BibTeX2HTML

A translator of BibTeX bibliographies into HTML

Version 1.99 — March 25, 2023

Jean-Christophe Filliâtre and Claude Marché
http://www.lri.fr/~filliatr/bibtex2html

Contents

1 Introduction

BibTeX2HTML is a collection of tools for producing automatically HTML documents from bibliographies written in the BibTeX format. It consists in three command line tools:

- `bib2bib` is a filter tool that reads one or several bibliography files, filters the entries with respect to a given criterion, and outputs the list of selected keys together with a new bibliography file containing only the selected entries;

- `bibtex2html` is a translator that reads a bibliography file and outputs two HTML documents that contains respectively the cited bibliography in a nice presentation, and the original BibTeX file augmented with several transparent HTML links to allow easy navigation. `bibtex2html` can handle *any* BibTeX style file, including those producing multiple bibliographies.

- `aux2bib` reads a `.aux` file as produced by LaTeX and writes to standard output a BibTeX file containing exactly the BibTeX entries refereed in the `.aux` file.

2 The `bibtex2html` command tool

bibtex2html is a BibTeX to HTML translator. It is invocated as

`bibtex2html` [options] [*file.bib*]

where the possible options are described below and where *file.bib* is the name of the BibTeX file, which must have a *.bib* suffix. If this file is not given, then entries are input from standard input.

Then two HTML documents are created (unless option `-nobibsource` is selected or input is standard input, see below) :

- *file.html* which is the bibliography in HTML format ;

- *file_ bib.html* which contains all the entries in ASCII format.

`bibtex` is called on *file.bib* in order to produce the a LaTeX document, and then this LaTeX document is translated into an HTML document. The BibTeX file *file.bib* is also parsed in order to collect additional fields (abstract, url, ps, http, etc.) that will be used in the translation.

If input is standard input and option `--output` is not given, the first file is output to standard output, and the second is not created.

2.1 Additional fields and automatic web links

The main interest of `bibtex2html` with respect to a traditional LaTeX to HTML translator is the use of additional fields in the BibTeX database and the automatic insertion of web links.

A link is inserted:

- at each cross-reference inside the bibliography entries;

- when the `\url` LaTeX macro is used in the text;

- for each BibTeX field whose name is "ftp", "http", "url", "ps" ,"dvi", "rtf", "pdf", "documenturl", "urlps" or "urldvi". The name of this link depends on the nature of the link:

  - it is the file suffix, whenever this suffix is `.dvi`, `.ps`, `.pdf`, `.rtf`, `.txt` or `.html`, possibly followed by a compression sufix, `.gz`, `.Z` or `.zip`;
  - otherwise the name of the link is either `http` or `ftp` depending on the protocol.

  You can insert web link for other fields and/or specify alternative names for the links using the options `-f` and `-nf` (see below).

2.1.1 Abstracts

If a BibTeX entry contains a field `abstract` then its contents is quoted right after the bibliography entry .

This behavior may be suppressed with the option `--no-abstract`.

If you want both versions with and without abstracts, use the option `--both`. In that case, links named "Abstract" will be inserted from the page without abstracts to the page with abstracts,

2.1.2 Keywords

If a BibTeX entry contains a field `keywords` then its contents is displayed after the bibliography entry (and after the abstract if any).

This behavior may be suppressed with the option `--no-keywords`.

2.2 Command line options

Most of the command line options have a short version of one character (e.g. `-r`) and an easy-to-remember/understand long version (e.g. `--reverse-sort`).

2.2.1 General aspect of the HTML document

`-t` ***string***, `--title` ***string***

specify the title of the HTML file (default is the file name).

`--header` ***string***

give an additional header for the HTML document.

`--footer` ***string***

give an additional footer for the HTML document.

`-s` ***string***, `--style` ***string***

use BibTeX style *string* (plain, alpha, etc.). Default style is plain.

`-noabstract`, `--no-abstract`

do not print the abstracts (if any).

