

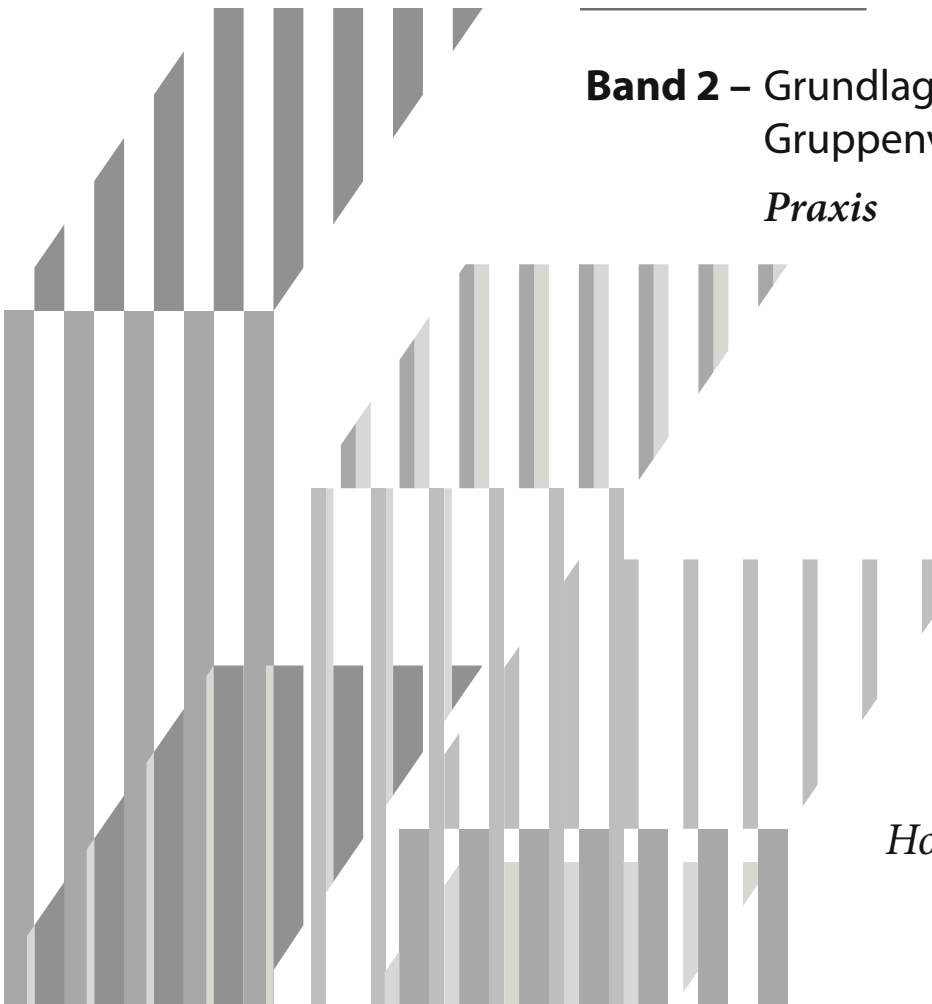
MODERNE STRUKTURIERTE PROGRAMMIERUNG

Objektorientiert und algorithmisch

ENTWERFEN | IMPLEMENTIEREN | TESTEN

Band 2 – Grundlagen
Gruppenverarbeitungen
Praxis

Horst van Bremen



Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über <http://dnb.dnb.de> abrufbar.

© 2022 Dipl.-Ing. Horst van Bremen

Gestaltung:	Katharina Schmitt
Titel- und Einbandgrafik:	Monika Sylvia Gomm † 1971
Englisch-Korrektorat:	David Roseveare
Verlag:	HMG® Verlags-GmbH, Neue Strasse 74, 32657 Lemgo Mail-Adresse: office@hmg-verlag.de Website: www.hmg-verlag.de
Herstellung:	BoD – Books on Demand GmbH, In de Tarpen 42, 22848 Norderstedt
Version / Datum:	Version 1.0.4 vom 2.5.2023
Literaturverzeichnis:	siehe Kapitel „11.1 Literaturverzeichnis“ auf Seite 411 ff.
Rechtliches:	siehe Kapitel „11.2 Rechtliche Hinweise“ auf Seite 413 ff.
ISBN des Paperback-Buchs:	978-3-910566-02-6
ISBN der Paperback-Buchreihe:	978-3-910566-00-2
ISBN des Hardcover-Buchs:	978-3-910566-12-5
ISBN der Hardcover-Buchreihe:	978-3-910566-10-1

