Preface

Several kinds of tasks occur repeatedly when working with text files. You might want to extract certain lines and discard the rest. Or you may need to make changes wherever certain patterns appear, but leave the rest of the file alone. Writing single-use programs for these tasks in languages such as C, C++, or Pascal is time-consuming and inconvenient. Such jobs are often easier with *awk*. The *awk* utility interprets a special-purpose programming language that makes it easy to handle simple data-reformatting jobs.

The GNU implementation of *awk* is called *gawk*; it is fully compatible with the System V Release 4 version of *awk*. *gawk* is also compatible with the POSIX specification of the *awk* language. This means that all properly written *awk* programs should work with *gawk*. Thus, we usually don't distinguish between *gawk* and other *awk* implementations.

Using *awk* allows you to:

- Manage small, personal databases
- Generate reports
- Validate data
- Produce indexes and perform other document preparation tasks
- Experiment with algorithms that you can adapt later to other computer languages

9 October 2001 01:40

In addition, *gawk* provides facilities that make it easy to:

- Extract bits and pieces of data for processing
- Sort data
- Perform simple network communications

This book teaches you about the *awk* language and how you can use it effectively. You should already be familiar with basic system commands, such as *cat* and *ls*,* as well as basic shell facilities, such as input/output (I/O) redirection and pipes.

Implementations of the *awk* language are available for many different computing environments. This book, while describing the *awk* language in general, also describes the particular implementation of *awk* called *gawk* (which stands for "GNU awk"). *gawk* runs on a broad range of Unix systems, ranging from 80386 PC-based computers up through large-scale systems, such as Crays. *gawk* has also been ported to Mac OS X, MS-DOS, Microsoft Windows (all versions) and OS/2 PCs, Atari and Amiga microcomputers, BeOS, Tandem D20, and VMS.

History of awk and gawk

The name *awk* comes from the initials of its designers: Alfred V. Aho, Peter J. Weinberger, and Brian W. Kernighan. The original version of *awk* was written in 1977 at AT&T Bell Laboratories. In 1985, a new version made the programming language more powerful, introducing user-defined functions, multiple input streams, and computed regular expressions. This new version became widely available with Unix System V Release 3.1 (SVR3.1). The version in SVR4 added some new features and cleaned up the behavior in some of the "dark corners" of the language. The specification for *awk* in the POSIX Command Language and Utilities standard further clarified the language. Both the *gawk* designers and the original Bell Laboratories *awk* designers provided feedback for the POSIX specification.

Paul Rubin wrote the GNU implementation, *gawk*, in 1986. Jay Fenlason completed it, with advice from Richard Stallman. John Woods contributed parts of the code as well. In 1988 and 1989, David Trueman, with help from me, thoroughly reworked *gawk* for compatibility with the newer *awk*. Circa 1995, I became the primary maintainer. Current development focuses on bug fixes, performance improvements, standards compliance, and occasionally, new features.

---

* These commands are available on POSIX-compliant systems, as well as on traditional Unix-based systems. If you are using some other operating system, you still need to be familiar with the ideas of I/O redirection and pipes.

In May of 1997, Jürgen Kahrs felt the need for network access from *awk*, and with a little help from me, set about adding features to do this for *gawk*. At that time, he also wrote the bulk of *TCP/IP Internetworking with gawk* (a separate document, available as part of the *gawk* distribution). Chapter 14, *Internetworking with gawk*, is condensed from that document. His code finally became part of the main *gawk* distribution with *gawk* Version 3.1.

See Appendix A, *The Evolution of the awk Language*, for a complete list of those who made important contributions to *gawk*.

A Rose by Any Other Name

The *awk* language has evolved over the years. Full details are provided in Appendix A. The language described in this book is often referred to as "new *awk*" (*nawk*).

Because of this, many systems have multiple versions of *awk*. Some systems have an *awk* utility that implements the original version of the *awk* language and a *nawk* utility for the new version.* Others have an *oawk* version for the "old *awk*" language and plain *awk* for the new one. Still others only have one version, which is usually the new one.†

All in all, this makes it difficult for you to know which version of *awk* you should run when writing your programs. The best advice I can give here is to check your local documentation. Look for *awk*, *oawk*, and *nawk*, as well as for *gawk*. It is likely that you already have some version of new *awk* on your system, which is what you should use when running your programs. (Of course, if you're reading this book, chances are good that you have *gawk*!)

