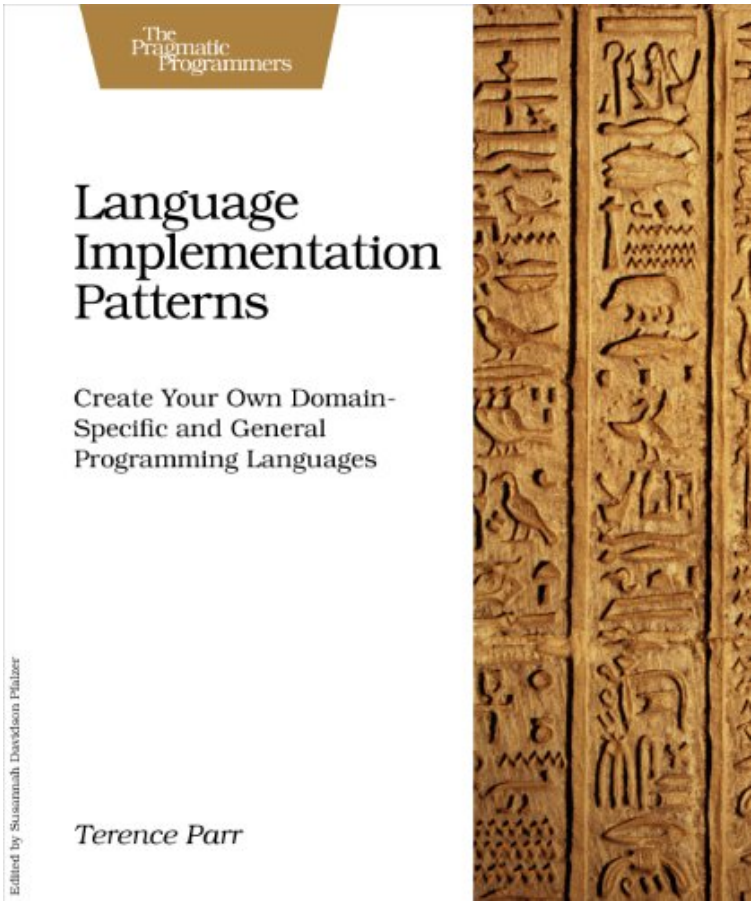
# Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages (Pragmatic Programmers)

*Von Terence Parr*
*ePub | *DOC | audiobook | ebooks | Download PDF*






Produktinformation -Verkaufsrang: #272486 in eBooksVerffentlicht am: 2009-12-31Erscheinungsdatum: 2012-11-06File Name: B00A376HGG | File size: 28.Mb

**Von Terence Parr : Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages (Pragmatic Programmers)** before purchasing it in order to gage whether or not it would be worth my time, and all praised Language Implementation Patterns: Create Your Own Domain-Specific and General Programming Languages (Pragmatic Programmers):

KurzbeschreibungLearn to build configuration file readers, data readers, model-driven code generators, source-to-

source translators, source analyzers, and interpreters. You don't need a background in computer science--ANTLR creator Terence Parr demystifies language implementation by breaking it down into the most common design patterns. Pattern by pattern, you'll learn the key skills you need to implement your own computer languages.Knowing how to create domain-specific languages (DSLs) can give you a huge productivity boost. Instead of writing code in a general-purpose programming language, you can first build a custom language tailored to make you efficient in a particular domain.The key is understanding the common patterns found across language implementations. Language Design Patterns identifies and condenses the most common design patterns, providing sample implementations of each.The pattern implementations use Java, but the patterns themselves are completely general. Some of the implementations use the well-known ANTLR parser generator, so readers will find this book an excellent source of ANTLR examples as well. But this book will benefit anyone interested in implementing languages, regardless of their tool of choice. Other language implementation books focus on compilers, which you rarely need in your daily life. Instead, Language Design Patterns shows you patterns you can use for all kinds of language applications.You'll learn to create configuration file readers, data readers, model-driven code generators, source-to-source translators, source analyzers, and interpreters. Each chapter groups related design patterns and, in each pattern, you'll get hands-on experience by building a complete sample implementation. By the time you finish the book, you'll know how to solve most common language implementation problems.Pressestimmen""Throw away your compiler theory book! Terence Parr shows how to write practical parsers, translators, interpreters, and other language applications using modern tools and design patterns. Whether you're designing your own DSL or mining existing code for bugs or gems, you'll find example code and suggested patterns in this clearly written book about all aspects of parsing technology.""--Guido van Rossum, Creator of the Python language""This text is excellent. The exposition plus the examples makes otherwise complex ideas very clear and accessible. Well done!""--Tom Nurkkala, Associate Professor, Computer Science and Engineering, Taylor UniversityKurzbeschreibungLearn to build configuration file readers, data readers, model-driven code generators, source-to-source translators, source analyzers, and interpreters. You don't need a background in computer science--ANTLR creator Terence Parr demystifies language implementation by breaking it down into the most common design patterns. Pattern by pattern, you'll learn the key skills you need to implement your own computer languages.Knowing how to create domain-specific languages (DSLs) can give you a huge productivity boost. Instead of writing code in a general-purpose programming language, you can first build a custom language tailored to make you efficient in a particular domain.The key is understanding the common patterns found across language implementations. Language Design Patterns identifies and condenses the most common design patterns, providing sample implementations of each.The pattern implementations use Java, but the patterns themselves are completely general. Some of the implementations use the well-known ANTLR parser generator, so readers will find this book an excellent source of ANTLR examples as well. But this book will benefit anyone interested in implementing languages, regardless of their tool of choice. Other language implementation books focus on compilers, which you rarely need in your daily life. Instead, Language Design Patterns shows you patterns you can use for all kinds of language applications.You'll learn to create configuration file readers, data readers, model-driven code generators, source-to-source translators, source analyzers, and interpreters. Each chapter groups related design patterns and, in each pattern, you'll get hands-on experience by building a complete sample implementation. By the time you finish the book, you'll know how to solve most common language implementation problems.