



**HAL**  
open science

## **BibTeX, mlBibTeX and Bibliography Styles**

Jean-Michel Hufflen

► **To cite this version:**

Jean-Michel Hufflen. BibTeX, mlBibTeX and Bibliography Styles. Biuletyn GUST, 2006, pp.76–80. <hal-00644459>

**HAL Id: hal-00644459**

**<https://hal.science/hal-00644459v1>**

Submitted on 24 Nov 2011

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization

# BIB<sub>TEX</sub>, MIBIB<sub>TEX</sub> and Bibliography Styles\*

Jean-Michel HUFFLEN

LIFC (FRE CNRS 2661)

University of Franche-Comté

16, route de Gray

25030 BESANÇON CEDEX

FRANCE

[hufflen@lifc.univ-fcomte.fr](mailto:hufflen@lifc.univ-fcomte.fr)

<http://lifc.univ-fcomte.fr/~hufflen>

## Abstract

The first part of this talk about BIB<sub>TEX</sub> will focus on some difficult points related to syntax of bibliography files, e.g., the specification of person and organisation names. In addition, we show how some successors of BIB<sub>TEX</sub> — BIB<sub>TEX</sub>8, Bibulus, MIBIB<sub>TEX</sub> programs — improve them. In a second part, we explain how bibliography styles are built. Some demonstrations of the BIB<sub>TEX</sub> program are given as part of this talk, and some technical points could be made clearer by using some functions belonging to MIBIB<sub>TEX</sub>.

**Keywords** Bibliographies, bibliography styles, BIB<sub>TEX</sub>, MIBIB<sub>TEX</sub>.

## Streszczenie

W pierwszej części tej prezentacji o BIB<sub>TEX</sub>-u skoncentrujemy się na kilku trudnych elementach składni plików bibliograficznych, np., specyfikacji nazw osób i instytucji. Dodatkowo pokażemy, jak niektórzy następcy BIB<sub>TEX</sub>-a — programy BIB<sub>TEX</sub>8, Bibulus, MIBIB<sub>TEX</sub> — poprawiają sytuację. W drugiej części wyjaśnimy, jak należy budować style bibliograficzne. Zademonstrujemy kilka przykładów użycia BIB<sub>TEX</sub>-a i wyjaśnimy niektóre problemy techniczne przy użyciu funkcji należących do MIBIB<sub>TEX</sub>-a.

**Słowa kluczowe** Bibliografie, style bibliograficzne, BIB<sub>TEX</sub>, MIBIB<sub>TEX</sub>.

## Introduction

This talk is about BIB<sub>TEX</sub> [11], the bibliography processor usually associated with L<sup>A</sup>T<sub>E</sub>X. It extends Gabriela Grusza's, also given at the Bacho<sub>TEX</sub> 2006 conference, by going thoroughly into some particular and difficult points: role of braces within bibliographical entries (in .bib files), specification of person and organisation names, test of bibliography styles (.bst files). In addition, we show how some successors of BIB<sub>TEX</sub> — BIB<sub>TEX</sub>8 [10, § 13.1.1], Bibulus [17], MIBIB<sub>TEX</sub><sup>1</sup> [4] programs — improve them.

The second edition of *The L<sup>A</sup>T<sub>E</sub>X Companion* [10] includes a detailed chapter (Ch. 13) about the BIB<sub>TEX</sub> program, with some descriptions of tools that help in managing BIB<sub>TEX</sub> databases (§ 13.4). A less detailed description, but in Polish, can also be found in [1, § 6]. So, we only give the broad

outlines of our demonstration in this paper. Sometimes, we use MIBIB<sub>TEX</sub>'s functions in order to make clearer some features of BIB<sub>TEX</sub>: displaying bibliographical data as they are processed — by means of XML<sup>2</sup>-like syntax — and testing some parts of a bibliography style. MIBIB<sub>TEX</sub> is written in Scheme [8] and runs with at least three Scheme interpreters: MIT Scheme [3], bigloo [13], and PLT Scheme [2, 12]. The parts related to XML use SXML<sup>3</sup>, Oleg B. Kiselyov's implementation [9].

