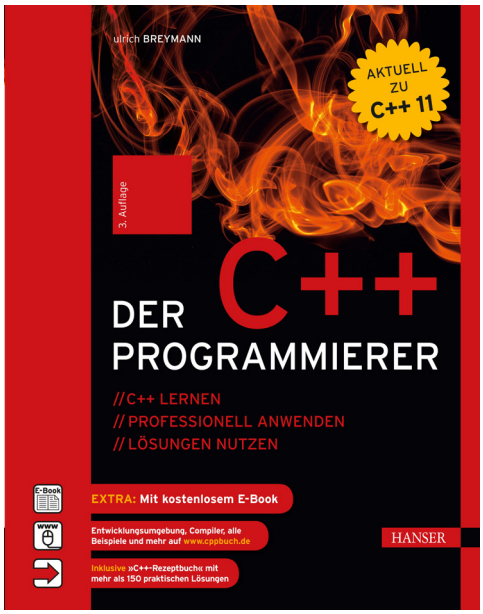
HANSER



Vorwort

zu

„Der C++-Programmierer“

(3. Auflage)

von Ulrich Breymann

ISBN (Buch): 978-3-446-43894-1
ISBN (E-Book): 978-3-446-43953-5

Weitere Informationen und Bestellungen unter
http://www.hanser-fachbuch.de/978-3-446-43894-1
sowie im Buchhandel

Vorwort

Diese Auflage dieses Buchs unterscheidet sich von der vorherigen durch die vollständige Umstellung auf den 2011 verabschiedeten ISO C++-Standard. Das Buch ist konform zum neuen C++-Standard, ohne den Anspruch auf Vollständigkeit zu erheben – das Standard-Dokument [ISOC++] umfasst allein schon mehr als 1300 Seiten. Sie finden in diesem Buch eine verständliche und mit vielen Beispielen angereicherte Einführung in die Sprache. Im Teil »Das C++ Rezeptbuch« gibt es zahlreiche Tipps und Lösungen für typische Aufgaben, die in der täglichen Praxis anfallen. Es gibt konkrete, sofort umsetzbare Lösungsvorschläge. Zahlreiche Algorithmen für praxisnahe Problemstellungen helfen bei der täglichen Arbeit. Auf größtmögliche Portabilität wird geachtet: Die Beispiele funktionieren unter Linux genau so wie unter Windows. Die problembezogene Orientierung lässt die in die Sprache einführenden Teile kürzer werden. Damit wird das Lernen erleichtert, und die Qualität des Buchs, auch als Nachschlagewerk dienen zu können, bleibt erhalten.

Für wen ist dieses Buch geschrieben?

Dieses Buch ist für alle geschrieben, die einen kompakten und gleichzeitig gründlichen Einstieg in die Konzepte und Programmierung mit C++ suchen. Es ist für Anfänger[1] gedacht, die noch keine Programmiererfahrung haben, aber auch für Programmierer, die diese Programmiersprache kennen lernen möchten. Beiden Gruppen und auch C++-Erfahrenen dient das Buch als ausführliches Nachschlagewerk.

Die ersten 11 Kapitel führen in die Sprache ein. Es wird sehr schnell ein Verständnis des objektorientierten Ansatzes entwickelt. Die sofortige praktische Umsetzung des Gelernten steht im Vordergrund. C++ wird als Programmiersprache unabhängig von speziellen Produkten beschrieben; C-Kenntnisse werden nicht vorausgesetzt. Das Buch eignet sich zum Selbststudium und als Begleitbuch zu einer Vorlesung oder zu Kursen. Die vielen Beispiele sind leicht nachzuvollziehen und praxisnah umsetzbar. Klassen und Objekte, Templates und Exceptions sind Ihnen bald keine Fremdworte mehr. Es gibt mehr als 90

[1] Geschlechtsbezogene Formen meinen hier und im Folgenden stets Männer *und* Frauen.

Übungsaufgaben – mit Musterlösungen im Anhang. Durch das Studium dieser Kapitel werden aus Anfängern bald Fortgeschrittene.

Diesen und anderen Fortgeschrittenen und Profis bietet das Buch kurze Einführungen in die Themen Thread-Programmierung, Netzwerk-Programmierung mit Sockets einschließlich eines kleinen Webservers, Datenbankanbindung, grafische Benutzungsoberflächen und mehr. Dabei wird durch Einsatz der Boost-Library und des Qt-Frameworks größtmögliche Portabilität erreicht.

