Exploring Languages with Interpreters and Functional Programming Chapter Index

H. Conrad Cunningham

27 April 2022

Contents

Professor of Computer and Information Science
University of Mississippi
214 Weir Hall
P.O. Box 1848
University, MS 38677
(662) 915-7396 (dept. office)

Browser Advisory: The HTML version of this textbook requires a browser that supports the display of MathML. A good choice as of April 2022 is a recent version of Firefox from Mozilla.

Instability Warning: This version of ELIFP is in work beginning in January 2022. The author may change its structure and content without warning. No changes are planned to the 2018 version upon which the this version is based.

The stable version is on the Fall 2018 CSci 450 course website.

Feedback Request: The author plans to publish this textbook eventually. He invites anyone using this book to give him feedback on its current structure and content: to point out typos and other errors or suggest improvements and extensions. He can be contacted at hcc AT cs DOT olemiss DOT edu.

Exploring Languages with Interpreters and Functional Programming

- **Single file of partial textbook:** as HTML
  as PDF

Chapter 0: Preface

- **Chapter:** as HTML
  as PDF
- Slides: NONE

Chapter 1: Evolution of Programming Languages

- **Chapter:** as HTML
  as PDF
- **Slides:** (HTML) Evolving Computer Hardware Affects Programming Languages
  (HTML) History of Programming Languages

Chapter 2: Programming Paradigms

- **Chapter:** as HTML
  as PDF
- **Slides:** (HTML) Programming Paradigm
  (HTML) Programming Paradigms Scala Version

Chapter 3: Object-Based Paradigms

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Object-Based Paradigms

Chapter 4: First Haskell Programs

- **Chapter:** as HTML
  as PDF

- Slides: (HTML) First Haskell Functions

Chapter 5: Types

- **Chapter:** as HTML
  as PDF
- **Slides:** (HTML) Type System Concepts
  No slides yet for 5.3

Chapter 6: Procedural Abstraction

- **Chapter:** as HTML
  as PDF
- **Slides:** (HTML) Top-Down Stepwise Refinement
  [(HTML) Modular Design and Programming
- Other: (Haskell) Quick Overview of Basic Haskell

Chapter 7: Data Abstraction

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Using Data Abstraction

Chapter 8: Evaluation Model

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Evaluation of Functional Programs

Chapter 9: Recursion Styles and Efficiency

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Recursion Styles

Chapter 10: Simple Input and Output (FUTURE)

- as HTML as PDF
- Slides: NONE YET

Chapter 11: Software Testing Concepts

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 12: Testing Haskell Programs

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 13: List Programming

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) List Programming

Chapter 14: Infix Operators and List Examples

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Infix Operations and List Examples

Chapter 15: Higher-Order Functions

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Higher-Order List Programming

Chapter 16: Haskell Function Concepts

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Haskell Function Concepts

Chapter 17: Higher-Order Function Examples

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Higher Order Function Examples

Chapter 18: More List Processing

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) More List Processing

Chapter 19: Systematic Generalization

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 20: Problem Solving

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 21: Algebraic Data Types

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Algebraic Data Types
- **Domain Specific Languages** as HTML
  as PDF

Chapter 22: Data Abstraction Revisited

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 23: Overloading and Type Classes

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Overloading and Type Classes
- **Movable Objects case study:** MovableObjects module
  NamedObjects module
  NamedMovableObjects module
  NamedMovableTest module skeleton

Chapter 24: FUTURE

< - Chapter: : as HTML : as PDF

- Slides: NONE YET

Chapter 25: Proving Haskell Laws

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 26: Program Synthesis

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 27: Text Processing

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 28: Type Inference

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 29: Models of Reduction

TODO: The figure drawing need to be changed. Current ones only appear in PDF.

