

NAME

bibclean – prettyprint and syntax check BibTeX and Scribe bibliography data base files

SYNOPSIS

```
bibclean [ -author ] [ -copyleft ] [ -copyright ] [ -error-log filename ] [ -help ] [ '-?' ]
[ -init-file filename ] [ -ISBN-file filename ] [ -keyword-file filename ]
[ -max-width nnn ] [ -[no-]align-equals ] [ -[no-]brace-protect ] [ -[no-]check-values ]
[ -[no-]debug-match-failures ] [ -[no-]delete-empty-values ] [ -[no-]file-position ]
[ -[no-]fix-accents ] [ -[no-]fix-braces ] [ -[no-]fix-font-changes ] [ -[no-]fix-initials ]
[ -[no-]fix-math ] [ -[no-]fix-names ] [ -[no-]German-style ] [ -[no-]keep-linebreaks ]
[ -[no-]keep-parbreaks ] [ -[no-]keep-preamble-spaces ] [ -[no-]keep-spaces ]
[ -[no-]keep-string-spaces ] [ -[no-]parbreaks ] [ -[no-]prettyprint ]
[ -[no-]print-ISBN-table ] [ -[no-]print-keyword-table ] [ -[no-]print-patterns ] [
-[no-]quiet ]
[ -[no-]read-init-files ] [ -[no-]remove-OPT-prefixes ] [ -[no-]scribe ]
[ -[no-]trace-file-opening ] [ -[no-]warnings ] [ -output-file filename ] [ -version ]
<infile or bibfile1 bibfile2 bibfile3 ...
>outfile
```

All options can be abbreviated to a unique leading prefix.

An explicit file name of “-” represents standard input; it is assumed if no input files are specified.

On VAX VMS and IBM PC DOS, the leading “-” on option names may be replaced by a slash, “/”; however, the “-” option prefix is always recognized.

DESCRIPTION

bibclean prettyprints input BibTeX files to *stdout*, or to a user-specified file, and checks the brace balance and bibliography entry syntax as well. It can be used to detect problems in BibTeX files that sometimes confuse even BibTeX itself, and importantly, can be used to normalize the appearance of collections of BibTeX files.

Here is a summary of the formatting actions:

- BibTeX items are formatted into a consistent structure with one *field* = “*value*” pair per line, and the initial @ and trailing right brace in column 1.
- Tabs are expanded into blank strings; their use is discouraged because they inhibit portability, and can suffer corruption in electronic mail.
- Long string values are split at a blank and continued onto the next line with leading indentation.
- A single blank line separates adjacent bibliography entries.
- Text outside BibTeX entries is passed through verbatim.
- Outer parentheses around entries are converted to braces.
- Personal names in *author* and *editor* field values are normalized to the form “P. D. Q. Bach”, from “P.D.Q. Bach” and “Bach, P.D.Q.”.
- Hyphen sequences in page numbers are converted to en-dashes.
- Month values are converted to standard BibTeX string abbreviations.
- In titles, sequences of upper-case characters at brace level zero are braced to protect them from being converted to lower-case letters by some bibliography styles.
- CODEN, ISBN (International Standard Book Number) and ISSN (International Standard Serial Number) entry values are examined to verify the checksums of each listed number, and correct ISBN hyphenation is automatically supplied.

The standardized format of the output of **bibclean** facilitates the later application of simple filters, such as **bibcheck**(1), **bibdup**(1), **bibextract**(1), **bibindex**(1), **bibjoin**(1), **biblabeled**(1), **biblook**(1), **biborder**(1), **bibsort**(1), **citefind**(1), and **citetags**(1), to process the text, and also is the one expected by the GNU Emacs BibTeX support functions.

OPTIONS

Command-line switches may be abbreviated to a unique leading prefix, and letter case is *not* significant. All options are parsed before any input bibliography files are read, no matter what their order on the command line. Options that correspond to a yes/no setting of a flag have a form with a prefix "no-" to set the flag to *no*. For such options, the last setting determines the flag value used. That is significant when options are also specified in initialization files (see the **INITIALIZATION FILES** manual section).

The leading hyphen that distinguishes an option from a filename may be doubled, for compatibility with GNU and POSIX conventions. Thus, **-author** and **--author** are equivalent.

To avoid confusion with options, if a filename begins with a hyphen, it must be disguised by a leading absolute or relative directory path, e.g., */tmp/-foo.bib* or *./-foo.bib*.