Softwareentwicklung ist nicht nur Programmierung: Einführend werden anhand von Beispielen unter anderem die Automatisierung der Programmerzeugung mit Make, die Dokumentationserstellung mit Doxygen und die Versionskontrolle behandelt. Das Programmdesign wird durch konkrete Umsetzungen von UML[2]-Mustern nach C++ unterstützt. Das integrierte »C++-Rezeptbuch« mit mehr als 150 praktischen Lösungen, das sehr umfangreiche Register und das detaillierte Inhaltsverzeichnis machen das Buch zu einem praktischen Nachschlagewerk für alle, die sich mit der Softwareentwicklung in C++ beschäftigen.

Übersicht

Schwerpunkt von Teil I ist die Einführung in die Programmiersprache. Die anschließenden Teile gehen darüber hinaus und konzentrieren sich auf die verschiedenen Probleme der täglichen Praxis. Die in dieser Übersicht verwendeten Begriffe mögen Ihnen zum Teil noch unbekannt sein – in den betreffenden Kapiteln werden sie ausführlich erläutert.

Teil I – Einführung in C++

Das *Kapitel 1* vermittelt zunächst die Grundlagen, wie ein Programm geschrieben und zum Laufen gebracht wird. Es folgen einfache Datentypen und Anweisungen zur Kontrolle des Programmablaufs. Die Einführung der C++-Datentypen `vector` und `string` beendet das Kapitel. *Kapitel 2* beschäftigt sich mit der einfachen Ein- und Ausgabe, auch mit Dateien. Das *Kapitel 3* zeigt Ihnen, wie Sie Funktionen schreiben. Makros, Templates für Funktionen und die modulare Gestaltung von Programmen folgen.

Objektorientierung ist der Schwerpunkt von *Kapitel 4*. Dabei geht es nicht nur um die Konstruktion von Objekten, sondern auch um den Weg von der Problemstellung zu Klassen und Objekten. Zeiger, einfache Arrays (C-Arrays) und Zeichenketten sowie die Erzeugung von Objekten zur Laufzeit sind Inhalt von *Kapitel 5*. Dazu kommen mehrdimensionale C-Arrays und das Schreiben und Lesen von Binärdaten in bzw. aus Dateien. Auf der Basis von Zeigern führt *Kapitel 6* das Thema Objektorientierung fort. Dabei lernen Sie kennen, wie eine String-Klasse funktioniert, und wie Sie Klassen-Templates und Templates mit einer variablen Anzahl von Parametern konstruieren. Das *Kapitel 7* zeigt Ihnen das Mittel objektorientierter Sprachen, um Generalisierungs- und Spezialisierungsbeziehungen auszudrücken: die Vererbung mit ihren Möglichkeiten. Strategien zur Fehlerbehandlung mit Exceptions finden Sie in *Kapitel 8*. Das *Kapitel 9* zeigt, wie Sie Operatorsymbolen wie + und - eigene Bedeutungen zuweisen können und in welchem Zusammenhang das sinnvoll ist. Sie lernen, »intelligente« Zeiger (Smart Pointer) zu konstruieren und Objekte als Funktionen einzusetzen. *Kapitel 10* beschreibt ausführlich die

[2] UML ist eine grafische Beschreibungssprache für die objektorientierte Programmierung.

Ein- und Ausgabemöglichkeiten, die in Kapitel 2 nur einführend gestreift werden, einschließlich der Fehlerbehandlung und der Formatierung der Ausgabe. Eine Einführung in die Standard Template Library (STL) bietet *Kapitel 11*. Es zeigt, wie die Komponenten (Container, Iteratoren und Algorithmen) zusammenwirken. Die STL und ihre Wirkungsweise bilden die Grundlage eines sehr großen Teils der C++-Standardbibliothek.

Die Kapitel 1 bis 11 sind für ein gutes Verständnis von C++ unverzichtbar. Reguläre Ausdrücke (*Kapitel 12*) und die Programmierung paralleler Abläufe mit Threads (*Kapitel 13*) sind dazu nicht notwendig – hierbei handelt es sich um Ergänzungen, wie sie von vielen Programmiersprachen angeboten werden.

Teil II – Bausteine komplexer Anwendungen

Ein Programm benötigt eine Möglichkeit, mit der Außenwelt zu kommunizieren. Tastatur und Konsole allein reichen für komplexe Anwendungen in der Regel nicht aus. Mausbedienung und Bildschirmgrafik sind heute Standard bei Desktop-Anwendungen. Das *Kapitel 14* zeigt, wie grafische Benutzungsschnittstellen konstruiert werden. Wie ein Programm die Verbindung mit dem Internet aufnehmen kann, dokumentiert das *Kapitel 15*. Und wohin mit den ganzen Daten, die bei Programmende nicht verloren gehen sollen? In *Kapitel 16* lernen Sie, wie ein Programm an eine Datenbank angebunden wird. Die genannten Themen sind so umfangreich, dass sie selbst Bücher füllen. Dieser Teil bietet Ihnen daher nur einen Einstieg.

