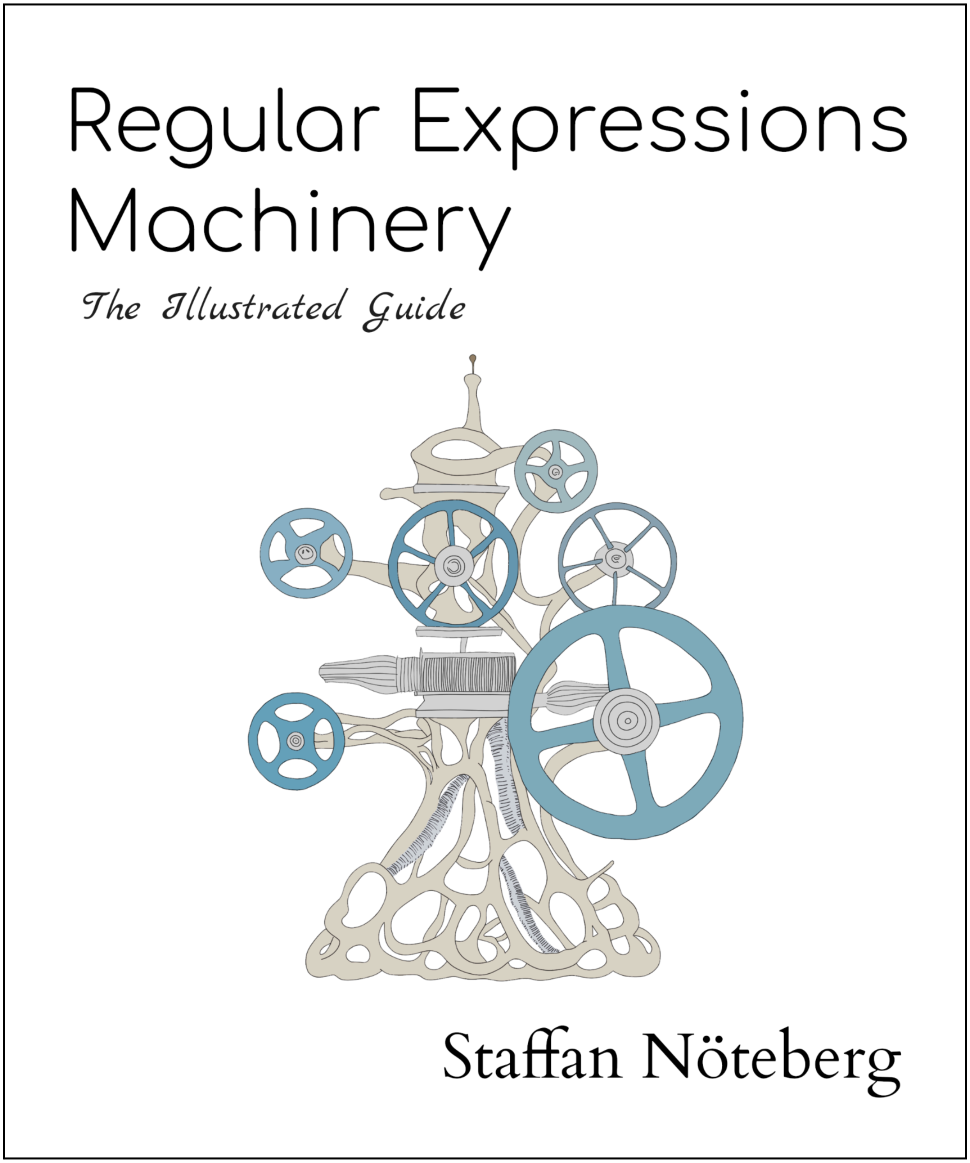

Regular Expressions Machinery

The Illustrated Guide

Staffan Nöteberg

This short excerpt is from ***Regular Expressions Machinery: The Illustrated Guide*** by Staffan Nöteberg. You can find more information or purchase an ebook copy at https://www.pragprog.com.

While every effort has been made to ensure the accuracy of the information contained herein, the author and the publisher assumes no responsibility for any errors or omissions, or for any damages resulting from the use of this publication.