- author** Display an author credit on the standard error unit, *stderr*, and then terminate with a success return code. Sometimes an executable program is separated from its documentation and source code; this option provides a way to recover from that.
- copyleft** Display copyright information on the standard error unit, *stderr*, and then terminate with a success return code.
- copyright** Display copyright information on the standard error unit, *stderr*, and then terminate with a success return code.
- error-log filename** Redirect *stderr* to the indicated file, which will then contain all of the error and warning messages. This option is provided for those systems that have difficulty redirecting *stderr*.
- help** or **-?** Display a help message on *stderr*, giving a usage description, similar to this section of the manual pages, and then terminate with a success return code.
- ISBN-file filename** Provide an explicit ISBN-range initialization file. It will be processed *after* any system-wide and job-wide ISBN initialization files found on the **PATH** (for VAX VMS, **SYSSYSTEM**) and **BIBINPUTS** search paths, respectively, and may override them. The ISBN initialization file name can be changed at compile time, or at run time through a setting of the environment variable **BIBCLEANISBN**, but defaults to *.bibclean.isbn* on UNIX, and *bibclean.isb* elsewhere. For further details, see the **ISBN INITIALIZATION FILES** manual section.
- init-file filename** Provide an explicit value pattern initialization file. It will be processed *after* any system-wide and job-wide initialization files found on the **PATH** (for VAX VMS, **SYSSYSTEM**) and **BIBINPUTS** search paths, respectively, and may override them. It in turn may be overridden by a subsequent file-specific initialization file. The initialization file name can be changed at compile time, or at run time through a setting of the environment variable **BIBCLEANINI**, but defaults to *.bibcleanrc* on UNIX, and to *bibclean.ini* elsewhere. For further details, see the **INITIALIZATION FILES** manual section.
- keyword-file filename** Provide an explicit keyword initialization file. It will be processed *after* any system-wide and job-wide keyword initialization files found on the **PATH** (for VAX VMS, **SYSSYSTEM**) and **BIBINPUTS** search paths, respectively, and may override them. The keyword initialization file name can be changed at compile time, or at run time through a setting of the environment

variable **BIBCLEANKEY**, but defaults to *.bibclean.key* on UNIX, and *bibclean.key* elsewhere. For further details, see the **KEYWORD INITIALIZATION FILES** manual section.

–max-width *nnn*

bibclean normally limits output line widths to 72 characters, and in the interests of consistency, that value should not be changed. Occasionally, special-purpose applications may require different maximum line widths, so this option provides that capability. The number following the option name can be specified in decimal, octal (starting with 0), or hexadecimal (starting with 0x). A zero or negative value is interpreted to mean unlimited, so **–max-width 0** can be used to ensure that each field/value pair appears on a single line.

When **–no-prettyprint** requests **bibclean** to act as a lexical analyzer, the default line width is unlimited, unless overridden by this option.

When **bibclean** is prettyprinting, line wrapping will be done only at a space. Consequently, a long non-blank character sequence may result in the output exceeding the requested line width.

When **bibclean** is lexing, line wrapping is done by inserting a backslash-newline pair when the specified maximum is reached, so no line length will ever exceed the maximum.

–[no-]align-equals

With the positive form, align the equals sign in key/value assignments at the same column, separated by a single space from the value string. Otherwise, the equals sign follows the key, separated by a single space. Default: *no*.

–[no-]brace-protect

Protect uppercase and mixedcase words at brace-level zero with braces to prevent downcasing by some **BIBTEX** styles. Default: *yes*.

–[no-]check-values

With the positive form, apply heuristic pattern matching to field values in order to detect possible errors (e.g., “*year = "192"*” instead of “*year = "1992"*”), and issue warnings when unexpected patterns are found.

That checking is usually beneficial, but if it produces too many bogus warnings for a particular bibliography file, you can disable it with the negative form of this option. Default: *yes*.

–[no-]debug-match-failures

With the positive form, print out a warning when a value pattern fails to match a value string.

That is helpful in debugging new patterns, but because the output can be voluminous, you should use this option only with small test files, and initialization files that eliminate all patterns apart from the ones that you are testing. Default: *no*.

–[no-]delete-empty-values

With the positive form, remove all field/value pairs for which the value is an empty string. That is helpful in cleaning up bibliographies generated from text editor templates. Compare this option with **–[no-]remove-OPT-pre-fixes** described below. Default: *no*.

–[no-]file-position

With the positive form, give detailed file position information in warning and error messages. Default: *no*.

–[no-]fix-accents

With the positive form, normalize **T_EX** accents in annotes, authors, booktitles, editors, notes, remarks, and titles. Default: *no*.

–[no-]fix-braces

With the positive form, normalize bracing in annotes, authors, booktitles, editors, notes, remarks, and titles, by removing unnecessary levels of braces. Default: *no*.

–[no-]fix-font-changes

With the positive form, supply an additional brace level around font changes in titles to protect against downcasing by some \TeX styles. Font changes that already have more than one level of braces are not modified.

For example, if a title contains the Latin phrase $\text{\em Dictyostelium Discoideum}$ or $\text{\em {D}ictyostelium {D}iscoideum}$, then downcasing will incorrectly convert the phrase to lower-case letters. Most \TeX users are surprised that bracing the initial letters does not prevent the downcase action. The correct coding is $\text{\em {D}ictyostelium {D}iscoideum}$. However, there are also legitimate cases where an extra level of bracing wrongly protects from downcasing. Consequently, **bibclean** will normally *not* supply an extra level of braces, but if you have a bibliography where the extra braces are routinely missing, you can use this option to supply them.