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 30: Infinite Data Structures

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Future Chapter 31

Future Chapter 32

Future Chapter 33

Future Chapter 34

Future Chapter 35

Future Chapter 36

Future Chapter 37

Future Chapter 38

Future Chapter 39

Chapter 40: Language Processing (FUTURE)

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

Chapter 41: Calculator: Concrete Syntax

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Calculator Concrete Syntax
- **Expression Tree Case Study** as HTML
  as PDF
- Fall 2016 Lua Expression Language 1 interpreter folder

Chapter 42: Calculator: Abstract Syntax & Evaluation

- **Chapter:** [as HTML
  as PDF
- Slides: (HTML) Calculator: Abstract Syntax & Evaluation

Chapter 43: Calculator: Modular Structure

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Calculator: Modular Structure

Chapter 44: Calculator: Parsing

- **Chapter:** as HTML
  as PDF
- Slides: (HTML) Calculator: Parsing

Chapter 45: Parser Combinators

- **Chapter:** as HTML
  as PDF
- Slides: None Yet

Chapter 46: Calculator: Compilation

- **Chapter:** as HTML
  as PDF
- Slides: None Yet

Chapter 47 Imperative Core Language

- **Chapter:** as HTML
  as PDF
- Fall 2017-18 ELI ImpCore interpreter modules (prefix syntax) code mostly works but needs a bit of update to match recent changes to ELI Calculator
  - REPL module
  - Recursive descent parser module
  - Lexical analyzer module
  - Abstract Syntax module
  - Evaluator module
  - Environments module
  - Values module
  - Test Imp Core(in work, not current)
- Fall 2016 Lua Imperative Core interpreter folder
- Fall 2013 Kamin Interpreter in Lua Toolset (KILT) folder
- Kamin-Budd Interpreters original source code folder

Chapter 80: (Appendix) Review of Relevant Mathematics

- **Chapter:** as HTML
  as PDF
- Slides: NONE YET

UNDER DEVELOPMENT

OLD Chapter 4: List Programming Supplements

Most of the content of this old chapter went into new chapters 13 and 14, but some was moved to earlier chapters.

- Possible future material on modular programming based on:
  - Modular Design
  - Data Abstraction
  - Cookie Jar Abstract Data Type

Future Chapter? Using Algebraic Data Types

- **TODO**: Regular Expressions using algebraic data types
- Framework Design Using Function Generalization: A Binary Tree Traversal Case Study

Future Chapter? Domain Specific Languages

- **Domain Specific Languages** as HTML
  as PDF

Future Chapters? Games

- Wizard's Adventure game (Elixir)
- Dice of Doom (Elixir)

Other Possible Topics

- **Software Patterns:** [as HTML]{<../Patterns/Patterns_index.html>)
  [as PDF]{<../Patterns/Patterns_index.pdf>)

Acknowledgements

I began this effort in Summer 2016 by adapting previous materials from my courses on Functional Programming (primarily), Multiparadigm Programming, Object-Oriented Programming, Software Architecture, Software Families, and Software Language Engineering.

I added new materials in Spring and Summer 2017 to draft the 2017 version of the textbook titled *Introduction to Functional Programming Using Haskell*.

In Spring and Summer 2018, I began work on an updated 2018 version of the textbook, now titled *Exploring Languages with Interpreters and Functional Programming*. I broke several of the longer chapters into 2-4 new chapters or appendices. I incorporated new material from my Spring 2018 Software Language

Engineering class (e.g. Type Concepts). I also wrote new chapters including the two new chapters on Software Testing.

I retired from the full-time faculty in 2019, so I am no longer made changes driven by my teaching schedule, but I do plan to continue to evolve the textbook.

The 2022 version is a work in progress. I plan to adapt some class materials from 2018 and before that has not yet been incorporated, to make improvements in existing chapters, and to complete missing sections and chapters.

I maintain this textbook as text files in Pandoc's dialect of Markdown using embedded LaTeX markup for the mathematical formulas and then translate the documents to HTML, PDF, and other formats as needed.

References