Teil III – Praktische Methoden und Werkzeuge der Softwareentwicklung

Die Entwicklung von Programmen besteht nicht nur im Schreiben von Code. Die Compilation eines Projekts mit vielen Programmdateien und Abhängigkeiten kann schnell ein komplexer Vorgang werden. Die Automatisierung dieses Prozesses mit dem Tool *make* ist Thema von *Kapitel 17*. Programme sind nicht auf Anhieb fehlerfrei. Sie müssen getestet werden: *Kapitel 18* stellt ein Werkzeug für den Unit-Test vor und zeigt den praktischen Einsatz. *Kapitel 19* demonstriert ein Werkzeug zur automatischen Dokumentationserstellung und geht kurz auf die Versionsverwaltung und die Projektverwaltung ein.

Teil IV – Das C++-Rezeptbuch: Tipps und Lösungen für typische Aufgaben

Sichere Programmentwicklung ist die Überschrift des *Kapitels 20*. Sie finden dort Regeln zum Design von Methoden und mehrere Tipps zur defensiven Programmierung, die die Risiken falscher Algorithmen oder falscher Benutzung vermindern. Auch gibt es Tipps zur exception-sicheren Beschaffung von Speicher und zur Thread-Programmierung. *Kapitel 21* zeigt Rezepte, wie Sie bestimmte UML-Muster in C++-Konstruktionen umwandeln können. *Kapitel 22* erklärt den Unterschied zwischen Wert- und Referenzsemantik und die Auswirkung auf die Geschwindigkeit von C++-Programmen. Es werden Empfehlungen gegeben und am Beispiel konkretisiert, wie die Performanz deutlich verbessert werden kann. *Kapitel 23* erweitert die Grundlagen des Kapitels 17 um praktische Rezepte zur automatischen Ermittlung von Abhängigkeiten zwischen Programmdateien, Makefiles für Verzeichnisbäume, die automatische Erzeugung von Makefiles und statischer und dynamischer Bibliotheken. Algorithmen für viele verschiedene Aufgaben finden Sie in *Kapitel 24*. Wegen der Vielzahl empfiehlt sich ein Blick in das Inhaltsverzeichnis, um einen Überblick zu gewinnen. Der C++-Standard bietet für viele Datei- und Verzeichnis-

operationen keine Unterstützung an. *Kapitel 25* enthält deshalb fertige Rezepte zum Anlegen, Löschen und Lesen von Verzeichnissen und mehr auf der Basis der Boost-Library. Ergänzt wird dies durch das formatierte Lesen und Schreiben von Daten und die Abspeicherung binärer Daten als Block.

Teil V – Die C++-Standardbibliothek

In mehreren Kapiteln wird die C++-Standardbibliothek in Kürze beschrieben. Die Inhalte dieses Teils sind: Hilfsfunktionen und -klassen, Container, Iteratoren, Algorithmen, Einstellung nationaler Besonderheiten, String, Speicherverwaltung, Funktionen der Programmiersprache C.

Anhang

Der Anhang enthält unter anderem verschiedene hilfreiche Tabellen und die Lösungen der Übungsaufgaben.

Wo finden Sie was?

Bei der Programmentwicklung wird häufig das Problem auftauchen, etwas nachschlagen zu müssen. Es gibt die folgenden Hilfen:

Erklärungen zu Begriffen sind im *Glossar* ab Seite 957 aufgeführt.

Es gibt ein recht umfangreiches *Stichwortverzeichnis* ab Seite 971 und ein sehr detailliertes *Inhaltsverzeichnis*.

Software zum Buch

Auf der Webseite *http://www.cppbuch.de/* finden Sie die Software zu diesem Buch. Sie enthält unter anderem einen Compiler, eine Integrierte Entwicklungsumgebung sowie alle Programmbeispiele und die Lösungen zu den Aufgaben. Sie finden dort auch weitere Hinweise, Errata und nützliche Links.

Zu guter Letzt

Allen Menschen, die dieses Buch durch Hinweise und Anregungen verbessern halfen, sei an dieser Stelle herzlich gedankt. Frau Brigitte Bauer-Schiewek und Frau Irene Weilhart vom Hanser Verlag danke ich für die sehr gute Zusammenarbeit.

Bremen, im Januar 2014

Ulrich Breymann