

O'REILLY®

第4版
A GNU Manual



Effective awk Programming

高效awk编程 (影印版)

東南大學出版社

Arnold Robbins 著

第4版

高效awk编程 (影印版)

Effective awk Programming



Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo

O'REILLY®

O'Reilly Media, Inc. 授权东南大学出版社出版

南京 东南大学出版社

图书在版编目(CIP)数据

高效 awk 编程:第4版:英文/(美)罗宾斯(Robbins, A.)著. —影印本. —南京:东南大学出版社,2015.9

书名原文:Effective awk Programming, 4E

ISBN 978-7-5641-5912-2

I. ①高… II. ①罗… III. ①UNIX 操作系统—英文 IV. ①TP316.81

中国版本图书馆 CIP 数据核字(2015)第 165784 号

江苏省版权局著作权合同登记

图字:10-2015-240 号

© 2015 by O'Reilly Media, Inc.

Reprint of the English Edition, jointly published by O'Reilly Media, Inc. and Southeast University Press, 2015. Authorized reprint of the original English edition, 2015 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由 O'Reilly Media, Inc. 出版 2015。

英文影印版由东南大学出版社出版 2015。此影印版的出版和销售得到出版权和销售权的所有者——O'Reilly Media, Inc. 的许可。

版权所有,未得书面许可,本书的任何部分和全部不得以任何形式重制。

高效 awk 编程 第4版(影印版)

出版发行:东南大学出版社

地 址:南京四牌楼2号 邮编:210096

出 版 人:江建中

网 址:<http://www.seupress.com>

电子邮件:press@seupress.com

印 刷:常州市武进第三印刷有限公司

开 本:787毫米×980毫米 16开本

印 张:35

字 数:685千字

版 次:2015年9月第1版

印 次:2015年9月第1次印刷

书 号:ISBN 978-7-5641-5912-2

定 价:98.00元

本社图书若有印装质量问题,请直接与营销部联系。电话(传真):025-83791830

To my parents, for their love, and for the wonderful example they set for me.

*To my wife Miriam, for making me complete. Thank you for building your life together
with me.*

*To our children Chana, Rivka, Nachum, and Malka, for enrichening our lives in
innumerable ways.*

Foreword to the Third Edition

Arnold Robbins and I are good friends. We were introduced in 1990 by circumstances—and our favorite programming language, `awk`. The circumstances started a couple of years earlier. I was working at a new job and noticed an unplugged Unix computer sitting in the corner. No one knew how to use it, and neither did I. However, a couple of days later, it was running, and I was root and the one-and-only user. That day, I began the transition from statistician to Unix programmer.

On one of many trips to the library or bookstore in search of books on Unix, I found the gray `awk` book, a.k.a. Alfred V. Aho, Brian W. Kernighan, and Peter J. Weinberger's *The AWK Programming Language* (Addison-Wesley, 1988). `awk`'s simple programming paradigm—find a pattern in the input and then perform an action—often reduced complex or tedious data manipulations to a few lines of code. I was excited to try my hand at programming in `awk`.

Alas, the `awk` on my computer was a limited version of the language described in the gray book. I discovered that my computer had “old `awk`” and the book described “new `awk`.” I learned that this was typical; the old version refused to step aside or relinquish its name. If a system had a new `awk`, it was invariably called `nawk`, and few systems had it. The best way to get a new `awk` was to ftp the source code for `gawk` from `prep.ai.mit.edu`. `gawk` was a version of new `awk` written by David Trueman and Arnold, and available under the GNU General Public License.

(Incidentally, it's no longer difficult to find a new `awk`. `gawk` ships with GNU/Linux, and you can download binaries or source code for almost any system; my wife uses `gawk` on her VMS box.)

My Unix system started out unplugged from the wall; it certainly was not plugged into a network. So, oblivious to the existence of `gawk` and the Unix community in general, and desiring a new `awk`, I wrote my own, called `mawk`. Before I was finished, I knew

about `gawk`, but it was too late to stop, so I eventually posted to a `comp.sources.news` group.

A few days after my posting, I got a friendly email from Arnold introducing himself. He suggested we share design and algorithms and attached a draft of the POSIX standard so that I could update `mawk` to support language extensions added after publication of *The AWK Programming Language*.

Frankly, if our roles had been reversed, I would not have been so open and we probably would have never met. I'm glad we did meet. He is an `awk` expert's `awk` expert and a genuinely nice person. Arnold contributes significant amounts of his expertise and time to the Free Software Foundation.

This book is the `gawk` reference manual, but at its core it is a book about `awk` programming that will appeal to a wide audience. It is a definitive reference to the `awk` language as defined by the 1987 Bell Laboratories release and codified in the 1992 POSIX Utilities standard.

