# BibTeXformat

Benjamin Bulheller
www.bulheller.com
webmaster@bulheller.com

March 9, 2011

## Contents

## Licence

# Recent Changes

## March 7th 2011

- The configuration files now have the extension `.cfg` instead of none at all. Users keeping their old data should add the extension at least to the main configuration file (the other files are defined therein, so they are fine). If the file is not found, the extension `.cfg` is added and a warning issued.
- The `infile` instruction was added to the syntax of `-aopfix` (Section 7.4, page 31) and `-typefix` (Section 7.2, page 26) configuration blocks.

## February 5th 2011

- The option `-fieldregex` allows to use one or more regular expressions on specific BibTeX fields. This is more selective than the usual substitutions that are used on every complete line of the library. See Section 6.3, page 23.
- The option `-fileregex` performs regular expression tasks on the file paths given in `file` fields. The function can be used, for example, to turn relative paths in absolute ones and vice versa. See Section 6.3, page 23.
- The option `-fpapers` complements the `-f` option. While the latter forces all labels to be re-created – thus overwriting all existing ones – `-fpapers` only forces the change of existing labels with the standard format of Papers (*Author:2000p1749*). This preserves all user-defined labels. See Section 4, page 16.
- With the `-aopfix` option it is possible to fix citations of ahead-of-print articles. Depending on multiple conditions such as *missing pages* and *doi present*, one can effect necessary changes to correct the citation. See Section 7.4, page 31.
- The replacement of fields during the reference type correction was fixed and cannot lead to double field entries anymore.
- A bug was fixed concerning command line parameters that are also defined in the configuration file. Now the command line has highest precedence and overrides settings in the configuration file.
- The bibliography of this manual has now been treated with the actual script that is being documented!

## June 28th 2010

### IMPORTANT: Command Line Parameter Change

- The predefined substitutions of common symbols and accented characters are no longer done automatically but require the new `-defsubst` option (for *def*ault *subst*itutions). The old `-subst` option is independent of the new option and only carries out the substitutions defined by the user in the substitutions file. See Section 6. A list has been added to the manual providing an overview of the default substitutions, See Section 6.1, page 21.
- The use of fields without any delimiters (that is braces or double quotes) has been implemented. This facilitates the use of BibTeX macros/strings, which must not be enclosed in any delimiters. See Section 5.2, page 19.

- The feature to ignore BibTeX fields (`-s` option) was greatly extended. Apart from the *general* list that works on all items, there can now be other lists to configure reference types individually. It is now possible to configure, for example, a separate list for `@misc` items. See Section 5.4, page 21.
- A reference type correction for `@inproceedings` was added. All default type corrections are listed in Table 1, page 27. Additionally, the type corrections can now be redefined and new ones defined in the configuration file. See Section 7.2.2, page 28.
- The BibTeX logo in this manual has been corrected to its proper looks (it used to be BibTeX before but nobody ever complained).

## March 19th 2010

### IMPORTANT: Bugfix Affecting the Cite Key Format

A *really* stupid bug was fixed concerning the cite key generation. The cite key format if page numbers are used (`-pn`) is *Author:Year:Page*. If a page range is given, the start page is extracted and used, however, due to the bug the page was ignored if only the start page was given in the first place. This went unnoticed for several months as most citation databases provide the proper citation with the range – I had one case out of 450.

The correction of this bug causes cite keys of papers given without a page range to change. I apologize for any inconvenience.

## January 12th 2010

### IMPORTANT: Changes to Command Line Options and Configuration File

JabRef uses the `file` field to link to external files while Papers uses `local-url`. The script can convert between both formats. However, since many users of the script use other programs, naming the command line options with respect to the mentioned applications was stupid and rather confusing. Therefore, the options and variables in the configuration file have been renamed. Be assured that extensive self-flagellation has been carried out.

Changes to the command line options (possibly to be changed in the `formatlib` script):

- `-jabpdf` → `-local2file`
- `-pappdf` → `-file2local`

Changes in the configuration file:

- `$PapPDF` → `$LocalURL`
- `$JabPDF` → `$FileURL`

While these changes should be considered, the script can deal with the old versions as well to maintain backwards-compatibility.

### Changelog

- `-filecheck` option added to check linked files for existence.

- `-filedir` option added to define the folder linked files are contained in.
- The conversion between `local-url` and `file` fields was fixed.

## Papers

### Export to BibTeX

The script is of general use for BibTeX libraries exported from programs such as BibDesk[2], JabRef[3], and EndNote.[4] However, in 2009 and 2010, lots of features have been added specifically to improve the work with libraries exported from Papers.[1] In conjunction with `bibtex-format` it is possible to 'fix' the exported library without going via EndNote or BibDesk, and the features of the script go far beyond what is possible using other programs. This is a quick'n'dirty list of the features that users of Papers will need and should have a closer look at:

- Papers does not allow a user-defined cite key format. The script can generate the cite keys using the "Author:Year:Page" scheme. See Section 4, page 16.
- Although Papers sometimes exports `@books` and other reference types different from `@article`, these entries cannot be trusted. For example, mandatory fields such as `publisher` may be missing. The script lists all used reference types and provides a possibility to replace the type including the respective fields. See Section 7.2, page 26.
- Author names with multiple parts (e.g., "John van Doe Jr.") are exported incorrectly and can lead to wrong citations. The script can check all author names for such problems (`-autcheck`) and can be configured to correct the names (`-autfix`). Initials that are not separated by dots or spaces (e.g. XY Author) and names in capitals are a problem that cannot be fixed by the script and have to be corrected in Papers directly. See Section 7.1, page 24.

  A Papers-related problem that cannot even be detected by the script is also related to the initials. For each author there are the fields *Firstname* and *Initials*. If *Firstname* contains "Jimmy Joe" then *Initials* should contain *JJ*. However, it may be that the initials are correct but the first names are given incompletely, for example, *Jimmy*. Since only the *Firstname* field is exported, BibTeX will miss out on the second initial. Therefore, it is imperative that the first names and initials are checked for validity. If there are more initials than first names then it points to a possible error.
- It is not possible in Papers to exclude specific fields for the export. Some fields cause problems, for example, the `note` field may be always cited if it is present (this depends on the used BibTeX style). The script offers the possibility to filter unwanted fields, see Section 5.4, page 21.
- Papers does not warn about missing journal abbreviations. The script provides a robust way to add and also switch between different abbreviation styles. See Section 8, page 8.

## Import from BibTeX

Papers can import BibTeX libraries but needs absolute file paths to linked PDFs. If the program used to create the library used relative paths instead, the `-fileregex` option can be used to correct this. See Section 6.3, page 23.