## Role of braces

In BIB<sub>TEX</sub>, braces serve two purposes. First, they can be used to surround the value associated with a field name, as shown in Fig. 1. In this case, these braces can be replaced by double quotes ('"'). The use of double quotes inside the value of a field is allowed only if they are surrounded by braces.

\* Title in Polish: *BIB<sub>TEX</sub>, MIBIB<sub>TEX</sub> i style bibliograficzne*.

<sup>1</sup> Multi-Lingual BIB<sub>TEX</sub>.

<sup>2</sup> eXtensible Markup Language.

<sup>3</sup> Scheme implementation of XML.

```
@INPROCEEDINGS{zemianski2002,
  AUTHOR = {Andrzej Zemian{\'}{n}{ski}},
  TITLE = {Waniliowe plantacje
           {Wroc{\l}awia}},
  BOOKTITLE = {Zajdel 2002},
  PAGES = {99-164},
  PUBLISHER = {Fabryka S{\l}\'}{o}{w},
  ADDRESS = {Lublin},
  YEAR = 2002}

@BOOK{bera1979,
  AUTHOR = {Paul B{\'}{e}{ra}},
  TITLE = {Ceux d'ailleurs},
  PUBLISHER = {Fleuve Noir},
  VOLUME = 900,
  SERIES = {Anticipation},
  YEAR = 1979}
```

(MIBIB $\TeX$  uses an additional field, LANGUAGE, unrecognised by BIB $\TeX$ .)

**Figure 1:** Examples of bibliographical entries.

Second, braces are used to group some characters, that is, to view them as an atomic value. This value does not obey case changes. For example, the word ‘*Wroclawia*’ in the title given in Fig. 1 is surrounded by braces, so it remains capitalised even if this title is to be put using lower-case characters. For example, such a specification:

```
TITLE = {The \LaTeX\ Companion}
```

would cause an error if a case change is applied. The correct way to specify such a title is:

```
TITLE = {The {\LaTeX} Companion}
```

As another example, let us consider the second entry of Fig. 1 and build a ‘References’ section using the alpha bibliography style, that is, references are labelled by the first three letters of the author’s last name. If we remove the braces surrounding the accented letter, the generated key for ‘Paul Béra’ will be B\’79, that is, an incorrect key. We have to write B{\’e}ra or B{\’{e}}ra in order for this key to be ‘[Bér79]’.

This ‘trick’ is less used with BIB $\TeX$ 8 since this program deals with 8-bit codes. So:

```
Paul Béra      Andrzej Zemian{ń}ski
```

can be put directly within files encoded w.r.t. Latin 1 and Latin 2). However, BIB $\TeX$ 8 does not deal with Unicode[14], it only knows some ASCII<sup>4</sup> extensions like Latin 1, Latin 2, etc. Besides, using several encodings within a same job is impossible.

<sup>4</sup> American Standard Code for Information Interchange.

MIBIB $\TeX$  partially solves this problem. When it parses a .bib file, the result can be viewed as an XML tree: we can display either the list implementing the tree w.r.t. SXML conventions, or a string according to XML syntax. We explain how to do that in Fig. 2, and the two kinds of results are given in Fig. 3 & 4 for the authors’ names of Fig. 1. It can be viewed that some  $\TeX$  commands for accents have been expanded. In fact, the functions belonging to MIBIB $\TeX$ ’s kernel could be used with Unicode strings, but as far as we know, the only encoding recognised by most Scheme interpreters is Latin 1. That is why ‘\’{e}’ is expanded, not ‘\’{n}’<sup>5</sup>. Nevertheless, we hope that this situation is temporary: Scheme’s future versions should deal with Unicode. In addition, the expansion of  $\TeX$  commands will be controlled by *pattern* specifications, as sketched in [5].