`-nokeywords`, `--no-keywords`

do not print the keywords (if any).

`-both`, `--both`

produce both pages with and without abstracts. If the BibTeX file is foo.bib then the two pages will be respectively foo.html and foo_abstracts.html (The suffix may be different, see option `--suffix`). Links are inserted from the page without abstracts to the page with abstracts.

`-nokeys`, `--no-keys`

do not print the cite keys. Note: this option implicitly suppresses the use of HTML tables to format the entries; to enforce the use of tables, use option `-use-table` (passing it *after* option `-nokeys` on the command line).

`-use-keys`, `--use-keys`

use the cite keys from the BibTeX input file (and not the ones generated by the BibTeX style file).

`-rawurl`, `--raw-url`

print URLs instead of files' types.

`-heveaurl`, `--hevea-url`

interpret the macro `\url` as HeVeA's one, i.e. with two arguments, the first one being the url and the second one the text to print. The default behavior is to interpret the macro `\url` as the one from the package `url`, which has only one argument (the url itself).

`-f` ***field***, `--field` ***field***

add a web link for that BibTeX field.

`-nf` ***field string***, `--named-field` ***field string***

similar to `-f` but specifies the way to display the link (e.g. `-nf springer "At Springer's"`).

`-note` ***field***, `--note` ***field***

declare that a field must be treated like the `abstract` field, i.e. is an annotation to be displayed as a text paragraph below the entry.

`-multiple`, `--multiple`

make a separate web page for each entry. *Beware: this option produces as many HTML files as BibTeX entries!*

`-single`, `--single`

produce a single document, inserting each BibTeX entry (the input) right after its BibTeX output

`-bg` ***color***, `--background` ***color***

set the background color of the HTML file (default is none).

`-css` ***file***, `--style-sheet` ***file***

set a style sheet file for the HTML document (default is none).

`-dl`, `--dl`

use HTML `DL` lists instead of HTML tables to format entries.

`-unicode`, `--unicode`

use Unicode entities for the following macros :

```
\models \curlyvee \curlywedge \bigcirc \varepsilon
\not{\models}
```

`-html-entities`, `--html-entities`

use HTML entities for the following macros :

```
\= \Im \Leftarrow \Re \Rightarrow \aleph \ang \angle \approx
\ast \cdot \cdots \cong \copyright \cup \dagger \diamond \emptyset
\equiv \exists \forall \ge \geq \in \infty \int \land \lang
\lceil \le \leftarrow \leftrightarrow \leq \lfloor \longleftarrow
\longrightarrow \lor \lozenge \nabla \ne \neg \neq \ni \notin
\oplus \otimes \partial \perp \pm \prod \propto \rang \rceil
\rfloor \rightarrow \sim \simeq \sqrt \subset \subseteq
\sum \supset \supseteq \therefore \times \tm \to \vartheta
\vee \wedge \wp
```

2.2.2 Controlling the translation

`-m` ***file***, `--macros-from` ***file***

read the LaTeX macros in the given file. Note: `bibtex2html` does not handle macros arguments; arguments are simply discarded.

`-noexpand --no-expand`

do not expand the abbreviation strings, leave them in the output file.

2.2.3 Selecting the entries

`-citefile` ***filename***, `--citefile` ***filename***

Select only keys appearing in *filename*. To be used manually or in conjonction with `bib2bib`.

`-e` ***key***, `--exclude` ***key***

exclude an particular entry.

2.2.4 Sorting the entries

`-d, --sort-by-date`

sort by date.

`-a, --sort-as-bibtex`

sort as BibTeX (usually by author).

`-u, --unsorted`

unsorted i.e. same order as in .bib file (default).

`-r, --reverse-sort`

reverse the sort.

`--revkeys`

number entries in reverse order (i.e. from n to 1 in plain style).

2.2.5 Miscellaneous options

`-nodoc`, `--nodoc`

do not produce a full HTML document but only its body (useful to merge the HTML bibliography in a bigger HTML document).

