

Inhaltsverzeichnis

1.	Einleitung	11
1.1	Die Entstehung von Java.....	11
1.2	Die Java-Technologie: Programmiersprache, Compiler und Laufzeitumgebung.....	13
1.3	Programmieren lernen mit Java.....	15
2.	Notwendige Programme für das Programmieren in Java	17
2.1	Ein Texteditor für die Erstellung des Programmcodes.....	17
2.2	Java Development Kit: wichtiges Werkzeug zum Programmieren.....	19
2.3	Vorbereitungsmaßnahmen für die Verwendung von JDK	20
2.4	Die IDE installieren.....	25
3.	Die ersten Schritte mit Java	27
3.1	Hallo Welt – das erste eigene Programm schreiben	27
3.2	So lässt sich das Programm ausführen.....	29
3.3	Kommentare: weitere Informationen zum Programmcode hinzufügen.....	30
3.4	Übungsaufgabe: Erste Java-Programme selbst gestalten	32
4.	IDE: Mehr Effizienz beim Programmieren	36
4.1	Was ist eine IDE und welche Vorteile bietet sie?.....	36
4.2	Ein Programm mit NetBeans schreiben und ausführen	37
4.3	Weitere praktische Funktionen der IDE	42
5.	Grundlegende Programmfunktionen: Variablen und Operatoren	45
5.1	Die Aufgabe von Variablen in der Informatik	45
5.2	Variablen in das Java-Programm einfügen.....	46
5.3	Variablentypen: unterschiedliche Arten von Informationen abspeichern	49
5.4	Grundlegende Operatoren für Java-Programme	51
5.5	Datentypen verändern.....	54
5.6	Übungsaufgabe: Variablen und Operatoren verwenden.....	60

6.	Zusammengesetzte Datentypen in Java	63
6.1	Arrays	63
6.2	Strings	67
6.3	Java Collection: vielfältige weitere zusammengesetzte Datentypen	68
6.4	Listen.....	69
6.5	Sets.....	72
6.6	Queues.....	75
6.7	Stacks	77
6.8	Maps	78
6.9	Übungsaufgabe: zusammengesetzte Datentypen verwenden	80
7.	Verzweigungen: Entscheidungen mit if-Abfragen treffen	85
7.1	Der Aufbau einer einfachen if-Abfrage	85
7.2	Vergleichsoperatoren	86
7.3	Bedingungen mit logischen Operatoren verknüpfen	89
7.3	Weitere Alternativen mit else hinzufügen.....	91
7.4	Mehrere Auswahloptionen mit else if oder switch erzeugen	92
7.5	Übungsaufgabe: Verzweigungen erstellen	96
8.	Schleifen: einzelne Programmteile wiederholen	102
8.1	Die while-Schleife: der grundlegende Schleifentyp	102
8.2	Die fußgesteuerte do-while-Schleife	104
8.3	Eine feste Anzahl an Durchläufen mit einer for-Schleife vorgeben	106
8.4	Spezielle Schleifen für zusammengesetzte Datentypen.....	107
8.5	Übungsaufgabe: Schleifen verwenden	111
9.	Die Grundlagen der objektorientierten Programmierung	115
9.1	Was bedeutet objektorientierte Programmierung	115
9.2	Die Klasse: eine Vorlage für Objekte	117
9.3	Objekte von einer Klasse ableiten	118
9.4	Methoden: Aktionen mit Objekten durchführen	120
9.5	Übungsaufgabe: objektorientierte Programme erstellen	123
10.	Weiterführende Techniken der objektorientierten Programmierung	126
10.1	Die Daten kapseln.....	126
10.2	Vererbung: Klassen von übergeordneten Strukturen ableiten	130
10.3	Interfaces: feste Vorgaben für Klassen gestalten.....	133

10.4	Abstrakte Klassen in Java	135
10.5	Polymorphie in der objektorientierten Programmierung	136
10.6	Konstanten mit enums vorgeben	139

