

bjarne STROUSTRUP

// Der Erfinder von C++



Die **C++**
Programmiersprache

HANSER

Bleiben Sie auf dem Laufenden!



Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter



www.hanser-fachbuch.de/newsletter



Hanser Update ist der IT-Blog des Hanser Verlags mit Beiträgen und Praxistipps von unseren Autoren rund um die Themen Online Marketing, Webentwicklung, Programmierung, Softwareentwicklung sowie IT- und Projektmanagement. Lesen Sie mit und abonnieren Sie unsere News unter



www.hanser-fachbuch.de/update



Bjarne Stroustrup

Die C++-Programmiersprache

Aktuell zum C++ 11-Standard

aus dem Amerikanischen übersetzt von Frank Langenau

HANSER

Authorized translation from the English language edition, entitled C++ PROGRAMMING LANGUAGE, THE, 4th Edition, 0321563840 by BJARNE STROUSTRUP, published by Pearson Education, Inc, publishing as Addison-Wesley Professional, Copyright © 2013.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc. GERMAN language edition published by CARL HANSER VERLAG, Copyright © 2015

Alle in diesem Buch enthaltenen Informationen, Verfahren und Darstellungen wurden nach bestem Wissen zusammengestellt und mit Sorgfalt getestet. Dennoch sind Fehler nicht ganz auszuschließen. Aus diesem Grund sind die im vorliegenden Buch enthaltenen Informationen mit keiner Verpflichtung oder Garantie irgendeiner Art verbunden. Autoren und Verlag übernehmen infolgedessen keine juristische Verantwortung und werden keine daraus folgende oder sonstige Haftung übernehmen, die auf irgendeine Art aus der Benutzung dieser Informationen – oder Teilen davon – entsteht.

Ebenso übernehmen Autoren und Verlag keine Gewähr dafür, dass beschriebene Verfahren usw. frei von Schutzrechten Dritter sind. Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Buch berechtigt deshalb auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.



Bibliografische Information der Deutschen Nationalbibliothek:

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

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdruckes und der Vervielfältigung des Buches, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren) – auch nicht für Zwecke der Unterrichtsgestaltung – reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

© 2015 Carl Hanser Verlag München, www.hanser-fachbuch.de

Lektorat: Brigitte Bauer-Schiewek

Fachlektorat: André Wilms, Neukirchen-Vluyn

Copy editing: Regina Langenau, Chemnitz

Herstellung: Irene Weilharth

Umschlagdesign: Marc Müller-Bremer, www.rebranding.de, München

Umschlagrealisation: Stephan Rönigk

Gesamtherstellung: Kösel, Krugzell

Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702

Printed in Germany

Print-ISBN: 978-3-446-43961-0

E-Book-ISBN: 978-3-446-43981-8

Inhalt

Vorwort	XXXI
Teil I – Einführung	1
1 Vorbemerkungen	3
1.1 Zum Aufbau dieses Buchs	3
1.1.1 Einführung	4
1.1.2 Grundlegende Sprachmittel	4
1.1.3 Abstraktionsmechanismen	5
1.1.4 Die Standardbibliothek	7
1.1.5 Beispiele und Referenzen	8
1.2 Der Entwurf von C++	10
1.2.1 Programmierstile	12
1.2.2 Typprüfung	15
1.2.3 C-Kompatibilität	16
1.2.4 Sprache, Bibliotheken und Systeme	17
1.3 C++ lernen	19
1.3.1 In C++ programmieren	21
1.3.2 Ratschläge für C++-Programmierer	22
1.3.3 Ratschläge für C-Programmierer	23
1.3.4 Ratschläge für Java-Programmierer	24
1.4 Historische Anmerkungen	25
1.4.1 Chronik	26
1.4.2 Die frühen Jahre	27
1.4.2.1 Sprachfeatures und Bibliotheksinstrumente	28
1.4.3 Der 1998-Standard	30
1.4.3.1 Sprachfeatures	30
1.4.3.2 Die Standardbibliothek	31
1.4.4 Der 2011-Standard	32
1.4.4.1 Sprachfeatures	33
1.4.4.2 Standardbibliothek	34
1.4.5 Wofür wird C++ verwendet?	35