`-nobibsource`, `--nobibsource`

do not produce the `_bib.html` file. In that case, no "BibTeX entry" link are inserted in the HTML file.

`-suffix` ***string***, `--suffix` ***string***

give an alternate suffix *string* for both HTML files and links (default is `.html`).

`-fsuffix` ***string***, `--file-suffix` ***string***

give an alternate suffix *string* for HTML files (default is `.html`).

`-lsuffix` ***string***, `--link-suffix` ***string***

give an alternate suffix *string* for HTML links (default is `.html`).

`-o` ***file***, `--output` ***file***

specifies the output file. If *file* is `-`, then the standard output is selected.

`-c` ***command***, `--command` ***command***

specify the BibTeX command (default is `bibtex -min-crossrefs=1000`). May be useful for example if you need to specify the full path of the `bibtex` command.

`--print-keys`

print the BibTeX entries on the standard output (one per line), as selected and sorted by `bibtex2html`. This is useful if you want to use the selection and sorting facilities of `bibtex2html` in another program. Note: you may need to set also the `-q` option (quiet) to suppress the usual output.

`-i`, `--ignore-errors`

ignore BibTeX errors.

`-q`, `--quiet`

be quiet.

`-w`, `--warn-error`

stop at the first warning.

`-h`, `--help`

print a short usage and exit.

`-v`, `--version`

print the version and exit.

`-noheader`, `--no-header`

do not insert the `bibtex2html` command in the HTML document (default is to insert it as a comment at the beginning).

3 The `bib2bib` command line tool

`bib2bib` is a tool for extracting some entries from a list of bibliography files. It is invocated as

`bib2bib` [options] *file1.bib* $\cdots$ *filen.bib*

where the possible options are described below and where *file1.bib* $\cdots$ *filen.bib* are the names of the BibTeX files, which must have a *.bib* suffix. If no files at all are given on the command line, then input is taken from standard input.

The options allow to specify a filter condition to test against each references read from bib files. The result will be a new BibTeX file containing all the entries of the input files that satisfy the condition. Notice that this output file contains all the necessary informations: each string and each cross-reference needed will be also in that file.

Additionally, *bib2bib* may output a file containing all the keys of entries that satisfy the condition. This second file is suitable for input as option `-citefile` to `bibtex2html`.

3.1 Command line options

`-c` ***condition***

specify a condition for selecting the entries. The output will retain only the entries that satisfy this condition. If several such condition are given, then only the entries that satisfy all the conditions are selected. The syntax of conditions is given below, notice that it is better to escape shell expansions in that conditions, in other words, you should write conditions between quotes.

`-ob` ***filename***

specify the filename where the selected entries are output. If not given, it defaults to standard output.

`-oc` ***filename***

specify the filename where the list of selected keys is output. If not given, this file is not created.

Notice that the two output files above are suitable for use with bibtex2html. A typical use would be

`bib2bib -oc` *citefile* `-ob` *bibfile.bib* `-c` *condition* `file1.bib file2.bib ...`
`bibtex2html -citefile` *citefile* `bibfile.bib`

which will produce exactly the HTML file for the selected references.

`--expand`

expand all abbreviations in the output file.

`--expand-xrefs`

expand all crossrefs in the output file. Notice that the meaning of such an expansion is not completely obvious: it's better to let bibtex (via bibtex2html) handle the cross-references itself, depending on the style considered.

Notice that `bibtex2html` itself will expand the strings (by default, unless you specify the `-noexpand` option) but not the cross-references.

`--no-comment`

prevent generation of extra comments at beginning of output bib file.

`--remove` ***f***

remove all occurrences of field *f*. This option can be used several times to remove several fields.

`--rename` ***f1 f2***

rename all occurrences of field *f1* into `f2`. This option can be used several times to rename several fields. Beware that if an entry already has both fields `f1` and `f2`, this will result in two fields *f2*, and BibTeX styles usually take only the first occurrence into account.