11.	Vorgefertigte Klassen und Methoden für Java-Programme	143
------------	--	------------

11.1	Die Vorteile vorgefertigter Bibliotheken	143
11.2	Anwendungsbeispiel: eine Bibliothek nutzen	144
11.3	Die Dokumentation der Java-Bibliotheken	146
11.4	Übungsaufgabe: Programme mit vorgefertigten Bibliotheken	148

12.	Fehler im Programm	151
------------	---------------------------	------------

12.1	Verschiedene Arten von Fehlern	151
12.2	Syntaxfehler beheben	152
12.3	Laufzeitfehler durch Ausnahmen abfangen	153
12.4	Logische Fehler durch das Debugging finden	158

13.	Praxisbeispiel: das Warenlager in einem Supermarkt verwalten	162
------------	---	------------

13.1	Die Programmfunktionen und Strukturen	162
13.2	Die Klassen für die verschiedenen Produkte erstellen	163
13.3	Das Hauptprogramm gestalten	166
13.4	Die einzelnen Programmfunktionen programmieren	169

14.	Daten dauerhaft speichern: in Dateien	183
------------	--	------------

14.1	Daten in eine Datei schreiben	184
14.2	Daten aus einer Datei einlesen	186
14.3	XML-Dateien verwenden	188
14.4	Übungsaufgabe: Daten in Dateien speichern	194

15.	Datenbanken: eine sichere und effiziente Alternative für die Datenspeicherung	198
------------	--	------------

15.1	Der Aufbau einer Datenbank	198
15.2	Java DB: eine sinnvolle Wahl für Java-Programme	200
15.3	Datenbanken über die grafische Benutzeroberfläche erzeugen	200
15.4	Tabellen per SQL-Befehl erzeugen	207
15.5	Werte in die Tabelle einfügen	209
15.6	Daten abfragen	210
15.7	Inhalte ändern oder löschen	212

15.8	Die Datenbank in das Java-Programm einbinden	213
15.9	SQL-Befehle im Java-Programm verwenden	216
15.10	Übungsaufgabe: mit Datenbanken arbeiten.....	219

16.	Grafische Benutzeroberflächen mit JavaFX gestalten	225
------------	---	------------

16.1	Geeignete Hilfsmittel für User Interfaces	225
16.2	Das erste Programm mit einer grafischen Benutzeroberfläche erstellen	226
16.3	Weitere Elemente in das Fenster einfügen.....	232
16.4	Scene Builder: eine weitere Möglichkeit für die Erstellung von GUIs.....	238
16.5	Übungsaufgabe: grafische Benutzeroberflächen selbst gestalten	245

17.	Anwendungsbeispiel: Programm für die Personalverwaltung	250
------------	--	------------

17.1	Welche Funktionen soll das Programm erfüllen?	250
17.3	Mitarbeiter hinzufügen	256
17.4	Mitarbeiter entfernen.....	264
17.5	Gehalt anpassen	268
17.6	Das Personal anzeigen	271

18.	Multithreading für eine präzise Steuerung der Abläufe	279
------------	--	------------

18.1	Was ist ein Thread?.....	279
18.2	Threads in Java erzeugen	280
18.3	Verschiedene Zustände eines Threads	285
18.4	Die Reihenfolge bei der Ausführung	286
18.5	Prioritäten festlegen.....	288
18.6	Gemeinsam verwendete Variablen	289
18.7	Den Ablauf mit Monitoren steuern	292
18.8	Beispiel für die Verwendung von Threads.....	294
18.9	Übungsaufgabe: Threads erzeugen.....	299

19.	JSON – ein Objekt für die Übermittlung von Daten	305
------------	---	------------

19.1	Was steckt hinter dem Begriff JSON?.....	305
19.2	Die Vorbereitungsmaßnahmen.....	306
19.3	Mit dem JSONObject arbeiten	306
19.4	Das JSONArray	309
19.5	JSONTokener und CDL	311
19.6	Daten mit JSON schreiben und lesen	313
19.7	Übungsaufgabe: mit JSON arbeiten	317

