

Allgemeines

<i>Dozent:</i>	Prof. Dr. Roland Feindor / Prof. Dr. Reiner Hüttl
<i>Verantwortlich:</i>	Prof. Dr. Roland Feindor / Prof. Dr. Reiner Hüttl
<i>Studiengang:</i>	Bachelor, Diplom
<i>Kennung:</i>	Bachelor / Diplom: Pflichtfach im Grundstudium
<i>Voraussetzungen:</i>	Programmieren I
<i>Sprache:</i>	Deutsch
<i>Lehrform:</i>	2 SWS seminaristischer Unterricht und 2 SWS Übung
<i>Arbeitsaufwand:</i>	60 Stunden Präsenzzeit, 90 Stunden Selbststudium
<i>Leistungspunkte:</i>	5
<i>Medienform:</i>	Vortrag mit Laptop mit Beamer (.ppt), Overhead-Projektor, Tafelarbeit, Übungen mit SEU, ausgearbeitetes Skript (.pdf)
<i>Prüfung:</i>	Programmierprojekt (Studienarbeit im Team), und Klausur, 90 Min., mit Unterlagen laut Aushang

Lernziele und Inhalt

Richtziel

Die Studierenden lernen die grundlegenden Programmierkonzepte und alle wichtigen Elemente der objektorientierten Programmierung sicher kennen und beherrschen. Mithilfe einer modernen Software-Entwicklungsumgebung sollen sie auch komplexere Projekte objektorientiert entwerfen, erstellen, testen und analysieren können und dabei einen qualitätsorientierten Programmierstil sicher verwenden.

Inhaltsübersicht

Am Beispiel von Java werden die Studenten in die Grundzüge der objektorientierten Programm-Entwicklung eingeführt. Sie lernen dabei alle wichtigen Elemente der objektorientierten Programmierung kennen (Klassen, Objekte, Interfaces, Vererbung, Polymorphie, Behälter und Iteratoren, Ausnahmen). Es werden auch allgemeine Programmierkonzepte vermittelt (Rekursion, Tests, Dokumentation, Sortieren, Programmierregeln), die von der Programmiersprache weitgehend unabhängig sind. Die Vorlesung wird von Übungen an modernen PCs mit aktuellen Entwicklungsumgebungen begleitet. Hier ist neben den Übungsaufgaben auch ein kleines Pflicht-Projekt im Team mit einer Präsentation der Ergebnisse vorgesehen.

Inhalt

- 1. Einführung*
Software-Entwicklungsumgebung, Vergleich zu C, Datentypen, Kontrollstrukturen, Java-Umgebung, Pakete, Tests mit JUnit, Rekursion
- 2. Klassen und Objekte*
Grundprinzipien der OOP, Sichtbarkeit, Konstruktoren, Überladen von Methoden, statische Methoden und Variablen, This, Strings, Arrays, Wrapper-Klassen, Parameterübergabe, Klasse Object, Dokumentation mit JavaDoc, Graphische Oberflächen mit Swing
- 3. Interfaces und Vererbung*
Interface definieren, Interface verwenden, Fabrik-Muster, Vererbung, abstrakte Klassen, Polymorphie und dynamisches Binden
- 4. Behälter und Iteratoren*
Array, Vector, Collection, List, Iterator, Set, Map, Sortieren mit Merge-Sort und Quick-Sort

5. *Fehler und Ausnahmen*
geprüfte und ungeprüfte Ausnahmen, lokale Behandlung und Weitergabe, Return-Codes, Pre- und Postconditions
6. *Programmierregeln und Konventionen*

Literatur

Besonders empfohlen

1. Ullenboom, Ch.: Java ist auch eine Insel www.galileocomputing.de/openbook/javainsel4/
2. Arnold, K. ; Gosling, J. / Holmes, D.: *Die Programmiersprache Java*. Addison Wesley
3. Kröckertskoth, T.: *Java 2, Grundlagen und Einführung*. RRZN-Handbuch, Hannover
4. Campione, M.; Walrath, K. ; Huml, A.: *The Java Tutorial*. Addison Wesley
5. Krüger, G.: *Handbuch der Java-Programmierung*. Addison Wesley

Zusätzliche Literatur

6. Gosling, J.; Joy, B.; Steele, G.: *The Java Language Specification*. Addison Wesley
7. Eckel, B.: *Thinking in Java*. Prentice Hall
8. Flanagan, D.: *Java in a Nutshell*. O'Reilly
9. Bloch, J.: *Effektiv Java programmieren*. Addison Wesley
10. <http://java.sun.com> insbesondere <http://java.sun.com/docs/books/>