Example:

`bib2bib --remove abstract --remove copyright --rename x-pdf url` *bibfile.bib*

removes all `abstract` and `copyright` fields and rename all `x-pdf` fields into name `url`.

`-s` ***f***

sorts the entries of the bibliography with respect to the given field *f*, which may also be `$key` or `$type` to refer to the key or to the entry type, as for filter conditions. It may also be `$date`, to ask for sorting from oldest to newest, as for option `-d` of bibtex2html.

This option may be used several times to specify a lexicographic order, such as by author, then by type, then by date:

`bib2bib -s 'author' -s '$type' -s '$date'` *bibfile.bib*

When sorting, the resulting bibliography will always contains the comments first, then the preambles, then the abbreviations, and finally the regular entries. Be warned that such a sort may put cross-references before entries that refer to them, so be cautious.

`-r`

reverses the sort order.

`-q, --quiet`

be quiet.

`-w, --warn-error`

stop at the first warning.

`--php-output` ***file***

outputs the bib file as a two-dimensional array in PHP syntax, in *file*.

3.2 Filter conditions

A filter condition is a boolean expression that is evaluated against a BibTeX entry to decide whether this entry should be selected. A condition is either:

- a *comparison* between two *expressions*, written as e_1 *op* e_2 ;
- a matching of a field name with respect to a *regular expression*, written as $field : regexp$;

- a conjunction of two conditions, written as c_1 `and` c_2 (or c_1 `&` c_2) ;
- a disjunction of two conditions, written as c_1 `or` c_2 (or c_1 `|` c_2);
- a negation of a condition, written as `not` c (or `!` c) ;
- a test of existence of a field, written as `exists` f (or `?` f) where f is a field name ;

where an expression is either:

- a string constant, written either between double quotes or single quotes ;
- an integer constant ;
- a field name ;
- the special ident `$key` which corresponds to the key of an entry.
- the special ident `$type` which corresponds to the type of an entry (ARTICLE, INPROCEEDINGS, etc.). Notice that an entry type is always written in uppercase letters.

Comparison operators are the usual ones: `=`, `<`, `>`, `<=`, `>=` and `<>`.

The field names are any sequences of lowercase or uppercase letters (but no distinction is made between lowercase and uppercase letters).

Be careful when writing conditions in a shell command: the shell must not interpret anything in a condition, such as `$key`. So usually you need to put conditions inside quote characters that forbid shell interpretation: single quotes under Unix shell, or double quotes under Microsoft Windows shell. This is why strings in conditions may be put indifferently between single or double quotes: you will use the ones which are not the ones you use to forbid shell interpretation. In examples below, we will use Unix convention, so under Windows you have to permute the use of single and double quotes.

Note that within `Makefile`s you have to escape the `$` character in `$key` or `$type` (using `$$key` and `$$type` instead, at least with GNU make).

Regular expressions must be put between single or double quotes, and must follow the GNU syntax of regular expressions, as for example in GNU Emacs. Any character other than `$^.*+?[]` matches itself, see Table 1 for the meaning of the special characters.

Notice that if several conditions are given with option `-c` on the command line, then they are understood as the conjunction of them, in other words

`bib2bib -c '`c_1`' ⋯ -c '`c_n`'`

is equivalent to

`bib2bib -c '`c_1 `and ⋯ and` c_n`'`

Table 2 shows a formal grammar for conditions.

Remarks on evaluation of conditions

- According to BibTeX conventions, entry types, keys and field names have to be considered case insensitive, that is no distinction is made between uppercase and lowercase letters. Inside `bib2bib`, these are always converted to uppercase, so you may take this into account when writting conditions (see below).