1.5	Ratschläge	37
1.6	Literaturhinweise	38
2	Ein Rundreise durch C++: Die Grundlagen	43
2.1	Einführung	43
2.2	Die Grundlagen	44
2.2.1	Hello, World!	44
2.2.2	Typen, Variablen und Arithmetik	46
2.2.3	Konstanten	48
2.2.4	Tests und Schleifen	49
2.2.5	Zeiger, Arrays und Schleifen	51
2.3	Benutzerdefinierte Typen	53
2.3.1	Strukturen	53
2.3.2	Klassen	55
2.3.3	Aufzählungen	57
2.4	Modularität	58
2.4.1	Separate Kompilierung	59
2.4.2	Namespaces	60
2.4.3	Fehlerbehandlung	61
2.4.3.1	Ausnahmen	62
2.4.3.2	Invarianten	63
2.4.3.3	Statische Assertionen	64
2.5	Nachbemerkung	65
2.6	Ratschläge	65
3	Eine Rundreise durch C++: Abstraktionsmechanismen	67
3.1	Einführung	67
3.2	Klassen	68
3.2.1	Konkrete Typen	68
3.2.1.1	Ein arithmetischer Typ	69
3.2.1.2	Ein Container	71
3.2.1.3	Container initialisieren	72
3.2.2	Abstrakte Typen	73
3.2.3	Virtuelle Funktionen	76
3.2.4	Klassenhierarchien	77
3.3	Kopieren und verschieben	81
3.3.1	Container kopieren	81
3.3.2	Container verschieben	83
3.3.3	Ressourcenverwaltung	85
3.3.4	Operationen unterdrücken	86
3.4	Templates	87
3.4.1	Parametrisierte Typen	87
3.4.2	Funktions-Templates	88

3.4.3	Funktionsobjekte	89
3.4.4	Variadische Templates	92
3.4.5	Alias	93
3.5	Ratschläge	94
4	Eine Rundreise durch C++: Container und Algorithmen	95
4.1	Bibliotheken	95
4.1.1	Überblick über die Standardbibliothek	96
4.1.2	Header und Namespace der Standardbibliothek	97
4.2	Strings	98
4.3	Stream-Ein-/Ausgabe	100
4.3.1	Ausgabe	100
4.3.2	Eingabe	101
4.3.3	Ein-/Ausgabe von benutzerdefinierten Typen	102
4.4	Container	104
4.4.1	vector	104
4.4.1.1	Elemente	106
4.4.1.2	Bereichsüberprüfung	106
4.4.2	list	107
4.4.3	map	109
4.4.4	unordered_map	110
4.4.5	Überblick über Container	110
4.5	Algorithmen	112
4.5.1	Iteratoren verwenden	113
4.5.2	Iteratortypen	115
4.5.3	Stream-Iteratoren	116
4.5.4	Prädikate	118
4.5.5	Überblick über Algorithmen	118
4.5.6	Containeralgorithmen	119
4.6	Ratschläge	120
5	Eine Rundreise durch C++: Nebenläufigkeit und Dienstprogramme	121
5.1	Einführung	121
5.2	Ressourcenverwaltung	121
5.2.1	unique_ptr und shared_ptr	122
5.3	Nebenläufigkeit	124
5.3.1	Tasks und Threads	125
5.3.2	Argumente übergeben	126
5.3.3	Ergebnisse zurückgeben	127
5.3.4	Daten gemeinsam nutzen	127
5.3.4.1	Warten auf Ereignisse	129
5.3.5	Kommunizierende Tasks	130

