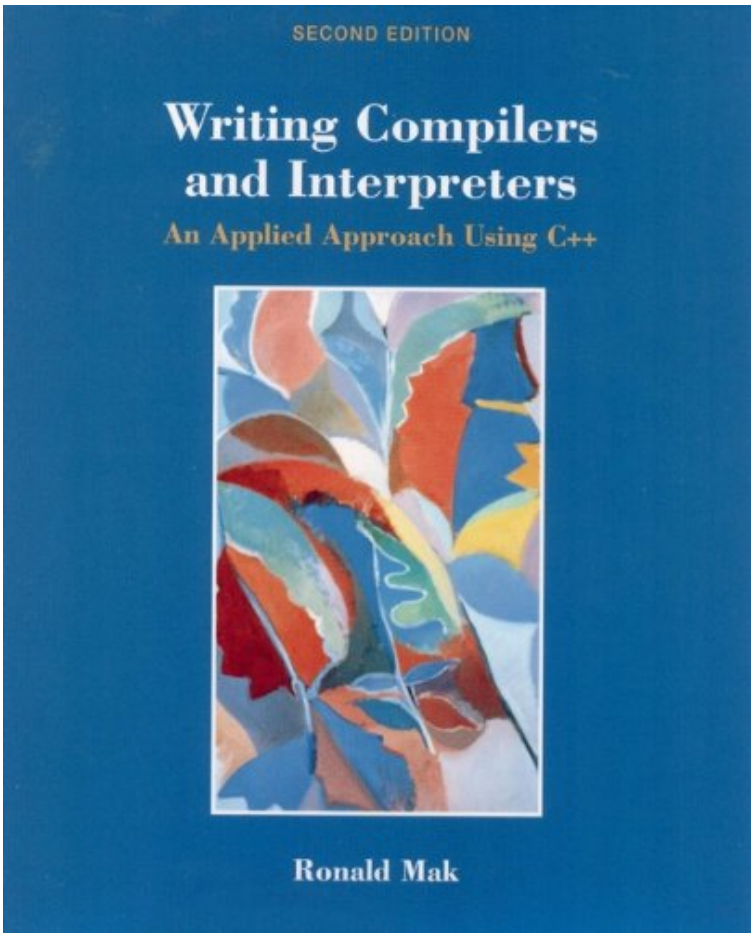
Writing Compilers and Interpreters: An Applied Approach Using C++

Von Ronald Mak
**Download PDF | ePub | DOC | audiobook | ebooks*



Produktinformation -Verkaufsrang: #1020841 in eBooksVerffentlicht am: 2008-05-05Erscheinungsdatum: 2008-05-05File Name: B000V5WH5K | File size: 47.Mb

Von Ronald Mak : Writing Compilers and Interpreters: An Applied Approach Using C++ before purchasing it in order to gage whether or not it would be worth my time, and all praised Writing Compilers and Interpreters: An Applied Approach Using C++:

KundenrezensionenHilfreichste Kundenrezensionen2 von 2 Kunden fanden die folgende Rezension hilfreich. Schritt fr Schritt zum Pascal-CompilerVon Dr. Chrilly DonningerIch schreibe - aus Spass an der Sache - fast fr jedes Projekt eine eigene "Little Language". Das nennt sich heute hochtrabend "Domain-Specific-Language" (DSL). Ich habe das Buch auf der Suche nach Tricks fr eine neue Skriptsprache durchgelesen.Es fngt mit einem pretty-printer an und endet mit einem - weitgehend vollstndigen - LL(1) Pascal Compiler. In jeden Schritt prsentiert der Autor den vollstndigen sehr gut dokumentierten und lesbaren C++ Kode (und nicht nur einzelne Kode-Fragemente). Der C++ Stil ist "C with Classes". Der Author verzichtet z.B. auf Templates, Exceptions und die STL. Wobei ich mir nicht mehr sicher bin, ob es diese C++ Features im Erscheinungsjahr (1996) berhaupt schon gab. Meines Erachtens wrden sie sie den Kode aber

