A

The Evolution of the awk Language

This book describes the GNU implementation of *awk*, which follows the POSIX specification. Many long-time *awk* users learned *awk* programming with the original *awk* implementation in Version 7 Unix. (This implementation was the basis for *awk* in Berkeley Unix, through 4.3-Reno. Subsequent versions of Berkeley Unix, and systems derived from 4.4BSD-Lite, use various versions of *gawk* for their *awk*.) This chapter briefly describes the evolution of the *awk* language, with cross-references to other parts of the book where you can find more information.

Major Changes Between V7 and SVR3.1

The *awk* language evolved considerably between the release of Version 7 Unix (1978) and the new version that was first made generally available in System V Release 3.1 (1987). This section summarizes the changes, with cross-references to further details:

- The requirement for `;` to separate rules on a line (see the section "awk Statements Versus Lines" in Chapter 1, *Getting Started with awk*).
- User-defined functions and the `return` statement (see the section "User-Defined Functions" in Chapter 8, *Functions*).
- The `delete` statement (see the section "The delete Statement" in Chapter 7, *Arrays in awk*).
- The `do-while` statement (see the section "The do-while Statement" in Chapter 6, *Patterns, Actions, and Variables*).
- The built-in functions `atan2`, `cos`, `sin`, `rand`, and `srand` (see the section "Numeric Functions" in Chapter 8).

- The built-in functions `gsub`, `sub`, and `match` (see the section "String-Manipulation Functions" in Chapter 8).
- The built-in functions `close` and `system` (see the section "Input/Output Functions" in Chapter 8).
- The `ARGC`, `ARGV`, `FNR`, `RLENGTH`, `RSTART`, and `SUBSEP` built-in variables (see the section "Built-in Variables" in Chapter 6).
- The conditional expression using the ternary operator `?:` (see the section "Conditional Expressions" in Chapter 5, *Expressions*).
- The exponentiation operator `^` (see the section "Arithmetic Operators" in Chapter 5) and its assignment operator form `^=` (see the section "Assignment Expressions" in Chapter 5).
- C-compatible operator precedence, which breaks some old *awk* programs (see the section "Operator Precedence (How Operators Nest)" in Chapter 5).
- Regexps as the value of `FS` (see the section "Specifying How Fields Are Separated" in Chapter 3, *Reading Input Files*) and as the third argument to the `split` function (see the section "String-Manipulation Functions" in Chapter 8).
- Dynamic regexps as operands of the `~` and `!~` operators (see the section "How to Use Regular Expressions" in Chapter 2, *Regular Expressions*).
- The escape sequences `\b`, `\f`, and `\r` (see the section "Escape Sequences" in Chapter 2). (Some vendors have updated their old versions of *awk* to recognize `\b`, `\f`, and `\r`, but this is not something you can rely on.)
- Redirection of input for the `getline` function (see the section "Explicit Input with getline" in Chapter 3).
- Multiple `BEGIN` and `END` rules (see the section "The BEGIN and END Special Patterns" in Chapter 6).
- Multidimensional arrays (see the section "Multidimensional Arrays" in Chapter 7).

Changes Between SVR3.1 and SVR4

The System V Release 4 (1989) version of Unix *awk* added these features (some of which originated in *gawk*):

- The `ENVIRON` variable (see the section "Built-in Variables" in Chapter 6).
- Multiple *–f* options on the command line (see the section "Command-Line Options" in Chapter 11, *Running awk and gawk*).

- The *–v* option for assigning variables before program execution begins (see the section "Command-Line Options" in Chapter 11).
- The –– option for terminating command-line options.
- The `\a`, `\v`, and `\x` escape sequences (see the section "Escape Sequences" in Chapter 2).
- A defined return value for the `srand` built-in function (see the section "Numeric Functions" in Chapter 8).
- The `toupper` and `tolower` built-in string functions for case translation (see the section "String-Manipulation Functions" in Chapter 8).
- A cleaner specification for the `%c` format-control letter in the `printf` function (see the section "Format-Control Letters" in Chapter 4, *Printing Output*).
- The ability to dynamically pass the field width and precision (`"%*.*d"`) in the argument list of the `printf` function (see the section "Format-Control Letters" in Chapter 4).
- The use of regexp constants, such as `/foo/`, as expressions, where they are equivalent to using the matching operator, as in `$0 ~ /foo/` (see the section "Using Regular Expression Constants" in Chapter 5).
- Processing of escape sequences inside command-line variable assignments (see the section "Assigning Variables on the Command Line" in Chapter 5).