`.`	matches any character except newline
`[..]`	character set; ranges are denoted with `-`, as in `[a-z]`; an initial `^`, as in `[^0-9]`, complements the set
`^`	matches the beginning of the string matched
`$`	matches the end of the string matched
`\r`	matches a carriage return
`\n`	matches a linefeed
`\t`	matches a tabulation
`\b`	matches word boundaries
`\`*ddd*	matches character with ASCII code *ddd* in decimal
`\`*char*	quotes special character *char* among `$^.*+?[]\`
regexp`*`	matches *regexp* zero, one or several times
regexp`+`	matches *regexp* one or several times
regexp`?`	matches *regexp* once or not at all
regexp1 `|` *regexp2*	alternative between two regular expressions, this operator has low priority against `*`, `+` and `?`
`\(` *regexp* `\)`	grouping regular expression

Table 1: Syntax of regular expressions

$$
\begin{array}{rcl}
\mathit{Cond} & \rightarrow & \mathit{Cond}\ \texttt{and}\ \mathit{Cond} \mid \mathit{Cond}\ \texttt{or}\ \mathit{Cond} \mid \texttt{not}\ \mathit{Cond} \mid \texttt{exists}\ \mathit{Id} \\
\mathit{Cond} & \rightarrow & \mathit{Cond}\ \texttt{\&}\ \mathit{Cond} \mid \mathit{Cond}\ \texttt{|}\ \mathit{Cond} \mid \texttt{!}\ \mathit{Cond} \mid \texttt{?}\ \mathit{Id} \\
 & & \mid \mathit{Expr}\ \mathit{Comp}\ \mathit{Expr} \mid \mathit{Expr} : \mathit{String} \mid \texttt{(}\ \mathit{Cond}\ \texttt{)} \\
\mathit{Comp} & \rightarrow & \texttt{=} \mid \texttt{>} \mid \texttt{<} \mid \texttt{>=} \mid \texttt{<=} \mid \texttt{<>} \\
\mathit{Expr} & \rightarrow & \mathit{Id} \mid \mathit{String} \mid \mathit{Int} \mid \texttt{\$key} \mid \texttt{\$type} \\
\mathit{Id} & \rightarrow & [\texttt{a} - \texttt{z}\texttt{A} - \texttt{Z}]^{+} \\
\mathit{String} & \rightarrow & \texttt{"([\^{}"\textbackslash] | \textbackslash" | \textbackslash\textbackslash)*"} \mid \texttt{'([\^{}'\textbackslash] | \textbackslash' | \textbackslash\textbackslash)*'} \\
\mathit{Integer} & \rightarrow & [\texttt{0} - \texttt{9}]^{+}
\end{array}
$$

Table 2: Syntax of conditions

- On the other hand, case matters when comparing strings, or matching them against regular expressions. For example, `title : "Computer"` may return `true` if the title contains the word `Computer` with a capital letter, whereas `title : "computer"` would return `false`.
- A consequence of the two previous remarks, is that if you want for example to check equality of the entry type and a string value, put the value in uppercase, as for example `$type = "INPROCEEDINGS"`, otherwise the condition would be always false.
- When performing a comparison with an non-existent field, the result is always false ; beware that this means that for example `not (f = "value")` and `f <> "value"` are not equivalent: for an entry that does not have a field `f`, the first condition is true whereas the second is false.
- As usual, `not` has higher priority than `and`, which itself has higher priority than `or`. `and` and `or` associate to the left.

- Comparison using `>`, `<`, `>=` and `<=` may only be used between integer values. In any other case, a warning is displayed and the result is false.
- There is a special handling for strings containing LaTeX accented letters (or for backward compatibility, ISO-Latin1 accented characters): each variant of writing such letters are considered the same, and equivalent to their HTML entity form, for example strings `"Filli\^atre"`, `"Filli{\^a}tre"`, `"Filli\^{a}tre"` and `"Filliâtre"` are considered identical and indeed equal to `"Filli&acirc;tre"`. Note that when using such a string as a regular expression, there is no need to escape the backslash, since interpretation of LaTeX accenting commands is made before interpretation into a regexp. Using HTML entities for matching accented names is thus considered as the safest method.

3.3 Examples

Here are some examples to help you writing the filter conditions you are interested in.