5.3.5.1	future und promise	131
5.3.5.2	packaged_task	132
5.3.5.3	async()	133
5.4	Kleine Hilfskomponenten	134
5.4.1	Zeit	134
5.4.2	Typfunktionen	135
5.4.2.1	iterator_traits	135
5.4.2.2	Typprädikate	137
5.4.3	pair und tuple	138
5.5	Reguläre Ausdrücke	139
5.6	Mathematik	140
5.6.1	Mathematische Funktionen und Algorithmen	140
5.6.2	Komplexe Zahlen	140
5.6.3	Zufallszahlen	141
5.6.4	Vektorarithmetik	143
5.6.5	Numerische Grenzen	144
5.7	Ratschläge	144
Teil II – Grundlegende Sprachelemente		145
6	Typen und Deklarationen	147
6.1	Der ISO-C++-Standard	147
6.1.1	Implementierungen	149
6.1.2	Der grundlegende Quellzeichensatz	149
6.2	Typen	150
6.2.1	Fundamentale Typen	150
6.2.2	Boolesche Typen	151
6.2.3	Zeichentypen	153
6.2.3.1	Vorzeichenbehaftete und vorzeichenlose Zeichen	155
6.2.3.2	Zeichenlitterale	156
6.2.4	Ganzzahltypen	158
6.2.4.1	Ganzzahlilitterale	158
6.2.4.2	Typen von Ganzzahliliteralen	159
6.2.5	Gleitkommatypen	160
6.2.5.1	Gleitkommalitterale	160
6.2.6	Präfixe und Suffixe	161
6.2.7	void	162
6.2.8	Größen	162
6.2.9	Ausrichtung	165
6.3	Deklarationen	166
6.3.1	Die Struktur von Deklarationen	168
6.3.2	Mehrere Namen deklarieren	169
6.3.3	Namen	170
6.3.3.1	Schlüsselwörter	171

6.3.4	Gültigkeitsbereiche	172
6.3.5	Initialisierung	174
6.3.5.1	Fehlende Initialisierer	177
6.3.5.2	Initialisierungslisten	178
6.3.6	Einen Typ herleiten: auto und decltype()	179
6.3.6.1	Der Typspezifizierer auto	179
6.3.6.2	auto und {} -Listen	180
6.3.6.3	Der Spezifizierer decltype()	181
6.4	Objekte und Werte	182
6.4.1	L-Werte und R-Werte	182
6.4.2	Lebensdauer von Objekten	183
6.5	Typalias	184
6.6	Ratschläge	185
7	Zeiger, Arrays und Referenzen	187
7.1	Einführung	187
7.2	Zeiger	187
7.2.1	void*	188
7.2.2	nullptr	189
7.3	Arrays	190
7.3.1	Array-Initialisierer	191
7.3.2	Stringlitterale	192
7.3.2.1	Rohe Zeichen-Strings	194
7.3.2.2	Größere Zeichensätze	195
7.4	Zeiger auf Arrays	196
7.4.1	Navigieren in Arrays	198
7.4.2	Mehrdimensionale Arrays	200
7.4.3	Arrays übergeben	201
7.5	Zeiger und const	203
7.6	Zeiger und Besitz	205
7.7	Referenzen	206
7.7.1	L-Wert-Referenzen	208
7.7.2	R-Wert-Referenzen	211
7.7.3	Referenzen auf Referenzen	214
7.7.4	Zeiger und Referenzen	215
7.8	Ratschläge	217
8	Strukturen, Unions und Aufzählungen	219
8.1	Einführung	219
8.2	Strukturen	219
8.2.1	Layout einer Struktur	221
8.2.2	Namen von Strukturen	222
8.2.3	Strukturen und Klassen	224

8.2.4	Strukturen und Arrays	225
8.2.5	Typäquivalenz	227
8.2.6	Plain Old Data	228
8.2.7	Felder	230
8.3	Unions	231
8.3.1	Unions und Klassen	233
8.3.2	Anonyme Unions	234
8.4	Aufzählungen	237
8.4.1	Aufzählungsklassen	237
8.4.2	Einfache Aufzählungen	241
8.4.3	Unbenannte Aufzählungen	243
8.5	Ratschläge	243
9	Anweisungen	245
9.1	Einführung	245
9.2	Zusammenfassung der Anweisungen	245
9.3	Deklarationen als Anweisungen	247
9.4	Auswahanweisungen	248
9.4.1	if -Anweisungen	248
9.4.2	switch -Anweisungen	250
9.4.2.1	Deklarationen in case -Zweigen	252
9.4.3	Deklarationen in Bedingungen	252
9.5	Schleifenanweisungen	253
9.5.1	Bereichsbasierte for -Anweisungen	254
9.5.2	for -Anweisungen	255
9.5.3	while -Anweisungen	256
9.5.4	do -Anweisungen	257
9.5.5	Schleifen verlassen	257
9.6	goto -Anweisungen	258
9.7	Kommentare und Einrückungen	259
9.8	Ratschläge	261
10	Ausdrücke	263
10.1	Einführung	263
10.2	Ein Taschenrechner	263
10.2.1	Der Parser	264
10.2.2	Eingabe	268
10.2.3	Low-level-Eingabe	272
10.2.4	Fehlerbehandlung	274
10.2.5	Das Rahmenprogramm	274
10.2.6	Header	275
10.2.7	Befehlszeilenargumente	276
10.2.8	Eine Anmerkung zum Stil	277