## 1 Introduction

The applications JabRef[3] and BibDesk[2] offer a very sophisticated management of BibTeX databases and users handling their files with either program won't have any pressing need for `bibtexformat`. However, the possibility to easily replace over 100 symbols and accented characters, as well as some other tasks, which can be performed by a single command, may come in handy nevertheless. The script will be particularly helpful for users who primarily use EndNote[4] or Papers,[1] create the BibTeX database via the export function and don't want to go via JabRef or BibDesk to do stuff that always needs to be done to the database after the export.

Some features of `bibtexformat`:

- Over 100 symbols and accented characters are replaced with the correct BibTeX commands (exclusively Mac OS). This includes escaping symbols such as % and &, and is easily extendible for individual needs (e.g. to automatically format something like $\pi \rightarrow \pi^*$). See Section 6, page 21.

- During the processing, each single BibTeX field is available fully parsed and all kinds of actions can be easily implemented. The library `ParseBibTeX.pm` can be used independently of `bibtexformat` to create own scripts using the parsed content that is returned with a simple command. See Section 10, page 35.

- The cite keys (labels) can be created in a consistent fashion and it is possible to keep existing keys as well as to force their replacement. See Section 4, page 16.

- The journal titles can be abbreviated using EndNote term lists (available for every scientific subject). The use of multiple lists is also possible, for example, the newest EndNote list and a personal one with abbreviations not contained in the former. This process also warns about problems such as mistyped journal titles and unknown abbreviations (often due to typos or slightly different titles like "and" vs. "&"), something that EndNote used to silently ignore – at least until v9.0. See Section 8, page 32.

- Unneeded BibTeX entries can be filtered out in order to make the database more compact and find entries easier when searching through it. The list of ignored items is easily expandable, by default it filters out e.g. *abstract*, *keywords* and *note*. See Section 5.4, page 21.

- The BibTeX library can be sorted alphabetically according to the cite keys (which usually start with the author of the publication). See Section 5.3, page 20.

- Abbreviated page ranges can be corrected, e.g. 291–8 → 291–298. See Section 7.3, page 31.

- The library can be formatted nicely, e.g. all equal signs be aligned, multiline entries combined into a single line and field descriptors indented. See Section 5.1, page 18.

- Particularly for use in combination with the publication manager Papers a BibTeX type substitution was implemented. The replacement of `@article` with BibTeX types such as `@book` and `@thesis` can be triggered, including the exchange of several field names (e.g. institution instead of affiliation). See Section 7.2, page 26.

- Links to external files can be converted between `local-url` and `file`. The existence of the linked files can be checked to detect dead links in the library. See Sections 9.2 and 9.3, page 35.

- The author name format is checked for correct division into first names, last names, *von* and *Jr.* part. Automatic substitutions can be defined to solve these problems (if needed, for example, for regular export from Papers). See Section 7.1, page 24.

The general workflow when working with BibTeX is to import new references into the reference manager and export the database to BibTeX format. If any of the features stated above is not satisfactorily implemented in the used software, a simple script run will apply the changes requested via command line options. See Section 2.2 on how to use the script quickest on a regular basis using a runscript based on the user's preferences. Due to its many options, `bibtexformat` is best run by a runscript after creating one's preferred combination of options.

# 2 Setup and Configuration

## 2.1 Installation

The script is written in Perl, which is an interpreted language and, therefore, does not require a compilation. Linux and Mac OS have Perl installed by default, Windows users need to install ActivePerl[5] and may need to do some adjustments (e.g. the path to the Perl executable in the first line of the script).

If you are not used to the terminal you should use the installation routine, which is included in the zip archive. Uncompress the archive, open a terminal session (Applications → Utilities → Terminal), change into the directory with the files and run the script `install`, for example:

```
cd Downloads
cd bibtexformat
./install
```

A possible problem is that the file permissions were not preserved in the archive. Under Linux and Mac OS a file has to *be* executable, it is not implicitly assumed to be due to its extension. If you get a `Permission denied` error, try making all script executable:

```
chmod +x bibtexformat formatlib install
```

Then run `./install` again. Under Windows the file extension `.pl` may need to be added to satisfy Microsoft's extension fetishism.

The installation is divided into single steps and in each one the user is told what is needed and going to be done if `y` is answered. So if the step is skipped the user knows what has to be done manually.

```
leslie@texmac:~/Downloads/bibtexformat/> ./install

bibtexformat - BibTeX Library Handling
--------------------------------------

(c) Benjamin Bulheller, please see source for licence
    http://www.bulheller.com

Usage:   install  <directory>
  e.g.   install  ~/bin/bibtexformat

         If no directory is provided, ~/bin/bibtexformat is chosen by default.


 =>  This installation script copies the following files:

     to /Users/leslie/bin/bibtexformat:
        bibtexformat          (the script itself)
        formatlib             (run script with the user's favorite parameters)
        bibtexformat.pdf      (the manual)
        abbreviations         (example file for journal abbreviations)
        configuration         (contains user-defined settings)
        authors               (author name substitutions)
        ParseBibTeX.pm        (Perl library for parsing the BibTeX database)
        DebugPrint.pm         (Perl library required by the script)
        GetParameters.pm      (Perl library required by the script)

     Do you want to proceed? ("y" to proceed, any other key to abort)
```

If you are comfortable using the command line you may want to do the steps needed yourself. Apart from the script itself, three Perl libraries are required. Three configuration files and a runscript are provided as templates to be extended to the user's liking. All is contained in the archive and to install it manually, do the following:

- Unzip the archive (→ folder `bibtexformat`). The two contained Perl libraries are required and the script needs to find them during execution. They should be kept in the same folder as the `bibtexformat` script as this makes later updates easier.
- Copy the folder to a permanent location (e.g. `~/bin/bibtexformat`).
- Add this folder to the search path in `~/.bashrc` or `~/.profile` by adding the line `PATH=$PATH:~/bin/bibtexformat`.
- Correct the path to the configuration file (the variable `$ScriptLocation` under "Configuration Variables" at the beginning of the `bibtexformat` script).
- Under Linux and Mac OS the file needs to be made executable, see above.

You have to close the current terminal session to update the search path. If the script is found and no problems occur with the required Perl libraries, issuing the command `bibtexformat` will display the usage information.

## 2.2 Regular Usage

### 2.2.1 The `formatlib` Script

The `bibtexformat` script is controlled via command line parameters to keep it flexible and easily configurable by the user. However, this also means that one may need four or five different switches for the preferred settings. A small bash script can reduce this to just a single command.

