

# Inhalt

Materialien zum Buch .....	12
<b>1 Einleitung</b> .....	<b>15</b>
<b>1.1 Was ist Objektorientierung?</b> .....	<b>15</b>
<b>1.2 Hallo liebe Zielgruppe</b> .....	<b>16</b>
<b>1.3 Was bietet dieses Buch (und was nicht)?</b> .....	<b>18</b>
1.3.1 Bausteine des Buches .....	18
1.3.2 Crosscutting Concerns: übergreifende Anliegen .....	21
1.3.3 Die Rolle von Programmiersprachen .....	23
<b>1.4 Warum überhaupt Objektorientierung?</b> .....	<b>24</b>
1.4.1 Gute Software: Was ist das eigentlich? .....	25
1.4.2 Die Rolle von Prinzipien .....	26
1.4.3 Viele mögliche Lösungen für ein Problem .....	27
<b>2 Die Basis der Objektorientierung</b> .....	<b>29</b>
<b>2.1 Die strukturierte Programmierung als Vorläufer der Objektorientierung</b> .....	<b>30</b>
<b>2.2 Die Kapselung von Daten</b> .....	<b>33</b>
<b>2.3 Polymorphie</b> .....	<b>35</b>
<b>2.4 Die Vererbung</b> .....	<b>36</b>
2.4.1 Vererbung der Spezifikation .....	36
2.4.2 Vererbung von Umsetzungen (Implementierungen) .....	37
<b>3 Die Prinzipien des objektorientierten Entwurfs</b> .....	<b>41</b>
<b>3.1 Prinzip 1: Prinzip einer einzigen Verantwortung</b> .....	<b>42</b>
<b>3.2 Prinzip 2: Trennung der Anliegen</b> .....	<b>47</b>

<b>3.3</b>	<b>Prinzip 3: Wiederholungen vermeiden</b> .....	49
<b>3.4</b>	<b>Prinzip 4: offen für Erweiterung, geschlossen für Änderung</b> ...	52
<b>3.5</b>	<b>Prinzip 5: Trennung der Schnittstelle von der Implementierung</b> .....	55
<b>3.6</b>	<b>Prinzip 6: Umkehr der Abhängigkeiten</b> .....	58
3.6.1	Umkehrung des Kontrollflusses .....	62
<b>3.7</b>	<b>Prinzip 7: Mach es testbar</b> .....	64

## **4 Die Struktur objektorientierter Software** 67

---

<b>4.1</b>	<b>Die Basis von allem: das Objekt</b> .....	67
4.1.1	Eigenschaften von Objekten: Objekte als Datenkapseln	69
4.1.2	Operationen und Methoden von Objekten .....	76
4.1.3	Kontrakte: ein Objekt trägt Verantwortung .....	81
4.1.4	Die Identität von Objekten .....	83
4.1.5	Objekte haben Beziehungen .....	85
<b>4.2</b>	<b>Klassen: Objekte haben Gemeinsamkeiten</b> .....	86
4.2.1	Klassen sind Modellierungsmittel .....	87
4.2.2	Kontrakte: die Spezifikation einer Klasse .....	91
4.2.3	Klassen sind Datentypen .....	95
4.2.4	Klassen sind Module .....	105
4.2.5	Sichtbarkeit von Daten und Methoden .....	108
4.2.6	Klassenbezogene Methoden und Attribute .....	115
4.2.7	Singleton-Methoden: Methoden für einzelne Objekte ...	120
<b>4.3</b>	<b>Beziehungen zwischen Objekten</b> .....	121
4.3.1	Rollen und Richtung einer Assoziation .....	123
4.3.2	Navigierbarkeit .....	124
4.3.3	Multiplizität .....	124
4.3.4	Qualifikatoren .....	129
4.3.5	Beziehungsklassen, Attribute einer Beziehung .....	130
4.3.6	Implementierung von Beziehungen .....	132
4.3.7	Komposition und Aggregation .....	133
4.3.8	Attribute .....	136
4.3.9	Beziehungen zwischen Objekten in der Übersicht .....	137

<b>4.4</b>	<b>Klassen von Werten und Klassen von Objekten .....</b>	<b>137</b>
4.4.1	Werte in den objektorientierten Programmiersprachen .....	138
4.4.2	Entwurfsmuster »Fliegengewicht« .....	141
4.4.3	Aufzählungen (Enumerations) .....	144
4.4.4	Identität von Objekten .....	147