### Person and organisation names

When BIB $\TeX$  processes the value of an AUTHOR or EDITOR field, it divides a family name into four fields: *First* (for a first name), *von* (for a particle), *Last* (for a last name), *Junior* and recognizes these components regarding to the following possible syntaxes [11, § 4]:

- *First von Last*
- *von Last, First*
- *von Last, Junior, First*

As suggested by the cases used within this terminology, the words belonging to the *von* field are supposed to use only lowercase characters, whereas the words belonging to other fields are supposed to be capitalised. In addition:

- there is at least a *Last* part, so the last word of a part ending with the *Last* part belongs to this field, even it uses only lower-case characters;
- BIB $\TeX$  considers that the *First* and *von* parts are as big as possible. Examples:

```
– Henry Rider Haggard
   First      Last
– Jean de la Fontaine du Bois Joli
   First      von      Last
```

Using braces can force another splitting way:

```
{Federico} {Moreno Torroba}
   First      Last
```

MIBIB $\TeX$  provides a nicer way to specify the parts of a name using *keywords*, for example:

<sup>5</sup> In fact, Polish names are partially expanded: for example, the publisher’s name for the zemianski2002 entry would get ‘Fabryka S{\l}ów’ because ‘ó’ belongs to the Latin 1 encoding.

```

((log-output-p-pv 'screen)) ; All the messages are not put in the .log file, but displayed at
                             ; the screen.
((bibtexkey-alist-pv 'extend)) ; All the bibliographical entries are retained. This expression
                             ; is evaluated if the .tex file contains the command \cite{*}.

(define example-flag ; We use 'example-' as prefix, in order to avoid name conflicts
  (s-parse-bib-file "example.bib")) ; with the other definitions. As another example, the prefix
                                   ; used for the functions transforming .bib entries to SXML
                                   ; trees is 's-'. Several .bib files can be processed, in turn.

(define example-sxml-tree
  (and example-flag ; If parsing succeeds, it results in a non-false value. The value of this
    (s-get-sxml-mlbiblio-tree))) ; variable example-sxml-tree is the representation in SXML
                                ; of the contents of the example.bib file (see Figure 3).

(write example-sxml-tree) ; write can be replaced by a pretty-print procedure: pp in MIT
                          ; Scheme [3, § 14.5] or bigloo [13, § 4.2], pretty-print in PLT
                          ; Scheme [12, § 32].

(srl:sxml->xml example-sxml-tree) ; Returns the value of example-sxml-tree, as a string,
                                  ; according to an XML-like syntax.

((s-preambles-pv 'get)) ; Returns the concatenation of all the @preamble commands processed,
                        ; as a string.
((s-string-def-alist-pv 'get) example-string) ; If example-string's value — which must be
                                               ; a string — has been defined by a @string command, returns the
                                               ; corresponding value. Otherwise, returns #f, the false value.

((sxpath "//author") example-sxml-tree) ; Getting the values of all the AUTHOR fields. The
                                         ; argument of sxpath is an XPath [15] expression, the
                                         ; result is a list whose elements are the selected nodes.

(for-each pp ((sxpath "//author") example-sxml-tree)) ; Using a pretty-print function
                                                      ; (cf. infra) for each selected node.

(for-each (lambda (sxml-subtree) ; Displaying each selected node as a
  (display (srl:sxml->xml sxml-subtree))) ; string, according to an XML-like syntax.
  ((sxpath "//author") example-sxml-tree))

```

**Figure 2:** How to display and check the XML tree that result from parsing a .bib file.

first => Federico, last => Moreno Torroba  
 a mixed notation being allowed:

Bois Joli, first => Jean,  
 von => de la Fontaine du

that is, you can use a keyword for a part that has not been specified yet (you cannot define a part twice). Specific abbreviations of first names can be specified by the `abbr` keyword:

Henry Rider Haggard, `abbr => H. Rider`

In BIB<sub>T</sub>E<sub>X</sub>, the only way to specify an organisation name as author is to give only a *Last* part. MIBIB<sub>T</sub>E<sub>X</sub>'s keywords allow a nicer way to specify such a name and the key for sorting it:

org => Bacho\TeX, `sortingkey => BachoTeX`

Last but not at least, MIBIB<sub>T</sub>E<sub>X</sub> has two connectors for multiple names of authors or editors: 'and' between co-authors (or co-editors)—like in BIB<sub>T</sub>E<sub>X</sub>—and 'with' between collaborators. For example:

Frank Mittelbach and Michel Goossens with  
 Johannes Braams with David Carlisle with  
 Rowley, Chris A. with Christine Detig with  
 Joachim Schrod

—and see the bibliography of this article—the co-authors are to be put before other collaborators.