Changes Between SVR4 and POSIX awk

The POSIX Command Language and Utilities standard for *awk* (1992) introduced the following changes into the language:

- The use of *–W* for implementation-specific options (see the section "Command-Line Options" in Chapter 11).
- The use of `CONVFMT` for controlling the conversion of numbers to strings (see the section "Conversion of Strings and Numbers" in Chapter 5).
- The concept of a numeric string and tighter comparison rules to go with it (see the section "Variable Typing and Comparison Expressions" in Chapter 5).
- More complete documentation of many of the previously undocumented features of the language.

The following common extensions are not permitted by the POSIX standard:

- `\x` escape sequences are not recognized (see the section "Escape Sequences" in Chapter 2).
- Newlines do not act as whitespace to separate fields when `FS` is equal to a single space (see the section "Examining Fields" in Chapter 3).
- Newlines are not allowed after `?` or `:` (see the section "Conditional Expressions" in Chapter 5).
- The synonym `func` for the keyword `function` is not recognized (see the section "Function Definition Syntax" in Chapter 8).
- The operators `**` and `**=` cannot be used in place of `^` and `^=` (see the section "Arithmetic Operators" and section "Assignment Expressions" in Chapter 5).
- Specifying `-Ft` on the command line does not set the value of `FS` to be a single tab character (see the section "Specifying How Fields Are Separated" in Chapter 3).
- The `fflush` built-in function is not supported (see the section "Input/Output Functions" in Chapter 8).

Extensions in the Bell Laboratories awk

Brian Kernighan, one of the original designers of Unix *awk*, has made his version available via his home page (see the section "Other Freely Available awk Implementations" in Appendix B, *Installing gawk*). This section describes extensions in his version of *awk* that are not in POSIX *awk*:

- The `-mf` *`N`* and `-mr` *`N`* command-line options to set the maximum number of fields and the maximum record size, respectively (see the section "Command-Line Options" in Chapter 11). As a side note, his *awk* no longer needs these options; it continues to accept them to avoid breaking old programs.
- The `fflush` built-in function for flushing buffered output (see the section "Input/Output Functions" in Chapter 8).
- The `**` and `**=` operators (see the section "Arithmetic Operators" and section "Assignment Expressions" in Chapter 5).
- The use of `func` as an abbreviation for `function` (see the section "Function Definition Syntax" in Chapter 8).

The Bell Laboratories *awk* also incorporates the following extensions, originally developed for *gawk*:

- The `\x` escape sequence (see the section "Escape Sequences" in Chapter 2).
- The */dev/stdin*, */dev/stdout*, and */dev/stderr* special files (see the section "Special Filenames in gawk" in Chapter 4).
- The ability for `FS` and for the third argument to `split` to be null strings (see the section "Making Each Character a Separate Field" in Chapter 3).
- The `nextfile` statement (see the section "Using gawk's nextfile Statement" in Chapter 6).
- The ability to delete all of an array at once with `delete` *`array`* (see the section "The delete Statement" in Chapter 7).

Extensions in gawk Not in POSIX awk

The GNU implementation, *gawk*, adds a large number of features. This section lists them in the order they were added to *gawk*. They can all be disabled with either the *––traditional* or *––posix* options (see the section "Command-Line Options" in Chapter 11).

Version 2.10 of *gawk* introduced the following features:

- The AWKPATH environment variable for specifying a path search for the *–f* command-line option (see the section "Command-Line Options" in Chapter 11).
- The `IGNORECASE` variable and its effects (see the section "Case Sensitivity in Matching" in Chapter 2).
- The */dev/stdin*, */dev/stdout*, */dev/stderr*, and */dev/fd/N* special filenames (see the section "Special Filenames in gawk" in Chapter 4).

Version 2.13 of *gawk* introduced the following features:

- The `FIELDWIDTHS` variable and its effects (see the section "Reading Fixed-Width Data" in Chapter 3).
- The `systime` and `strftime` built-in functions for obtaining and printing timestamps (see the section "Using gawk's Timestamp Functions" in Chapter 8).
- The *–W lint* option to provide error and portability checking for both the source code and at runtime (see the section "Command-Line Options" in Chapter 11).
- The *–W compat* option to turn off the GNU extensions (see the section "Command-Line Options" in Chapter 11).
- The *–W posix* option for full POSIX compliance (see the section "Command-Line Options" in Chapter 11).