10.3	Zusammenfassung der Operatoren	278
10.3.1	Ergebnisse	282
10.3.2	Reihenfolge der Auswertung	283
10.3.3	Operatorrangfolge	284
10.3.4	Temporäre Objekte	285
10.4	Konstante Ausdrücke	287
10.4.1	Symbolische Konstanten	289
10.4.2	const -Typen in konstanten Ausdrücken	290
10.4.3	Literale Typen	290
10.4.4	Referenzargumente	291
10.4.5	Adresse konstanter Ausdrücke	292
10.5	Implizite Typkonvertierung	292
10.5.1	Heraufstufungen	293
10.5.2	Konvertierungen	293
10.5.2.1	Integrale Konvertierungen	294
10.5.2.2	Gleitkommakonvertierungen	294
10.5.2.3	Zeiger- und Referenzkonvertierungen	295
10.5.2.4	Zeiger-auf-Member-Konvertierungen	295
10.5.2.5	Boolesche Konvertierungen	295
10.5.2.6	Gleitkomma-Ganzzahl-Konvertierungen	296
10.5.3	Übliche arithmetische Konvertierungen	297
10.6	Ratschläge	297
11	Auswahloperationen	299
11.1	Diverse Operatoren	299
11.1.1	Logische Operatoren	299
11.1.2	Bitweise logische Operatoren	299
11.1.3	Bedingte Ausdrücke	301
11.1.4	Inkrementieren und Dekrementieren	301
11.2	Freispeicher	303
11.2.1	Speicherverwaltung	305
11.2.2	Arrays	308
11.2.3	Speicherplatz beschaffen	309
11.2.4	Überladen von new	310
11.2.4.1	nothrow new	312
11.3	Listen	313
11.3.1	Implementierungsmodell	313
11.3.2	Qualifizierte Listen	315
11.3.3	Unqualifizierte Listen	315
11.4	Lambda-Ausdrücke	317
11.4.1	Implementierungsmodelle	318
11.4.2	Alternativen für Lambda-Ausdrücke	319
11.4.3	Erfassung	321
11.4.3.1	Lambda und Lebensdauer	323

11.4.3.2	Namen von Namespaces	324
11.4.3.3	Lambda und this	324
11.4.3.4	Veränderbare Lambda-Ausdrücke	324
11.4.4	Aufruf und Rückgabe	325
11.4.5	Der Typ eines Lambda-Ausdrucks	325
11.5	Explizite Typumwandlung	326
11.5.1	Konstruktion	328
11.5.2	Benannte Typumwandlungen	329
11.5.3	C-Typumwandlungen	331
11.5.4	Funktionale Typumwandlung	331
11.6	Ratschläge	332
12	Funktionen	333
12.1	Funktionsdeklarationen	333
12.1.1	Warum Funktionen?	334
12.1.2	Bestandteile einer Funktionsdeklaration	334
12.1.3	Funktionsdefinitionen	335
12.1.4	Werte zurückgeben	337
12.1.5	Inline-Funktionen	339
12.1.6	constexpr -Funktionen	340
12.1.6.1	constexpr und Referenzen	341
12.1.6.2	Bedingte Auswertung	342
12.1.7	[[noreturn]] -Funktionen	342
12.1.8	Lokale Variablen	343
12.2	Argumentübergabe	344
12.2.1	Referenzargumente	345
12.2.2	Array-Argumente	347
12.2.3	Listenargumente	349
12.2.4	Nicht angegebene Anzahl von Argumenten	350
12.2.5	Standardargumente	354
12.3	Überladene Funktionen	356
12.3.1	Automatische Auflösung von Überladungen	356
12.3.2	Überladen und Rückgabetyt	358
12.3.3	Überladen und Gültigkeitsbereiche	359
12.3.4	Auflösung für mehrere Argumente	360
12.3.5	Manuelle Auflösung von Überladungen	360
12.4	Vor- und Nachbedingungen	361
12.5	Zeiger auf Funktion	363
12.6	Makros	367
12.6.1	Bedingte Übersetzung	370
12.6.2	Vordefinierte Makros	371
12.6.3	Pragmas	372
12.7	Ratschläge	372

