden Artefakte inerhalb des Refactoring-Projekts erfordern eine ausführliche Projektplanung, mithilfe des Pflichtenhefts und der Meilensteinplanung.

### Risiken

Eine umfangreiche Dokumentation der Anforderungen und zu implementierenden Features ist für ein einwandfreies Ergebnis verpflichten, um Missverständnisse zu vermeiden und einen reibungslosen Ablauf zu gewährleisten.

### Schätzung des Aufwands

Die Projektumsetzung erfolgt in einem Zeitrahmen von 4 Stunden mit insgesamt 3 Meilensteinen, deren Zeitaufwand jeweils zwischen 1,5 und 1 Stunde liegen. Die Projektplanung nimmt hierbei den größten Anteil in Anspruch.

## 2. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.1 | **Nichttechnischer Titel** | | State of The Art | | |
| **Quellen** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

Der vorleigende Code ist bereits in einer alten Java-Version entstanden. Inzwischen gibt es Modernisierungsmöglichkeiten. Hierzu zählt das Ersetzen des Prinzips von Vector und Enumeration. Dies betrifft die „Customer“-Klasse.  
Die Vektor-Variable muss hierzu durch eine ArrayList<Rental> ersetzt werden. Daraufhin muss in der   
„addRental(Rental arg)“-Methode .addElement(arg) durch .add(arg) ersetzt werden.  
Abschließend kann die while-Schleife durch eine for-each-Schleife ersetzt werden. Dies spart einige Zeilen Code.

### Testhinweise

Ein Testdurchlauf der JUnit4-Tests muss nach den Änderungen durchgeführt werden.

**3.2.3 Schätzung des Aufwands**  
Umsetzungsaufwand: 3 min, Testaufwand: 2 min, Gesamt: 5 min

## 3. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.2 | **Nichttechnischer Titel** | | Refactoring anhand von Extract Method | | |
| **Quellen** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

In diesem Schritt soll die Refactoring-Methode Extract Method verwendet werden, um die Methode *statement()* aus der Klasse *Customer* zu vereinfachen. Am besten eignet sich für diesen Schritt das Switch-Statement.Es soll aus der Methode *Statement()* der Teil dees Switch-Statements kopiert werden und dann gelöscht und mit *thisAmount = amountFor(each*) ersetzt werden*.* Jetzt soll eine neue Methode *amountFor()* geschrieben werden, die das Switch-Statement ersetzen soll, in die der zuvor kopierte Code eingefügt werden muss. Danach werden die Variablennamen in der neuen Methode gändert: *each -> aRental*

*thisAmount -> result*

Somit ist diese Refactoring Methode abgeschlossen.

### Risiken

Es ist wichtig, bei dieser Methode darauf zu achten, nichts auszulassen, also auch alle Variablen zu verändern, um beim Complilieren nicht auf Fehler zu stoßen und den Code schlussendlich zu verschlechtern.

### Schätzung des Aufwands

Für die Umsetzung dieser Methode wird ein Aufwand von circa 6 Minuten eingeplant.

## 4. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.3 | **Nichttechnischer Titel** | | Refactoring anhand von Move Method | | |
| **Quellen** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

Für den nächsten Refactoring-Schritt wird die Methode Move Method verwendet. Hierbei soll die Methode *amountFor()* aus der Klasse *Customer* in die Klasse *Rental* geschrieben werden, da sie keine Informationen aus der Klasse *Customer* verwendet sondern aus der Klasse *Rental*. Hierfür muss der Code der Methode *amountFor()* kopiert werden und in die Klasse *Rental* als eine neue Methode mit angepasstem Namen *getCharge()* eingefügt werden. In der Methode statement() aus der Klasse Customer muss nun noch der Ausdruck *thisAmount = amountFor(each)*

zu *thisAmount = each.getCharge()*

geändert werden.

### Schätzung des Aufwands

Für die Umsetzung dieser Methode wird ein Aufwand von circa 5 Minuten eingeplant.

## 5. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.4 | **Nichttechnischer Titel** | | Refactoring anhand von Replace Temp with Query | | |
| **Quellen** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

Im nächsten Schritt soll mit der Refactoring-Methode „Replace Temp with Query“ gearbeitet werden. In der Klasse Customer sollen in der Methode statement() die Variablen totalAmount und frequentRenterPoints gelöscht werden.   
Es sollen zwei neue private Methoden getTotalCharge() und getTotalFrequentRenterPoints() erstellt werden, die die Variablen ersetzen. Somit werden überall im Quellcode die beiden Variablen durch die jeweiligen Methodenaufrufe ersetzt. Weiterhin werden die Methoden getCharge() und frequentRenterPoints() von der Klasse Rental in die Klasse Movie verschoben.

### Schätzung des Aufwands

Für die Umsetzung dieser Methode wird ein Aufwand von circa 8 Minuten eingeplant.

### Risiken

Da die Variablen durch Methoden ersetzt werden, die die Variablen berechnen und zurückliefern, muss gewährleistet sein dass die Berechnung richtig aufgestellt wird.