On the other hand, the novice `awk` programmer can study a wealth of practical programs that emphasize the power of `awk`'s basic idioms: data-driven control flow, pattern matching with regular expressions, and associative arrays. Those looking for something new can try out `gawk`'s interface to network protocols via special `/inet` files.

The programs in this book make clear that an `awk` program is typically much smaller and faster to develop than a counterpart written in C. Consequently, there is often a payoff to prototyping an algorithm or design in `awk` to get it running quickly and expose problems early. Often, the interpreted performance is adequate and the `awk` prototype becomes the product.

The new `pgawk` (profiling `gawk`) produces program execution counts. I recently experimented with an algorithm that for n lines of input exhibited $\sim Cn^2$ performance, while theory predicted $\sim Cn \log n$ behavior. A few minutes poring over the `awk` `prof.out` profile pinpointed the problem to a single line of code. `pgawk` is a welcome addition to my programmer's toolbox.

Arnold has distilled over a decade of experience writing and using `awk` programs, and developing `gawk`, into this book. If you use `awk` or want to learn how, then read this book.

—Michael Brennan
Author of mawk
March 2001

Foreword to the Fourth Edition

Some things don't change. Thirteen years ago I wrote: "If you use `awk` or want to learn how, then read this book." True then, and still true today.

Learning to use a programming language is about more than mastering the syntax. One needs to acquire an understanding of how to use the features of the language to solve practical programming problems. A focus of this book is many examples that show how to use `awk`.

Some things do change. Our computers are much faster and have more memory. Consequently, speed and storage inefficiencies of a high-level language matter less. Prototyping in `awk` and then rewriting in `C` for performance reasons happens less, because more often the prototype is fast enough.

Of course, there are computing operations that are best done in `C` or `C++`. With `gawk` 4.1 and later, you do not have to choose between writing your program in `awk` or in `C/C++`. You can write most of your program in `awk` and the aspects that require `C/C++` capabilities can be written in `C/C++`, and then the pieces glued together when the `gawk` module loads the `C/C++` module as a dynamic plug-in. Chapter 16 has all the details, and, as expected, many examples to help you learn the ins and outs.

I enjoy programming in `awk` and had fun (re)reading this book. I think you will, too.

—Michael Brennan
Author of mawk
October 2014

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. Such jobs are often easy 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`; if you invoke it with the proper options or environment variables, it is fully compatible with the POSIX¹ specification of the `awk` language and with the Unix version of `awk` maintained by Brian Kernighan. This means that all properly written `awk` programs should work with `gawk`. So most of the time, we don't distinguish between `gawk` and other `awk` implementations.

Using `awk` you can:

- 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

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

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

1. The 2008 POSIX standard is accessible online (<http://www.opengroup.org/onlinepubs/9699919799/>).

- Profile and debug awk programs
- Extend the language with functions written in C or C++

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`,² 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 Intel-architecture PC-based computers up through large-scale systems. `gawk` has also been ported to Mac OS X, Microsoft Windows (all versions), and OpenVMS.³

History of awk and gawk

Recipe for a Programming Language

1 part `egrep`

1 part `snobol`

2 parts `ed`

3 parts C

Blend all parts well using `lex` and `yacc`. Document minimally and release.

After eight years, add another part `egrep` and two more parts C. Document very well and release.

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 (1987). The version in System V Release 4 (1989) 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.

2. These utilities 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.
3. Some other, obsolete systems to which `gawk` was once ported are no longer supported and the code for those systems has been removed.

Both the `gawk` designers and the original `awk` designers at Bell Laboratories provided feedback for the POSIX specification.

Paul Rubin wrote `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 1994, I became the primary maintainer. Current development focuses on bug fixes, performance improvements, standards compliance, and, occasionally, new features.

In May 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* (<https://www.gnu.org/software/gawk/manual/gawkinet/gawkinet.html>) (a separate document, available as part of the `gawk` distribution). His code finally became part of the main `gawk` distribution with `gawk` version 3.1.

John Haque rewrote the `gawk` internals, in the process providing an `awk`-level debugger. This version became available as `gawk` version 4.0 in 2011.

See “Major Contributors to `gawk`” on page 465 for a full list of those who have 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`.” By analogy, the original version of `awk` is referred to as “old `awk`.”

On most current systems, when you run the `awk` utility you get some version of new `awk`.⁴ If your system’s standard `awk` is the old one, you will see something like this if you try the test program:

```
$ awk 1 /dev/null
error→ awk: syntax error near line 1
error→ awk: bailing out near line 1
```

In this case, you should find a version of new `awk`, or just install `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`.

4. Only Solaris systems still use an old `awk` for the default `awk` utility. A more modern `awk` lives in `/usr/xpg6/bin` on these systems.

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 how to write programs in 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.

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, it notes any `gawk` features that are not in the POSIX standard for `awk`.