## **5 Vererbung und Polymorphie** 157

---

<b>5.1</b>	<b>Die Vererbung der Spezifikation .....</b>	<b>157</b>
5.1.1	Hierarchien von Klassen und Unterklassen .....	158
5.1.2	Unterklassen erben die Spezifikation von Oberklassen ...	159
5.1.3	Das Prinzip der Ersetzbarkeit .....	163
5.1.4	Abstrakte Klassen, konkrete Klassen und Schnittstellenklassen .....	169
5.1.5	Vererbung der Spezifikation und das Typsystem .....	178
5.1.6	Sichtbarkeit im Rahmen der Vererbung .....	185
<b>5.2</b>	<b>Polymorphie und ihre Anwendungen .....</b>	<b>196</b>
5.2.1	Dynamische Polymorphie am Beispiel .....	197
5.2.2	Methoden als Implementierung von Operationen .....	202
5.2.3	Anonyme Klassen .....	211
5.2.4	Single und Multiple Dispatch .....	213
5.2.5	Die Tabelle für virtuelle Methoden .....	231
<b>5.3</b>	<b>Die Vererbung der Implementierung .....</b>	<b>242</b>
5.3.1	Überschreiben von Methoden .....	245
5.3.2	Das Problem der instabilen Basisklassen .....	253
5.3.3	Problem der Gleichheitsprüfung bei geerbter Implementierung .....	258
<b>5.4</b>	<b>Mehrfachvererbung .....</b>	<b>265</b>
5.4.1	Mehrfachvererbung: Möglichkeiten und Probleme .....	265
5.4.2	Delegation statt Mehrfachvererbung .....	273
5.4.3	Mixin-Module statt Mehrfachvererbung .....	275
5.4.4	Die Problemstellungen der Mehrfachvererbung .....	279
<b>5.5</b>	<b>Statische und dynamische Klassifizierung .....</b>	<b>294</b>
5.5.1	Entwurfsmuster »Strategie« statt dynamischer Klassifizierung .....	295
5.5.2	Dynamische Änderung der Klassenzugehörigkeit .....	300

## **6 Persistenz** 305

---

<b>6.1</b>	<b>Serialisierung von Objekten</b> .....	305
<b>6.2</b>	<b>Speicherung in Datenbanken</b> .....	306
6.2.1	Relationale Datenbanken .....	306
6.2.2	Struktur der relationalen Datenbanken .....	307
6.2.3	Begriffsdefinitionen .....	307
<b>6.3</b>	<b>Abbildung auf relationale Datenbanken</b> .....	313
6.3.1	Abbildung von Objekten in relationalen Datenbanken ...	313
6.3.2	Abbildung von Beziehungen in relationalen Datenbanken .....	317
6.3.3	Abbildung von Vererbungsbeziehungen auf eine relationale Datenbank .....	321
<b>6.4</b>	<b>Normalisierung und Denormalisierung</b> .....	326
6.4.1	Die erste Normalform: es werden einzelne Fakten gespeichert .....	327
6.4.2	Die zweite Normalform: alles hängt vom ganzen Schlüssel ab .....	329
6.4.3	Die dritte Normalform: keine Abhängigkeiten unter den Nichtschlüsselspalten	332
6.4.4	Die vierte Normalform: Trennung unabhängiger Relationen .....	336
6.4.5	Die fünfte Normalform: einfacher geht's nicht .....	338

## **7 Abläufe in einem objekt-orientierten System** 343

---

<b>7.1</b>	<b>Erzeugung von Objekten mit Konstruktoren und Prototypen</b>	344
7.1.1	Konstruktor: Klassen als Vorlagen für ihre Exemplare	344
7.1.2	Prototypen als Vorlagen für Objekte .....	348
7.1.3	Entwurfsmuster »Prototyp« .....	354
<b>7.2</b>	<b>Fabriken als Abstraktionsebene für die Objekterzeugung</b> .....	355
7.2.1	Statische Fabriken .....	359
7.2.2	Abstrakte Fabriken .....	362
7.2.3	Konfigurierbare Fabriken .....	367
7.2.4	Registraturen für Objekte .....	371
7.2.5	Fabrikmethoden .....	375

