Standard Kurs Java Refactoring

Dieser Java Refactoring Kurs zeigt wie man die interne Struktur von Java Programmen via Refactoring Schritt für Schritt verbessert.

Information

Kurscode: JREFDas Angebot Firmenkurs finden Sie hier.Das Angebot Online Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

Information

Kurscode: JREFKursdauer: 2 TageKurspreis Firmenkurs Inhouse: sFr. 4199.- (3-5 Teilnehmer)sFr. 5879.- (6-8 Teilnehmer)sFr. 6999.- (9-12 Teilnehmer)Inhouse Firmenkurse werden vor Ort bei Ihnen durchgeführt. Je nach Ort und Distanz können zusätzliche Kosten für Übernachtung und Anfahrt anfallen.Das Angebot Standard Kurs finden Sie hier.Das Angebot Online Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

Information

Kurscode: JREFDas Angebot Firmenkurs finden Sie hier.Das Angebot Standard Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

Einleitung

Mit "Refactoring" fasst man alle Aktivitäten der Software-Entwicklung zusammen, die den Programmcode bestehender Anwendungen verbessern, ohne die Funktionsweise dieser Programme zu ändern.Bestehende Programme werden in der Regel oft angepasst und umgebaut. Dabei soll der Programmcode nicht immer unverständlicher und schwieriger, sondern einfacher und strukturierter werden. Zudem muss das Design beim Umbau des Programmcodes im Auge behalten werden.Heute sind klare Regeln für ein schrittweises Redesign bekannt. Diese werden allgemein unter dem Begriff "Refactoring" zusammengefasst.Dieses Kurs bietet ein vertiefte Einführung in das Refactoring mit Java. Eine Fallstudie zeigt die Vorteile von Refactoring deutlich auf.Zahlreiche Übungen und Beispiele begleiten diesen Kurs.In diesem Kurs setzen wir die Entwicklungsumgebung Eclipse oder auf Wunsch IntellJ ein.

Ihr Nutzen

- Bestehenden Code auf schlechte Gewohnheiten (Bad Smells) hin prüfen können.
- Diesen Code via Refactoring Schritt für Schritt verbessern können.
- Die Refactoring Denkweise anhand einer Fallstudie nachvollziehen können.
- Das Denken und Handeln in Refactoring fördern.

Verwandte Kurse

- Java SE Einführung (JEGL)
- Java SE für nicht OO Programmierer (JNOO)
- Java Advanced
- Java Design Patterns

Voraussetzungen

Gute Grundkenntnisse von Java analog der JEGL (Java Einführung) und JPF2 (Java Vertiefung).

Teilnehmerkreis

Java-Entwickler, welche bestehende Software warten, (Applikations- bzw.) Wartungsverantwortliche, Java-Softwareingenieure, die für das Redesign von grösseren Applikationen verantwortlich sind sowie Klassendesigner und Softwarearchitekten.

Unterlagen

- Tutorial
- Code Walks
- Internet / Intranet

Inhalt

- Einführung
  - Was ist Refactoring
  - Warum Refactoring
  - Bad Code Smell
  - Clean Code
- Eine erste Fallstudie
- Refactoring Prinzipien
  - Wann ist Refactoring angesagt?
  - Probleme mit Refactoring
  - Refactoring und Design
- Schritte des Refactoring
- Codierung ohne Refactoring
- Schlechte Gewohnheiten (Bad Smell)
- Methoden Refactoring
  - Extract Method
  - Inline Method
  - Extract Variable
  - Inline Temp
  - Replace Temp with Query
  - Split Temporary Variable
  - Remove Assignments to Parameters
  - Replace Method with Method Object
  - Substitute Algorithm
- Object Refactoring
  - Move Method
  - Move Field
  - Extract Class
  - Inline Class
  - Hide Delegate
  - Remove Middle Man
  - Introduce Foreign Method
  - Introduce Local Extension
- Data Refactoring
  - Change Value to Reference
  - Change Reference to Value
  - Duplicate Observed Data
  - Self Encapsulate Field
  - Replace Data Value with Object
  - Replace Array with Object
  - Change Unidirectional Association to Bidirectional
  - Change Bidirectional Association to Unidirectional
  - Encapsulate Field
  - Encapsulate Collection
  - Replace Magic Number with Symbolic Constant
  - Replace Type Code with Class
  - Replace Type Code with Subclasses
  - Replace Type Code with State/Strategy

  - Replace Subclass with Fields
- Ausdrücke vereinfachen
  - Decompose Conditional
  - Consolidate Conditional Expression
  - Consolidate Duplicate Conditional Fragments
  - Remove Control Flag
  - Replace Nested Conditional with Guard Clauses
  - Replace Conditional with Polymorphism
  - Introduce Null Object
  - Introduce Assertion
- Methodenaufrufe vereinfachen
  - Add Parameter
  - Remove Parameter
  - Rename Method
  - Separate Query from Modifier
  - Parameterize Method
  - Introduce Parameter Object
  - Preserve Whole Object
  - Remove Setting Method
  - Replace Parameter with Explicit Methods
  - Replace Parameter with Method Call
  - Hide Method
  - Replace Constructor with Factory Method
  - Replace Error Code with Exception
  - Replace Exception with Test
- Generalisieren
  - Pull Up Field
  - Pull Up Method
  - Pull Up Constructor Body
  - Push Down Field
  - Push Down Method
  - Extract Subclass
  - Extract Superclass
  - Extract Interface
  - Collapse Hierarchy
  - Form Template Method
  - Replace Inheritance with Delegation
  - Replace Delegation with Inheritance

Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

https://www.simtech-ag.ch/Constant