Vorwort

Der hier vorliegende zweite Band der ersten Staffel dieser Buchreihe versammelt alle Programmbeispiele, die auf der Basis der JSP/MSP-Entwürfe im 1. Band vom Autor geschrieben wurden, sowie Testkapitel und Testprotokolle. Das ist zumindest unüblich. Normalerweise enthalten Bücher über Programmierung zwar Code-Beispiele in Form kurzer Abschnitte, die englisch Snippets – also Schnipsel oder Bruchstück – genannt werden, aber den Code ganzer Programme – und erst recht Tests hiervon – hat der Autor noch in keinem Fachbuch gesehen. Im vorliegenden Fall gibt es aber für das ungewöhnliche Vorgehen gute Gründe:

- Anfänger tun sich häufig schwer damit, etwas, das sie theoretisch verstanden haben, in ein funktionierendes Programm umzusetzen. Die dafür erforderliche Erfahrung braucht ihre Zeit, um sich zu entwickeln. Um so besser und schneller geht es, wenn Beispiele vorliegen, an denen sich die eigenen Entwürfe orientieren können. Die Download-Option auf www.hmg-verlag.de erspart das Abtippen.
- In Jahrzehnten des Ausprobierens und des Scheiterns bildeten sich Kriterien dafür, wie eine professionelle Programmierung aussehen kann, nicht muss. Ein Beispiel: In den allerersten ALGOL-Programmen des Autors hießen die Variablen A1, A2, A3 und so weiter, was diesen Code von vornherein nahezu unverständlich machte. Wie wichtig es ist, passende Namen zu wählen, stellte sich erst nach und nach heraus. Hier möchte der zweite Band stilbildend sein.
- Große Unsicherheiten bestehen anfangs auch bei der Modularisierung, sowie bei der OO-Programmierung, ganz besonders aber bei der Zuordnung von Code-Snippets zu Methoden. Wie man hier geschickt vorgeht, das lehrt auch JSP/MSP nicht. Hinsichtlich dieses Aspekts wird im zweiten Band angestrebt, zumindest gute Lösungen zu zeigen, ohne den Anspruch auf Vorbildlichkeit zu erheben.
- Vollständig ungewöhnlich ist es, JSP/MSP-Entwürfe in OO-Sprachen zu realisieren. Die strukturierte, datenbasierte Methode von Michael A. Jackson wurde zwar primär für die algorithmische Programmierung konzipiert, eignet sich aber auch für OO-Implementierungen. Das Zusammenspiel „echter“ OO-Klassen mit den neu eingeführten Process Control-Klassen, die sich den JSP-typischen Baumstrukturen verdanken, kann anhand der hier vorgelegten Beispiele genau studiert werden.
- Die Aufteilung der Buchreihe in Paare von Methoden- und Praxis-Bänden ist nicht nur dem Umfang der Bände geschuldet, sondern sie dient hauptsächlich dazu, die Entwürfe und ihre Umsetzungen nebeneinanderlegen zu können. Durch die konsequente Übernahme der Operations- und Kontroll-Bezeichnungen in die Programme und durch die ausführliche Kommentierung kann die Ableitung des Codes aus den Struktogrammen gut nachvollzogen werden. Die Test-Abschnitte unterstützen das Verständnis.
- Code-externe Kommentare, Anmerkungen und teilweise recht detaillierte Darstellungen dienen dazu, den Implementierungen eine eigene Semantik zu geben. Auch gut kommentierter Code ist selten wirklich selbsterklärend. Die Zusatztexte sollen die verbleibende Verständnislücke schließen helfen.
- Um „den Wald“ zu schonen, wurden Redundanzen nach Kräften vermieden. Wiederverwendeter Code wird nicht wiederholt, sondern es wird auf dessen erstes Auftreten verwiesen.