If you think that you need this option, it is *strongly* recommended that you apply **bibclean** to your bibliography file with and without **–fix-font-changes**, then compare the two output files to ensure that extra braces are not being supplied in titles where they should not be present. You will have to decide which of the two output files is the better choice, then repair the incorrect title bracing by hand.

Because font changes in titles are uncommon, except for cases of the type which this option is designed to correct, it should do more good than harm. Default: *no*.

–[no-]fix-initials

With the positive form, insert a space after a period following author initials. Default: *yes*.

–[no-]fix-math

With the positive form, improve readability of math mode in titles by inserting spaces around operators, deleting other unnecessary space, and removing braces around single-character subscripts and superscripts. Default: *no*.

–[no-]fix-names

With the positive form, reorder *author* and *editor* name lists to remove commas at brace level zero, placing first names or initials before last names. Default: *yes*.

–[no-]German-style

With the positive form, interpret quote characters ["] inside *braced* value strings at brace level 1 according to the conventions of the \TeX style file *german.sty*, which overloads quote to simplify input and representation of German umlaut accents, sharp-s (es-zet), ligature separators, invisible hyphens, raised/lowered quotes, French guillemets, and discretionary hyphens. Recognized character combinations will be braced to prevent \TeX from interpreting the quote as a string delimiter.

Quoted strings receive no special handling from this option, and because German nouns in titles must anyway be protected from the downcasing operation of most \TeX bibliography styles, German value strings that use the overloaded quote character can always be entered in the form "{...}", without the need to specify this option at all.

Default: *no*.

–[no-]keep-linebreaks

Normally, line breaks inside value strings are collapsed into a single space, so that long value strings can later be broken to provide lines of reasonable length.

With the positive form, linebreaks are preserved in value strings. If **–max-width** is set to zero, this preserves the original line breaks. Spacing *outside* value strings remains under **bibclean**'s control, and is not affected by this option.

- Default: *no*.
- [no-]keep-parbreaks** With the positive form, preserve paragraph breaks (either formfeeds, or lines containing only spaces) in value strings. Normally, paragraph breaks are collapsed into a single space. Spacing *outside* value strings remains under **bibclean**’s control, and is not affected by this option. Default: *no*.
- [no-]keep-preamble-spaces** With the positive form, preserve all whitespace in @Preamble{...} entries. Default: *no*.
- [no-]keep-spaces** With the positive form, preserve all spaces in value strings. Normally, multiple spaces are collapsed into a single space. This option can be used together with **–keep-linebreaks**, **–keep-parbreaks**, and **–max-width 0** to preserve the form of value strings while still providing syntax and value checking. Spacing *outside* value strings remains under **bibclean**’s control, and is not affected by this option. Default: *no*.
- [no-]keep-string-spaces** With the positive form, preserve all whitespace in @String{...} entries. Default: *no*.
- [no-]parbreaks** With the negative form, a paragraph break (either a formfeed, or a line containing only spaces) is not permitted in value strings, or between field/value pairs. That may be useful to quickly trap runaway strings arising from mismatched delimiters. Default: *yes*.
- [no-]prettyprint** Normally, **bibclean** functions as a prettyprinter. However, with the negative form of this option, it acts as a lexical analyzer instead, producing a stream of lexical tokens. See the **LEXICAL ANALYSIS** manual section for further details. Default: *yes*.
- [no-]print-ISBN-table** With the positive form, print the ISBN-range table on *stderr*, then terminate with a success return code.
- That action is taken after all command-line options are processed, and before any input files are read (other than those which are values of command-line options).
- The format of the output ISBN-range table is acceptable for input as an ISBN initialization file (see the **ISBN INITIALIZATION FILES** manual section). Default: *no*.
- [no-]print-keyword-table** With the positive form, print the keyword initialization table on *stderr*, then terminate with a success return code.
- That action is taken after all command-line options are processed, and before any input files are read (other than those which are values of command-line options).
- The format of the output table is acceptable for input as a keyword initialization file (see the **KEYWORD INITIALIZATION FILES** manual section). Default: *no*.
- [no-]print-patterns** With the positive form, print the value patterns read from initialization files as they are added to internal tables. Use this option to check newly-added patterns, or to see what patterns are being used.
- When **bibclean** is compiled with native pattern-matching code (the default), these patterns are the ones that will be used in checking value strings for valid syntax, and all of them are specified in initialization files, rather than hard-coded into the program. For further details, see the **INITIALIZATION FILES** manual section. Default: *no*.