Throughout this book, whenever we refer to a language feature that should be available in any complete implementation of POSIX *awk*, we simply use the term *awk*. When referring to a feature that is specific to the GNU implementation, we use the term *gawk*.

Using This Book

The term *awk* refers to a particular program as well as to the language you use to tell this program what to do. When we need to be careful, we call the language "the *awk* language," and the program "the *awk* utility." This book explains both the *awk* language and how to run the *awk* utility. The term *awk program* refers to a program written by you in the *awk* programming language.

---

* Of particular note is Sun's Solaris, where */usr/bin/awk* is, sadly, still the original version. Use */usr/xpg4/bin/awk* to get a POSIX-compliant version of *awk* on Solaris.

† Often, these systems use *gawk* for their *awk* implementation!

Primarily, this book explains the features of *awk*, as defined in the POSIX standard. It does so in the context of the *gawk* implementation. While doing so, it also attempts to describe important differences between *gawk* and other *awk* implementations.* Finally, any *gawk* features that are not in the POSIX standard for *awk* are noted.

This book has the difficult task of being both a tutorial and a reference. If you are a novice, feel free to skip over details that seem too complex. You should also ignore the many cross-references; they are for the expert user and for the online info version of the document.

There are sidebars scattered throughout the book. They add a more complete explanation of points that are relevant, but not likely to be of interest on first reading. All appear in the index, under the heading "advanced features."

Most of the time, the examples use complete *awk* programs. In some of the more advanced sections, only the part of the *awk* program that illustrates the concept currently being described is shown.

While this book is aimed principally at people who have not been exposed to *awk*, there is a lot of information here that even the *awk* expert should find useful. In particular, the description of POSIX *awk* and the example programs in Chapter 12, *A Library of awk Functions*, and in Chapter 13, *Practical awk Programs*, should be of interest.

Chapter 1, *Getting Started with awk*, provides the essentials you need to know to begin using *awk*.

Chapter 2, *Regular Expressions*, introduces regular expressions in general, and in particular the flavors supported by POSIX *awk* and *gawk*.

Chapter 3, *Reading Input Files*, describes how *awk* reads your data. It introduces the concepts of records and fields, as well as the `getline` command. I/O redirection is first described here.

Chapter 4, *Printing Output*, describes how *awk* programs can produce output with `print` and `printf`.

Chapter 5, *Expressions*, describes expressions, which are the basic building blocks for getting most things done in a program.

Chapter 6, *Patterns, Actions, and Variables*, describes how to write patterns for matching records, actions for doing something when a record is matched, and the built-in variables *awk* and *gawk* use.

---

* All such differences appear in the index under the entry "differences in *awk* and *gawk*."

Chapter 7, *Arrays in awk*, covers *awk*'s one-and-only data structure: associative arrays. Deleting array elements and whole arrays is also described, as well as sorting arrays in *gawk*.

Chapter 8, *Functions*, describes the built-in functions *awk* and *gawk* provide, as well as how to define your own functions.

Chapter 9, *Internationalization with gawk*, describes special features in *gawk* for translating program messages into different languages at runtime.

Chapter 10, *Advanced Features of gawk*, describes a number of *gawk*-specific advanced features. Of particular note are the abilities to have two-way communications with another process, perform TCP/IP networking, and profile your *awk* programs.

Chapter 11, *Running awk and gawk*, describes how to run *gawk*, the meaning of its command-line options, and how it finds *awk* program source files.

Chapter 12, *A Library of awk Functions*, and Chapter 13, *Practical awk Programs*, provide many sample *awk* programs. Reading them allows you to see *awk* solving real problems.

Chapter 14, *Internetworking with gawk*, provides an in-depth discussion and examples of how to use *gawk* for Internet programming.

Appendix A, *The Evolution of the awk Language*, describes how the *awk* language has evolved since first release to present. It also describes how *gawk* has acquired features over time.

Appendix B, *Installing gawk*, describes how to get *gawk*, how to compile it under Unix, and how to compile and use it on different PC operating systems. It also describes how to report bugs in *gawk* and where to get three other freely available implementations of *awk*.

Appendix C, *Implementation Notes*, describes how to disable *gawk*'s extensions, as well as how to contribute new code to *gawk*, how to write extension libraries, and some possible future directions for *gawk* development.

Appendix D, *Basic Programming Concepts*, provides some very cursory background material for those who are completely unfamiliar with computer programming. Also centralized there is a discussion of some of the issues surrounding floating-point numbers.

Appendix E, *GNU General Public License*, and Appendix F, *GNU Free Documentation License*, present the licenses that cover the *gawk* source code and this book, respectively.

The Glossary defines most, if not all, the significant terms used throughout the book. If you find terms that you aren't familiar with, try looking them up here.

Typographical Conventions

The following typographical conventions are used in this book:

Italic
: Used to show generic arguments and options; these should be replaced with user-supplied values. Italic is also used to highlight comments in examples. In the text, italic indicates commands, filenames, options, and the first occurrences of important terms.

`Constant width`
: Used for code examples, inline code fragments, and variable and function names.

`Constant width italic`
: Used in syntax summaries and examples to show replaceable text; this text should be replaced with user-supplied values. It is also used in the text for the names of control keys.

`Constant width bold`
: Used in code examples to show commands or other text that the user should type literally.

`$`, `>`
: The `$` indicates the standard shell's primary prompt. The `>` indicates the shell's secondary prompt, which is printed when a command is not yet complete.

[ ] Surround optional elements in a description of syntax. (The brackets themselves should never be typed.)

When you see the owl icon, you know the text beside it is a note.

On the other hand, when you see the turkey icon, you know the text beside it is a warning.

Dark Corners

Until the POSIX standard (and *The Gawk Manual*), many features of *awk* were either poorly documented or not documented at all. Descriptions of such features (often called "dark corners") are noted in this book with "(d.c.)". They also appear in the index under the heading "dark corner."

Any coverage of dark corners is, by definition, something that is incomplete.

The GNU Project and This Book

The Free Software Foundation (FSF) is a nonprofit organization dedicated to the production and distribution of freely distributable software. It was founded by Richard M. Stallman, the author of the original Emacs editor. GNU Emacs is the most widely used version of Emacs today.

The GNU* Project is an ongoing effort on the part of the Free Software Foundation to create a complete, freely distributable, POSIX-compliant computing environment. The FSF uses the "GNU General Public License" (GPL) to ensure that their software's source code is always available to the end user. A copy of the GPL is included in this book for your reference (see Appendix E). The GPL applies to the C language source code for *gawk*. To find out more about the FSF and the GNU Project online, see the GNU Project's home page at *http://www.gnu.org*. This book may also be read from their documentation web site at *http://www.gnu.org/manual/gawk/*.

Until the GNU operating system is more fully developed, you should consider using GNU/Linux, a freely distributable, Unix-like operating system for Intel 80386, DEC Alpha, Sun SPARC, IBM S/390, and other systems.† There are many books on GNU/Linux. One that is freely available is *Linux Installation and Getting Started* by Matt Welsh (Specialized Systems Consultants). Another good book is *Learning Debian GNU/Linux* by Bill McCarty (O'Reilly). Many GNU/Linux distributions are often available in computer stores or bundled on CD-ROMs with books about Linux. (There are three other freely available, Unix-like operating systems for 80386 and other systems: NetBSD, FreeBSD, and OpenBSD. All are based on the 4.4-Lite Berkeley Software Distribution, and they use recent versions of *gawk* for their versions of *awk*.)

The book you are reading is actually free—at least, the information in it is free to anyone. The machine-readable source code for the book comes with *gawk*; anyone may take this book to a copying machine and make as many copies as they like. (Take a moment to check the Free Documentation License in Appendix F.)

---

* GNU stands for "GNU's not Unix."

† The terminology "GNU/Linux" is explained in the Glossary.

Although you could just print it out yourself, bound books are much easier to read and use. Furthermore, part of the proceeds from sales of this book go back to the FSF to help fund development of more free software. In keeping with the GNU Free Documentation License, O'Reilly & Associates is making the DocBook version of this book available on their web site (*http://www.oreilly.com/catalog/awkprog3*). They also contributed significant editorial resources to the book, which were folded into the Texinfo version distributed with *gawk*.

The book itself has gone through a number of previous editions. Paul Rubin wrote the very first draft of *The GAWK Manual*; it was around 40 pages in size. Diane Close and Richard Stallman improved it, yielding a version that was around 90 pages long and barely described the original, "old" version of *awk*.

I started working with that version in the fall of 1988. As work on it progressed, the FSF published several preliminary versions (numbered 0.*x*). In 1996, Edition 1.0 was released with *gawk* 3.0.0. SSC published the first two editions of *Effective awk Programming*, and the FSF published the same two editions under the title *The GNU Awk User's Guide*.