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 and HTML versions (<http://www.gnu.org/software/gawk/manual/>) of the book.

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.

Most of the time, the examples use complete `awk` programs. Some of the more advanced sections show only the part of the `awk` program that illustrates the concept being described.

Although 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 10 and Chapter 11 should be of interest.

This book is split into several parts, as follows:

- Part I, *The awk Language*, describes the `awk` language and the `gawk` program in detail. It starts with the basics, and continues through all of the features of `awk`. It contains the following chapters:
 - Chapter 1, *Getting Started with awk*, provides the essentials you need to know to begin using `awk`.
 - Chapter 2, *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 3, *Regular Expressions*, introduces regular expressions in general, and in particular the flavors supported by POSIX `awk` and `gawk`.

- Chapter 4, *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. Network I/O is also briefly introduced here.
- Chapter 5, *Printing Output*, describes how `awk` programs can produce output with `print` and `printf`.
- Chapter 6, *Expressions*, describes expressions, which are the basic building blocks for getting most things done in a program.
- Chapter 7, *Patterns, Actions, and Variables*, describes how to write patterns for matching records, actions for doing something when a record is matched, and the predefined variables `awk` and `gawk` use.
- Chapter 8, *Arrays in awk*, covers `awk`'s one and only data structure: the associative array. Deleting array elements and whole arrays is described, as well as sorting arrays in `gawk`. The chapter also describes how `gawk` provides arrays of arrays.
- Chapter 9, *Functions*, describes the built-in functions `awk` and `gawk` provide, as well as how to define your own functions. It also discusses how `gawk` lets you call functions indirectly.
- Part II, *Problem Solving with awk*, shows how to use `awk` and `gawk` for problem solving. There is lots of code here for you to read and learn from. This part contains the following chapters:
 - Chapter 10, *A Library of awk Functions*, provides a number of functions meant to be used from main `awk` programs.
 - Chapter 11, *Practical awk Programs*, provides many sample `awk` programs.
 Reading these two chapters allows you to see `awk` solving real problems.
- Part III, *Moving Beyond Standard awk with gawk*, focuses on features specific to `gawk`. It contains the following chapters:
 - Chapter 12, *Advanced Features of gawk*, describes a number of advanced features. Of particular note are the abilities to control the order of array traversal, have two-way communications with another process, perform TCP/IP networking, and profile your `awk` programs.
 - Chapter 13, *Internationalization with gawk*, describes special features for translating program messages into different languages at runtime.
 - Chapter 14, *Debugging awk Programs*, describes the `gawk` debugger.
 - Chapter 15, *Arithmetic and Arbitrary-Precision Arithmetic with gawk*, describes advanced arithmetic facilities.
 - Chapter 16, *Writing Extensions for gawk*, describes how to add new variables and functions to `gawk` by writing extensions in C or C++.

- Part IV, *Appendices*, provides the following appendices, including the GNU General Public License:
 - Appendix A, *The Evolution of the awk Language*, describes how the awk language has evolved since its first release to the present. It also describes how gawk has acquired features over time.
 - Appendix B, *Installing gawk*, describes how to get gawk, how to compile it on POSIX-compatible systems, and how to compile and use it on different non-POSIX systems. It also describes how to report bugs in gawk and where to get other freely available awk implementations.
 - Appendix C, *GNU General Public License*, presents the license that covers the gawk source code.

The version of this book distributed with gawk contains additional appendices and other end material. To save space, we have omitted them from the printed edition. You may find them online, as follows:

- The appendix on implementation notes (http://www.gnu.org/software/gawk/manual/html_node/Notes.html) describes how to disable gawk's extensions, how to contribute new code to gawk, where to find information on some possible future directions for gawk development, and the design decisions behind the extension API.
- The appendix on basic concepts (http://www.gnu.org/software/gawk/manual/html_node/Basic-Concepts.html) provides some very cursory background material for those who are completely unfamiliar with computer programming.
- The glossary (http://www.gnu.org/software/gawk/manual/html_node/Glossary.html) defines most, if not all, of the significant terms used throughout the book. If you find terms that you aren't familiar with, try looking them up here.
- The GNU FDL (http://www.gnu.org/software/gawk/manual/html_node/GNU-Free-Documentation-License.html) is the license that covers this book.

Some of the chapters have exercise sections; these have also been omitted from the print edition but are available online.

Typographical Conventions

This book is written in Texinfo (<http://www.gnu.org/software/texinfo/>), the GNU documentation formatting language. A single Texinfo source file is used to produce both the printed and online versions of the documentation. Because of this, the typographical conventions are slightly different than in other books you may have read.

