The Malevolent Ingenuity of Preprocessor Metaprogramming in C

Temirkhan Myrzamadi

September 18, 2020

ii

Contents

0.1 Foreword

The macro system is probably the most contradictory feature of C: it provides a way to extend the host programming language with new syntactical constructs, thus forcing some ill-formed programs fail to compile, but nevertheless is an increasingly treacherous technology, being able to potentially cause a numerous amount of bugs out of nothing.

This book was derived by my continuous experiments with the preprocessor: I have implemented some of the concepts of functional programming, including algebraic data types (ADTs) with exhaustive pattern matching, parametric polymorphism, higher-kinded types, and generalised ADTs, as well as from the world of object orientation: classes, dynamic dispatch, and all the coherent stuff: type introspection, type-safe error handling facilities and more.

So what I want to say: writing macros is an art. Macros, overall, can even be considered as computer languages on their own, with custom syntax, given by their signatures, and semantics, defined by their bodies. In this book, I have accomplished my best concerning the development of macros, and I do believe that it will advance a reader to a completely new dimension of metaprogramming, show how much more beautiful macro interfaces can be, and develop the intuition in such a tricky topic.

0.2 Audience

This book is targeted mostly at programmers who feel comfortable with the
C programming language. No additional prerequsites are required, though,
some experience with macros might be helpful.

0.3   Outline

So to sum up everything discussed above, the aim of this book is to show how to achieve all the benefits of the macro system of C and to overcome its drawbacks as much as possible. It comprises of lots of examples, case studies, various techniques and tricks. All the explanations and code blocks are C11-compliant, unless otherwise stated.

Now let's give a short outline of all the chapters: