

The Bib_TE_X macro package for WinEdt
(aka. bibMacros)

<http://www.winedt.org/Config/menus/00bibMacros.pdf>

R Schlicht
w.m.l@gmx.net

Revision 2.1 — 7th June 2007

Contents

Contents	1
1 Overview	2
2 Installation	3
3 Commands	4
3.1 On Fields	4
3.2 On Entries	5
3.3 On a Bibliography	5
3.4 Help	8
4 Options	9
4.1 What is an Entry?	9
4.2 Formatting Entries	10
<i>Remove Empty Fields? ◇ Remove Empty Lines? ◇ The Case ◇ Alignment ◇</i> <i>Wrapping Lines ◇ Delimiters ◇ Comma after Last Field?</i>	
4.3 Generating a Key	13
<i>Parts which constitute the Key ◇ Fine-Tuning ◇ Extra Label ◇ Conditional Rules ◇</i> <i>Always generate a Key? ◇ Prompt the Key?</i>	
4.4 Configuration and Resource Files	18
<i>Local Configurations ◇ Bibliography Templates ◇ Bibliography Styles ◇ String File</i>	
A Todo & Bugs	22
B History	23

1 Overview

This set of macros assists you in editing and managing Bib_TE_X files in WinEdt. It comes with two interfaces: Either a Popup Menu or a Main Menu. Both contain the following commands:

Command	Description	Shortcut
New Field	Create a new field	N
Delete Field	Delete the current field (and copy it to the Clipboard)	D
Delete Field Content	Delete the content of the current field	L
Strip Delimiters	Remove delimiters	T
Strip Inner Braces	Remove braces inside a field	P
Expand String	Complete the current string (as defined in a @STRING entry)	S
Complete Field	Complete the field content from the current file	F
New Bibliography Entry	Submenu ▷ Insert a new Bib _T E _X entry	B
Delete Bibliography Entry	Delete the complete entry (and copy it to the Clipboard)	E
Generate Key	Suggest a key for the current entry and insert it (overwriting any existing key)	K
Clean Entry	Remove empty fields, align the lines and call Generate Key, if the key is empty	C
Clean Bibliography	Do this for the whole bibliography or for the current selection	G
Sort Bibliography	Sort a bibliography by two definable criteria	O
Extract From Aux	Extract all \cite'd (or \nocite'd) entries into a new bibliography file	A
Extract Annotations	Extract 'annotate' fields into separate files	X
Merge Annotations	Merge annotation files into a single database	M
Find Entry	Find all entries containing a specified string (possibly a Regular Expression)	Y
Info for current Field	What's this field supposed to contain?	I
Bib _T E _X Documentation	Open Bib _T E _X manual	U
bibMacros Manual	Open bibMacros manual	R
Open cfg file	Open the configuration file	<i>⟨space⟩</i>

2 Installation

To install the package, unzip the file `bibMacros.zip` in `%b\Macros\LaTeX` (where `%b` is WinEdt's local directory, e.g. `C:\Documents and Settings\{user}\Application Data\WinEdt Team\WinEdt`), and execute the macro `_bibMacros_Install.edt` (by choosing **Macros | Execute Macro...**). That's all!

The following configuration and resource files may be edited by you:

<code>bibMacros.cfg</code>	(main configuration file)
<code>bibMacrosFields.cfg</code>	(entry definitions)
<code>bibMacrosFields_apacite.cfg</code>	(entry definitions for the apacite package)
<code>bibMacrosFields_biblatex.cfg</code>	(entry definitions for the biblatex package)
<code>bibMacrosFields_jurabib.cfg</code>	(entry definitions for the jurabib package)
<code>BMStrings.bib</code>	(@STRING definitions)

A detailed survey of all options is given in the section “**Options**”.

If you have been using WinEdt's default popup menu “**BibTeX Items**” before, you must disable it in order to use the `bibMacros` popup menu, because they are sharing the same shortcut (`ALT-B`). You don't need the old menu anymore, since now you only have to press `ALT-B` twice to get a similar – actually better – version of the “**BibTeX Items**” menu. Of course you could also choose a different shortcut.

Should you want to uninstall the package one day, simply remove the **Menus** (in **Options | Menu Setup ...**) and delete the files to eliminate all traces.

The `bibMacros` package was written for WinEdt 5.3 or later and should not be run with a previous version.

Unless you are using WinEdt 5.5, which provides an internal MUI, it is strongly recommended that you also install the external MUI plug-in, which is available from <http://www.winedt.org/Plugins/mui.php>. If it is installed, it will be used by the commands **Sort Bibliography** and **Find Entry**.

The latest version of the `bibMacros` package can always be found at:
<http://www.winedt.org/Config/menus/BibTeX.php>

3 Commands

Terminology:

- A “bibliography entry” is everything inside @...{.....}
- The “type” is the string between @...{ (e. g. “BOOK”)
- A “field” is a pair of field “name” and field “value” (e. g.: author = {...})
- The “key” is the assigned string for an entry, i. e. what’s inside the \cite{...} command

You can invoke all commands from anywhere within the current field resp. the current entry.

3.1 On Fields

New Field will present you a list of fields to be added to the current entry. The new field will be inserted after the current one (or in the current line if the line is empty). You will only be shown fields which are (1) not already in the current entry and (2) are allowed in that entry.

→ customizable in `bibMacrosFields.cfg` resp. `bibMacrosFields_<bibStyle>.cfg`

→ see sections “[Bibliography Templates](#)” and “[Bibliography Styles](#)”

Delete Field will delete the complete field (name and value) and copy it to the clipboard (should you want to insert it somewhere else).

Delete Field Content will delete the value of a field, format it according to your preferences and (should you want that) change the delimiters.

→ see section “[Delimiters](#)”

Strip Delimiters will remove delimiters from the current field value (e. g. for the ‘year’ field or for fields which include @STRINGS).

Strip Inner Braces will remove braces inside the current field (beginning with brace-level one, i. e. the outermost braces).

Expand String will search in the current file for a @STRING definition and replace the abbreviation with the complete string. You can also specify additional files which the macro should search for expansions.

→ see section “[String File](#)”

Complete Field will search backwards for contents of the current field and snatch it from there. So that, e. g. if you want to add a few books by the same author to your database, you don’t have to retype the name.

3.2 On Entries

New Bibliography Entry will insert a new entry template after the current entry. If you choose “Generic”, all fields of the current bibliography style will be included in the entry template. “Other” asks you for the name of an entry, which has to be defined in `bibMacrosFields.cfg`.

→ customizable in `bibMacrosFields.cfg` resp. `bibMacrosFields_<bibStyle>.cfg`

→ see sections “Bibliography Templates” and “Bibliography Styles”

Delete Bibliography Entry will delete the current entry and copy it to the clipboard (should you want to insert it somewhere else).

Generate Key will suggest a key for the current entry and insert it (replacing any existing key).

→ for customization options see (the long) section “Generating a Key”

Clean Entry will remove empty fields in the current entry, align the lines according to your preferences, check for missing commas, change the delimiter types if you wish, and call **Generate Key**, if the key is empty.

E. g., if you invoked **Clean Entry** on the following entry (which, I admit, is made up to look extremely horrible):

```
@MASTERSTHESIS {*,
  author      = {{{\{E}}douard Masterly},
  title       = {Mastering Thesis Writing},
  school      = "Stanford
               University"
  year        = "1988",
  note        =   "*" }
```

you would get this result:

```
@MASTERSTHESIS{Masterly-MastThesWrit:88,
  author      = {{{\{E}}douard Masterly},
  title       = {Mastering Thesis Writing},
  school      = {Stanford University},
  year        = 1988
}
```

→ for customization options see section “Formatting Entries”

3.3 On a Bibliography

Clean Bibliography will clean all selected entries or, if nothing’s selected, all entries in the current file.

Sort Bibliography will sort the complete bibliography or the entries in the current selection.

New (2.0)!

With WinEdt 5.5 and its new internal MUI, you will be shown an advanced dialog (see figure 3.1). For older WinEdt versions, you should get the external MUI plug-in (available from <http://www.winedt.org/Plugins/mui.php>), which will show a similar dialog.

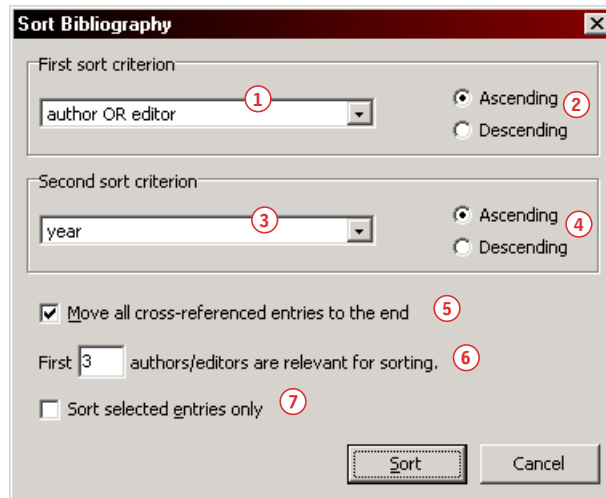


Figure 3.1:
Sort Bibliography dialog

You can specify two sort criteria (① and ③). These can be field names, like `title`, `year` etc., possibly a Regular Expression (like `author|editor`). Furthermore, there are two special keys: `KEY` (the cite key) and `TYPE` (the type of entry).

You can set the sorting order for each criterion (② and ④).

Additionally, all cross-referenced entries can be moved to the end of the file (⑤), which is required by Bib \TeX .

If the bibliography is to be sorted by author or editor, you can set the maximum number of names relevant for sorting (⑥).

Finally, you may also select part of a bibliography file, and sort this part only (⑦). Thus, you could even sort by more than two criteria. This option is only shown if something is actually selected.

Extract From Aux will extract all `\cite'd` (or `\nocite'd`) entries into a new file. It has to be invoked from the main \TeX file, and the `.aux` file must exist.

→ further documentation can be found in the macro file `bibAuxExtract.edt`.

Extract Annotations extracts all `'annotate'` or `'annotation'` fields into separate files.

→ further documentation can be found in the macro file `bibAnnoteExtract.edt`.

Merge Annotations merges annotation files into a single database. The `'annotate'` resp. `'annotation'` field is used by some bibliography styles. If you've never heard of it, these macros are probably useless for you. The path to which or from which annotation files are extracted resp. merged, can be set in `bibMacros.cfg` or at the end of the bibliography file itself.

→ further documentation can be found in the macro file `bibAnnoteMerge.edt`.

The last two macros can be invoked either from the main \TeX file, in which case they will work on all `\bibliography's`, or from a Bib \TeX file.

New (2.1)!

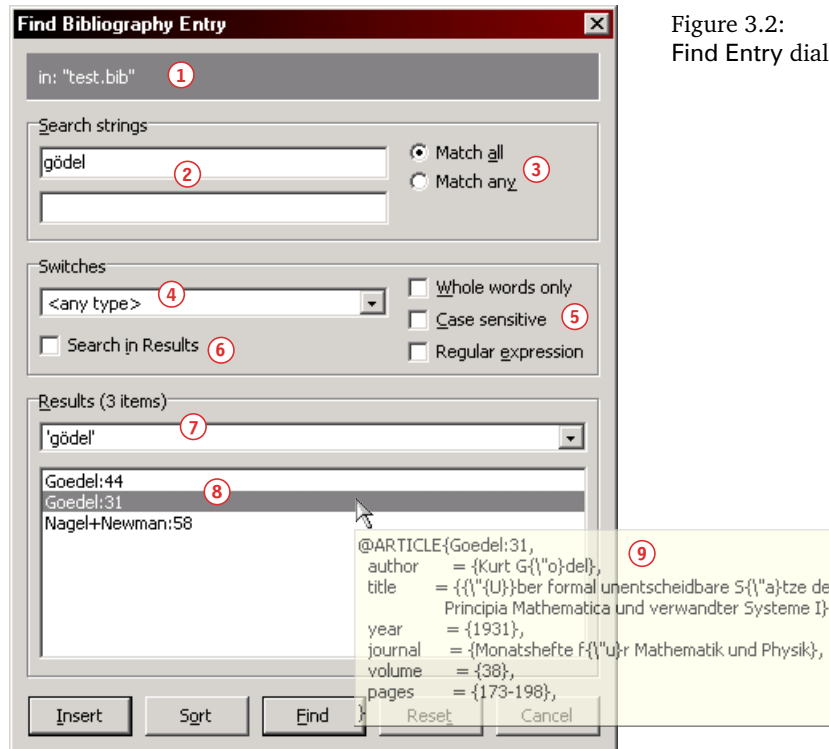


Figure 3.2:
Find Entry dialog

Find Entry will present you a list of entries and insert the key or the entry itself.

New (2.0)!

When using WinEdt 5.5 with its new (internal) MUI, you'll see the advanced dialog above (figure 3.2); for previous WinEdt versions you need the external MUI plug-in, which will show a similar but less powerful dialog.

If you invoke the command from your main \TeX file, all appropriate bibliographies will be searched (you have to have \LaTeX ed the file once); if you invoke it from a Bib \TeX file, only this file will be searched (1).

You can specify one or two search strings (2), and choose whether both or either of them have to be matched (3). You also have the possibility to restrict the search to an entry type (4). Additionally, you may specify whether you only want to search for whole words, whether search should be case sensitive and whether the search string is a regular expression (5). Searching for an empty string will show all entries of the bibliography. Note that international characters do not pose a problem even if they are spelled out in \LaTeX 's notation in the bibliography entry ('Gödel').

Search may also be confined to the results of a previous query (6), which are stored for each bibliography file and may be selected from a drop-down list (7). The results can also be sorted alphabetically.

If you select an item from the results list (8), a tool tip will pop up showing the complete entry (9), so that you can verify that it's the one you are looking for. To insert the key at the current point, double-click on it, click on Insert or simply press Enter. If you right-click on a selected key, the complete entry will be inserted.

3.4 Help

Info for Current Field will prompt you with a short description of the purpose of the current field.

→ customizable in `bibMacrosFields.cfg` resp. `bibMacrosFields_<bibStyle>.cfg`

→ see sections “[Bibliography Templates](#)” and “[Bibliography Styles](#)”

BibTeX Documentation will open the file `<TEXMF>\doc\bibtex\bttdoc.dvi` in the DVI viewer.

bibMacros Manual will open the manual (this file).

Open cfg file will open the configuration file `bibMacros.cfg`.

4 Options

The following presents a detailed survey of all options for the bibMacros package. These options can be changed in the file bibMacros.cfg or locally to a bibliography file (see section “[Local Configurations](#)”).

You don’t have to read everything or even learn it by heart. The package comes with somewhat sensible defaults, so that you can start using it right away. If you have upgraded the bibMacros package, you should at least read the section “[History](#)” to check whether any options have changed.

4.1 What is an Entry?

By default, bibliography entries (we’re only talking about the ‘@’ here) have to start at the beginning of the line. All other entries will be ignored. (This is also the default for WinEdt’s gather interface, and it’s sensible because you can differentiate between the following cases).

Example:

```
@BOOK{...,  
  ...  
}
```

will be found, while

```
  @STRING{...}
```

won’t be.

Admissible options:

```
Assign("bib_beginLine", "<"); // default  
Assign("bib_beginLine", "");
```

Bib_T_E_X is case-insensitive with respect to the cite key. I. e. it allows you to `\cite{yourKey}` in the _T_E_X file, while the entry is defined as, say, a `@BOOK{YOURKEY}` in the Bib_T_E_X database.

If you find the Extract From Aux macro too slow, *and you are sure that you never use different letter-cases in the cite keys*, you can disable case-insensitivity.

Admissible options:

```
Assign("bib_keyCS", "0"); // case-insensitive (default)  
Assign("bib_keyCS", "1"); // case-sensitive (not recommended)
```

4.2 Formatting Entries

relevant for: [New Field](#) · [Delete Field Content](#) · [New Entry](#) · [Generate Key](#) · [Clean Entry](#)

4.2.1 Remove Empty Fields?

By default, empty fields are removed when cleaning an entry:

Admissible options:

```
Assign("bib_emptyFields","remove"); // remove empty fields (default)
Assign("bib_emptyFields","leave"); // leave empty fields
```

4.2.2 Remove Empty Lines?

Empty lines inside an entry can also be removed when cleaning it, if you wish.

Admissible options:

```
Assign("bib_emptyLines","remove"); // remove empty lines (default)
Assign("bib_emptyLines","leave"); // leave empty lines
```

4.2.3 The Case

Case of the Type

Determine the case of the entry type (e. g. 'BOOK', 'ARTICLE', etc.): '@B00K', '@Book' or '@book'.

If you leave the option empty, the case will not be changed, and for new entries, it will be in upper case.

Admissible options:

```
Assign("bib_typeCase","U"); // upper case (default)
Assign("bib_typeCase","L"); // lower case
Assign("bib_typeCase","I"); // initial capital
Assign("bib_typeCase",""); // don't change anything
```

Case of the Field Names

Determine the case of the field names (e. g. 'author', 'title', etc.): 'AUTHOR', 'Author' or 'author'.

(This does not have any effect on newly created entry templates, since here, required fields will appear in upper case, optional fields in lower case.)

Admissible options:

```
Assign("bib_fieldCase","U"); // upper case
Assign("bib_fieldCase","L"); // lower case (default)
Assign("bib_fieldCase","I"); // initial capital
Assign("bib_fieldCase",""); // don't change
```

4.2.4 Alignment

Alignment of the Fields

You can specify three numbers:

- the column to which the field *names* are aligned,
- the column to which the field *values* are aligned (which should be at least as long as the longest field name plus the spaces before and two afterwards), and
- the column to which *continued lines* are aligned (relative to the column number of the field value).

Example:

```
author      = {\'\{E\}douard Masterly},
title       = {Mastering Thesis
              Writing},
```

will be achieved by:

```
Assign("bib_alignName","2");
Assign("bib_alignValue","17");
Assign("bib_alignContValue","3"); // negative values also allowed
```

Alignment of the equal sign

The equal sign “=” can either be aligned to the beginning of the field *value* or to the end of the field *name*.

Example:

```
author      = {...}
publisher   = {...}
```

or:

```
author =      {...}
publisher =   {...}
```

Admissible options:

```
Assign("bib_alignEqual","value"); // align to value (default)
Assign("bib_alignEqual","name");  // align to name
```

Alignment of the Key

By default, the key will follow immediately after the type of the entry. But you can also have it aligned relative to the field value. (Note that the opening brace will be aligned, not the key itself, because otherwise you'd get unwanted spaces in front of the items in WinEdt's Gather Interface.)

Example:

```
@ARTICLE{Sim94,
  author   = {A. Simoni\v{c}},
  ...
}
```

or:

```
@ARTICLE {Sim94,
  author   = {A. Simoni\v{c}},
  ...
}
```

Admissible options:

```
Assign("bib_alignKey","no"); // don't align the key (default)
Assign("bib_alignKey","0"); // align key to the column of bib_alignValue
Assign("bib_alignKey","2"); // align key to bib_alignValue+2
Assign("bib_alignKey","-3"); // align key to value-3 (the above example)
```

etc.

4.2.5 Wrapping Lines

Lines can be wrapped to a specific length. This will also unwrap lines, i. e. line breaks will be removed if the field contents fit in one line.

Admissible options:

```
Assign("bib_wrap","no"); // don't wrap lines (default)
Assign("bib_wrap","90"); // wrap lines at column 90
```

etc.

4.2.6 Delimiters

Braces or Quotes

If you clean an entry or if you delete a field value, field delimiters can automatically be changed, so that all your fields are either surrounded by "quotes" or by {braces}. (Of course only those fields which have been enclosed in delimiters originally).

This option also tells the bibMacros what to use to surround new fields.

Admissible options:

```
Assign("bib_fieldDelim","braces"); // change delimiters to braces (default)
Assign("bib_fieldDelim","quotes"); // change delimiters to quotation marks
Assign("bib_fieldDelim","leave"); // leave the delimiters in peace
// (and put new fields in {braces})
```

Numerical Fields

Furthermore, it is possible to either automatically strip delimiters from purely numerical fields (e.g. ‘year’), to always add them, or to leave them as they are (which can of course lead to inconsistent bracing):

Admissible options:

```
Assign("bib_numericFieldDelim","remove"); // remove delimiters from numerical fields
Assign("bib_numericFieldDelim","add"); // add delimiters, if not already there
Assign("bib_numericFieldDelim","leave"); // leave delimiters
```

4.2.7 Comma after Last Field?

When cleaning an entry you can have the (optional) comma after the last field automatically removed. Also, new entry templates can be created without the last comma. (You don’t have to worry about forgetting to add the comma when appending another field afterwards, since both creating a new field and cleaning the entry will check whether any commas are missing.)

Admissible options:

```
Assign("bib_lastComma","yes"); // comma after last field (default)
Assign("bib_lastComma","no"); // no comma after last field
```

4.3 Generating a Key

relevant for: [Generate Key](#) · [Clean Entry](#)

If you’re tired of making up keys for your bibliography entries; if you want them to be consistently constructed; or if you aren’t sure whether a key already exists, you can let the `bibMacros` package automatically create the key for you.

NOTE: The macros rely on the “Gather Interface” when checking whether a key already exists. Now, in WinEdt’s default setup, all open bibliography files will be searched for entries, unless the tree is built (in which case only the bibliography files in the tree will searched). So, if you have another bibliography file opened, keys from that file will also be considered as unavailable for the current entry. If you don’t want that, either close all unrelated bibliography files, or else build the tree.

4.3.1 Parts which constitute the Key

You have to specify which fields shall be used to construct a key by putting them in {curly braces}. This can also be a WinEdt Regular Expression (e.g. a disjunction like {author|editor}, or something like {@{book}title}). Everything outside braces will be taken verbatim.

Examples:

```
Assign("bib_key","{author|editor}-{title}:{year}"); // default
Assign("bib_key","myBib-{author|editor}:{year}");
Assign("bib_key","{title}-{crossref|journal}({volume}.{number}:p{pages})");
```

You can take any field name that can appear in a bibliography entry. If a field does not appear in the entry, the verbatim part *before* the respective field will be skipped. E.g. if the entry does not

contain a title, your key will look like this: 'Masterly:88' instead of 'Masterly-MastThesWrit:88'. The '-' has been skipped.

4.3.2 Fine-Tuning

For each field used for key generation you can define ten options, which are to be constructed following the syntax:

```
bib_{<field(s)>}_{<option>}
```

where *<field(s)>* must be part of the definition in `bib_key`, and *<option>* is one of the following (default values shown on the right):

nrwords	3
case	<i><empty></i>
etc	<i><empty></i>
skip	<i><empty></i>
alt	<i><empty></i>
delim	<i><empty></i>
firstletter	"\$Alpha+Numeric\$", i.e. <i><a letter or number></i>
letters	"+\$Alpha+Numeric\$", i.e. <i><letters or numbers></i>
minletters	3
extractletters	4

Here are examples for each option:

`nrwords` The maximum number of words from one field that should be put in the key.

```
Assign("bib_{author|editor}_nrwords", "2");
```

would only use the first two names to construct the key.

`case` The case of the words.

```
Assign("bib_{title}_case", "I");
```

would capitalize all words of the title. Further options are: U for uppercase, L for lowercase, and an empty string to not change capitalization.

`etc` A string to append to represent words omitted in the construction of the key.

```
Assign("bib_{author|editor}_etc", "ETAL");
Assign("bib_{title}_etc", "...");
```

`alt` A string to be used if the field is missing in the entry.

```
Assign("bib_{year}_alt", "ND");
```

`skip` A (WinEdt RegEx) string which effects a premature end of searching in the field.

```
Assign("bib_{title}_skip", "[.:]");
```

would skip the part after a '.' or a ':' (because it's the subtitle).

`delim` A string to put between words of one field.

```
Assign("bib_{title}_delim", "-");
```

would put a '-' between all (parts of) words taken from the title field.

A word will only be included in the key, if it matches the following three criteria:

`firstletter` The first letter (WinEdt RegEx).

```
Assign("bib_{title}_firstletter", "$Upper$");
```

Words have to start with a capital letter.

```
Assign("bib_{title}_firstletter", "?");
```

Anything can be the first letter.

`letters` All other letters (WinEdt RegEx).

```
Assign("bib_{title}_letters", "@{$Alpha$|-}");
```

The hyphen is part of the word.

```
Assign("bib_{title}_letter", "*");
```

Anything can be a letter.

Note: You don't have to specify special characters like '{', '}', '\', etc., since they will be removed anyway (as they are not allowed in Bib_T_E_X labels).

`minletters` Minimum number of letters.

```
Assign("bib_{title}_minletters", "4");
```

Skip words like 'and' and 'in' in a title, for instance.

If a word matches the above criteria, you can finally define:

`extractletters` How many characters should be extracted from the word.

```
Assign("bib_{title}_extractletters", "5");
```

would extract the first 5 letters (maximum), while a negative number:

```
Assign("bib_{year}_extractletters", "-2");
```

would extract the last 2 letters (in this case digits). This number is independent from the minimum number of letters, i. e. the value can be bigger or smaller (or equal).

Resume

You can customize the above ten options for each field type that is part of your definition for key generation. For instance if your definition was:

```
Assign("bib_key", "{author|editor}:{year}");
```

you would have the possibility to define the following options:

<code>bib_{author editor}_nrwords</code>	<code>bib_{year}_nrwords</code>
<code>bib_{author editor}_case</code>	<code>bib_{year}_case</code>
<code>bib_{author editor}_etc</code>	<code>bib_{year}_etc</code>
<code>bib_{author editor}_skip</code>	<code>bib_{year}_skip</code>
<code>bib_{author editor}_alt</code>	<code>bib_{year}_alt</code>
<code>bib_{author editor}_delim</code>	<code>bib_{year}_delim</code>
<code>bib_{author editor}_firstletter</code>	<code>bib_{year}_firstletter</code>
<code>bib_{author editor}_letters</code>	<code>bib_{year}_letters</code>
<code>bib_{author editor}_minletters</code>	<code>bib_{year}_minletters</code>
<code>bib_{author editor}_extractletters</code>	<code>bib_{year}_extractletters</code>

If you don't define all (or any) of the above options, defaults will be taken. So, if you find all this confusing, your best bet might be to define only the `bib_key` option, and have a look at how the default fine-tuning options look like. You then only need to change those options which you don't like.

→ for an example see the file `BMStrings.bib`

4.3.3 Extra Label

If the key already exists, an extra label can be appended automatically. E. g., if you already have a database entry 'Abel:1997', the next entry from the same author in the same year could be 'Abel:1997a'. (Note: This will be done automatically, i. e. you will not even be prompted the extra label. Instead, WinEdt will gently beep.)

The extra labels can either be alphabetical or numerical.

Labels will either only be appended if necessary, i. e. if the key 'Abel:1997' already exists; or you can have them always added, so that the first key will be e. g. 'Abel:1997a', regardless whether there is another entry from Abel in 1997.

In any case, if there is a key with a first extra label ('a' or '1') in your database, the macro will not insert a key without a label, but will continue with 'b' resp. '2' etc.

Admissible options:

```
Assign("bib_extraLabel","no"); // no extra label
Assign("bib_extraLabel","{a}"); // alphabetical labels will be appended, if necessary
Assign("bib_extraLabel","{1}"); // numerical labels will be appended, if necessary
Assign("bib_extraLabel","+a"); // alphabetical labels will always be appended
Assign("bib_extraLabel","+1"); // numerical labels will always be appended
```

Everything outside curly braces will be taken verbatim, so that your extra label can also consist of additional characters. E. g.:

```
Assign("bib_extraLabel","({a})"); // would produce a key like 'Abel:1997(b) '
Assign("bib_extraLabel","-+1"); // would produce a key like 'Abel:1997-2'
```

etc.

4.3.4 Conditional Rules

You can also specify conditional rules for the key generation. E. g., you can have different sets of rules for different types of entries or a specific rule if a certain field is part of the entry. These sets of rules follow the syntax:

`[<condition>=<rule>]`

where `<condition>` can be a field name or an entry type and `<rule>` is a key generation rule as described above. Conditions can also be made up of disjunctions ('|' = OR). The conditional rules will be applied if and only if no other rule has been successful in constructing a key. If the condition is left empty, the rule will always be applied, if no other rule has been successful in constructing a key.

If there is a last rule not in square brackets, it will always be executed. That's why you can still use the above examples or your old rules. The conditional rules are just an extension (almost: you do

have to escape the characters '[' and ']' with '\ now, if you want them literally in the key). Note, that the order of the rules is important.

Here are some examples:

- `Assign("bib_key", "[key={key}][{author|editor}{year}]");`

means that

1. if the entry includes a 'key' field, this field will be taken;
2. if the key (the cite key) is still empty, 'author' or 'editor' and the 'year' fields will be taken.

- `Assign("bib_key", "[editor={editor}(ed.)][author={author}]{year}");`

means that

1. if there's an 'editor' field, take it (appending '(ed.)');
2. else, if there's an 'author' field, take it;
3. regardless, what the key already consists of, the year will be appended.

- `Assign("bib_key", "[ARTICLE={journal}-{volume}-{number}]>[INCOLLECTION|INPROCEEDINGS={author}{booktitle}{pages}]>[{author|editor}{title}]");`

means that

1. if the entry is an @ARTICLE, take 'journal', 'volume' and 'number';
2. if the entry is of type @INCOLLECTION or @INPROCEEDINGS, take the 'author', 'booktitle' and 'pages' fields;
3. else, if none of the above matched, take 'author' or 'editor' and 'title'.

4.3.5 Always generate a Key?

You can always have a key generated when cleaning an entry or the whole bibliography, that is even if a key already exists. This option is probably only useful when you want to import bibliography databases.

Admissible options:

```
Assign("bib_keyAlways", "yes");
Assign("bib_keyAlways", "no"); // Default
```

4.3.6 Prompt the Key?

You can either be prompted the key before it is inserted or have it selected after it's been inserted.

Admissible options:

```
Assign("bib_keyPrompt", "yes"); // Default
Assign("bib_keyPrompt", "no");
```

4.4 Configuration and Resource Files

4.4.1 Local Configurations

What makes this packages even more flexible is the possibility to define any of the options locally for a bibliography database. They will override the options from the configuration file `bibMacros.cfg`. Put this line:

```
// bibMacros:
```

to the end of the bibliography file, followed by any number of

```
Assign("<option>", "<value>");
```

directives (which Bib_T_E_X itself will ignore). These options will only be valid for the current bibliography file.

Alternatively, you may use the following syntax: Start the configuration block with:

```
// bibMacros::
```

(note the two colons!), and then insert the options *without* the “bib_” prefix like this:

```
<option> = <value>
```

You can end the local configuration block with:

```
// bibMacros-end
```

→ for an example see the file `BMStrings.bib`

If you don’t want to clutter your bibliography files with these settings, you can also specify a different configuration file which will be read after `bibMacros.cfg` by putting these two lines:

```
// bibMacros::
```

```
confFile = <your\configuration.file>
```

to the end of a bibliography file.

Another possibility is to put the configuration in the main _T_E_X file, which will then apply to all bibliography files (but may be overridden by settings local to the respective files):

```
% bibMacros:
```

```
% <option> = <value>
```

```
% bibMacros-end
```

4.4.2 Bibliography Templates

relevant for: [New Field](#) · [New Entry](#) · [Info for Current Field](#)

New entry templates will be created on the fly. They are defined in the file `bibMacrosFields.cfg` resp. `bibMacrosFields_<bibStyle>.cfg`. The appearance depends on the options described in the above section “[Formatting Entries](#)”.

If you want to insert a new field, you’ll get a list of possible fields. Possible here means (1) that it’s not already in the entry, and (2) that it is applicable for the current entry. Which fields are applicable for each entry type is also defined in the file `bibMacrosFields.cfg`.

Here’s how: E. g. this part of `bibMacrosFields.cfg`:

```
Assign("bib_ARTICLE",!">
+AUTHOR,>
+TITLE,>
+YEAR,>
+JOURNAL,>
+volume,>
+number,>
+pages,>
+month,>
note,abstract,keywords,source,annotate,crossref,key,url>
");
```

defines that:

1. a new @ARTICLE entry template will consist of all fields which are preceded by a '+', i. e.: AUTHOR, TITLE, YEAR, JOURNAL, volume, number, pages, month (the capitalization is only for your convenience to show you which fields are required for that entry);
2. all other fields are optional (but make sense) in an @ARTICLE entry. They will be offered, if you want to insert a new field.

So, if you want to change the set of fields that an entry template consists of, remove or add the '+'s in front of the respective field names. If you want to be offered different fields when inserting a new field, remove or add the field names themselves.

If you don't want delimiters around certain fields, you can append '-' to these fields. For instance, if you don't want delimiters around the 'year' field, the definition would look like this:

```
Assign("bib_ARTICLE",!">
+AUTHOR,>
+TITLE,>
+YEAR-,>
// ... etc.
```

Furthermore, it is possible to have the fields filled with default contents, when you create a new entry or a new field. For example, you might use the @MISC field to reference content from the internet. That is, you want to put the URL in the 'howpublished' field. To achieve this, your @MISC entry definition could look like this:

```
Assign("bib_MISC",">
+howpublished<WWW: \url{*}>,>
+author,>
// ... etc.
```

If you want to fill a field with a default content *and* you don't want delimiters around it, put the '-' before the '(content)'.

You can also define 'alias' entries. E. g. if want to have the possibility to create both a complete @INCOLLECTION entry template and an @INCOLLECTION that crossreferences another entry and thence can inherit a lot of its fields from this entry. Of course, these two entry types cannot have the same variable name. But you can override the name of the variable (which normally specifies the type of entry, e. g. as above for @ARTICLE) by putting the string '=INCOLLECTION' in the entry type definition.

→ see the example in `bibMacrosFields.cfg`

If you have defined a new entry template, the easiest way to assign a menu item to it in the **New Bibliography Entry** submenu is to copy an existing item for some other entry and then change the value in the **Macro** field for your new entry:

```
Assign("BM_newEntry", "<templateName>");
```

4.4.3 Bibliography Styles

relevant for: [New Field](#) · [New Entry](#) · [Info for Current Field](#)

If you are using a BibTeX style that requires or allows completely different fields in the entries than the default styles, you can create new entry definitions for this style either in the default entry template configuration file `bibMacrosFields.cfg` or in a new file named after the style: `bibMacrosFields_<bibStyle>.cfg` (e. g. `bibMacrosFields_apacite.cfg`). You have to append the name of the style to all variable names.

If you don't define an entry for the particular style, the default definition will be taken to create a new entry, to create a new field, or to inform you of the current field's purpose.

You can load these non-standard entry definitions instead of the default ones if you put this:

```
Assign("bib_bibStyle", "<bibStyle>");
```

either in the global configuration file, or in the local options at the end of your respective bibliography file.

For example, this package provides templates for the L^AT_EX and BibTeX style `jurabib`. Entries for this style are called: `bib_ARTICLEjurabib` instead of `bib_ARTICLE`, etc. It would be loaded with:

```
Assign("bib_bibStyle", "jurabib");
```

→ for examples see the files `bibMacrosFields_jurabib.cfg`, `bibMacrosFields_apacite.cfg` and `bibMacrosFields_biblatex.cfg`

4.4.4 String File

relevant for: [Expand String](#)

You can expand a @STRING if you have defined it in the same bibliography file.

Example: If you have this:

```
title = STOC,
```

in an entry, it will be expanded to:

```
title = {Symposium on the Theory of Computing},
```

if you have defined the @STRING

```
@STRING{STOC = "Symposium on the Theory of Computing"}
```

in your bibliography file. For speed's sake, you should put all your @STRING definitions to the beginning of your files. If your bibliography database is very large, it is also recommended to put the line

```
// bibEndStrings
```

after all @STRING definitions in your file (even if you haven't defined any @STRINGs there!), which will speed up the expansion because everything after that line will be ignored in the search.

The file `BMStrings.bib` contains those @STRING definitions that are provided by the standard bibliography styles, to which you can add your own ones. You can also specify other file(s) to be searched for @STRING definitions, too, by saying e. g.:

```
Assign("bib_stringFile", "C:\texmf\bibtex\bib\beebe\type.bib");
```

or even:

```
Assign("bib_stringFile", >  
      "%P\strings.bib">  
      {C:\texmf\bibtex\bib\ams\mrabbrev.bib});
```

(For multiple files, braces around the filename are necessary!)

A Todo & Bugs

- If entries contain tab stops, lots of things don't work
- Redo parsing of fields (Clean Entry)
- Possibility to specify paths when merging or extracting annotations
- @STRING concatenations all have to be on one line. E. g., for the field:

```
month =          may # "/"  
           # jun,
```

'jun' will not get expanded, and cleaning the entry may have undesired results.

- Option: Always expand @STRINGs when cleaning
- Option: Sort fields when cleaning
- Option: Remove duplicates when cleaning the bibliography
- Possibility to change cross-referencing entries when changing a key (really?)
- bibMacros wants you!

You can help improving this macro package by criticizing it. If you find bugs, missing fields or typos, or if you think the defaults are stupid, please tell me. If you don't understand the documentation or if you are not able to achieve something you think should be achievable, please ask me. If you would like to contribute (entry) configurations for different styles, send them to me:

R Schlicht <w.m.1@gmx.net>

B History

Revision 2.1 (20070607)

- changes in `bibMacros.cfg`:

Option	previous	2.1	
<code>bib_annotatePath</code>	<i><not implemented></i>	<i><relative path></i>	→ 3.3

- Find Entry:
 - will show the searched bibliographies
- Extract/Merge Annotations:
 - option to set the path for extracted/merged annotation files
- entry definitions for the `biblatex` package completed (`bibMacrosFields_biblatex.cfg`)

Revision 2.0 (20070505)

- Sort Bibliography:
 - adapted to WinEdt 5.5's new internal MUI
- Find Entry:
 - adapted to WinEdt 5.5's new internal MUI
 - possibility to search within results
 - international characters don't pose a problem anymore
- entry definitions for the `biblatex` package (`bibMacrosFields_biblatex.cfg`)

Revision 1.9 (20061106)

- changes in `bibMacros.cfg`:

Option	previous	1.9	
<code>bib_emptyFields</code>	<i><not implemented></i>	"remove", "leave"	→ 4.2
<code>bib_{field}_alt</code>	<i><not implemented></i>	<i><string></i>	→ 4.3
<code>bib_{field}_case</code>	<i><not implemented></i>	"U", "L", "I", <i><empty></i>	→ 4.3

- Sort Bibliography:
 - possibility to specify different sort order for the criteria (thanks to Sebastien Nguyen <Sebastien.Nguyen@limsi.fr>)
 - possibility to specify maximum number of authors relevant for sorting (suggested by Sebastien Nguyen <Sebastien.Nguyen@limsi.fr>)
- Generate Key:
 - new option: use string for missing field (suggested by David Huffer <David.Huffer@csosa.gov>)
 - new option: change case of the words (suggested by Dominik Wujastyk <ucgadkw@uc1.ac.uk>)

- Clean Entry:
 - new option: do not remove empty fields
(suggested by Sebastien Nguyen <Sebastien.Nguyen@limsi.fr>)
- new, alternative syntax for local configuration settings
(‘<option> = <value>’ instead of ‘Assign("bib_<option>", "<value>");’)

Revision 1.8 (20040413)

- several minor bug fixes collected over the last 10 months

Revision 1.7 (20030611)

- Sort Bibliography:
 - possibility to sort cross-referenced entries last
(suggested by Jürgen Dix <dix@cs.man.ac.uk>)
- Find Entry:
 - uses the MUI plug-in (if it’s installed; it is available from: <http://www.winedt.org/Plugins/mui.php>)
 - find string in certain entry types only
 - insert the key or the entry
 - options: Whole Words Only, Case-Sensitive, Regular Expression
- package can be installed in %b (default) or %B
- @STRINGs will be ignored when cleaning the bibliography
- more bug fixes

Revision 1.6 (20021001)

- parentheses () as entry delimiters are supported rudimentarily
- Sort Bibliography:
 - uses the MUI plug-in (if it’s installed; it is available from: <http://www.winedt.org/Plugins/mui.php>)
 - will ignore @STRING and @PREAMBLE entries
- Generate Key:
 - if the entry crossreferences another one, missing fields will be used from the crossreferenced entry to construct the key (only from the current file, though)
- New Entry / New Field:
 - possibility to define default field contents
 - possibility to omit delimiters around fields

Revision 1.5 (20020710)

- changes in bibMacros.cfg:

Option	previous	1.5	
bib_wrap	<not implemented>	“no”, or <positive number>	→ 4.2.5
bib_numericFieldDelim	“remove”, “leave”	“remove”, “leave”, “add”	→ 4.2.6

- Sort Bibliography:
 - will expand @STRINGs, (suggested by Chris Sims <sims@Princeton.EDU> translate special characters, (thanks to Jürgen Dix <dix@cs.man.ac.uk> and parse names before sorting
 - possibility to sort up to 10 000 entries (when using WinEdt 20020223)

- Clean Entry:
 - option to always add delimiters to purely numerical fields
 - option to wrap lines (suggested by Juan Fiol <fiolj@umr.edu>)
- Generate Key:
 - will expand @STRINGS before generating the key
 - won't append an extra label anymore, if called twice on the same entry (thanks to Juan Fiol <fiolj@umr.edu>)

Revision 1.4 (20020528)

- changes in bibMacros.cfg:

Option	previous	1.4	
bib_key	"<rule>"	"[<condition>=<rule>]..." or "<rule>"	→ 4.3.4
bib_keyAlways	<not implemented>	"yes" or "no"	→ 4.3.5
- Generate Key
 - conditional rules for key generation (suggested by Mark Ainsworth <ma@maths.strath.ac.uk>)
- Clean Entry:
 - option to always generate a key
- reintroduced the possibility to have external entry template configuration files.
 - bibMacrosFields_jurabib.cfg
 - bibMacrosFields_apacite.cfg
- this pdf manual

Revision 1.3 (20020412)

- changes in bibMacros.cfg:

Option	previous	1.3	
bib_keyCS	<not implemented>	"1" or "0"	→ 4.1
bib_extraLabel	<not implemented>	"no", "{a}", "{1}", "{+a}", "{+1}" etc.	→ 4.3.3
bib_keyPrompt	<not implemented>	"yes" or "no"	→ 4.3.6
bib_stringFile	"{<bibfile.bib>}"	"{<bibfile.bib>}"	→ 4.4.4
- new command Sort Bibliography
- new command Strip Inner Braces (suggested by G. M. Ambrosi <ambrosi@uni-trier.de>)
- Generate Key:
 - new option bib_extraLabel to automatically append a label to ambiguous keys (suggested by Patrick Wallman <patrik.wallman@chemeng.lth.se>)
 - new option bib_keyPrompt to turn off prompting the key before inserting it
- Find Bibliography Entry:
 - new command "show!"
 - "reset!" instead of "reset"
 - "reset!<string>" instead of "reset:<string>"
- Extract From Aux:
 - is compatible with 'harvard' and 'chicago' packages now (thanks to Jan Odegard <odegard@rice.edu>)
 - possibility to append new entries to an existing *-minimal.bib file
 - new switch bib_keyCS to turn off case-sensitive key search (suggested by Jürgen Dix <dix@cs.man.ac.uk>)
- Extract From Aux, Extract / Merge Annotations, Find Entry:
 - automatically search WinEdt's Tree and the TeXMF directories for unknown files (suggested by Jürgen Dix <dix@cs.man.ac.uk>)

Revision 1.2 (20020117)

- changes in bibMacros.cfg:

Option	previous	1.2	
bib_alignKey	“No”, <i><positive></i> or <i><negative number></i>	“no”, <i><positive></i> or <i><negative number></i>	→ 4.2.4
bib_fieldDelim	“brackets”, “quotes”, “leave”	“braces”, “quotes”, “leave”	→ 4.2.6
bib_lastComma	<i><not implemented></i>	“yes”, “no”	→ 4.2.7

- new entry templates are created on the fly (independent from WinEdt templates)
bibMacrosFields.cfg changed, merged with bibMacrosFieldsjurabib.cfg
- lots of improvements and bugfixes

Revision 1.1 (20011230)

first release : <http://www.winedt.org/Config/menus/BibTeX.php>

Disclaimer: The author provides the bibMacros package ‘as is’, without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The entire risk as to the quality and performance of the bibMacros package is with you. Should the bibMacros package prove defective, you assume the cost of all necessary servicing, repair, or correction.

Last modified: 7th June 2007
Bug reports and suggestions to:
R Schlicht <w.m.l@gmx.net>