Version 2.14 of *gawk* introduced the following feature:

- The `next file` statement for skipping to the next datafile (see the section "Using gawk's nextfile Statement" in Chapter 6).

Version 2.15 of *gawk* introduced the following features:

- The `ARGIND` variable, which tracks the movement of `FILENAME` through `ARGV` (see the section "Built-in Variables" in Chapter 6).
- The `ERRNO` variable, which contains the system error message when `getline` returns −1 or `close` fails (see the section "Built-in Variables" in Chapter 6).
- The */dev/pid*, */dev/ppid*, */dev/pgrpid*, and */dev/user* filename interpretation (see the section "Special Filenames in gawk" in Chapter 4).
- The ability to delete all of an array at once with `delete` *`array`* (see the section "The delete Statement" in Chapter 7).
- The ability to use GNU-style long-named options that start with −− (see the section "Command-Line Options" in Chapter 11).
- The −−*source* option for mixing command-line and library-file source code (see the section "Command-Line Options" in Chapter 11).

Version 3.0 of *gawk* introduced the following features:

- `IGNORECASE` changed, now applying to string comparison as well as regexp operations (see the section "Case Sensitivity in Matching" in Chapter 2).
- The `RT` variable that contains the input text that matched `RS` (see the section "How Input Is Split into Records" in Chapter 3).
- Full support for both POSIX and GNU regexps (see Chapter 2).
- The `gensub` function for more powerful text manipulation (see the section "String-Manipulation Functions" in Chapter 8).
- The `strftime` function acquired a default time format, allowing it to be called with no arguments (see the section "Using gawk's Timestamp Functions" in Chapter 8).
- The ability for `FS` and for the third argument to `split` to be null strings (see the section "Making Each Character a Separate Field" in Chapter 3).
- The ability for `RS` to be a regexp (see the section "How Input Is Split into Records" in Chapter 3).
- The `next file` statement became `nextfile` (see the section "Using gawk's nextfile Statement" in Chapter 6).

- The *--lint-old* option to warn about constructs that are not available in the original Version 7 Unix version of *awk* (see the section "Major Changes Between V7 and SVR3.1" earlier in this appendix).
- The *-m* option and the `fflush` function from the Bell Laboratories research version of *awk* (see the section "Command-Line Options" in Chapter 11; also see the section "Input/Output Functions" in Chapter 8).
- The *--re-interval* option to provide interval expressions in regexps (see the section "Regular Expression Operators" in Chapter 2).
- The *--traditional* option was added as a better name for *--compat* (see the section "Command-Line Options" in Chapter 11).
- The use of GNU Autoconf to control the configuration process (see the section "Compiling gawk for Unix" in Appendix B).
- Amiga support.

Version 3.1 of *gawk* introduced the following features:

- The `BINMODE` special variable for non-POSIX systems, which allows binary I/O for input and/or output files (see the section "Using gawk on PC Operating Systems" in Appendix B).
- The `LINT` special variable, which dynamically controls lint warnings (see the section "Built-in Variables" in Chapter 6).
- The `PROCINFO` array for providing process-related information (see the section "Built-in Variables" in Chapter 6).
- The `TEXTDOMAIN` special variable for setting an application's internationalization text domain (see the section "Built-in Variables" in Chapter 6, and Chapter 9, *Internationalization with gawk*).
- The ability to use octal and hexadecimal constants in *awk* program source code (see the section "Octal and Hexadecimal Numbers" in Chapter 5).
- The `|&` operator for two-way I/O to a coprocess (see the section "Two-Way Communications with Another Process" in Chapter 10, *Advanced Features of gawk*).
- The */inet* special files for TCP/IP networking using `|&` (see the section "Using gawk for Network Programming" in Chapter 10).
- The optional second argument to `close` that allows closing one end of a two-way pipe to a coprocess (see the section "Two-Way Communications with Another Process" in Chapter 10).
- The optional third argument to the `match` function for capturing text-matching subexpressions within a regexp (see the section "String-Manipulation Functions" in Chapter 8).