-[no-]quiet	This option is the opposite of -[no-]warning ; it exists for user convenience, and for compatibility with other programs that use -q for quiet operation, without warning messages.
-[no-]read-init-files	With the negative form, suppress loading of system-, user-, and file-specific initialization files. Initializations will come <i>only</i> from those files explicitly given by -init-file filename options. Default: <i>yes</i> .
-[no-]remove-OPT-prefixes	With the positive form, remove the “OPT” prefix from each field name where the corresponding value is <i>not</i> an empty string. The prefix “OPT” must be entirely in upper-case to be recognized. This option is for bibliographies generated with the help of the GNU Emacs BIB_TE_X editing support, which generates templates with optional fields identified by the “OPT” prefix. Although the function <i>M-x bibtex-remove-OPT</i> normally bound to the keystrokes <i>C-c C-o</i> does the job, users often forget, with the result that BIB_TE_X does not recognize the field name, and ignores the value string. Compare this option with -[no-]delete-empty-values described above. Default: <i>no</i> .
-[no-]scribe	With the positive form, accept input syntax conforming to the SCRIBE document system. The output will be converted to conform to BIB_TE_X syntax. See the SCRIBE BIBLIOGRAPHY FORMAT manual section for further details. Default: <i>no</i> .
-[no-]trace-file-opening	With the positive form, record in the error log file the names of all files which bibclean attempts to open. Use this option to identify where initialization files are located. Default: <i>no</i> .
-[no-]warnings	With the positive form, allow all warning messages. The negative form is <i>not</i> recommended because it may mask problems that should be repaired. Default: <i>yes</i> .
-output-file filename	Supply an alternate output file to replace <i>stdout</i> . If the filename cannot be opened for output, execution terminates immediately with a nonzero exit code.
-version	Display the program version number on <i>stderr</i> , and then terminate with a success return code. That will also include an indication of who compiled the program, the host name on which it was compiled, the time of compilation, and the type of string-value matching code selected, when that information is available to the compiler.

ERROR RECOVERY AND WARNINGS

When **bibclean** detects an error, it issues an error message to both *stderr* and *stdout*. That way, the user is clearly notified, and the output bibliography also contains the message at the point of error.

Error messages begin with a distinctive pair of queries, *??*, beginning in column 1, followed by the input file name and line number. If the **-file-position** option was specified, they also contain the input and output positions of the current file, entry, and value. Each position includes the file byte number, the line number, and the column number. In the event of a runaway string argument, the entry and value positions should precisely pinpoint the erroneous bibliography entry, and the file positions will indicate where it was detected, which may be rather later in the files.

Warning messages identify possible problems, and are therefore sent only to *stderr*, and not to *stdout*, so they never appear in the output file. They are identified by a distinctive pair of percents, *%%*, beginning in column 1, and as with error messages, may be followed by file position messages if the **-file-position** option was specified.

For convenience, the first line of each error and warning message sent to *stderr* is formatted according to the expectations of the GNU Emacs *next-error* command. You can invoke **bibclean** with the Emacs *M-x compile*<RET>*bibclean filename.bib >filename.new* command, then use the *next-error* command, normally bound to *C-x* ‘ (that’s a grave, or back, accent), to move to the location of the error in the input file.

If error messages are ignored, and left in the output bibliography file, they will precipitate an error when the bibliography is next processed with **BIB_TEX**.

After issuing an error message, **bibclean** then resynchronizes its input by copying it verbatim to *stdout* until a new bibliography entry is recognized on a line in which the first non-blank character is an at-sign (@). That ensures that nothing is lost from the input file(s), allowing corrections to be made in either the input or the output files. However, if **bibclean** detects an internal error in its data structures, it will terminate abruptly without further input or output processing; that kind of error should never happen, and if it does, it should be reported immediately to the author of the program. Errors in initialization files, and running out of dynamic memory, will also immediately terminate **bibclean**.

INITIALIZATION FILES

bibclean can be compiled with one of three different types of pattern matching; the choice is made by the installer at compile time:

- The original version uses explicit hand-coded tests of value-string syntax.
- The second version uses regular-expression pattern-matching host library routines together with regular-expression patterns that come entirely from initialization files.
- The third version uses special patterns that come entirely from initialization files.

The second and third versions are the ones of most interest here, because they allow the user to control what values are considered acceptable. However, command-line options can also be specified in initialization files, no matter which pattern matching choice was selected.

When **bibclean** starts, it searches for initialization files, finding the first one in the system executable program search path (on UNIX and IBM PC DOS, **PATH**) and the first one in the **BIBINPUTS** search path, and processes them in turn. Then, when command-line arguments are processed, any additional files specified by **-init-file filename** options are also processed. Finally, immediately before each *named* bibliography file is processed, an attempt is made to process an initialization file with the same name, but with the extension changed to *.ini*. The default extension can be changed by a setting of the environment variable **BIB-CLEANEXT**. That scheme permits system-wide, user-wide, session-wide, and file-specific initialization files to be supported.

When input is taken from *stdin*, there is no file-specific initialization.

For precise control, the **-no-read-init-files** option suppresses all initialization files except those explicitly named by **-init-file filename** options, either on the command line, or in requested initialization files.

Recursive execution of initialization files with nested **-init-file** options is permitted; if the recursion is circular, **bibclean** will finally get a non-fatal initialization file open failure after opening too many files. That terminates further initialization file processing. As the recursion unwinds, the files are all closed, then execution proceeds normally.

