Introduction

It can be argued that the current free-software movement is the most important thing happening in computing today. We are in the midst of a major shift from all software being proprietary and closely held by individual companies to a large body of software that can be freely acquired and used by anyone for any purpose. Free software now includes not only programming language compilers and linkers, but numerous utilities, graphical user interface environments, and even entire operating systems.

Add all this to the fact that virtually all free software is compiled by GCC, and it can be argued that GCC is the most important piece of software in the world. Of course, programs are written in many languages, and there are compilers for these languages, but for the most part these compilers are written and compiled using GCC. At some point, all free software harks back to GCC. Some computer companies have begun to drop support for their own compilers and simply install GCC instead. It's free for the taking and is constantly being extended and maintained.

With the addition of the two latest languages to the GCC family—Java and Ada—the GCC compiler is spreading its wings even further. This brings the total number of active languages in GCC to six: C, C++, Objective-C, Fortran, Java, and Ada. Development is in progress on other languages, such as COBOL, and they will be added to GCC if there is enough support behind them.

Milestones

The GNU Project was launched in 1984 for the purpose of developing a free operating system. Richard Stallman is the founder of the GNU Project and the original author of GCC.

The initial release of the first beta of GCC, release number 0.9, was on March 22, 1987. The first actual release, version 1.0, was on May 23, 1987. In all there have been 108 releases from the very beginning to the release on which this book is based—version 3.1, released on May 5, 2002. That's an average of one release every 1.7 months for the last 15 years.

What's Inside?

The purpose of this book is to provide information to those wishing to use GCC for software development. A good bit of information can be found about GCC internals that can be used to get you started in the direction of working inside the compiler, but the main idea behind this book is to guide you through the steps of installing and using the compiler to develop software. Any way that you care to measure software, GCC is huge. And like most huge software systems, it contains useful features that you can use only if you discover that they exist, determine exactly what it is they do, and figure out how to use them. That's the primary purpose of this book.

The book is divided into three parts. Part I, "The Free Software Compiler," serves as an introduction to the fundamentals of the compiler and includes instructions you can follow to download and install it. Part II, "Using the Compiler Collection," contains detailed instructions for using the compiler. A chapter is dedicated to each of the six programming languages, with several examples of each. Special chapters are included to describe the preprocessor and techniques for linking objects produced from different languages. Part III, "Peripherals and Internals," includes chapters on linking, debugging, cross-compiling, makefiles, and the GNU assembler. Part III also contains information on the inner workings of both the front end and back end of the compiler.

GCC is the world's champion in the number of command-line options available. These options are listed alphabetically in Appendix D and cross-referenced in Appendix C. Chapter 21 contains even more command-line options—the ones that have to do with the specific computer hardware for which the compiler is generating code.

To give you a better idea of the topics covered in this book, here's a short description of each chapter:

- Chapter 1 is a general introduction to the fundamental concepts of GCC, including a list of its parts and the languages it compiles.
- Chapter 2 contains procedures you can use to install GCC.
- Chapter 3 describes the workings of the preprocessor and how you can employ it to process the source code of a language.

- Chapter 4 contains examples of compiling and linking C.
- Chapter 5 contains examples of compiling and linking C++.
- Chapter 6 contains examples of compiling and linking Objective-C.
- Chapter 7 contains examples of compiling and linking Fortran.
- Chapter 8 contains examples of compiling and linking Java.
- Chapter 9 contains examples of compiling and linking Ada.
- Chapter 10 contains examples of mixing two languages to create a single executable.
- Chapter 11 explains how the internationalization facilities can be employed in your compiled program to allow its displayed strings to be modified to fit a locale.
- Chapter 12 contains examples of producing and using static and shared libraries.
- Chapter 13 explains the fundamentals of using the GNU debugger.
- Chapter 14 describes the use of `make` and its associated utilities.
- Chapter 15 discusses the GNU assembler and describes how you can use it in conjunction with GCC.
- Chapter 16 describes the process required to configure GCC to compile and link programs to be executed on another computer.
- Chapter 17 describes how GCC can be used to produce code for an embedded system.
- Chapter 18 contains examples of generating useful output from the compiler other than object code.
- Chapter 19 describes the rudiments of using `lex` and `yacc` to create a language front end for GCC.
- Chapter 20 describes the content of the intermediate language produced by the compiler front end and read by the compiler back end.
- Chapter 21 contains a list of the command-line options that apply versions of GCC running on specific hardware.
- Appendix A contains a copy of the GNU Public License.
- Appendix B lists the environment variables that effect GCC.
- Appendix C is a cross-reference of the command-line options by category.
- Appendix D is an alphabetical listing of the command-line options.
- Appendix E is a glossary.