Es ist dem Autor bewusst, dass es vermutlich ebensoviele Meinungen über die „richtige“ Programmierung wie Programmierer(innen) gibt. Daher ist es möglich, dass die hier gezeigten Implementierungen auf heftige Kritik oder gar auf völlige Ablehnung stoßen. Ganz sicher gibt es auch an den nicht immer konsequent umgesetzten Namenskonventionen und anderen Formalia dies und jenes zu bemängeln. Insofern hofft der Autor auf die Gnade derjenigen, die es damit ganz besonders streng halten. Allen anderen mögen die Kapitel des zweiten Bands hoffentlich von großem Nutzen sein.

Lemgo, den 2.5.2023

Übersichtsverzeichnis Band 2

1	Example1 in COBOL II und C	1
2	Example1 in Java	31
3	SonnetAnalysis in C	53
4	SonnetAnalysis in Java	73
5	ProcessWordList in C	101
6	ProcessWordList in Java	123
7	EvaluateSportsDS in C	141
8	File Services in Java	205
9	EvaluateSportsDS in Java	247
10	MailSportResults in Java	317
11	Anhang	411

Inhaltsverzeichnis Band 2

1	Example1 in COBOL II und C	1
1.1	COBOL-II-Code von Example1	1
1.2	COBOL-II-Code-Durchgang	5
1.2.1	Programmkopf	5
1.2.2	Input-Output Section	6
1.2.3	Working Storage	7
1.2.4	Procedure Division	10
1.2.5	Schlussbemerkungen zur COBOL-II-Version	15
1.3	C-Code von Example1	15
1.3.1	main()	17
1.3.2	readTextin()	21
1.3.3	itoa()	22
1.3.4	substr()	24
1.4	C-Code-Durchgang	25
1.4.1	Programmvorspann und main()-Funktion	25
1.4.2	readTextin()	28
1.4.3	itoa()	28
1.4.4	substr()	28
2	Example1 in Java	31
2.1	Java-Klassendiagramm von Example1	31
2.2	Java-Code von Example1	33
2.2.1	Klasse TextData	33
2.2.2	Klasse Total	35
2.2.3	Klasse Conversion	35
2.2.4	Klasse Example1	36
2.2.5	Klasse FileService	38
2.3	Java-Code-Durchgang	43
2.3.1	Utility-Klasse FileService	43
2.3.2	Hinweis zur Exception-Behandlung	44
2.3.3	main()	44
2.3.4	analyzeText()	49
2.4	Verarbeitung von diakritischen Zeichen	50
2.5	Bewertung des Java-Codes von Example1	51
3	SonnetAnalysis in C	53
3.1	Globale Deklarationen	53
3.2	main()	54
3.3	checkLCA_usage()	58
3.4	countLineGroup()	59
3.5	processLineGroup()	60
3.6	readTextin()	60
3.7	Code-Durchgang	62
3.7.1	Überblick	62

Band 2

3.7.2 Gruppenschlüssel-Struktur	64
3.7.3 main()	65
3.7.4 checkLCA_usage()	68
3.7.5 countLineGroup()	68
3.7.6 processLineGroup()	69
3.7.7 readTextin()	69
3.8 Laufergebnisse	71
4 SonnetAnalysis in Java	73
4.1 OO-Vorbereitungen	73
4.2 Realisierung verteilter Lese-Operationen	77
4.3 Implementierungsumgebungen	78
4.4 BlueJ-Klassendiagramm	80
4.5 Klasse SonnetAnalysis	80
4.6 Klasse FileService	82
4.7 Klasse PartHandler	82
4.7.1 processPartGroups()	84
4.7.2 processPlusPartGroups()	85
4.7.3 printAnalysisResults()	85
4.7.4 checkLCA_usage()	86
4.8 Klasse LCA_Entry	87
4.9 Klasse GroupHandler	88
4.9.1 processLineGroup()	88
4.9.2 processPlusLineGroup()	89
4.10 Klasse TL_Util	90
4.10.1 createTextLine()	90
4.10.2 getXXX() / hasXXX() / isXXX() methods	91
4.11 Klasse GroupKey	92
4.12 Erweitertes Klassendiagramm	94
4.13 Code-Durchgang	95
4.13.1 Überblick	95
4.13.2 main()	96
4.13.3 PartHandler & LCA_Entry	96
4.13.4 GroupHandler	97
4.13.5 TL_Util	97
4.13.6 GroupKey	98
4.14 Laufergebnisse	98
5 ProcessWordList in C	101
5.1 Globale Deklarationen	101
5.2 main()	102
5.3 buildGroupLine()	105
5.4 readSortWL()	107
5.5 printfDetails()	109
5.6 itoa()	110
5.7 Code-Durchgang	112
5.7.1 Überblick	112
5.7.2 Gruppenschlüssel-Struktur	112
5.7.3 readSortWL()	113