13	Ausnahmenbehandlung	375
13.1	Fehlerbehandlung	375
13.1.1	Ausnahmen	376
13.1.2	Herkömmliche Fehlerbehandlung	378
13.1.3	Durchhangeln	379
13.1.4	Alternative Ansichten von Ausnahmen	380
13.1.4.1	Asynchrone Ereignisse	380
13.1.4.2	Ausnahmen, die keine Fehler sind	380
13.1.5	Wann Sie keine Ausnahmen verwenden können	381
13.1.6	Hierarchische Fehlerbehandlung	382
13.1.7	Ausnahmen und Effizienz	384
13.2	Ausnahmegarantien	386
13.3	Ressourcenverwaltung	388
13.3.1	finally	391
13.4	Invarianten erzwingen	393
13.5	Ausnahmen auslösen und abfangen	398
13.5.1	Ausnahmen auslösen	398
13.5.1.1	noexcept -Funktionen	400
13.5.1.2	Der Operator noexcept	400
13.5.1.3	Ausnahmespezifikationen	401
13.5.2	Ausnahmen abfangen	402
13.5.2.1	Ausnahmen erneut auslösen	403
13.5.2.2	Jede Ausnahme abfangen	404
13.5.2.3	Mehrere Handler	405
13.5.2.4	try -Blöcke in Funktionen	405
13.5.2.5	Beendigung	407
13.5.3	Ausnahmen und Threads	409
13.6	Eine vector -Implementierung	410
13.6.1	Ein einfacher vector	410
13.6.2	Speicher explizit darstellen	414
13.6.3	Zuweisung	417
13.6.4	Größe ändern	419
13.6.4.1	reserve()	420
13.6.4.2	resize()	421
13.6.4.3	push_back()	421
13.6.4.4	Abschließende Gedanken	422
13.7	Ratschläge	423
14	Namespaces	425
14.1	Kompositionsprobleme	425
14.2	Namespaces	426
14.2.1	Explizite Qualifizierung	428
14.2.2	using -Deklarationen	429
14.2.3	using -Direktiven	430

14.2.4	Argumentabhängige Namensauflösung	431
14.2.5	Namespaces sind offen	434
14.3	Modularisierung und Schnittstellen	435
14.3.1	Namespaces als Module	436
14.3.2	Implementierungen	438
14.3.3	Schnittstellen und Implementierungen	440
14.4	Komposition mit Namespaces	442
14.4.1	Komfort vs. Sicherheit	442
14.4.2	Namespace-Alias	443
14.4.3	Namespaces zusammensetzen	444
14.4.4	Komposition und Auswahl	445
14.4.5	Namespaces und Überladen	446
14.4.6	Versionsverwaltung	448
14.4.7	Verschachtelte Namespaces	450
14.4.8	Unbenannte Namespaces	451
14.4.9	C-Header	452
14.5	Ratschläge	453
15	Quelldateien und Programme	455
15.1	Separate Übersetzung	455
15.2	Binden	456
15.2.1	Dateilokale Namen	459
15.2.2	Header-Dateien	460
15.2.3	Die Eine-Definition-Regel	462
15.2.4	Header der Standardbibliothek	464
15.2.5	Binden mit Nicht-C++-Code	465
15.2.6	Binden und Zeiger auf Funktionen	467
15.3	Header-Dateien verwenden	468
15.3.1	Organisation mit einzeltem Header	468
15.3.2	Organisation mit mehreren Header-Dateien	472
15.3.2.1	Andere Taschenrechnermodule	475
15.3.2.2	Header verwenden	477
15.3.3	Include-Wächter	478
15.4	Programme	479
15.4.1	Initialisierung von nichtlokalen Variablen	479
15.4.2	Initialisierung und Nebenläufigkeit	480
15.4.3	Programmbeendigung	481
15.5	Ratschläge	483
Teil III	– Abstraktionsmechanismen	485
16	Klassen	487
16.1	Einführung	487
16.2	Grundlagen von Klassen	488