The idea is to define all required parameters in this runscript instead of typing them while executing `bibtexformat`. The runscript holds all parameters that are needed every time, for example,

```
#!/bin/bash

bibtexformat $@ -s -labels -pn -f -format -sort -abb ~/bin/abbreviations
```

An example script (called `formatlib`) is provided. The variable `$@` contains the whole command as given to the script and it is possible to pass, for example, `-o outfile.bib` on to `formatlib`.

### 2.2.2 Exporting from Papers via AppleScript

The provided AppleScript requires only little configuration. With a single click, it exports the Papers library to a BibTeX file and runs the `formatlib` on it. Since the terminal output of shell commands ran via AppleScript is discarded, the `-log` option is forced. ***The AppleScript is not (and will never be) a way to avoid the command line entirely!*** Before using it on a regular basis, make sure that all runs smoothly in the terminal, the output of the script has to be checked carefully.

To configure the AppleScript, the temporary file and output file have to be defined, both are given relative to the user's home folder. Since AppleScript does not provide a way to wait for a process to finish, two delays have to be set to wait for the export in Papers and the execution of `bibtexformat` to complete. The processing time depends on the machine speed, the size of the library, and the number of substitutions and corrections to be performed. Therefore, these two delays have to be adapted to the individual user.

Since Papers does not support AppleScript directly (as of v1.9), the menu items are accessed via explicit text references such as *File* or *Export*. It is, therefore, dependent on the language of Mac OS and Papers and the references to menu items have to be translated if a different language than English is used. This is indicated in the source code of the script.

## 2.3 Configuration

The `bibtexformat` script can be configured via an external configuration file. Its path and filename are defined at the beginning of `bibtexformat`. The path to the configuration file is basically the only setting that has to be done in the source of the script itself. The included install script takes care of this setting.

```
#################################################
# Configuration Variables
#################################################

# the path to the script, its libraries, configuration file, etc.
my $ScriptLocation = "$ENV{HOME}/bin/bibtexformat";

# all configuration variables should be changed in the file defined here
my $Configuration = "$ScriptLocation/configuration";
```

For every run, the configuration file defined in the lines above is read. Alternatively, the user may define a different file via the `-conf` option. Hence, it is possible to have multiple configurations for different purposes and choose the required one via the command line parameter. The original file is still evaluated, and, therefore, the file provided via `-conf` may even be just a partial one with a subset of the possible options, since all missing settings will simply be taken from the main configuration file. The command line parameters always have precedence over values given in the configuration file(s):

1. Lowest precedence: Initial values defined in the source code;
2. Configuration file defined by `$Configuration`;
3. Additional configuration file given via `-conf`;
4. Highest precedence: Command line parameters.

Apart from the settings in the configuration file, which are documented directly in there, some features of `bibtexformat` require the user to set up some files before they can be used:

- The abbreviation of journal names requires the EndNote[4] term list of the respective research field. See Section 8, page 8.
- The script can replace strings with the respective LaTeX code (e.g., `alpha` $\rightarrow$ `$\alpha$` $= \alpha$, or `H2O` $\rightarrow$ `H$_2$O` $= H_2O$). These substitutions have to be added to a specific file, see Section 6.2, page 22.
- Correction of incorrectly formatted author names (if the library was exported by Papers) has to be configured in the respective file. See Section 7.1, page 24.
- Replacement of `@article` with BibTeX types that are not supported by Papers[1] must be prepared in Papers. See Section 7.2, page 26.

## 3 General Usage – The Command Line Options

The usage information is displayed if the script is invoked without any or an unknown command line parameter. In general the usage is

```
bibtexformat library.bib -o library.new.bib [other options]
```

The extension `.bib` may be omitted, it is added automatically. Multiple libraries can be processed in a single run. If no output file is defined via `-o` then the input file is overwritten and a backup of the original file is created (unless this feature is disabled in the configuration file).

The command line parameters are loosely separated into groups of similar tasks. Some features, which are executed by an option such as `-labels`, can be further configured by the options given in the secondary lists (while, for example, `-pn` has no effect without the main option `-labels`).

- `-o`
  This defines an **o**utput file name to which the resulting library is written to.

  If omitted, the input file extension is changed to `.old.bib` and the original filename is used for output. The creation of the backup file can be disabled in the configuration file (variable `$BackupLibrary`). It is strongly recommended to write to a different file unless the library is produced by a program such as Papers or EndNote and can be recreated easily. The script changes the library and, using the `-s` option, may even delete information from it.

Multiple runs on the same library have been tested without problems but are generally not recommended. A backup file should be created if multiple runs are required.

If the given filename for output starts with a dot, it is regarded as the extension for the output file. This is a shortcut for single file processing and crucial for multiple file processing:

```
bibtexformat Library.bib -o .new.bib
```

This saves the output to `Library.new.bib`.

- `-s`
  "**S**hort library", filters out unneeded items such as abstract, keywords, etc. To edit the ignore list, check the configuration block `-> IgnoreFields` in the configuration file (Section 2.3, page 10). Simply add or remove items there.

- `-conf`
  Defines a **conf**iguration file to read the configuration of the script. See Section 2.3.

- `-log`
  Writes all terminal output to a .log file instead of STDOUT/STDERR.

- `-sort`
  Sorts the items alphabetically according to their BibTeX label/cite key. See Section 5.3, page 20.

---

- `-quotes`
  Changes the field delimiters (curly braces by default in Papers) to double quotes (`{...}` → `"..."`). This does not affect fields without delimiters, e.g. a field entry that is a BibTeX `@string`, or fields that contain just single numbers and are allowed to lack any delimiter. See Section 5.2, page 19.

- `-braces`
  Changes the field delimiters to curly braces (`"..."` → `{...}`). See Section 5.2, page 19.

---

- `-labels`
  Enables the creation of cite keys for each entry without a label (existing ones are not overwritten unless `-f` or `-fpapers` is given). The template is *FirstAuthor:Year*. Lower case letters are added if necessary to avoid ambiguous labels. See Section 4, page 16.

  - `-pn`
    Uses the first **p**age **n**umber to create unambiguous cite keys/labels of the type *Author:Year:Page*. This is highly recommended as the label is not going to change when new references are added (as it would be the case if references are numbered). The script always checks for double labels and warns the user if they occur.

  - `-fy`
    By default only the last two digits of the publication year are taken for the labels. This was decided to make the labels shorter and since the year is not necessarily given with four digits. Using `-fy`, the **f**ull **y**ear with all given digits is taken to create the cite keys.

  - `-sep`
    This defines the **sep**arator of the BIBTEX labels and is set to a colon by default, e.g. "Author:Year".
  - `-f`
    **F**orces the generation of cite keys, even if already defined. Since it is possible that previously created cite keys have been used in LATEX documents already, the script does not touch them by default.
  - `-fpapers`
    **F**orces the generation of cite keys but only overwrites existing ones that match the default format of **Papers** (*Author:2000:p7281*). User-defined keys are preserved.

---

- `-filecheck`
  Checks the external files linked via `file` and `local-url` for existence or zero file size. See Section 9.1, page 34.

- `-filedir`
  Allows to provide a path to the location of linked files. This requires relative paths in the library. See Section 9.1, page 34.

- `-local2file`
  This converts all links to external files as defined by Papers and BibDesk (using `local-url`) to links for JabRef (using `file`). See Section 9.2, page 35.

- `-file2local`
  This converts all links to external files as defined by JabRef (using `file`) to links for Papers or BibDesk (using `local-url`). See Section 9.3, page 35.

---

- `-autcheck`
  Check all author names for correct division into first names, last names, *von* and *Jr.* part. Warnings are also printed for suspicious entries. If multiple libraries are processed, the results are combined and duplicates removed. See Section 7.1, page 24.

- `-autfix`
  Performs the corrections of author names as defined in the respective file, which is in turn defined in the configuration file. See Section 7.1, page 24.

- `-autmax`
  Shortens the number of authors to the given number (BIBTEX adds *et al.*). This is intended for users who have to use style files that do not support this automatically. See Section 7.1.2, page 25.

- `-autlist`
  Prints a list of all authors in order to check them for typos or similar but not equal spelling (e.g. "III" vs. "3rd"). If multiple libraries are processed, the results are combined and duplicates removed. If the variable `$ShowSum` is true ("1") in the configuration file, the number of occurrences is printed next to the author name and the list is sorted according to these numbers. Otherwise the list is sorted alphabetically. See Section 7.1.1, page 24.

---

- `-rangefix`
  Expands abbreviated page ranges (e.g. $291 - 8 \rightarrow 291 - 298$).

- `-protitle`
  Protects the title case by enclosing it in double braces, `{{...}}`. That way, BibTeX will not change the case of the title.

- `-typelist`
  Prints a list of all found reference types (e.g., `@article`, `@book`). See Section 7.2, page 26.

- `-typereset`
  Replaces all BibTeX reference types with `@article`. This is needed for exports by Papers, since the types are often set incorrectly. See Section 7.2, page 26.

- `-typecheck`
  Checks each BibTeX item whether the required fields of its entry type (e.g. `@article`, `@book`) are present. No changes are done to the file, only warnings are given as these errors need to be corrected in the reference manager.

- `-typefix`
  Replaces the BibTeX type (e.g. `@article`) with another to correct entries exported by Papers. See Section 7.2, page 26.

- `-joulist`
  Prints a list of all journals in order to check them for typos or similar but not equal spelling (e.g. "The Journal..." vs. "Journal"). If multiple libraries are processed, the results are combined and duplicates removed. If the variable `$ShowSum` is true ("1") in the configuration file, the number of occurrences is printed next to the journal name and the list sorted according to these numbers. Otherwise the list is sorted alphabetically.

- `-aopfix`
  To fix the citations of ahead-of-print articles that do not have their final page numbers yet. The DOI value can be added to an arbitrary field so that it is contained in the citation. See Section 7.4, page 31.

---

- `-defsubst`
  Enables default substitutions that are defined in the source code, replacing the most common symbols and accented characters. See Section 6, page 21.

- `-subst`
  Enables user-defined substitutions of symbols, words, or phrases. See Section 6, page 21.

- `-fieldregex`
  Performs regular expressions given by the command line on specific fields:

  ```
  ... -fieldregex year "from" "to" author "from" "to" ...
  ```

  Regular expression commands such as `^` and `$` work fine to refer to the beginning and end of the entry, respectively. The symbols have to be used without escaping them. See Section 6, page 21.

- `-fileregex`
  Uses regular expressions given by the command line on each individual filename within "`file`" fields:

```
... -fileregex "from1" "to1" "from2" "to2" ...
```

The regular expression commands `^` and `$` match the beginning and end of the *filenames*, respectively. Hence, this can be used to change relative paths to absolute path by using

```
... -filereges "^" "/Users/me/Publications/"
```

See Section 6, page 21.

---

- `-nl`
  Defines the number of empty lines ("**n**ew**l**ines") between BIBTEX items, the default is 2.

- `-format`
  This effects a reformatting of the library. The default parameters for this can be changed in the configuration file or via the command line. See Section 5, page 18.

  - `-lb`
    **l**eading **b**lanks before a field descriptor, the default is 3.
  - `-ep`
    the **e**qual sign **p**osition (including leading blanks), the default is 15.
  - `-lc`
    Formats the BIBTEX type identifiers ("`@article`") and field keywords ("`author`") **l**ower**c**ase.
  - `-uc`
    Formats the BIBTEX type identifiers ("`@article`") and field keywords ("`author`") **u**pper**c**ase.
  - `-combine`
    By default entries spread over multiple lines (e.g. title or abstract) are preserved. Via `-combine` the entries can be contracted to a single line.
  - `-wrap`
    Wraps entries spread over multiple lines at a given column number, e.g. `-wrap 80`.

---

- `-abb`
  In the configuration file, one or more files holding journal abbreviations can be defined with the `@AbbFiles` variable. Using `-abb`, additional files can be added to this list. See Section 8, page 32.

- `-abb1`, `-abb2`, `-full`
  To abbreviate the journal titles. A journal name can have the full title and two different abbreviations (usually with and without periods), of which the first abbreviation is used by default if a file is given via `-abb`. Using the parameters `-abb1`, `-abb2`, or `-full` one of the abbreviations can be specifically selected or all abbreviated titles replaced with their full version. See Section 8, page 32.

The script reads the library and – depending on the parameters – may remove information from it. Especially if the file was changed manually for some reason, that is if it was not just exported from EndNote and the like, it is strongly recommended to make use of the `-o` option and write the output to a new file instead of overwriting the input.

It was taken care that replaced symbols are not replaced again if the `bibtexformat` is used on an already processed file. This concerns, for example, dollar signs, braces, backslashes and other characters that are used to typeset symbols in LaTeX. However, there may be cases that have not been considered yet what would result into symbols getting escaped twice and BibTeX getting confused by that. Therefore, it is generally recommended to avoid such problems by writing the output to a new file and using all needed parameters in one command instead of multiple runs. Again, it should make no problems running `bibtexformat` multiple times on a file but if you do so, check whether everything worked as expected.

Please see Section 2.2 on how to use the script quickest on a regular basis using a runscript based on the user's preferences. Due to its many options, `bibtexformat` is best run by a runscript.

## 4 Cite Key Generation

If the `-labels` option is given, the script creates the cite keys based on either of two predefined schemes, `Author:Year` or `Author:Year:Page`. There are at least as many cite key format philosophies as there are BibTeXers. The system can actually be chosen to one's liking but two rules should *definitely* be obeyed:

1. The system *must* ensure that the cite keys are unique, double labels would lead to citation mistakes.
2. The system should be *future proof*, that is, the cite keys should not change in the future when more references are added. This is utmost problematic when e.g. two or more reports have to be combined using the same references by with different labels.

A very common system is `Author:Year` (in fact so common that `bibtexformat` uses it by default). To keep the labels short, only the last two digits of the year are used. This can be avoided with `-fy` option, which causes `bibtexformat` to use the full year instead. In order to obey rule 1, letters are added to the label if more than one paper with the same first author per year is found, for example,

```
Author:03a
Author:03b
Author:03c
```

However, on the one hand this is bad because the label of a particular paper can hardly be memorized for large libraries and it takes time to look it up. On the other hand, rule 2. is extremely jeopardized. If a new paper of the same author and year is added to the top of the file, the labels for following references will change. The worst thing is that there is no way to detect wrong citations. The citation of `Author:03b` is still valid since the label exists, although the actual (prior) `Author:03b` reference is now called `Author:03c`. Therefore, such a labelling system is *much* discouraged! You have been warned...

A *much* better solution is to include the first page number of the paper, a common format being `Author:Year:Page`. The probability that the same author publishes two papers in a year with the same first page number is very small (thus ensuring rule 1) and the label will never change in the future (ensuring rule 2). A handy side effect is that just from the paper itself (or even the citation of it in another paper) one can instantly figure out the label and save an enormous amount of time.

The `Author:Year:Page` labelling style can be selected by issuing the option `-pn` (**p**age **n**umber). The subroutine generating the label for each reference is `CreateLabels`. At that point, each reference is already split up into its separate fields like this:

```
'Author' => 'Anfinsen',
'Entry' => [
             '@article{Anfinsen:1973p286,',
             '   author     = {Christian Anfinsen},',
             '   journal    = {Science},',
             '   title      = {Principles that govern the folding
                                of protein chains},',
             '   number     = {4096},',
             '   pages      = {223--230},',
             '   volume     = {181},',
             '   year       = {1973},',
             '}'
           ],
'FirstPage' => '223',
'Journal' => 'Science',
'Label' => 'Anfinsen:1973p286',
'Title' => 'Principles that govern the folding of protein chains',
'Year' => '73'
```

The routine loops over each reference in the Library and each field is accessible in the {`Fields`} hash, e.g. `$Item->{Fields}{author}`, `$Item->{Fields}{journal}`.

The separator between the single parts of the labels is a colon (`:`) by default and can be changed using the option `-sep`. Note that if given via the command line, the separator has to be enclosed in quotation marks: `-sep ":"`.

If neither author, year, nor page is defined and the label just an empty string, the script uses the journal name (if possible) and strips all blanks from it. If not even a journal name is defined the ultimate and last resort is using the label `Misc`. A warning will be issued if this happens.

By default, existing cite keys are not changed. If all labels are to be changed, then this can be **f**orced using the `-f` option. In order to preserve all user-defined labels but overwrite all cite keys created by Papers (*Author:2000p7826*), the `-fpapers` option may be used.

# 5 Database Formatting

## 5.1 General Format

If the `-format` option is given, the subroutine `FormatLibrary` tries to format the whole library nicely by indenting the BibTeX items (e.g. *author*) and aligning the equal signs of the fields. The default values for the format are

- 3 leading blanks (option `-lb`) to indent items;
- equal sign position in column 15 (option `-ep`);

These settings lead to entries of a format like this:

```
@article{Anfinsen:73:223,
   author      = {Christian Anfinsen},
   journal     = {Science},
   title       = {Principles that govern the folding of protein chains},
   number      = {4096},
   pages       = {223--230},
   volume      = {181},
   year        = {1973},
}
```

After processing, the library is written to the output file by `sub WriteBibTeX`, which can be found in the library `ParseBibTeX.pm`. After each entry, empty lines are added according to the value of the variable `$Options->{nl}` (set in the configuration file) or the number given via the command line option `-nl`. Mind that the adding of empty lines between entries strictly speaking is not part of the database formatting and, therefore, does not require the `-format` option. To disable the adding of empty lines, the respective value needs to be set to 0.

Some libraries contain an inconsistent letter case format for the type and field descriptors (`@ARTICLE`, `title`, `YEAR`). This can be switched to a consistent fashion, using `-uc` to switch all keywords to uppercase, and `-lc` to use lowercase letters.

By default, entries spread over multiple lines are kept that way. This does not affect the indentation, the additional lines will be indented to the correct column:

```
@article{Chen,
   title       = {The Earliest Events in Protein Folding: A Structural
                  Requirement for Ultrafast Folding in Cytochrome c},
   abstract    = {The folding dynamics of reduced cytochrome c (red-cyt c)
                  obtained from tuna heart, which contains a Trp residue.},
```

The `-combine` option causes multiple line entries to be contracted in a single line, which some users prefer and is handy with editors capable of switching line wrapping on and off. The script is also able to wrap entries to a specific maximum width, for example, column 80. This can be requested with the `-wrap #` option, with `#` being the column at which the lines are to be wrapped (at the latest, depending on the blanks on the line).

## 5.2 Field Delimiters

### 5.2.1 Braces vs. Quotation Marks

Using the options `-braces` or `-quotes`, the field delimiters of the library can be changed to braces or quotes, respectively.

BibTeX fields (everything between a field descriptor followed by an equal sign, e.g. `author =` and the comma) may be delimited using braces, double quotation marks or not at all in the case of single numbers:

```
author = "John Doe",
title  = {A Life as an Unknown, {AKA} a John Doe},
pages  = {12--24},
volume = 2
```

For the `author` and `title` fields it does not matter if braces or quotes are used as delimiters. While the `pages` field *has* to be delimited (it is not a plain number), the `volume` field *may* be delimited but does not have to be. Chapter 8 of *Tame the BeaST*[6] is a recommended read on that matter.

The brace depth controls whether BibTeX may change the letter case. Since the acronym "AKA" is surrounded by braces, BibTeX would not change the case to lowercase, if the style was configured to format the title lowercase. However, the brace level is not affected by the topmost delimiters of the field and it does not matter if braces or quotes are used.

Things become quite complicated, if quotation marks are contained in the field value. If quotes are used as field delimiters, the one within the field value have to be braced, otherwise the entry would turn invalid:

```
title  = {A Life as an Unknown, {AKA} a John Doe},       % correct
title  = "A Life as an Unknown, {AKA} a {"}John Doe{"}",  % correct
title  = "A Life as an Unknown, {AKA} a {"John Doe"}",    % correct
title  = "A Life as an Unknown, {AKA} a "John Doe"",      % WRONG
```

If you needed German umlauts, protect the letter case *and* have quotes as delimiters, then things would turn really nasty. The correct BibTeX for such a character within a quotes field value would be
{ { \"{A}} }
The { } are required to shield the quotation marks and the { } protect the letter case (or vice verca). To keep the conversion as robust as possible, `bibtexformat` will not change the delimiters to quotation marks if any quotes are found within the field's value.

### 5.2.2 BibTeX Strings

BibTeX features the possibility to define abbreviations (`@strings`) for regularly used strings. Very often, such definitions are contained in the style file, for example to turn `JACS` into `Journal of the American Chemical Society` or the required abbreviation. Two rules have to be followed:

- `@strings` must not be enclosed in braces or quotation marks;
- if contained in a field's value, `@strings` must be enclosed in hash symbols (`#`) and the field's delimiters properly closed and opened.

Even multiple values are possible, as long as each `@string` is enclosed with hash symbols:

```
@string{string1  = "First Name"}
@strind{string2  = "Last Name"}

<[...]>
author = "I myself and "# string1 # string2 #" and somebody else".
<[...]>
```

`bibtexformat` is able to handle both the `@string` definitions and the use of strings within the fields. This includes the change of the field delimiters. If a field did not have any delimiters in the first place, `bibtexformat` will not add any. The following example was taken from Chapter 11 of *Tame the BeaST*[6] (which is strongly recommended for its discussion of the author name format issue). The example demonstrates the use of multiple BIBTEX `@strings` within a field:

```
@string{goossens   = "Goossens, Michel"}
@string{mittelbach = "Mittelbach, Franck"}
@string{samarin    = "Samarin, Alexander"}
@string{AW         = "Addison-Wesley"}

@book{companion,
   author    = goossens #" and "# mittelbach #" and "# samarin,
   title     = "The {{\LaTeX}} {C}ompanion",
   year      = 1993,
   publisher = AW,
   month     = "December",
}
```

The `year` and `publisher` fields are unaffected by the `-quotes` or `-braces` option. In the above case, the `author` field also remains untouched because no delimiters are at the beginning and end of the field's value. There is a special case that has to be considered, when a `@string` is in the middle of the field:

```
author = "Me and "# myself #" and I",
```

Changing the surrounding quotation marks would create an invalid entry as the two quotation marks before and after the `#` would need to be changed as well. It would take a considerable number of conditions and regular expressions to process such cases in a robust way. In order to avoid slowing down the processing and keep the routine robust, `bibtexformat` will not change the quotation marks to braces in the above case. To sum up the special cases:

- Quotation marks will not be changed into braces if more than two `"` and multiple hash symbols (`#`) are found.
- Braces will not be changed into quotation marks if any `"` are found.

## 5.3 Sorting

The subroutine `SortLibrary` sorts the entries according to the cite keys. In most cases the cite key starts with the first author. By default the sequence of the entries is maintained; the option `-sort` must be given to enable sorting.

Every reference is checked whether a cite key is present and if the cite keys are unique. If either of those problems is encountered, sorting is disabled and the user warned about it.

### 5.4 Filtering Unnecessary Fields

The reference managers EndNote,[4] BibDesk,[2] and Papers[1] are able to export all available information on a paper using the respective BibTeX field. However, most times only a few fields are actually required to cite the publication and others can even make problems with some style files, for example, the `url` and `note` entry. Via the command line option `-s` (*short*) all fields that are not required can be filtered out.

Entry fields that are to be ignored are defined in the configuration file (see Section 2.3) in the configuration block `-> IgnoreFields`:

```
-> IgnoreFields:
   general:  Note, Abstract, Keywords, local-url, url
   @article: Note, Abstract, Keywords, url
   @misc:    Note, Abstract, Keywords, local-url
END
```

The "`general`" list is used for all reference types that are not defined specifically (the lists are not additive). In the above example, the `local-url` field is kept for `@articles` (it is usually the PDF) and for `@misc` items the `url` field is kept because `@misc` is often used to cite websites.

The default list "`general`" is much longer than shown above. By adding items or removing them from this list, the ignored fields can be adapted to one's liking. The strings are compared case-insensitively (`ARTICLE` = `article`) and the `@` of the reference types is optional. The entry for a reference type may be spread over several lines and blanks can be added freely to make the block better readable.

**Legacy / Backwards-compatibility:** The list of items to be ignored used to be stored in an array (`@IgnoreItems = (qw/Note Abstract/)`. This array has been superseded by the `general` list. If an old configuration file is used, `@IgnoreItems` will simply be used as the `general` list and nothing needs to be changed. The script will only complain if a `general` entry in `-> IgnoreItems` is found additionally.

## 6 Substitutions of Characters and Strings

Windows uses the Symbol True Type font to insert symbols and this information gets lost when the file is exported as a BibTeX database text file. At least EndNote until v9 did not export the correct LaTeX codes. On Mac OS, the information of special characters is preserved. The symbols turn up as seemingly random ASCII characters but these do relate to the original symbol. This is used in the default substitutions performed by the script.

### 6.1 Default Substitutions

About 100 symbols and accented characters are automatically replaced with their respective BibTeX code. Additionally, percent signs are escaped unless they are already preceded by a backslash. These substitutions are hard coded in the script itself (`sub PredefinedSubstitutions`) because the processing speed is considerably faster within the script compared to the evaluated instructions in an external file.

Using the `-defsubst` option the automatic substitutions can be enabled. Note that this does *not* affect the user defined replacements in the substitutions file, which are enabled by giving the `-subst` option.

The following symbols are escaped or replaced with their respective BibTeX code (using `-defsubst`):

- escaping of ampersands, dollar signs and percent signs
- Ä, ä, Ö, ö, Ü, ü, ß
- Á, á, À, à, É, é, È, è, Í, í, Ì, ì, Ó, ó, Ò, ò, Ú, ú, Ù, ù
- Â, â, Ê, ê, Î, î, Ô, ô, Û, û
- Ç, ç, Q̧, q̧, Ş, ş, Ø, ø, æ, œ
- Ñ, ñ, Õ, õ, Š, ř, Ÿ, ÿ, Å, å
- $\Gamma$, $\Delta$, $\Theta$, $\Lambda$, $\Xi$, $\Pi$, $\Sigma$, $\Upsilon$, $\Phi$, $\Psi$, $\Omega$
- $\alpha$, $\beta$, $\gamma$, $\delta$, $\epsilon$, $\zeta$, $\eta$, $\theta$, $\iota$, $\kappa$, $\lambda$, $\mu$, $\nu$, $\xi$, $o$, $\pi$, $\rho$, $\sigma$, $\varsigma$, $\tau$, $\upsilon$, $\varphi$, $\chi$, $\psi$, $\omega$, $\vartheta$, $\phi$, $\varpi$

In some of the following routines the BibTeX commands have to be removed again (e.g. when the cite keys are created, translating `\'{e}` into `e`) but the routine is needed at the very beginning of processing the file to translate the ASCII gibberish into the 'base' letters like a, i, o, etc.

Users who do not require several of the standard replacements should copy only the needed lines of `sub PredefinedSubstitutions` and paste them into the substitution file. Then, in order to disable the standard substitutions and enable only the user-defined substitutions, the command line option `-subst` is required and `-defsubst` must not be given.

When escaping symbols it has to be taken care of not escaping them twice, especially if the script is run multiple times on the same file. The most problematic case are dollar signs and checking which ones create a math environment and, therefore, must not be escaped and which ones are real dollar sign would be complex and prone to errors. Hence, dollar signs are only replaced in the one case where it is definitely sure that it is not a math environment: when only one dollar sign is found in the entry.

As mentioned in the previous Section it is recommended not to run `bibtexformat` multiple times on the same library file. If you have to this, check now and then for unexpected results. Everything *should* be fine but there may be cases that have not been considered yet.

## 6.2 User-Defined Substitutions (File)

User-defined substitutions can be added to the file specified by the variable `$Substitutions` in the configuration file (`$ScriptLocation/substitutions` by default). These commands are executed if the `-subst` option is given. Executing these regular expressions from an external file takes much longer than adding them to the main script, however, the latter makes updates of the script a real pain.

The standard (default) replacements done by the script are defined in the subroutine `PredefinedSubstitutions`. If any particular substitution causes problems it can be commented out there. Replacing symbols the way it is done in the script basically works only under Mac OS and was tested with EndNote X1 (as mentioned above, Mac OS turns each symbol in a character string when exporting to text and this string is used in the substitutions).

If any additional replacements are required, add all new symbols into an EndNote field. You can add all at once, just make sure that they are well separated (e.g. by multiple blanks because some symbols end up as two characters in a plain text file and clear separators help dividing them up.

Now export the reference containing the symbols as BibTeX database and open it in together with the `substitutions`-file. Just copy one of the existing regular expressions and change it to your needs. Example:

```
$Line =~ s/‘A/\\‘\{A\}/g;    # An accent grave
$Line =~ s/H2O/H$_2\$O/g;    # H2O
```

All characters (there are often two or three), which are originating from the symbol in question, have to go between the first two slashes. The result comes between slashes two and three, mind that backslashes and brackets have to be escaped (that is, prepended with a backslash).

## 6.3 User-Defined Substitutions (Command Line)

**Substitutions in particular fields:** The option `-fieldregex` takes a list of strings defining the field name, the regular expression to search for and the one to substitute with. Multiple substitutions may be given (the overall number of strings given to `-fieldregex` must be divisible by three):

```
... -fieldregex title "from1" "to1" journal "from2" "to2" -o newlibrary.bib
```

The above would use `s/from1/to1/g` on all contents of *title* fields and `s/from2/to2/g` on all contents of *journal* fields.

These substitutions are performed after parsing the library and the actual contents of the fields are used. Substitutions triggered by `-defsubst` and `-subst` are done on a line-by-line basis; this makes the `-fieldregex` switch more powerful. In addition to specifying particular fields, it is possible to refer to the beginning and end of the field contents, for example

```
... -fieldregex local-url "^" "/Users/fred/PDFs"
```

This would add the given path to the beginning of all `local-url` fields. The shell takes care of many escapings that are required in the substitutions file, e.g. the slashes in the above example can be used without a leading backslash to escape them.

It is recommended to enclose the regular expression parameters in quotation marks as shown above. To delete a match (i.e. replace with an emtpy string), use `""`.

**Substitutions in `filenames`:** Entries in `file` fields are a bit more complicated than `local-url` fields as they can hold multiple file links:

```
file = {Comment:filename:type; Comment:filename:type;}
```

While it is possible to add absolute paths to the beginning of all `local-url` entries by using `-fieldregex "^" "/Users/..."`, this would not work with `file`-fields since “^” would refer only to the very beginning of the field. For this, the `-fileregex` option can be used, which works on each individual file name in `file`-fields. It can take multiple pairs of regular expressions:

```
... -fileregex "from1" "to1" "from2" "to2" ...
```

# 7 Checks and Corrections

## 7.1 Author Name Corrections

### 7.1.1 Correct Name Part Separation

The correct interpretation of a simple name is much harder than it may appear without thinking thoroughly about it. Since first names sometimes have to be abbreviated, last name(s) always must be given in full and additions like 'von', 'de la', 'Jr.', etc. complicate the matter further, BibTeX needs to have the name properly formatted to separate the individual parts correctly. A short summary on author names:

- Multiple authors are separated by " and " (commas are possible but "and" is recommended and more commonly used).
- A name consists of some or all of the following parts:
  - First names (may be multiple names);
  - von part (as in "Ludwig van Beethoven" or "John de la Doe");
  - Last names (may be multiple names);
  - Jr part (as in "Disney, Jr." or "Shrek, 3rd").
- The first names are usually abbreviated (depending on the BibTeX style) "Lerner, Rick Orlando Francis" → "R.O.F. Lerner".

The "BibTeX-FAQ"[7] and "Tame the BeaST"[6] are two highly recommended summaries on such and all other kinds of BibTeX problems. And – although the document is quite aged by now – "BibTeXing" by Oren Patashnik himself.[8]

In Papers,[1] "first names" and "last names" can be separated fine. However, in versions until 1.9x (?), the names are exported incorrectly by the BibTeX exporter in a sequential fashion: "John Jim van Doe III" instead of "van Doe, III, John Jim". Due to this problem, it is *very* problematic if such names are used, as they will be interpreted incorrectly by BibTeX. Also note that this problem is not unique to the Papers[1] export but can also occur for EndNote,[4] BibDesk,[2] and JabRef[3] users, therefore, the author-format check is of general practicability.