To inquire about booking the author for podcasts, training sessions, and speaking engagements at conferences, please contact him directly at staffan.noteberg@rekursiv.se

PDF ISBN: 978-91-989983-0-6
EPUB ISBN: 978-91-989983-1-3
Book Version: V1.0—January, 2025

Introduction

> *"I define UNIX as 30 definitions of regular expressions living under one roof."*

Turing Award winner Donald Knuth said that, and it doesn't stop there. All mainstream programming languages host regular expressions as a domain-specific language (DSL) and there are numerous books about them. However, the book you're holding in your hand right now is very different.

Why You Might Find This Book Valuable

This book is written for those who want to deeply understand regular expressions. Maybe you've come across them in programming languages or tools but never really understood how they work under the hood. Even though the book is based on advanced mathematics, you don't need any prior knowledge.

Here are some examples of people who could benefit from the book:

- **Application developers:** If you regularly use regular expressions in languages like Java, C#, Python, or Ruby, you'll be able to write more efficient, accurate, and readable expressions after reading this book.
- **Data analysts:** If you work with text analysis or data mining, you can use regular expressions to identify patterns and extract information from large datasets.
- **Software developers focused on compilers and DSLs:** For those who develop compilers or domain-specific languages, this book provides insights into how regular expressions can be used to define and interpret languages.
- **System and network administrators:** If you administer systems or networks, you can use regular expressions to analyze log files, search for specific patterns, and automate tasks.

- **Computer science students:** If you're taking a course on automata theory or compiler design, this book will give you a concrete and illustrated introduction to regular expressions and the underlying automata theory.

By reading this book, you will:

- **Understand the theoretical basis of regular expressions:** You'll learn about finite automata, states, transitions, and how they relate to regular expressions.
- **Learn to construct and interpret complex regular expressions:** The book covers advanced concepts such as greediness, backtracking, quantifiers, and lookarounds.
- **Write more efficient and readable expressions:** You'll learn to avoid common pitfalls and write expressions that are easy to understand and maintain.
- **Use regular expressions in different contexts:** The book provides examples of how regular expressions can be used to solve problems in various domains.

Unlike many other books that focus solely on syntax, this book starts with a deep dive into the underlying automata theory. This provides you with a deeper understanding of how regular expressions work on a fundamental level. Clear illustrations and straightforward explanations make university-level mathematics accessible for everyone.

The goal of the book is to give you a deep understanding of regular expressions so that you can use them with confidence and efficiency in your work.

A Short Map of the Book

The book is divided into four parts, and my strong recommendation is that you read them in order.

Part I: The Automaton: We begin by looking at the theoretical foundation of regular expressions: finite automata. We will both delve into states and transitions, and also learn what deterministic and non-deterministic finite automata are, and why it is important when you develop your own regular expressions.

Part II: Two Operations and One Function: Here we go through the two operations and the only function needed to create regular expressions. Spoiler alert! The operations are concatenation and alternation. The function is the Kleene star. Can that really be enough to describe all regular expressions? You'll know when you have finished this PART II.

Part III: Syntactic Sugar, Abstractions, and Extensions: The third part is more like a traditional regex book. We look at both the basic and the more advanced features of regular expressions, such as quantifiers, groups, and lookarounds. We also examine some of the most common pitfalls.

Part IV: Test-Driven Regex Development: In the final part, we explore the concept of test-driven development (TDD) and how you can leverage it to write better regexes. By writing tests before you write the actual regexes, you can ensure that your regexes are correct and easy to maintain. We will also look at how to use TDD to refactor legacy regexes.

Again, I strongly recommend reading the book in the presented order. After all, that's why you're here: Your goal is to understand the regex machinery first and then apply that understanding to the syntax in Part III.

Time to lift the hood and look inside the engine. The exciting story of the automaton begins on the next page.

Get a Grip on the Regex Machinery

To effectively use regular expressions, you need to understand how the machinery works under the hood. It’s about taking control of the search process, controlling how the pattern is matched, and thus getting both faster and more accurate results.

In this illustrated guide, you gain precisely that understanding.

You can even get started without any prior knowledge of regular expressions. Before you know it, advanced tools like reluctant, lookbehind and nondeterministic finite automata will be at your fingertips as you write efficient and elegant regexes with ease.