- Positional specifiers in `printf` formats for making translations easier (see the section "Rearranging printf Arguments" in Chapter 9).
- The `asort` function for sorting arrays (see the section "Sorting Array Values and Indices with gawk" in Chapter 7).
- The `bindtextdomain` and `dcgettext` functions for internationalization (see the section "Internationalizing awk Programs" in Chapter 9).
- The `extension` built-in function and the ability to add new built-in functions dynamically (see the section "Adding New Built-in Functions to gawk" in Appendix C, *Implementation Notes*).
- The `mktime` built-in function for creating timestamps (see the section "Using gawk's Timestamp Functions" in Chapter 8).
- The `and`, `or`, `xor`, `compl`, `lshift`, `rshift`, and `strtonum` built-in functions (see the section "Bit-Manipulation Functions of gawk" in Chapter 8).
- The support for `next file` as two words was removed completely (see the section "Using gawk's nextfile Statement" in Chapter 6).
- The *––dump–variables* option to print a list of all global variables (see the section "Command-Line Options" in Chapter 11).
- The *––gen–po* command-line option and the use of a leading underscore to mark strings that should be translated (see the section "Extracting Marked Strings" in Chapter 9).
- The *––non–decimal–data* option to allow nondecimal input data (see the section "Allowing Nondecimal Input Data" in Chapter 10).
- The *––profile* option and *pgawk*, the profiling version of *gawk*, for producing execution profiles of *awk* programs (see the section "Profiling Your awk Programs" in Chapter 10).
- The *––enable–portals* configuration option to enable special treatment of pathnames that begin with */p* as BSD portals (see the section "Using gawk with BSD Portals" in Chapter 10).
- The use of GNU Automake to help in standardizing the configuration process (see the section "Compiling gawk for Unix" in Appendix B).
- The use of GNU `gettext` for *gawk*'s own message output (see the section "gawk Can Speak Your Language" in Chapter 9).
- BeOS support.
- Tandem support.

- The Atari port became officially unsupported.
- The source code now uses new-style function definitions, with *ansi2knr* to convert the code on systems with old compilers.

Major Contributors to gawk

This section names the major contributors to *gawk* and/or this book, in approximate chronological order:

- Dr. Alfred V. Aho, Dr. Peter J. Weinberger, and Dr. Brian W. Kernighan, all of Bell Laboratories, designed and implemented Unix *awk*, from which *gawk* gets the majority of its feature set.
- Paul Rubin did the initial design and implementation in 1986, and wrote the first draft (around 40 pages) of this book.
- Jay Fenlason finished the initial implementation.
- Diane Close revised the first draft of this book, bringing it to around 90 pages.
- Richard Stallman helped finish the implementation and the initial draft of this book. He is also the founder of the FSF and the GNU project.
- John Woods contributed parts of the code (mostly fixes) in the initial version of *gawk*.
- In 1988, David Trueman took over primary maintenance of *gawk*, making it compatible with "new" *awk*, and greatly improving its performance.
- Pat Rankin provided the VMS port and its documentation.
- Conrad Kwok, Scott Garfinkle, and Kent Williams did the initial ports to MS-DOS with various versions of MSC.
- Hal Peterson provided help in porting *gawk* to Cray systems.
- Kai Uwe Rommel provided the port to OS/2 and its documentation.
- Michal Jaegermann provided the port to Atari systems and its documentation. He continues to provide portability checking with DEC Alpha systems, and has done a lot of work to make sure *gawk* works on non-32-bit systems.
- Fred Fish provided the port to Amiga systems and its documentation.
- Scott Deifik currently maintains the MS-DOS port.
- Juan Grigera maintains the port to Win32 systems.
- Dr. Darrel Hankerson acts as coordinator for the various ports to different PC platforms and creates binary distributions for various PC operating systems. He is also instrumental in keeping the documentation up to date for the various PC platforms.

- Christos Zoulas provided the `extension` built-in function for dynamically adding new modules.
- Jürgen Kahrs contributed the initial version of the TCP/IP networking code and documentation, and motivated the inclusion of the `|&` operator.
- Stephen Davies provided the port to Tandem systems and its documentation.
- Martin Brown provided the port to BeOS and its documentation.
- Arno Peters did the initial work to convert *gawk* to use GNU Automake and `gettext`.
- Alan J. Broder provided the initial version of the `asort` function as well as the code for the new optional third argument to the `match` function.
- Arnold Robbins has been working on *gawk* since 1988, at first helping David Trueman, and as the primary maintainer since around 1994.