An initialization file may contain empty lines, comments from percent to end of line (just like **T_EX**), option switches, and field/pattern or field/pattern/message assignments. Leading and trailing spaces are ignored. That is best illustrated by a short example:

```
% This is a small bibclean initialization file

-init-file /u/math/bib/.bibcleanrc %% departmental patterns

chapter = "\"D\" "                %% 23

pages    = "\"D--D\" "            %% 23--27
```

```

volume  = "\"D \\an\\d D\" "      %% 11 and 12

year    = \
    "\"dddd, dddd, dddd\" " \
    "Multiple years specified."      %% 1989, 1990, 1991

-no-fix-names    %% do not modify author/editor lists

```

Long logical lines can be split into multiple physical lines by breaking at a backslash-newline pair; the backslash-newline pair is discarded. That processing happens while characters are being read, before any further interpretation of the input stream.

Each logical line must contain a complete option (and its value, if any), or a complete field/pattern pair, or a field/pattern/message triple.

Comments are stripped during the parsing of the field, pattern, and message values. The comment start symbol is not recognized inside quoted strings, so it can be freely used in such strings.

Comments on logical lines that were input as multiple physical lines via the backslash-newline convention must appear on the *last* physical line; otherwise, the remaining physical lines will become part of the comment.

Pattern strings must be enclosed in quotation marks; within such strings, a backslash starts an escape mechanism that is commonly used in UNIX software. The recognized escape sequences are:

\a	alarm bell (octal 007)
\b	backspace (octal 010)
\f	formfeed (octal 014)
\n	newline (octal 012)
\r	carriage return (octal 015)
\t	horizontal tab (octal 011)
\v	vertical tab (octal 013)
\ooo	character number octal <i>ooo</i> (e.g. \012 is linefeed). Up to 3 octal digits may be used.
\0xhh	character number hexadecimal <i>hh</i> (e.g., \0x0a is linefeed). <i>xhh</i> may be in either letter case. Any number of hexadecimal digits may be used.

Backslash followed by any other character produces just that character. Thus, \% gets a literal percent into a string (preventing its interpretation as a comment), \" produces a quotation mark, and \\ produces a single backslash.

An ASCII NUL (`\0`) in a string will terminate it; that is a feature of the C programming language in which **bibclean** is implemented.

Field/pattern pairs can be separated by arbitrary space, and optionally, either an equals sign or colon functioning as an assignment operator. Thus, the following are equivalent:

```

pages = "\"D--D\" "
pages : "\"D--D\" "
pages  "\"D--D\" "
    pages = "\"D--D\" "
    pages : "\"D--D\" "
pages  "\"D--D\" "

```

Each field name can have an arbitrary number of patterns associated with it; however, they must be specified in separate field/pattern assignments.

An empty pattern string causes previously-loaded patterns for that field name to be forgotten. That feature permits an initialization file to completely discard patterns from earlier initialization files.

Patterns for value strings are represented in a tiny special-purpose language that is both convenient and suitable for bibliography value-string syntax checking. While not as powerful as the language of regular-expression patterns, its parsing can be portably implemented in less than 3% of the code in a widely-used regular-expression parser (the GNU **regexp** package).

The patterns are represented by the following special characters:

<space>	one or more spaces
a	exactly one letter
A	one or more letters
d	exactly one digit
D	one or more digits
r	exactly one Roman numeral
R	one or more Roman numerals (i.e. a Roman number)
w	exactly one word (one or more letters and digits)
W	one or more space-separated words, beginning and ending with a word
.	one 'special' character, one of the characters <code><space>!#()*+,-./:;?[]~</code> , a subset of punctuation characters that are typically used in string values
:	one or more 'special' characters
X	one or more 'special'-separated words, beginning and ending with a word
\x	exactly one x (x is any character), possibly with an escape sequence interpretation given earlier
x	exactly the character x (x is anything but one of these pattern characters: <code>aAdDrRwW.:<space>\</code>)

The **X** pattern character is very powerful, but generally inadvisable, because it will match almost anything likely to be found in a `BIBTeX` value string. The reason for providing pattern matching on the value strings is to uncover possible errors, not mask them.

There is no provision for specifying ranges or repetitions of characters, but that can usually be done with separate patterns. It is a good idea to accompany the pattern with a comment showing the kind of thing it is expected to match. Here is a portion of an initialization file giving a few of the patterns used to match *number* value strings:

```
number =      "\"D\" "           %% 23
number =      "\"A AD\" "        %% PN LPS5001
number =      "\"A D(D)\\" "      %% RJ 34(49)
number =      "\"A D\" "         %% XNSS 288811
number =      "\"A D\\.D\" "      %% Version 3.20
number =      "\"A-A-D-D\" "      %% UMIAC-TR-89-11
number =      "\"A-A-D\" "        %% CS-TR-2189
number =      "\"A-A-D\\.D\" "    %% CS-TR-21.7
```

For a bibliography that contains only *article* entries, that list should probably be reduced to just the first pattern, so that anything other than a digit string fails the pattern-match test. That is easily done by keeping bibliography-specific patterns in a corresponding file with extension *.ini*, because that file is read automatically.

You should be sure to use empty pattern strings in the pattern file to discard patterns from earlier initialization files.

The value strings passed to the pattern matcher contain surrounding quotes, so the patterns should also. However, you could use a pattern specification like `"\D"` to match an initial digit string followed by anything else; the omission of the final quotation mark `\` in the pattern allows the match to succeed without checking that the next character in the value string is a quotation mark.