7.2.6	Erzeugung von Objekten als Singletons .....	384
7.2.7	Dependency Injection .....	393
<b>7.3</b>	<b>Objekte löschen</b> .....	404
7.3.1	Speicherbereiche für Objekte .....	404
7.3.2	Was ist eine Garbage Collection? .....	406
7.3.3	Umsetzung einer Garbage Collection .....	407
<b>7.4</b>	<b>Objekte in Aktion und in Interaktion</b> .....	419
7.4.1	UML: Diagramme zur Beschreibung von Abläufen .....	419
7.4.2	Nachrichten an Objekte .....	428
7.4.3	Iteratoren und Generatoren .....	428
7.4.4	Funktionsobjekte und ihr Einsatz als Eventhandler .....	440
7.4.5	Kopien von Objekten .....	450
7.4.6	Sortierung von Objekten .....	460
<b>7.5</b>	<b>Kontrakte: Objekte als Vertragspartner</b> .....	463
7.5.1	Überprüfung von Kontrakten .....	463
7.5.2	Übernahme von Verantwortung: Unterklassen in der Pflicht .....	465
7.5.3	Prüfungen von Kontrakten bei Entwicklung und Betrieb .....	478
<b>7.6</b>	<b>Exceptions: wenn der Kontrakt nicht eingehalten werden kann</b> .....	479
7.6.1	Exceptions in der Übersicht .....	480
7.6.2	Exceptions und der Kontrollfluss eines Programms .....	486
7.6.3	Exceptions im Einsatz bei Kontraktverletzungen .....	493
7.6.4	Exceptions als Teil eines Kontrakts .....	497
7.6.5	Der Umgang mit Checked Exceptions .....	502
7.6.6	Exceptions in der Zusammenfassung .....	509

## **8 Module und Architektur** 511

---

<b>8.1</b>	<b>Module als konfigurierbare und änderbare Komponenten</b> .....	511
8.1.1	Relevanz der Objektorientierung für die Software- architektur .....	511
8.1.2	Erweiterung von Modulen .....	513
<b>8.2</b>	<b>Die Präsentationsschicht: Model, View, Controller (MVC)</b> .....	520
8.2.1	Das Beobachter-Muster als Basis von MVC .....	520
8.2.2	MVC in Smalltalk: Wie es ursprünglich mal war .....	521
8.2.3	MVC: Klärung der Begriffe .....	522

8.2.4	MVC in Webapplikationen: genannt »Model 2« .....	527
8.2.5	MVC mit Fokus auf die Testbarkeit: Model-View-Presenter .....	529
<b>9</b>	<b>Aspekte und Objektorientierung</b> .....	<b>533</b>
<b>9.1</b>	<b>Trennung der Anliegen</b> .....	<b>533</b>
9.1.1	Kapselung von Daten .....	537
9.1.2	Lösungsansätze zur Trennung von Anliegen .....	538
<b>9.2</b>	<b>Aspektorientiertes Programmieren</b> .....	<b>545</b>
9.2.1	Integration von aspektorientierten Verfahren in Frameworks .....	545
9.2.2	Bestandteile der Aspekte .....	546
9.2.3	Dynamisches Crosscutting .....	547
9.2.4	Statisches Crosscutting .....	554
<b>9.3</b>	<b>Anwendungen der Aspektorientierung</b> .....	<b>556</b>
9.3.1	Zusätzliche Überprüfungen während der Übersetzung .....	557
9.3.2	Logging .....	558
9.3.3	Transaktionen und Profiling .....	559
9.3.4	Design by Contract .....	562
9.3.5	Introductions .....	565
9.3.6	Aspektorientierter Observer .....	566
<b>9.4</b>	<b>Annotations</b> .....	<b>569</b>
9.4.1	Zusatzinformation zur Struktur eines Programms .....	569
9.4.2	Annotations im Einsatz in Java und C# .....	571
9.4.3	Beispiele für den Einsatz von Annotations .....	573
<b>10</b>	<b>Objektorientierung am Beispiel: eine Webapplikation in JavaScript</b> .....	<b>579</b>
<b>10.1</b>	<b>OOP in JavaScript</b> .....	<b>581</b>
10.1.1	Objekte in JavaScript .....	582
10.1.2	Vererbung: JavaScript kennt keine Klassen .....	582
10.1.3	Datenkapselung durch Closures .....	585

<b>10.2 Die Anwendung im Überblick</b> .....	588
10.2.1 Architekturentscheidungen als Basis .....	588
10.2.2 Die Komponenten der Anwendung .....	592
<b>10.3 Das Framework</b> .....	593
10.3.1 Controller: zentrale Repräsentation von Diensten .....	595
10.3.2 Aktionen: Operationen auf Datenmodellen .....	602
10.3.3 Views: verschiedene Sichten auf die Daten .....	608
<b>10.4 Die Applikation</b> .....	611
10.4.1 Anwendungsfälle und das Design der Applikation .....	611
10.4.2 Eine eigene Ableitung des Controllers – und der Dienst »team_lesen« .....	613
10.4.3 Modelle zur Datenhaltung .....	618
10.4.4 Aktionen zur Durchführung von Fachlogik .....	622
10.4.5 Views für unterschiedliche Repräsentationen der Daten	625
<b>10.5 Ein Fazit – und was noch übrig bleibt</b> .....	635

## **Anhang** 637

---

<b>A Verwendete Programmiersprachen</b> .....	639
<b>B Glossar</b> .....	659
<b>C Die Autoren</b> .....	673
 Index .....	 675