5.7.4 printfDetails()	113
5.8 Testergebnisse mit den Worten von Shakespeares Sonett	113
6 ProcessWordList in Java	123
6.1 Einleitung	123
6.2 Klassendiagramm von ProcessWordList	124
6.3 Klasse ProcessWordList	125
6.4 Klasse GroupHandler	127
6.5 Klasse GroupKey	130
6.6 Hilfsklasse Trace	132
6.7 Klasse InputUtil	133
6.7.1 Klasse Counter	135
6.7.2 Klasse Counters	136
6.7.3 Klasse Conversion	137
6.8 Test der Java-Version von ProcessWordList	138
7 EvaluateSportsDS in C	141
7.1 Ergänzte Programmstrukturen	141
7.1.1 Top-Programmstruktur	141
7.1.2 Process R-School Group	142
7.1.3 Process Gender Group / Process Age Group	143
7.1.4 Gewinner-Listen-Auswertungen	144
7.1.5 Build Linked List	145
7.1.6 Compress Linked List	146
7.1.7 Add / Amend 1st Value Entries	147
7.1.8 Add Later Entry	148
7.2 Aufruf-Hierarchie	149
7.3 Moment mal!	150
7.3.1 Symbole und Bezeichnungen der JSP-Methode	150
7.3.2 Vergleich zu verwandten Symbolsprachen	150
7.3.3 Michael A. Jackson und seine „Codierknechte“	151
7.3.4 Grafisches	152
7.4 Entstehung der Aufrufhierarchie	152
7.5 C-Code von EvaluateSportsDS	155
7.5.1 Globale Deklarationen	155
7.5.2 main()	162
7.5.3 processCountryGroup()	165
7.5.4 processRSchoolGroup()	167
7.5.5 processGenderGroup()	173
7.5.6 processAgeGroup()	174
7.5.7 readSportsDS()	175
7.5.8 readSchoolDS()	178
7.5.9 buildLinkedList()	180
7.5.10 initializeLinkedList()	182
7.5.11 addPrefix_LL_Entry()	183
7.5.12 addOther_LL_Entry()	185
7.5.13 adjustChainEndIndex()	188
7.5.14 compressLinkedList()	189
7.5.15 displayPupilWinners()	192

Band 2

7.5.16	completePupilSpoRes()	195
7.5.17	displaySchoolWinners()	196
7.5.18	completeSchoolSpoRes()	199
7.5.19	displayLinkedList()	200
7.5.20	itoa() und substr()	202
8	File Services in Java	205
8.1	Klasse NameAndCount_AC	205
8.2	Klasse BR_RefNameAndCount	206
8.3	Klasse BW_RefNameAndCount	207
8.4	Klasse FS_Code	208
8.5	Klasse Trace	209
8.6	Klasse FileRegister	210
8.6.1	initReaders()	212
8.6.2	initWriters()	212
8.6.3	register()	213
8.6.4	getXXX() methods	216
8.6.5	handleResults()	218
8.6.6	incInRecCount()	219
8.6.7	incOutRecCount()	219
8.6.8	setXXX() methods	219
8.6.9	shutdown()	220
8.7	Klasse InFileUtil	225
8.7.1	open()	226
8.7.2	read()	229
8.7.3	close()	230
8.7.4	checkErrorLimitExceeded()	231
8.7.5	getXXX() methods	231
8.7.6	handleException()	232
8.7.7	setXXX() methods	233
8.7.8	shutdown()	234
8.8	Klasse OutFileUtil	236
8.8.1	open()	236
8.8.2	write()	239
8.8.3	close()	240
8.8.4	checkErrorLimitExceeded()	241
8.8.5	getXXX() methods	241
8.8.6	handleException()	242
8.8.7	setXXX() methods	243
8.8.8	shutdown()	243
9	EvaluateSportsDS in Java	247
9.1	Einleitung	247
9.2	Process Control-Klassen / Hilfsklassen	247
9.2.1	Klasse EvaluateSportsDS	247
9.2.2	Klasse StreamCode	252
9.2.3	Klasse Trace	253
9.2.4	Klasse Const	254
9.2.5	Klasse CountryGrpHandler	257