Because the value strings are intended to be processed by \TeX , the pattern matching ignores braces, and \TeX control sequences, together with any space following those control sequences. Spaces around braces are preserved. That convention allows the pattern fragment *A-AD-D* to match the value string *TN-K/slash 27-70*, because the value is implicitly collapsed to *TN-K27-70* during the matching operation.

bibclean's normal action when a string value fails to match any of the corresponding patterns is to issue a *warning* message something like this: *"Unexpected value in "year = "192""*. In most cases, that is sufficient to alert the user to a problem. In some cases, however, it may be desirable to associate a different message with a particular pattern. That can be done by supplying a message string following the pattern string. Format items `%%` (single percent), `%e` (entry name), `%f` (field name), `%k` (citation key), and `%v` (string value) are available to get current values expanded in the messages. Here is an example:

```
chapter = "\D:D\" "Colon found in ``%f = %v''" %% 23:2
```

To be consistent with other messages output by **bibclean**, the message string should *not* end with punctuation.

If you wish to make the message an error, rather than just a warning, begin it with a query (?), like this:

```
chapter = "\D:D\" "?Colon found in ``%f = %v''" %% 23:2
```

The query will not be included in the output message.

Escape sequences are supported in message strings, just as they are in pattern strings. You can use that to advantage for fancy things, such as terminal display mode control. If you rewrite the previous example as

```
chapter = "\D:D\" \
"?\033[7mColon found in ``%f = %v''\033[0m" %% 23:2
```

the error message will appear in inverse video on display screens that support ANSI terminal control sequences. Such practice is not normally recommended, because it may have undesirable effects on some output devices. Nevertheless, you may find it useful for restricted applications.

For some types of bibliography fields, **bibclean** contains special-purpose code to supplement or replace the pattern matching:

- *CODEN*, *ISBN* and *ISSN* field values are handled that way because their validation requires evaluation of checksums that cannot be expressed by simple patterns; no patterns are even used in these three cases.
- When **bibclean** is compiled with pattern-matching code support, *chapter*, *number*, *pages*, and *volume* values are checked only by pattern matching.
- *month* values are first checked against the standard \BibTeX month abbreviations, and only if no match is found are patterns then used.
- *year* values are first checked against patterns, then if no match is found, the year numbers are found and converted to integer values for testing against reasonable bounds.

Values for other fields are checked only against patterns. You can provide patterns for *any* field you like, even ones **bibclean** does not already know about. New ones are simply added to an internal table that is searched for each string to be validated.

The special field, *key*, represents the bibliographic citation key. It can be given patterns, like any other field. Here is an initialization file pattern assignment that will match an author name, a colon, a four-digit year, a colon, and an alphabetic string, in the BibNet Project style:

```
key = "A:dddd:A"                                %% Knuth:1986:TB
```

Notice that no quotation marks are included in the pattern, because the citation keys are not quoted. You can use such patterns to help enforce uniform naming conventions for citation keys, which is increasingly important as your bibliography data base grows.

ISBN INITIALIZATION FILES

bibclean contains a compiled-in table of ISBN ranges and country/language settings that is suitable for most applications.

However, ISBN data changes yearly, as new countries adopt ISBNs, and as publishers are granted new, or additional, ISBN prefixes.

Thus, from version 2.12, **bibclean** supports reading of run-time ISBN initialization files found on the **PATH** (for VAX VMS, **SYS\$SYSTEM**) and **BIBINPUTS** search paths, and then any specified by **-ISBN-file filename** options.

That feature makes it possible to incorporate new ISBN data without having to produce a new **bibclean** release and reinstall the software at end-user sites.

The format of an ISBN initialization file is similar to that of the **bibclean** initialization files described in the preceding section: comments begin with percent and continue to end of line, blank and empty lines are ignored, backslash-newline joins adjacent lines, and otherwise, lines are expected to contain a required pair of ISBN country/language-publisher prefixes forming a non-decreasing range, optionally followed by one or more words of text which are treated as the country/language group value. The latter value plays no part in ISBN validation, but its presence is strongly recommended, in order to make the ISBN table more understandable for humans.

Here is a short example:

```
%% The Faeroes got ISBN assignments between 1993 and 1998
99918-0          99918-3          Faeroes
99918-40         99918-61
99918-900        99918-938
```

It is not necessary to repeat the country names on succeeding entries with the same initial number (99918 in that example); that will be handled internally.

Data from ISBN files normally augments the compiled-in data. However, if the first prefix begins with a hyphen, then **bibclean** deletes the first entry in the table matching that first prefix (ignoring the leading hyphen):

```
%% Latvia got ISBN ranges between 1993 and 1998
%% so we remove the old placeholder, then add the
%% new ranges.
-9984-0          9984-9          This one is no longer valid

9984-00          9984-20          Latvia
9984-500         9984-770
9984-9000        9984-9984
```

KEYWORD INITIALIZATION FILES

bibclean contains a compiled-in table of keyword mappings that is suitable for most applications. The default settings merely adjust lettercase in certain keyword names, so that, for example, *isbn* is output as *ISBN*.

