2014

Exploring C++

Alice E. Fischer
University of New Haven, AFischer@newhaven.edu

Publisher Citation

Fischer, Alice E. (2009). Exploring C++.

Comments

The Anatomy of Programming Languages

Alice E. Fischer
University of New Haven

Frances S. Grodzinsky
Sacred Heart University

September 1992

Contents

Preface

This text is intended for a course in advanced programming languages or the structure of programming language and should be appropriate for students at the junior, senior, or master's level. It should help the student understand the principles that underlie all languages and all language implementations.

This is a comprehensive text which attempts to dissect language and explain how a language is really built. The first eleven chapters cover the core material: language specification, objects, expressions, control, and types. The more concrete aspects of each topic are presented first, followed by a discussion of implementation strategies and the related semantic issues. Later chapters cover current topics, including modules, object-oriented programming, functional languages, and concurrency constructs.

The emphasis throughout the text is on semantics and abstraction; the syntax and historical development of languages are discussed in light of the underlying semantical concepts. Fundamental principles of computation, communication, and good design are stated and are used to evaluate various language constructs and to demonstrate that language designs are improving as these principles become widely understood.

Examples are cited from many languages including Pascal, C, C++, FORTH, BASIC, LISP, FORTRAN, Ada, COBOL, APL, Prolog, Turing, Miranda, and Haskell. All examples are annotated so that a student who is unfamiliar with the language used can understand the meaning of the code and see how it illustrates the principle.

It is the belief of the authors that the student who has a good grasp of the structure of computer languages will have the tools to master new languages easily.

The specific goals of this book are to help students learn:

- To reason clearly about programming languages.
- To develop principles of communication so that we can evaluate the wisdom and utility of the decisions made in the process of language design.
- To break down language into its major components, and each component into small pieces so that we can focus on competing alternatives.
- To define a consistent and general set of terms for the components out of which programming languages are built, and the concepts on which they are based.

- To use these terms to describe existing languages, and in so doing clarify the conflicting terminology used by the language designers, and untangle the complexities inherent in so many languages.
- To see below the surface appearance of a language to its actual structure and descriptive power.
- To understand that many language features that commonly occur together are, in fact, independent and separable. To appreciate the advantages and disadvantages of each feature. To suggest ways in which these basic building blocks can be recombined in new languages with more desirable properties and fewer faults.
- To see the similarities and differences that exist among languages students already know, and to learn new ones.
- To use the understanding so gained to suggest future trends in language design.

Acknowledgement

The authors are indebted to several people for their help and support during the years we have worked on this project. First, we wish to thank our families for their uncomplaining patience and understanding.

We thank Michael J. Fischer for his help in developing the sections on lambda calculus, functional languages and logic. and for working out several sophisticated code examples. In addition, his assistance as software and hardware systems expert and TeX guru made this work possible.

Several reviewers read this work in detail and offered invaluable suggestions and corrections. We thank these people for their help. Special thanks go to Robert Fischer and Roland Lieger for reading beyond the call of duty and to Gary Walters for his advice and for the material he has contributed.

Finally, we thank our students at the University of New Haven and at Sacred Heart University for their feedback on the many versions of this book.

Parts of this manuscript were developed under a grant from Sacred Heart University.