3.3.1 Selecting entries of a given year

The following command reads input files `biblio1.bib` and `biblio2.bib`, and select only entries that appeared in 1999 :

```
bib2bib -oc cite1999 -ob 1999.bib -c 'year=1999' biblio1.bib biblio2.bib
```

The resulting file `cite1999` contains the list of keys selected. You can then produce the HTML file by

```
bibtex2html -citefile cite1999 1999.bib
```

You may also select references appeared after and/or before a given year. For example, references after 1997:

```
bib2bib -oc citeaft1997 -ob aft1997.bib -c 'year>1997' biblio.bib
```

or between 1990 and 1995:

```
bib2bib -oc cite90-95 -ob 90-95.bib -c 'year>=1990 and year<=1995' biblio.bib
```

3.3.2 Selecting references of a given author

The following command reads input files `biblio.bib` and select only entries whose (co)author is Donald Knuth:

```
bib2bib -oc knuth-citations -ob knuth.bib -c 'author : "Knuth"' biblio.bib
```

More complicated, if you would like to have only the references whose author is Knuth only, you may try

```
bib2bib -oc knuth-citations -ob knuth.bib \
 -c 'author : "^\(Donald \(E. \)?Knuth\|Knuth, Donald \(E.\)?\)$"' biblio.bib
```

or equivalently but missing the possible "`E.`":

```
bib2bib -oc knuth-citations -ob knuth.bib -c 'author = "Donald Knuth"
or author = "Knuth, Donald"' biblio.bib
```

3.3.3 Other examples

Any boolean combination of comparison and/or matching are possible. For example, the following command extract the references that appeared since 1995 and have lambda-calculus in their title, with anything between "lambda" and "calculus":

```
bib2bib -oc lambda -c 'year >= 1995 and title : "lambda.*calculus"' biblio.bib
```

for example, it will select a title containing `$\lambda$-calculus`.

3.4 Note on duplicates entries

`bib2bib` has the effect of merging several bib files into a single one. This process may result in duplicate entries in the resulting files, which is considered as erroneous by `bibtex`. Of course, this is not really a bug of `bib2bib` since it is of your own to take care not having entries with the same key.

However, there are two particular cases when this occurs naturally: when two bib files share common abbreviations, or when they share common cross-references.

In order to make `bib2bib` behaves correctly in such a case, it is designed as follows: for repeated abbrevs, the first abbrev is kept and the others are ignored, and for repeated regular entries, the last entry is kept and the others are ignored. With this behaviour, everything works well as soon as repeated abbrevs are really duplicate abbrevs of the same sentence, and repeated keys are really duplicate entries.

4 The `aux2bib` command line tool

`aux2bib` is a tool extracting the BibTeX references from a `.aux` file (as produced by LaTeX) and building the corresponding BibTeX file. It is invocated as

`aux2bib` *file.aux*

The BibTeX file is written on the standard output.

5 Frequently Asked Questions

1. **How may I tell bibtex2html to expand cross-references?**

   By default, all entries of the input BibTeX file are translated into HTML, including cross-references. Since the latter are there, bibtex will never expand them. If you want them to be expanded, you have to tell bibtex2html that crossref entries need not be in the resulting file. To do that you have to use the option `-citefile` to give the exact list of entries you want to see. If a cross-reference is not in that list, then its fields will be expanded into all entries that cross-refers to it. (Technically, this work because bibtex2html calls bibtex with option `-min-crossrefs=1000` by default.)

2. **When running**

   ```
   bib2bib -oc knuth-citations -ob knuth.bib -c 'author : "Knuth"' biblio.bib
   ```

   I get "Lexical error in condition: Unterminated string". What's going wrong?

   You are probably running `bib2bib` under Microsoft Windows, hence you should permute the use of single quotes and double quotes, as explained in Section 3.2:

   ```
   bib2bib -oc knuth-citations -ob knuth.bib -c "author : 'Knuth'" biblio.bib
   ```