From version 2.12, **bibclean** supports reading of run-time keyword initialization files found on the **PATH** (for VAX VMS, **SYS\$SYSTEM**) and **BIBINPUTS** search paths, and then any specified by **-keyword-file filename** options.

That feature makes it possible to incorporate special spellings of new keywords without having to produce a new **bibclean** release and reinstall the software at end-user sites.

The format of a keyword initialization file is similar to that of the other **bibclean** initialization files described in the preceding sections: comments begin with percent and continue to end of line, blank and empty lines are ignored, backslash-newline joins adjacent lines, and otherwise, lines are expected to contain a required pair of old and new keyword names.

Here is a short example:

```
%% We want special handling of MathReviews keywords
mrclass      MRclass
mrnumber     MRnumber
mrreviewer   MRreviewer
```

Data from keywords files normally augments the compiled-in data. However, if the first keyword begins with a hyphen, then **bibclean** deletes the first entry in the table matching that keyword (ignoring the leading hyphen):

```
%% Remove special handling of ISBN, ISSN, and LCCN values.
-issn        ISSN
-isbn        ISBN
-lccn        LCCN
```

Even though the second keyword in each deletion pair is not used, it still must be specified.

Notice that this feature can be used to regularize keyword names, but use it with care, in order to avoid producing duplicate key names in output B_IB_TE_X entries:

```
%% Map variations of keywords into a common name:
keys         keywords
keywds       keywords
keyword      keywords
keywrđ       keywords
keywrds      keywords
searchkey    keywords
```

LEXICAL ANALYSIS

When **-no-prettyprint** is specified, **bibclean** acts as a lexical analyzer instead of a prettyprinter, producing output in lines of the form

```
<token-number><tab><token-name><tab>"<token-value>"
```

Each output line contains a single complete token, identified by a small integer number for use by a computer program, a token type name for human readers, and a string value in quotes.

Special characters in the token value string are represented with ANSI/ISO Standard C escape sequences, so all characters other than NUL are representable, and multi-line values can be represented in a single line.

Here are the token numbers and token type names that can appear in the output when **-prettyprint** is specified:

```
0 UNKNOWN
1 ABBREV
2 AT
3 COMMA
4 COMMENT
5 ENTRY
6 EQUALS
7 FIELD
8 INCLUDE
9 INLINE
10 KEY
11 LBRACE
12 LITERAL
13 NEWLINE
```

```

14 PREAMBLE
15 RBRACE
16 SHARP
17 SPACE
18 STRING
19 VALUE

```

Programs that parse such output should also be prepared for lines beginning with the warning prefix, `%%`, or the error prefix, `??`, and for ANSI/ISO Standard C line number directives of the form

```
# line 273 "texbook1.bib"
```

which record the line number and file name of the current input file.

If a `-max-width nnn` command-line option was specified, long output lines will be wrapped at a backslash-newline pair, and consequently, software that processes the lexical token stream should be prepared to collapse such wrapped lines back into single lines.

As an example of the use of `-no-prettyprint`, the UNIX command pipeline

```

bibclean -no-prettyprint mylib.bib | \
awk '$2 == "KEY" {print $3}' | \
sed -e 's/"//g' | \
sort

```

will extract a sorted list of all citation keys in the file *mylib.bib*.

A certain amount of processing will have been done on the tokens. In particular, delimiters equivalent to braces will have been replaced by braces, and braced strings will have become quoted strings.

The LITERAL token type is used for arbitrary text that **bibclean** does not examine further, such as the contents of a `@Preamble{...}` or a `@Comment{...}`.

The UNKNOWN token type should never appear in the output stream. It is used internally to initialize token type variables.

SCRIBE BIBLIOGRAPHY FORMAT

bibclean's support for the SCRIBE bibliography format is based on the syntax description in the SCRIBE Introductory User's Manual, 3rd Edition, May 1980. SCRIBE was originally developed by Brian Reid at Carnegie-Mellon University, and is now marketed by Unilogic, Ltd.

The `BIBTEX` bibliography format was strongly influenced by SCRIBE, and indeed, with care, it is possible to share bibliography files between the two systems. Nevertheless, there are some differences, so here is a summary of features of the SCRIBE bibliography file format:

- (1) Letter case is not significant in field names and entry names, but case is preserved in value strings.
- (2) In field/value pairs, the field and value may be separated by one of three characters: `=`, `/`, or space. Space may optionally surround these separators.
- (3) Value delimiters are any of these seven pairs: `{ }` `[]` `()` `< >` `' '` `" "` `` ``.
- (4) Value delimiters may not be nested, even though with the first four delimiter pairs, nested balanced delimiters would be unambiguous.
- (5) Delimiters can be omitted around values that contain only letters, digits, sharp (`#`), ampersand (`&`), period (`.`), and percent (`%`).
- (6) Outside of delimited values, a literal at-sign (`@`) is represented by doubled at-signs (`@@`).
- (7) Bibliography entries begin with `@name`, as for `BIBTEX`, but any of the seven SCRIBE value delimiter pairs may be used to surround the values in field/value pairs. As in (4), nested delimiters are forbidden.
- (8) Arbitrary space may separate entry names from the following delimiters.
- (9) `@Comment` is a special command whose delimited value is discarded. As in (4), nested delimiters are forbidden.