There exist several 'tricks' to check if the parts of a name are actually what a user wanted to express: an example is using the display functions of MIBIB<sub>T</sub>E<sub>X</sub>, as shown in Fig. 3 & 4.

```
((author
  (name
    (personname (first "Andrzej")
                 (last "Zemia{\'}{n}ski"))))
 (author
  (name (personname (first "Paul")
                    (last "Béra")))))
```

**Figure 3:** Displaying authors' names as SXML subtrees.

### Test of bibliography styles

The `bst` language, used within `BIBTEX` to design bibliography styles, is based on handling a stack. This is an old-fashioned language, not modular, that is why there is some attempt to replace it by a more modern one: `Perl`<sup>6</sup> within `Bibulus` [17], `nbst`<sup>7</sup> [4], close to `XSLT`<sup>8</sup> [16] in `MLIBTEX`. However, this last program provides a compatibility mode that allows to run `bst` programs (without the multilingual features of `MLIBTEX`, of course). In addition, we can take advantage of `Scheme` as an interactive interpreter to run such a program step by step, as shown in Fig. 5. So these `MLIBTEX` functions can help users learn this language.

### Organising the making of a ‘References’ section

In `BIBTEX`, the whole process of making a `.bbl` file containing references is controlled by the bibliography style. First, the `.aux` file is analysed and the declarations of the bibliography style:

```
ENTRY INTEGERS STRINGS FUNCTION MACRO
```

are read. Then the `.bib` files are searched (`READ` statement of `bst`) — so all the identifiers used within such files must be defined at this time: macros belonging to the bibliography style and defining month and journal names, or user-defined abbreviations introduced in `.bib` files by means of `@string` commands — and the bibliography style is applied, by means of the statements:

```
SORT EXECUTE ITERATE
```

The source `.tex` file is never read by `BIBTEX`.

Such a *modus operandi* would not be suitable for `MLIBTEX`. The languages used throughout a document depend on this document, due to the static behaviour of the multilingual packages of `LATEX 2ε`

<sup>6</sup> Practical Extraction Report Language.

<sup>7</sup> New Bibliography STyles.

<sup>8</sup> eXtensible Stylesheet Language Transformations.

```
("<author>
  <name>
    <personname>
      <first>Andrzej</first>
      <last>Zemia{\'}{n}ski</last>
    </personname>
  </name>
</author>"
"<author>
  <name>
    <personname>
      <first>Paul</first>
      <last>Béra</last>
    </personname>
  </name>
</author>")
```

**Figure 4:** Displaying authors' names using XML syntax.

[7] (babel [10, Ch. 9] or polski [1, App. F]). The process is:

- read the `.aux` file,
- search `.bib` files (undefined symbols are marked and must be defined later, when the bibliography style is applied): the result is an SXML tree,
- if the compatibility mode is used, read the ‘old’ bibliography style, proceed like ‘old’ `BIBTEX` and stop,
- otherwise, search the preamble of the source `.tex` file for the languages used throughout the document and the way (multilingual packages) to handle them,
- assemble the parts of the bibliography style [6],
- apply this bibliography style, analogously to an `XSLT` program.

### Acknowledgements

Many thanks to Jerzy B. Ludwichowski, who has written the Polish translation of the abstract.

### References

- [1] Antoni DILLER: *L<sup>A</sup>T<sub>E</sub>X wiersz po wierszu*. Wydawnictwo Helio, Gliwice. Polish translation of *L<sup>A</sup>T<sub>E</sub>X Line by Line* with an additional annex by Jan Jelowicki. 2001.
- [2] Matthew FLATT: *PLT MzScheme: Language Manual. Version 299.100*. March 2005. <http://download.plt-scheme.org/doc/299.100/mred.pdf>.
- [3] Chris HANSON, THE MIT SCHEME TEAM *et al.*: MIT *Scheme Reference Manual*, 1st edition.

```

((log-output-p-pv 'screen))      ; Cf. Fig. 2.
((bbl-output-p-pv 'screen))      ; Id.

((b-bst-stack-pv 'reset))        ; Resetting the stack.
(define current-sxml-tree ...)    ; The SXML tree resulting from parsing .bib files (see Fig. 2).
(define current-entry ...)       ; The SXML subtree for the entry being processed.

(b-process-statement 27 current-sxml-tree current-entry) ; This function uses a token of the
; bst language, the current SXML tree, and the entry being processed. A number is pushed into
(b-process-statement 4 current-sxml-tree current-entry) ; the stack.
(b-process-statement '+ current-sxml-tree current-entry) ; Example of an operation.
((b-bst-stack-pv 'display))      ; Should be a stack whose unique element is 31.