nicht besser, sondern nur ineffizienter machen.Die schrittweise Einfhrung erzeugt notwendiger Weise etwas Redundanz. Das Buch hat 840 Seiten. N.Wirth: Compilerbau hat - bei vergleichbaren Inhalt - 120 Seiten. Wirth ist dafr extra-dry.LL(1) vulgo Recursive-Descent ist nicht so mchtig wie die von YACC/Bison untersttze LR() Grammatik. Fr DSLs ist das wesentlich einfachere LL(1) aber vollkommen ausreichend. Als Alternative bietet sich dass LL(*) Compiler Tool ANTLR an.Ich habe in dem Buch tatschlich ein paar nette Tricks gelernt. Das einzige was mir am Design nicht gefallen hat war die Implementierung der Symboltabelle als Binrbaum. Das kann man viel einfacher als Tabelle realisieren. Der Geschwindikeitsunterschied ist bei modernen Prozessoren irrelevant. Die mit Abstand zeitraubenste Routine ist das Einlesen des Programmierkodes von der Festplatte bzw die - im Buch nicht behandelte - Optimierung. Wenn man den Zugriff aber schon unbedingt beschleunigen will, muss man es als Hashtabelle implementieren. Binrbume sind ausserdem nur bei relativ zuflligem Input eine gute Datenstruktur. Variablennamen in einem Programm sind aber alles andere als zufllig (ein Problem mit dem auch Hashtabellen zu kmpfen haben). Fr DSLs die maximal ein paar 100 Zeilen lang sind, ist das aber alles vollkommen unntiger Firlefanz.1 von 1 Kunden fanden die folgende Rezension hilfreich. A good introduction to compiler basicsVon csmoThis book delivers exactly what it promises--a complete step-by-step example of writing 'a compiler'. The book is simply a description of one way to build one compiler (and interpreter, and debugger, and various useful utilities).The basics are well presented. First a topic is described, then source code is presented and explained. The results of test runs are shown, and then off to the next topic. Advanced topics, such as optimization, are intentially left out.When a person is ready to read a first book about compilers, this is a good one. All source code developed/described in the book is available on-line.0 von 0 Kunden fanden die folgende Rezension hilfreich. Good, Maybe a little too focused on one solution.Von gary.bell@cad.shorts.co.ukThe book describes step-by-step how the author would write a compiler for PASCAL. It could do with some more explanations of the logic behind some of the decisions,as it tends to quickly explain what the following C++ code does,before launching into pages of (well written) programming. If you have been tasked to write a specific compiler, then this book is probably what you want to get. If you are wanting to further your knowledge of the art, then you would be better looking at some of the more weighty volumes.

KurzbeschreibungQuickly master all the skills you need to build your own compilers and interpreters in C++Whether you are a professional programmer who needs to write a compiler at work or a personal programmer who wants to write an interpreter for a language of your own invention, this book quickly gets you up and running with all the knowledge and skills you need to do it right. It cuts right to the chase with a series of skill-building exercises ranging in complexity from the basics of reading a program to advanced object-oriented techniques for building a compiler in C++.Here's how it works:Every chapter contains anywhere from one to three working utility programs that provide a firsthand demonstration of concepts discussed, and each chapter builds upon the preceding ones. You begin by learning how to read a program and produce a listing, deconstruct a program into tokens (scanning), and how to analyze it based on its syntax (parsing). From there, Ron Mak shows you step by step how to build an actual working interpreter and an interactive debugger. Once you've mastered those skills, you're ready to apply them to building a compiler that runs on virtually any desktop computer.KurzbeschreibungQuickly master all the skills you need to build your own compilers and interpreters in C++Whether you are a professional programmer who needs to write a compiler at work or a personal programmer who wants to write an interpreter for a language of your own invention, this book quickly gets you up and running with all the knowledge and skills you need to do it right. It cuts right to the chase with a series of skill-building exercises ranging in complexity from the basics of reading a program to advanced object-oriented techniques for building a compiler in C++.Here's how it works:Every chapter contains anywhere from one to three working utility programs that provide a firsthand demonstration of concepts discussed, and each chapter builds upon the preceding ones. You begin by learning how to read a program and produce a listing, deconstruct a program into tokens (scanning), and how to analyze it based on its syntax (parsing). From there, Ron Mak shows you step by step how to build an actual working interpreter and an interactive debugger. Once you've mastered those skills, you're ready to apply them to building a compiler that runs on virtually any desktop computer.Synopsis Quickly master all the skills you need to build your own compilers and interpreters in C++ Whether you are a professional programmer who needs to write a compiler at work or a personal programmer who wants to write an interpreter for a language of your own invention, this book quickly gets you up and running with all the knowledge and skills you need to do it right. It cuts right to the chase with a series of skill-building exercises ranging in complexity from the basics of reading a program to advanced object-oriented techniques for building a compiler in C++. Here's how it works: Every chapter contains anywhere from one to three working utility programs that provide a firsthand demonstration of concepts discussed, and each chapter builds upon the preceding ones. You begin by learning how to read a program and produce a listing, deconstruct a program into tokens (scanning), and how to analyze it based on its syntax (parsing). From there, Ron Mak shows you step by step how to build an actual working interpreter and an interactive debugger. Once you've mastered those skills, you're ready to apply them to building a compiler that runs on virtually any desktop

computer.Visit the Wiley Computer Books Web page at: http://www wiley.com/compbooks/