- (10) The special form

```
@Begin{comment}
...
@end{comment}
```

permits encapsulating arbitrary text containing any characters or delimiters, other than “@End{comment}”. Any of the seven delimiter pairs may be used around the word “comment” following the “@Begin” or “@End”; the delimiters in the two cases need not be the same, and consequently, “@Begin{comment}”/“@End{comment}” pairs may *not* be nested.

- (11) The *key* field is required in each bibliography entry.
- (12) A backslashed quote in a string will be assumed to be a \TeX accent, and braced appropriately. While such accents do not conform to SCRIBE syntax, SCRIBE-format bibliographies have been found that appear to be intended for \TeX processing.

Because of that loose syntax, **bibclean**’s normal error detection heuristics are less effective, and consequently, SCRIBE mode input is not the default; it must be explicitly requested.

ENVIRONMENT VARIABLES

- BIBCLEANEXT** File extension of bibliography-specific initialization files. Default: *.ini*.
- BIBCLEANINI** Name of **bibclean** initialization files. Default: *.bibcleanrc* (UNIX), *bibclean.ini* (non-UNIX).
- BIBCLEANISBN** Name of **bibclean** ISBN initialization files. Default: *.bibclean.isbn* (UNIX), *bibclean.isb* (non-UNIX).
- BIBCLEANKEY** Name of **bibclean** keyword initialization files. Default: *.bibclean.key* (UNIX), *bibclean.key* (non-UNIX).
- BIBINPUTS** Search path for **bibclean** and \BibTeX input files. On UNIX, it is a colon-separated list of directories that are searched in order from first to last. It is not an error for a specified directory to not exist.
- On other operating systems, the directory names should be separated by whatever character is used in system search path specifications, such as a semicolon on IBM PC DOS.
- PATH** On Atari TOS, IBM PC DOS, IBM PC OS/2, Microsoft NT, and UNIX, search path for system executable files. The system-wide **bibclean** initialization file is searched for in that path.
- SYSS\$SYSTEM** On VAX VMS, search path for system executable files and the system-wide **bibclean** initialization file.

FILES

- *.bib* \BibTeX and SCRIBE bibliography data base files.
- *.ini* File-specific initialization files.
- .bibclean.isbn* UNIX system-wide and user-specific ISBN initialization files.
- .bibclean.key* UNIX system-wide and user-specific keyword initialization files.
- .bibcleanrc* UNIX system-wide and user-specific initialization files.
- bibclean.ini* Non-UNIX system-wide and user-specific initialization files.
- bibclean.isb* Non-UNIX system-wide and user-specific ISBN initialization files.
- bibclean.key* Non-UNIX system-wide and user-specific keyword initialization files.

SEE ALSO

bibcheck(1), bibdup(1), bibextract(1), bibindex(1), bibjoin(1), biblabel(1), biblex(1), biblook(1), biborder(1), bibparse(1), bibsearch(1), bibsort(1), bibtex(1), bibunlex(1), citefind(1), citesub(1), cite-tags(1), latex(1), scribe(1), tex(1).

AUTHOR

Nelson H. F. Beebe
 University of Utah
 Department of Mathematics, 110 LCB
 155 S 1400 E RM 233
 Salt Lake City, UT 84112-0090
 USA
 Tel: +1 801 581 5254
 FAX: +1 801 581 4148
 Email: beebe@math.utah.edu, beebe@acm.org, beebe@computer.org (Internet)
 URL: <http://www.math.utah.edu/~beebe>

COPYRIGHT

```
#####
#####
#####
###
###      bibclean: prettyprint and syntax check BibTeX and Scribe      ###
###                               bibliography data base files          ###
###
###                               Copyright (C) 1990--2013 Nelson H. F. Beebe      ###
###
### This program is covered by the GNU General Public License (GPL), ###
### version 2 or later, available as the file COPYING in the program ###
### source distribution, and on the Internet at                        ###
###
###                               ftp://ftp.gnu.org/gnu/GPL              ###
###
###                               http://www.gnu.org/copyleft/gpl.html    ###
###
### This program is free software; you can redistribute it and/or    ###
### modify it under the terms of the GNU General Public License as    ###
### published by the Free Software Foundation; either version 2 of    ###
### the License, or (at your option) any later version.               ###
###
### This program is distributed in the hope that it will be useful,    ###
### but WITHOUT ANY WARRANTY; without even the implied warranty of    ###
### MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the    ###
### GNU General Public License for more details.                      ###
###
### You should have received a copy of the GNU General Public         ###
### License along with this program; if not, write to the Free        ###
### Software Foundation, Inc., 59 Temple Place, Suite 330, Boston,     ###
### MA 02111-1307 USA                                                  ###
#####
#####
#####
```