```

**Figure 5:** Using MIBIB<sub>T</sub>E<sub>X</sub> to run a bst program step by step.

- 
- March 2002. Massachusetts Institute of Technology.
- [4] Jean-Michel HUFFLEN: “MIBIB<sub>T</sub>E<sub>X</sub>’s Version 1.3”. *TUGboat*, Vol. 24, no. 2, pp. 249–262. July 2003.
- [5] Jean-Michel HUFFLEN: “MIBIB<sub>T</sub>E<sub>X</sub>: beyond L<sub>A</sub>T<sub>E</sub>X”. In: Apostolos SYROPOULOS, Karl BERRY, Yannis HARALAMBOUS, Baden HUGUES, Steven PETER and John PLAICE, eds., *International Conference on T<sub>E</sub>X, XML, and Digital Typography*, Vol. 3130 of LNCS, pp. 203–215. Springer, Xanthi, Greece. August 2004.
- [6] Jean-Michel HUFFLEN: “Making MIBIB<sub>T</sub>E<sub>X</sub> Fit for a Particular Language. Example of the Polish Language”. *Biuletyn GUST*, Vol. 21, pp. 14–26. 2004.
- [7] Jean-Michel HUFFLEN: *Managing Languages within MIBIB<sub>T</sub>E<sub>X</sub>*. To appear. June 2005.
- [8] Richard KELSEY, William D. CLINGER, Jonathan A. REES, Harold ABELSON, Norman I. ADAMS IV, David H. BARTLEY, Gary BROOKS, R. Kent DYBVIK, Daniel P. FRIEDMAN, Robert HALSTEAD, Chris HANSON, Christopher T. HAYNES, Eugene Edmund KOHLBECKER, JR, Donald OXLEY, Kent M. PITMAN, Guillermo J. ROZAS, Guy Lewis STEELE, JR, Gerald Jay SUSSMAN and Mitchell WAND: “Revised<sup>5</sup> Report on the Algorithmic Language Scheme”. *HOSC*, Vol. 11, no. 1, pp. 7–105. August 1998.
- [9] Oleg E. KISELYOV: *XML and Scheme*. September 2005. <http://okmij.org/ftp/Scheme/xml.html>.
- [10] Frank MITTELBACH and Michel GOOSSENS, with Joannes BRAAMS, David CARLISLE, Chris A. ROWLEY, Christine DETIG and Joachim SCHROD: *The L<sub>A</sub>T<sub>E</sub>X Companion*. 2nd edition. Addison-Wesley Publishing Company, Reading, Massachusetts. August 2004.
- [11] Oren PATASHNIK: *BIB<sub>T</sub>E<sub>X</sub>ing*. February 1988. Part of BIB<sub>T</sub>E<sub>X</sub>’s distribution.
- [12] PLT: *PLT MzLib: Libraries Manual. Version 299.100*. March 2005. <http://download.plt-scheme.org/doc/299.100/mzlib.pdf>.
- [13] Manuel SERRANO: *Bigloo. A Practical Scheme Compiler. User Manual for Version 2.6c*. June 2004.
- [14] THE UNICODE CONSORTIUM: *The Unicode Standard Version 4.0*. Addison-Wesley. August 2003.
- [15] W3C: *XML Path Language (XPath). Version 1.0*. W3C Recommendation. Edited by James Clark and Steve DeRose. November 1999. <http://www.w3.org/TR/1999/REC-xpath-19991116>.
- [16] W3C: *XSL Transformations (XSLT). Version 1.0*. W3C Recommendation. Edited by James Clark. November 1999. <http://www.w3.org/TR/1999/REC-xslt-19991116>.
- [17] Thomas WIDMAN: “Bibulus—a Perl XML Replacement for BIB<sub>T</sub>E<sub>X</sub>”. In: *EuroT<sub>E</sub>X 2003*, pp. 137–141. ENSTB. June 2003.