Examples you would type at the command line are preceded by the common shell primary and secondary prompts, '\$' and '>'. Input that you type is shown **like this**. Output from the command, usually its standard output, appears like this. Error messages and other output on the command's standard error are preceded by the glyph "error→". For example:

```
$ echo hi on stdout
hi on stdout
$ echo hello on stderr 1>&2
error→ hello on stderr
```

In the text, almost anything related to programming, such as command names, variable and function names, and string, numeric and regexp constants appear in **this font**. Code fragments appear in the same font and quoted, 'like this'. Things that are replaced by the user or programmer appear in *this font*. Options look like this: **-f**. Filenames are indicated like this: */path/to/our file*. The first occurrence of a new term is usually its *definition* and appears in the same font as the previous occurrence of "definition" in this sentence.

Characters that you type at the keyboard look **like this**. In particular, there are special characters called "control characters." These are characters that you type by holding down both the **CONTROL** key and another key, at the same time. For example, a **Ctrl-d** is typed by first pressing and holding the **CONTROL** key, next pressing the **d** key, and finally releasing both keys.

For the sake of brevity, throughout this book, we refer to Brian Kernighan's version of **awk** as "BWK **awk**." (See "Other Freely Available **awk** Implementations" on page 485 for information on his and other versions.)



Notes of interest look like this.



Cautionary or warning notes look like this.

Dark Corners

Dark corners are basically fractal—no matter how much you illuminate, there's always a smaller but darker one.

—Brian Kernighan

Until the POSIX standard (and *Effective awk Programming*), 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.)”

But, as noted by the opening quote, any coverage of dark corners is by definition incomplete.

Extensions to the standard `awk` language that are supported by more than one `awk` implementation are marked “(c.e.)” for “common extension.”

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 GNU⁵ 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 its software’s source code is always available to the end user. 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 (<http://www.gnu.org>). This book may also be read from GNU’s website (<http://www.gnu.org/software/gawk/manual/>).

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`.

The book itself has gone through multiple previous editions. Paul Rubin wrote the very first draft of *The GAWK Manual*; it was around 40 pages long. Diane Close and Richard Stallman improved it, yielding a version that was around 90 pages 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. The FSF published the first two editions under the title *The GNU Awk User’s Guide*. SSC published two editions of the book under the title *Effective awk Programming*, and O’Reilly published the third edition in 2001.

This edition maintains the basic structure of the previous editions. For FSF edition 4.0, the content was thoroughly reviewed and updated. All references to `gawk` versions prior to 4.0 were removed. Of significant note for that edition was the addition of Chapter 14.

5. GNU stands for “GNU’s Not Unix.”

For FSF edition 4.1 (the fourth edition as published by O'Reilly), the content has been reorganized into parts, and the major new additions are Chapter 15 and Chapter 16.

This book will undoubtedly continue to evolve. If you find an error in the book, please report it! See “Reporting Problems and Bugs” on page 484 for information on submitting problem reports electronically.

How to Stay Current

You may have a newer version of `gawk` than the one described here. To find out what has changed, you should first look at the `NEWS` file in the `gawk` distribution, which provides a high-level summary of the changes in each release.

You can then look at the online version (<http://www.gnu.org/software/gawk/manual/>) of this book to read about any new features.

Using Code Examples

This book is here to help you get your job done. Most of the example programs in this book come in the `gawk` distribution and are marked in the files as being in the public domain. So, in general, you may use the code in this book in your programs and documentation. Incorporating a significant amount of prose or example code from this book into your product's documentation requires compliance with the GNU FDL.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: “*Effective awk Programming*, Fourth Edition, by Arnold Robbins (O'Reilly). Copyright 2015 Free Software Foundation, 978-1-491-90461-9.”

If you feel your use of code examples falls outside fair use or the permission given here, feel free to contact us at permissions@oreilly.com.

Safari® Books Online



Safari Books Online (www.safaribooksonline.com) is an on-demand digital library that delivers expert content in both book and video form from the world's leading authors in technology and business.

Technology professionals, software developers, web designers, and business and creative professionals use Safari Books Online as their primary resource for research, problem solving, learning, and certification training.

Safari Books Online offers a range of product mixes and pricing programs for organizations, government agencies, and individuals. Subscribers have access to thousands of

books, training videos, and prepublication manuscripts in one fully searchable database from publishers like O'Reilly Media, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, Course Technology, and dozens more. For more information about Safari Books Online, please visit us online.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at <http://bit.ly/effective-awk-programming-4e>.

To comment or ask technical questions about this book, send email to bookquestions@oreilly.com.

For more information about our books, courses, conferences, and news, see our website at <http://www.oreilly.com>.

Find us on Facebook: <http://facebook.com/oreilly>

Follow us on Twitter: <http://twitter.com/oreillymedia>

Watch us on YouTube: <http://www.youtube.com/oreillymedia>

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 Fenslon contributed many ideas and sample programs. Richard Mlynarik and Robert Chasell 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.