## 6. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.5 | **Nichttechnischer Titel** | | Refactoring anhand von Replace Type Code with State/Strategy | | |
| **Quellen** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | muss |

### Beschreibung

### In diesem Schritt wird die Refactoring-Methode „Replace Type Code with State/Strategy“ verwendet. In der Klasse Movie soll \_priceCode = priceCode; mit der Setter-Methode setPriceCode(priceCode); ausgetauscht werden. Dann soll eine neue abstrakte state-Klasse für den type code (Price) und Subklassen für das state-Objekt (ChildrensPrice, NewReleasePreice, RegularPrice) erstellt werden. Als nächstes soll ein abstract-query in der Superklasse erstellt werden, um den type code zurückzugeben und ein neues Feld in der alten Klasse für das state-Objekt erzeugt werden.

### Schätzung des Aufwands

Für die Umsetzung dieser Methode wird ein Aufwand von circa 8 Minuten eingeplant.

## 3.7 7. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.6 | **Nichttechnischer Titel** | | Refactoring anhand von Replace Switch with Polymorphism | | |
| **Quelle** | Lastenheft, Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | kann |

### 3.7.1 Beschreibung

### Um das switch-case-Statement mit Polymorphismus zu ersetzen, müssen zunächst einige Änderungen vorgenommen werden. Die Methode „getCharge(int daysRented)“ aus der Klasse „Movie“ muss in die Klasse „Price“ verlegt werden. Hierzu muss eine neue Methode „getCharge(int daysRented)“ in der Klasse „Movie“ eingeführt werden. Diese Methode greift auf die „Price“-Instanz aus der Variable in „Movie“ zu und führt dort die alte Methode auf, welche vorab von „Movie“ nach „Price“ verlegt wurde. Nachfolgend muss die der Polymorphismus genutzt werden. Hierzu muss das switch-case-Statement aus der „getCharge()“-Methode aus „Price“ auf die drei Unterklassen „RegularPrice“, „ChildrendsPrice“, „NewReleasePrice“ aufgeteilt werden. In die „Price“-Klasse muss daraufhin die „getCharge()“-Methode als abstract deklariert werden.

### 3.7.2 Risiken

Es kann ein Fehler bei der Aufteilung des switch-case-Statements entstehen. Dies muss durch sorgfältiges Arbeiten und Testläufe verhindert werden.

### 3.7.3 Testhinweise

Ein Testdurchlauf der JUnit4-Tests muss nach den Änderungen durchgeführt werden.

### 3.7.4 Schätzung des Aufwands

Umsetzungsaufwand: 5 min, Testaufwand: 2 min, Gesamt: 7 min

## 3.8 8. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.7 | **Nichttechnischer Titel** | | Werkzeugumgebung | | |
| **Quelle** |  | | **Verweise** |  | **Priorität** | soll |

### 3.8.1 Beschreibung

Als geeignetes Programmiertool wird die Entwicklungsumgebung IntelliJ verwendet und für die gemeinsame Arbeit, sowie Versionskontrolle wird Github verwendet.

## 9. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.8 | **Nichttechnischer Titel** | | Funktionale Erweiterungen | | |
| **Quelle** | Präsentation Martin Fowler | | **Verweise** |  | **Priorität** | kann |

### Beschreibung

Die Abrechnung soll um eine HTML-Ausgabe erweitert werden. Zum Testen werden die zuerst die Tests aus den Folien von Martin Fowler verwendet und falls benötigt im Nachgang ergänzt. Hierbei wird nach dem Prinzip Test Driven Development gearbeitet. *Wird in einer nächsten Versionierung ergänzt.*

## 9. Anforderung

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr. / ID** | 102.9 | **Nichttechnischer Titel** | | Form Template Method | | |
| **Quelle** |  | | **Verweise** |  | **Priorität** | muss |

### Beschreibung Um Form Template Method durchzusetzen, muss zunächst jeweils ein Statement Strategy für die Mehtode „statement()“ und „htmlStatement()“. Dies bedeutet es muss eine „TestStatement“ und „HtmlStatement“ Klasse erstellt werden, welche jeweils die Methoden „value(Customer aCustomer)“ beinhalten, die einen String zurückggeben, der die Aussgabe darstellt. Daraufhin müssen wieder einzelne Methoden aus der „value()“-Methode extrahiert werden (Extarct Methods). Anschließend müssen die einzelnen Methoden hochgezogen werden. Das heißt die Super-Klasse „Statement“ beinhaltet die „value(Customer aCustomer)“, die die unterschiedlichen abstrakten Methoden, die in den SubKlassen definiert werden, aufgerufen werden.

### Risiken Es können sich im Refactoring Fehler einschleichen. Aus diesem Grund muss geprüft werden, ob die TestCases vollständig sind (Test-Driven-Development).

**3.9.3 Testhinweise**

Ein Testdurchlauf der JUnit4-Tests muss nach den Änderungen durchgeführt werden.

**3.9.4 Schätzung des Aufwands**  
Umsetzungsaufwand: 10 min, Testaufwand: 5 min, Gesamt: 15 min

Die Genehmigung erfolgt...

|  |  |
| --- | --- |
| Datum: | 06.03.2021 |
| Unterschrift Auftraggeber: | M.Schanne |
| Unterschrift Projektleiter: | K.Heid |
| Weitere Unterschriften: | T.Schlegel, S.Braun, E.Heß |

# Anhang

Ihr Text



Diese Vorlage wurde erstellt von:  
vorlage-kostenlos.de

Mehr Informationen auf [www.sevdesk.de](https://sevdesk.de/?utm_source=template&utm_medium=referral&utm_campaign=pflichtenheft-vk)

[](https://sevdesk.de/?utm_source=template&utm_medium=referral&utm_campaign=pflichtenheft-vk)