This book presents complex concepts in a simple and visual way, with clear examples and practical applications. Whether you are a programmer, data analyst, or just want to get better at text processing, this book will take your knowledge to the next level.

Staffan Nöteberg is a bestselling author passionate about helping people become more efficient and focused. He is the author of these popular books:

- Pomodoro Technique Illustrated: The Easy Way to Do More in Less Time
- Monotasking: How to Focus Your Mind and Be More Productive
- Guiding Star OKRs: A New Approach to Setting and Achieving Goals

With a background in software development and an interest in productivity, Staffan combines his expertise with an ability to explain complex topics in an easy-to-understand way.

PDF ISBN: 978-91-989983-0-6

9 789198 998306

EPUB ISBN: 978-91-989983-1-3

9 789198 998313

Introduction

> *"I define UNIX as 30 definitions of regular expressions living under one roof."*

Turing Award winner Donald Knuth said that, and it doesn't stop there. All mainstream programming languages host regular expressions as a domain-specific language (DSL) and there are numerous books about them. However, the book you're holding in your hand right now is very different.

Why You Might Find This Book Valuable

This book is written for those who want to deeply understand regular expressions. Maybe you've come across them in programming languages or tools but never really understood how they work under the hood. Even though the book is based on advanced mathematics, you don't need any prior knowledge.

Here are some examples of people who could benefit from the book:

- **Application developers:** If you regularly use regular expressions in languages like Java, C#, Python, or Ruby, you'll be able to write more efficient, accurate, and readable expressions after reading this book.
- **Data analysts:** If you work with text analysis or data mining, you can use regular expressions to identify patterns and extract information from large datasets.
- **Software developers focused on compilers and DSLs:** For those who develop compilers or domain-specific languages, this book provides insights into how regular expressions can be used to define and interpret languages.
- **System and network administrators:** If you administer systems or networks, you can use regular expressions to analyze log files, search for specific patterns, and automate tasks.

- **Computer science students:** If you're taking a course on automata theory or compiler design, this book will give you a concrete and illustrated introduction to regular expressions and the underlying automata theory.

By reading this book, you will:

- **Understand the theoretical basis of regular expressions:** You'll learn about finite automata, states, transitions, and how they relate to regular expressions.
- **Learn to construct and interpret complex regular expressions:** The book covers advanced concepts such as greediness, backtracking, quantifiers, and lookarounds.
- **Write more efficient and readable expressions:** You'll learn to avoid common pitfalls and write expressions that are easy to understand and maintain.
- **Use regular expressions in different contexts:** The book provides examples of how regular expressions can be used to solve problems in various domains.

Unlike many other books that focus solely on syntax, this book starts with a deep dive into the underlying automata theory. This provides you with a deeper understanding of how regular expressions work on a fundamental level. Clear illustrations and straightforward explanations make university-level mathematics accessible for everyone.

The goal of the book is to give you a deep understanding of regular expressions so that you can use them with confidence and efficiency in your work.

A Short Map of the Book

The book is divided into four parts, and my strong recommendation is that you read them in order.

Part I: The Automaton: We begin by looking at the theoretical foundation of regular expressions: finite automata. We will both delve into states and transitions, and also learn what deterministic and non-deterministic finite automata are, and why it is important when you develop your own regular expressions.

Part II: Two Operations and One Function: Here we go through the two operations and the only function needed to create regular expressions. Spoiler alert! The operations are concatenation and alternation. The function is the Kleene star. Can that really be enough to describe all regular expressions? You'll know when you have finished this PART II.

Part III: Syntactic Sugar, Abstractions, and Extensions: The third part is more like a traditional regex book. We look at both the basic and the more advanced features of regular expressions, such as quantifiers, groups, and lookarounds. We also examine some of the most common pitfalls.

Part IV: Test-Driven Regex Development: In the final part, we explore the concept of test-driven development (TDD) and how you can leverage it to write better regexes. By writing tests before you write the actual regexes, you can ensure that your regexes are correct and easy to maintain. We will also look at how to use TDD to refactor legacy regexes.

Again, I strongly recommend reading the book in the presented order. After all, that's why you're here: Your goal is to understand the regex machinery first and then apply that understanding to the syntax in Part III.

Time to lift the hood and look inside the engine. The exciting story of the automaton begins on the next page.