Hochschule
Kaiserslautern
University of
Applied Sciences

PENTASYS
Unser Maßstab ist der Mensch

Hochschule Kaiserslautern
- Fachbereich Informatik und Microsystemtechnik -

Der Titel der Arbeit

Abschlussarbeit zur Erlangung des akademischen Grades
Bachelor of Science (B.Sc.)

vorgelegt von

Vorname Nachname
12345

Betreuer Hochschule:	Prof. Dr.
Betreuer PENTASYS:	B. Sc.

Abstract

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

> Dies ist ein Zitat.

verstand, scheinen nun doch vorueber zu Dies ist der Text sein.
siehe: http://janeden.net/die-praeambel

Inhaltsverzeichnis

Abbildungsverzeichnis **v**

Tabellenverzeichnis **vi**

1 Einführung **1**

2 Grundlagen **2**

3 Was ist Reactive Programming **3**

3.1 Reactive Programming - Ein neues Programmierparadigma? 3

3.1.1 Was versteht man unter Programmierparadigmen 3

3.1.2 Vergleich: Reactive Programming und Objekt orientierte Programmierung 3

3.2 Für was steht das Reactive im Kontext der Programmierung 3

3.2.1 Differenzierung zwischen Reactive Proramming und Reactive Systems . . 4

3.3 Reactive Programming vs. Functional Reactive Programming 4

3.4 Überblick über bekannte Frameworks und ihre Eigenschaften 4

3.4.1 Reactivex.io 4

3.4.2 Allgemeine Übersicht 4

3.4.3 Übersicht spezielle für die Entwicklung mit Java 4

3.5 Testen von reaktivem Code mit dem JUnit Framework 4

4 Einführung in Reactive Programming mit RxJava **5**

4.1 Wie funktioniert Reactive Programming 5

4.1.1 Syncronität 5

4.1.2 Parallelisierung 5

4.1.3 Push vs. Pull 5

4.2 Rx.Observable 5

4.3 Rx.Observer 5

4.3.1 Rx.Subscriber 6

4.4 Operationen und Transformationen 6

4.4.1 Exkursion: Streams API Java 8 6

4.4.2 Operation filter() 6

4.4.3 Transformation map() 6
4.4.4 Transformation flatMap() 6
4.4.5 Operation merge() 6
4.4.6 Operation zip() 6

Literaturverzeichnis **I**

Abbildungsverzeichnis

Tabellenverzeichnis

Listingverzeichnis

Kapitel 1

Einführung

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Kapitel 2

Grundlagen

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Kapitel 3

Was ist Reactive Programming

Historische Einführung. Wann und von wem wurde das erste Mal von RP gesprochen, wie war die weitere Entwicklung? Kapitelbeschreibung: Was folgt.

3.1 Reactive Programming - Ein neues Programmierparadigma?

Wie gliedert man einen Programmierstil ein? [1]

3.1.1 Was versteht man unter Programmierparadigmen

Erklärung was sind Paradigmen und wieso werden sie definiert.

3.1.2 Vergleich: Reactive Programming und Objekt orientierte Programmierung

Eventuell bessere mit Funktionaler Programmierung zu vergleichen. Muss noch genauer betrachtet werden. Soll die Unterschiede zu den gängigen, bekannten Methoden aufzeigen.

3.2 Für was steht das Reactive im Kontext der Programmierung

Was bedeutet der Begriff Reactive im eigentlichen Sinne, wie genau findet man dass in der Programmierung wieder?

[1] [Bai13]

3.2.1 Differenzierung zwischen Reactive Proramming und Reactive Systems

Das Manifest bezieht sich eigentlich auf die Systeme. Wo kommt nun RP in Spiel?

3.3 Reactive Programming vs. Functional Reactive Programming

FRP findet nun mal in Funktionalen Programmiersprachen statt. Java is jedoch OO und mit Java 8 und dem Rx Frameworks werden funktionale Eigenschaften in der OO Sprache eingebracht. Unterschiede müssen noch genau belegt[2] werden.

3.4 Überblick über bekannte Frameworks und ihre Eigenschaften

Überblick quer über die gängigen Programmiersprachen.

3.4.1 Reactivex.io

Kurze Erläuterung zu der Entstehung von Reactive Extensions

3.4.2 Allgemeine Übersicht

Rx Frameworks zu den jeweiligen Sprachen. Frameworks wie z.B. Akka[3].

3.4.3 Übersicht spezielle für die Entwicklung mit Java

RxJava. Reactive Streams Konvention. Java 9 Api Änderung bzgl. Reactive Streams.

Framework für JavaFX - RxJavaFX

Einführung und Eigenschaften erläutern

3.5 Testen von reaktivem Code mit dem JUnit Framework

Noch nichts genaues. Muss noch geschaut werden wie die Funktionalität von JUnit RP abdeckt.

[2] [Loh16]
[3] [Kar16]

Kapitel 4

Einführung in Reactive Programming mit RxJava

Beschreibung wieso RxJava. Beschreibung was beschrieben wird.

4.1 Wie funktioniert Reactive Programming

Einleitung zum Aufbau: Klassenübersicht des Frameworks mit Erklärung.

4.1.1 Syncronität

Sync vs. Async - was bringt RP in dieser Hinsicht

4.1.2 Parallelisierung

Concurrency vs. Parallelism - was tritt wie wann auf bzw. kann wie wann angewandt werden

4.1.3 Push vs. Pull

Wichtigster Unteschied. Observable als Gegenpart zu Interable - somit Push vs. Pull Vergleich.

4.2 Rx.Observable

Interface Übersicht. Nutzen und Anwendung anhand von Beispiel. Hot vs. Cold

4.3 Rx.Observer

Was kann Observer -> Interface Übersicht

4.3.1 Rx.Subscriber

Was ist speziell am Subscriber -> Interface Übersicht

4.4 Operationen und Transformationen

Erläuterung von den Stadien der Operation von Beginn über Mitte bis Ende.

4.4.1 Exkursion: Streams API Java 8

Beschreibung was Streams darstellen, wie sich Observables im Vergleich verhalten

4.4.2 Operation filter()

Beispiel und Perlenbild. Einsatz beschreiben

4.4.3 Transformation map()

Beispiel und Perlenbild. Einsatz beschreiben

4.4.4 Transformation flatMap()

Beispiel und Perlenbild. Einsatz beschreiben

4.4.5 Operation merge()

Beispiel und Perlenbild. Einsatz beschreiben

4.4.6 Operation zip()

Beispiel und Perlenbild. Einsatz beschreiben Eventuell noch mehr Operationen

Literaturverzeichnis

[Bai13] Bainomugisha, Carreton Andoni Lombide van Cutsem Tom Mostinckx Stijn Meuter Wolfgang d. Engineer: A survey on reactive programming. In: *ACM Computing Surveys* 45 (2013), Nr. 4, S. 1–34. http://dx.doi.org/10.1145/2501654.2501666. – DOI 10.1145/2501654.2501666. – ISSN 03600300

[Kar16] Karnok, Dávid: *Operator-fusion (Part 1)*. https://akarnokd.blogspot.de/2016/03/operator-fusion-part-1.html. Version: 2016 (Advanced Reactive Java)

[Loh16] Lohmüller, Jan C.: *Reactive Programming – Mehr als nur Streams und Lambdas*. https://www.informatik-aktuell.de/entwicklung/programmiersprachen/reactive-programming-mehr-als-nur-streams-und-lambdas.html. Version: 2016