BibTeX, mlBibTeX and Bibliography Styles

Jean-Michel Hufflen

BibTeX, MlBibTeX 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
http://lifc.univ-fcomte.fr/~hufflen

Abstract

The first part of this talk about BibTeX 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 BibTeX — BibTeX8, Bibulus, MlBibTeX programs — improve them. In a second part, we explain how bibliography styles are built. Some demonstrations of the BibTeX program are given as part of this talk, and some technical points could be made clearer by using some functions belonging to MlBibTeX.
Keywords Bibliographies, bibliography styles, BibTeX, MlBibTeX.

Streszczenie

W pierwszej części tej prezentacji o BibTeX-u skoncentrujemy się na kilku trudnych elementach składni plików bibliograficznych, np., specyfikacji nazw osób i instytucji. Dodatkowo pokażemy, jak niektórzy następcy BibTeX-a — programy BibTeX8, Bibulus, MlBibTeX — poprawiają sytuację. W drugiej części wyjaśnimy, jak należy budować style bibliograficzne. Zademonstrujemy kilka przykładów użycia BibTeX-a i wyjaśnimy niektóre problemy techniczne przy użyciu funkcji należących do MlBibTeX-a.
Słowa kluczowe Bibliografie, style bibliograficzne, BibTeX, MlBibTeX.

Introduction

This talk is about BibTeX [11], the bibliography processor usually associated with LaTeX. It extends Gabriela Grusza's, also given at the BachoTeX 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 BibTeX — BibTeX8 [10, § 13.1.1], Bibulus [17], MlBibTeX[1] [4] programs — improve them.

The second edition of *The LaTeX Companion* [10] includes a detailed chapter (Ch. 13) about the BibTeX program, with some descriptions of tools that help in managing BibTeX 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 MlBibTeX's functions in order to make clearer some features of BibTeX: displaying bibliographical data as they are processed — by means of XML[2]-like syntax — and testing some parts of a bibliography style. MlBibTeX 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[3], Oleg B. Kiselyov's implementation [9].

Role of braces

In BibTeX, 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: *BibTeX, MlBibTeX i style bibliograficzne*.

[1] **M**ulti-**L**ingual BibTeX.

[2] e**X**tensible **M**arkup **L**anguage.

[3] Scheme implementation of XML.

```
@INPROCEEDINGS{zemianski2002,
        AUTHOR = {Andrzej Zemia{\'{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}
```

(MlBibTeX uses an additional field, `LANGUAGE`, unrecognised by BibTeX.)

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 '*Wrocławia*' 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 BibTeX8 since this program deals with 8-bit codes. So:

```
Paul Béra      Andrzej Zemiański
```

can be put directly within files encoded w.r.t. Latin 1 and Latin 2). However, BibTeX8 does not deal with Unicode[14], it only knows some ASCII[4] extensions like Latin 1, Latin 2, etc. Besides, using several encodings within a same job is impossible.

[4] **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange.

MlBibTeX 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 MlBibTeX'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}`'[5]. 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 BibTeX 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;
- BibTeX considers that the *First* and *von* parts are as big as possible. Examples:
  - `Henry Rider` (First) `Haggard` (Last)
  - `Jean` (First) `de la Fontaine du` (von) `Bois Joli` (Last)

Using braces can force another splitting way:

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

MlBibTeX provides a nicer way to specify the parts of a name using *keywords*, for example:

[5] 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 duisplay 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 BibTeX, the only way to specify an organisation name as author is to give only a *Last* part. MlBibTeX'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, MlBibTeX has two connectors for multiple names of authors or editors: '`and`' between co-authors (or co-editors) — like in BibTeX — 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 MlBibTeX, 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.

```
("<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.

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[6] within Bibulus [17], nbst[7] [4], close to XSLT[8] [16] in MlBibTeX. However, this last program provides a compatibility mode that allows to run bst programs (without the multilingual features of MlBibTeX, 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 MlBibTeX 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 MlBibTeX. The languages used troughout a document depend on this document, due to the static behaviour of the multilingual packages of LaTeX 2ε [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.

[6] **P**ractical **E**xtraction **R**eport **L**anguage.
[7] **N**ew **B**ibliography **ST**yles.
[8] e**X**tensible **S**tylesheet **L**anguage **T**ransformations.

Acknowledgements

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

References

[1] Antoni Diller: *LaTeX wiersz po wierszu*. Wydawnictwo Helio, Gliwice. Polish translation of *LaTeX 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 MlBibTeX to run a bst program step by step.

March 2002. Massachusetts Institute of Technology.

[4] Jean-Michel Hufflen: "MlBibTeX's Version 1.3". TUGboat, Vol. 24, no. 2, pp. 249–262. July 2003.

[5] Jean-Michel Hufflen: "MlBibTeX: beyond LaTeX". In: Apostolos Syropoulos, Karl Berry, Yannis Haralambous, Baden Hugues, Steven Peter and John Plaice, eds., *International Conference on TeX, XML, and Digital Typography*, Vol. 3130 of LNCS, pp. 203–215. Springer, Xanthi, Greece. August 2004.

[6] Jean-Michel Hufflen: "Making MlBibTeX 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 MlBibTeX*. 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 Dybvig, 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^5 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 LaTeX Companion*. 2nd edition. Addison-Wesley Publishing Company, Reading, Massachusetts. August 2004.

[11] Oren Patashnik: *BibTeXing*. February 1988. Part of BibTeX'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 BibTeX". In: *EuroTeX 2003*, pp. 137–141. ENSTB. June 2003.