This edition maintains the basic structure of Edition 1.0, but with significant additional material, reflecting the host of new features in *gawk* Version 3.1. Of particular note is the section "Sorting Array Values and Indices with gawk" in Chapter 7, as well as the section "Bit-Manipulation Functions of gawk" in Chapter 8, all of Chapter 9 and Chapter 10, and the section "Adding New Built-in Functions to gawk" in Appendix C.

Effective awk Programming will undoubtedly continue to evolve. An electronic version comes with the *gawk* distribution from the FSF. If you find an error in this book, please report it! See the section "Reporting Problems and Bugs" in Appendix B for information on submitting problem reports electronically, or write to me in care of the publisher.

How to Contribute

As the maintainer of GNU *awk*, I am starting a collection of publicly available *awk* programs. For more information, see *ftp://ftp.freefriends.org/arnold/Awkstuff*. If you have written an interesting *awk* program, or have written a *gawk* extension that you would like to share with the rest of the world, please contact me (*arnold@gnu.org*). Making things available on the Internet helps keep the *gawk* distribution down to manageable size.

Acknowledgments

The initial draft of *The GAWK Manual* had the following acknowledgments:

> Many people need to be thanked for their assistance in producing this manual. Jay Fenlason contributed many ideas and sample programs. Richard Mlynarik and Robert Chassell gave helpful comments on drafts of this manual. The paper *A Supplemental Document for awk*, by John W. Pierce of the Chemistry Department at UC San Diego, pinpointed several issues relevant both to *awk* implementation and to this manual, that would otherwise have escaped us.

I would like to acknowledge Richard M. Stallman, for his vision of a better world and for his courage in founding the FSF and starting the GNU Project.

The following people (in alphabetical order) provided helpful comments on various versions of this book, up to and including this edition. Rick Adams, Nelson H.F. Beebe, Karl Berry, Dr. Michael Brennan, Rich Burridge, Claire Cloutier, Diane Close, Scott Deifik, Christopher ("Topher") Eliot, Jeffrey Friedl, Dr. Darrel Hankerson, Michal Jaegermann, Dr. Richard J. LeBlanc, Michael Lijewski, Pat Rankin, Miriam Robbins, Mary Sheehan, and Chuck Toporek.

Robert J. Chassell provided much valuable advice on the use of Texinfo. Karl Berry helped significantly with the TEX part of Texinfo.

I would like to thank Marshall and Elaine Hartholz of Seattle and Dr. Bert and Rita Schreiber of Detroit for large amounts of quiet vacation time in their homes, which allowed me to make significant progress on this book and on *gawk* itself.

Phil Hughes of SSC contributed in a very important way by loaning me his laptop GNU/Linux system, not once, but twice, which allowed me to do a lot of work while away from home. I would also like to thank Phil for publishing the first two editions of this book, and for getting me started as a technical author.

David Trueman deserves special credit; he has done a yeoman job of evolving *gawk* so that it performs well and without bugs. Although he is no longer involved with *gawk*, working with him on this project was a significant pleasure.

The intrepid members of the GNITS mailing list, and most notably Ulrich Drepper, provided invaluable help and feedback for the design of the internationalization features.

Nelson Beebe, Martin Brown, Scott Deifik, Darrel Hankerson, Michal Jaegermann, Jürgen Kahrs, Pat Rankin, Kai Uwe Rommel, and Eli Zaretskii (in alphabetical order) are long-time members of the *gawk* "crack portability team." Without their hard work and help, *gawk* would not be nearly the fine program it is today. It has been and continues to be a pleasure working with this team of fine people.

David and I would like to thank Brian Kernighan of Bell Laboratories for invaluable assistance during the testing and debugging of *gawk*, and for help in clarifying numerous points about the language. We could not have done nearly as good a job on either *gawk* or its documentation without his help.

Michael Brennan, author of *mawk*, contributed the Foreword, for which I thank him. Perhaps one of the most rewarding aspects of my long-term work with *gawk* has been the friendships it has brought me, both with Michael and with Brian Kernighan.

A special thanks to Chuck Toporek of O'Reilly & Associates for thoroughly editing this book and shepherding the project through its various stages.

I must thank my wonderful wife, Miriam, for her patience through the many versions of this project, for her proofreading, and for sharing me with the computer. I would like to thank my parents for their love, and for the grace with which they raised and educated me. Finally, I also must acknowledge my gratitude to G-d, for the many opportunities He has sent my way, as well as for the gifts He has given me with which to take advantage of those opportunities.

Arnold Robbins
Nof Ayalon
ISRAEL
March, 2001