20.	Netzwerkprogrammierung mit Java	323
20.1	Java für die Kommunikation in Netzwerken verwenden	323
20.2	Was ist ein Socket?	324
20.3	Einen einfachen Server erstellen	325
20.4	Einen einfachen Client programmieren	327
20.5	Die Funktionen des Servers und des Clients erweitern	329
20.6	RMI: Methoden auf einem Server abrufen	337
20.7	Anwendungsbeispiel für die Netzwerkprogrammierung	341
20.8	Übungsaufgabe: Programme für Netzwerke erstellen	350
21.	Lambda-Ausdrücke in Java	356
21.1	Hinführung: anonyme Klassen in Java	356
21.2	Lambda-Ausdrücke als Alternative zu anonymen Klassen	360
21.3	Lambda-Ausdrücke mit mehreren Befehlen und Übergabewerten	361
21.4	Der Zugriff auf die Variablen in Lambda-Ausdrücken	362
21.5	Lambda-Ausdrücke für Iterationen verwenden	364
21.6	Lambda-Ausdrücke in der Praxis verwenden	365
21.7	Übungsaufgabe: Lambda-Ausdrücke verwenden	372
22.	Generics	376
22.1	Generics: Wozu dient diese Technik und wie lässt sie sich anwenden?	376
22.2	Welche Datentypen eignen sich für Generics?	378
22.3	Generische Methoden erstellen	379
22.4	Generische Klassen	382
22.5	Einschränkungen für die Typen vorgeben	383
22.6	Wildcards verwenden	385
22.7	Anwendungsbeispiel für Generics	386
22.8	Übungsaufgabe: Mit Generics arbeiten	391
23.	Weitere Möglichkeiten für grafische Benutzeroberflächen	395
23.1	Das Layout vorgeben	395
23.2	Weitere Elemente in das Fenster einfügen	402
23.3	Farben und Grafiken hinzufügen	406
23.4	Transformationen, Animationen und weitere Effekte	412
23.5	Ein Programm durch eine ansprechende Gestaltung aufwerten	418
23.5	Übungsaufgabe: mit JavaFX arbeiten	422

24.	Die Arbeit in der Praxis: Versionsverwaltung mit Git	427
24.1	Was bedeutet Versionsverwaltung?	427
24.2	Git installieren und konfigurieren	428
24.3	Ein Projekt mit Git erstellen oder ein bestehendes Projekt nutzen	431
24.4	Die Versionsgeschichte	435
24.5	Zweige erstellen und zusammenführen	437
24.6	Weitere praktische Möglichkeiten: Git mit NetBeans verwenden	439
25.	Clean Code: besser lesbare Programme schreiben	444
25.1	Weshalb ist Clean Code wichtig?	444
25.2	Selbsterklärende Bezeichner verwenden	445
25.3	DRY: Auf Wiederholungen verzichten	446
25.4	KISS: einfachen und kurzen Code erzeugen	447
25.5	Least Surprise: Vorhersehbare Klassen und Methoden erzeugen	448
25.6	TDA: aussagekräftigen Code erstellen	449
25.7	Das SOLID-Prinzip	449
25.8	Clean Code oder Kommentare verwenden?	451
25.9	Die Boy-Scout-Regel für die Überarbeitung eines Programms	452
26.	Softwaretests mit JUnit	454
26.1	Was ist ein Softwaretest und welche Vorteile bietet er?	454
26.2	Verschiedene Arten von Tests	455
26.3	JUnit: Ein nützliches Framework für Software-Test	456
26.4	Einen Test mit JUnit vorbereiten und durchführen	457
26.5	Einen Test mit Parametern durchführen	464
26.6	Übungsaufgabe: einen Softwaretest durchführen	468
27.	Professionelle Programmierung in der Praxis	472
27.1	Das Praxisprojekt in einer Versionsverwaltung ablegen	472
27.2	Den Code bereinigen	473
27.3	Das Programm testen	482
28.	Der Weg zum Java-Programmierer: Wie geht es weiter?	485
	Glossar	488
	Index	492