9.2.6 Klasse R_SchoolGrpHandler	259
9.2.7 Klasse GSA_Entry	268
9.2.8 Klasse GenderGrpHandler	269
9.2.9 Klasse AgeGrpHandler	271
9.2.10 Klasse SpCoRK_Util	273
9.3 Rangeschlüsselklassen	278
9.3.1 Klasse PupilRank_1_AC	278
9.3.2 Klasse PupilRank_1	279
9.3.3 Klasse PupilRank_1_2_AC	281
9.3.4 Klasse PupilRank_1_2	283
9.3.5 Klasse PupilRank_1_3_AC	285
9.3.6 Klasse PupilRank_1_3	286
9.3.7 Klasse PupilRank_1_4_AC	287
9.3.8 Klasse PupilRank_1_4	289
9.4 Link-Listen-Klassen	290
9.4.1 Klasse Contestant_AC	290
9.4.2 Klasse Pupil	291
9.4.3 Klasse School	294
9.4.4 Klasse Contest_LL	296
9.5 Restliche Klassen	310
9.5.1 Hilfsklasse BuildStat	310
9.5.2 Klasse Conversion	312
10 MailSportResults in Java	317
10.1 Politisch unkorrekte Vorrede	317
10.2 Erste Schritte	318
10.2.1 Klasse MailSportResults beginnen	318
10.2.2 Klasse StreamCode	321
10.2.3 Klasse Trace	321
10.2.4 Klasse Const	322
10.2.5 Klasse ResultD_RK_Util	326
10.2.6 Klasse ResultD	330
10.2.7 Klasse SpCo_Rank_1_AC	331
10.2.8 Klasse SpCo_Rank_1	332
10.2.9 Klasse SpCo_Rank_1_2_AC	333
10.2.10 Klasse SpCo_Rank_1_2	335
10.2.11 Erster Test	336
10.3 Nationale und internationale Schul-Listen (1)	337
10.3.1 Klasse SchoolResD	340
10.3.2 Klasse S_L_Entry	342
10.4 Restliche einfache Klassen	346
10.4.1 Klasse L_S_T_E	346
10.4.2 Klasse SpoResD	351
10.4.3 Klasse MimeMessageUtil	354
10.4.4 Bestandsaufnahme / weiteres Vorgehen	357
10.5 Modellierungs-Einschub	359
10.5.1 Vorläufige Klassenliste	359
10.5.2 Zuordnung von Klassen zur Programmstruktur	359
10.5.3 Klassendiagramm mit Operationen / Kontrollen	365
10.6 Nationale und internationale Schul-Listen (2)	369

Band 2

10.6.1 Klasse SchoolList (1)	369
10.6.2 Klasse ResultGrpHandler	371
10.6.3 Zweiter Test	373
10.6.4 Klasse SchoolList (2)	378
10.6.5 Klasse CountryGrpHandler (1)	380
10.6.6 Dritter Test	382
10.7 E-Mails mit Schulergebnissen erzeugen	384
10.7.1 Klasse PupilResultHandler	384
10.7.2 Klasse SchoolGrpHandler (1)	386
10.7.3 Klasse CountryGrpHandler (2)	391
10.7.4 Vierter Test	391
10.8 Fertigstellung	397
10.8.1 Klasse SchoolList (3)	397
10.8.2 Klasse SchoolGrpHandler (2)	401
10.8.3 main() ergänzen	402
10.8.4 Fünfter und letzter Test	402
10.9 Ergänzungen des Klassendiagramms	404
10.10 Vorläufiger Abschied vom Schulsportwettbewerb	407
10.11 Manöverkritik	407
11 Anhang	411
11.1 Literaturverzeichnis	411
11.2 Rechtliche Hinweise	413
11.2.1 Geschützte Bezeichnungen	413
11.2.2 Haftungsausschluss	419
11.2.3 Zitate	419
11.2.4 Namen	420
11.2.5 Bild- und Grafiknachweis	420
11.2.6 Programme	420