If the script is given the `-autcheck` option, it checks all authors for the correct format (subroutine `CheckAuthors`). The routine checks for

- names with the von-part in the middle ("John von Doe");
- names with a not separated Jr-part ("John Doe Jr.");
- names with the Jr-part in the wrong place ("Doe, John, Jr.");
- names with the first names in wrong place ("J. J., Doe");
- potential double-names ("M. Sanchez Arroyo");
- names with empty fields ("Doe,");
- names with more than three comma-separated fields;
- names written only with capital letters (this also catches 'merged' initials, e.g. AB Author).

Mind that these checks cannot be deemed 100% robust. For example, the detection of incorrectly placed first names depends on the first names being abbreviated to distinguish them from the last names. The “potential double names”-warning is triggered, if an abbreviated name is followed by multiple non-abbreviated parts (“M. Sanchez Arroyo”). Lastly, the detection of the von- and Jr-part depends on predefined versions of those, and although over a dozen libraries from users (literally) all over the world have been checked for them, there may be more. Currently, the following strings are used to detect the von- and Jr-part:

```
    my $JrPart  = "(Jr|Sen|1st|2nd|3rd|I|II|III|IV)\.?";
    my $VonPart = "(von|von der|van|van der|del|de|de la|St|don|dos)";
```

Since the detection is, therefore, not perfect, it is strongly recommended to check the author names manually (visually...). Via the `-autlist` option, a list of all authors is printed. Since this list shows the author names exactly as read from the library by BIBTEX, it is safer to check the `-autlist` output than the respective list in Papers. A similar possibility exists for the list of journal titles, which can be requested via `-joulist`.

To solve the problem of incorrectly formatted author names, the script can be given a file with replacements to be performed, for example,

```
s/Lahari de Alwis/de Alwis, Lahari/
s/G N Phillips Jr/Phillips Jr, G N/
```

The file containing the substitution is called `authors` by default and defined in the configuration file (`$AuthorNames`). All substitutions found in this file are carried out if the `-autfix` option is given.

**CAUTION:** Initials have to be separated by blanks or dots, that is either *X Y Author*, *X.Y. Author*, or *X. Y. Author*. If it were written as *XY Author* BIBTEX would not differentiate between the initials and turn it into *X. Author*. Such cases cannot be fixed by `bibtexformat`, however, if the `-autcheck` option is given it will flag up the names containing capital letter sequences.

### 7.1.2 Maximum Number of Authors

BIBTEX directly supports the abbreviation of the author list with *et al.* and it is generally recommended to use this function. However, this has to be considered in the style file itself and can be tricky to add after it has been created. To create a personal style file, the package `custom-bib` by P.W. Daly is a fantastic and convenient possibility.[9]

`bibtexformat` provides the command line option `-autmax`, which allows to define a maximum number of authors and shortens the list of authors to that number if necessary. It adds the term *others* in that case and BIBTEX will change this to e.g. *et al.*, depending on the style file. To restrict the number of authors to 8 use the command

```
bibtextformat library.bib -auth 8
```

## 7.2 Publication Type Replacements

This feature has basically been tailor-made to tackle a problem with the BibTeX export feature of Papers until version 1.9 (?). The supported publication types were lacking important ones such as `@book`, `@phdthesis`, etc. and, in addition to that, many crucial item fields like `publisher` were missing as well. There is a field `type`, which supposedly causes a BibTeX item to be a `@book` rather than `@article` but this did not always work and could therefore not be trusted. Moreover, the content of the field is not exported to the library and hence another approach had to be devised.

The approach taken by `bibtexformat` is anything but perfect, however, for Papers-only users it proofed worthwhile as it spares the need to use a second program such as EndNote, JabRef, or BibDesk. The two-application approach that some users may prefer over the method described below is using Papers for all `@articles` and, for example, JabRef for maintaining a second BibTeX library comprising all `@books` and other non-`@article` items. BibTeX is able to use multiple libraries to generate one bibliography (`\bibliography{bib01,bib02,bib03}`).

To get a summary of all reference types found in the database, use the `-typelist` option. This provides an overview of all types and helps to decide which replacements may be necessary. The script prints a summary of all used reference types:

```
=> List of reference types
   ----------------------
       6 article
       2 book
       1 phdthesis
       1 misc
```

If many wrong types are found it is recommended to use the `-typereset` switch to change all reference types to `@article` to begin with. All types differing from `@article` should be carefully checked. If the letter case differs (e.g., `@Book` and `@book`), the `-lc` option can be used to format all keywords lowercase.

If the `-typefix` options is given, the script scans certain fields for keywords that trigger the substitution of the type and renaming of several field descriptors of the reference. All comparisons are done case-insensitively. See Table 1, page 27 for the corrections that are hard-coded in `bibtexformat`.

**Caution:** Papers sometimes exports types different from `@article`, however, the fields may be incorrectly assigned or missing. The `-typecheck` option can be used to check for mandatory fields. See Section 7.2.3.

### 7.2.1 Built-In Type Corrections

Table 1, page 27 shows all corrections that are hard-coded in the `bibtexformat` source code and carried out if the `-typefix` option is given (see `sub TypeConversion`). In the current version of the script, these settings can be more easily configured in the configuration file (see next Section) but for the sake of full backwards-compatibility, the old replacements have been kept. However, they can be overridden by a user-defined setting, that is if a replacement for a certain reference type is provided in the configuration file, the respective built-in settings are ignored for this type.

Table 1: Replacements of publication types (usually only needed for BibTeX export of Papers). ***Only one*** of the conditions needs to be true, the string comparisons are done case-insensitively.

| Change to | Triggered by | Changes done |
|---|---|---|
| `@book` | • BibTeX type is `@book`<br>• `type` is *book*<br>• `affiliation` is *book* | • `volume` → `address`<br>• `journal` → `publisher`<br>• `number` → `edition` |
| `@inbook` | • BibTeX type is `@inbook`<br>• `type` is *inbook*<br>• `affiliation` is *inbook* | • `volume` → `address`<br>• `journal` → `publisher`<br>• `number` → `edition` |
| `@phdthesis` | `journal` is<br>• *thesis*<br>• *dissertation*<br>• *Diplomarbeit* | • BibTeX type is `@phdthesis`<br>• `affiliation` → `school`<br>• `journal` → `type` |
| `@mastersthesis` | `journal` is *mastersthesis* | • Type set to `@phdthesis`<br>• `affiliation` → `school` |
| `@misc` | • BibTeX type is `@misc`,<br>• `type` is `misc`,<br>• `journal` is `misc`. | • `@article` → `@misc`<br>• `url` → `howpublished`<br>• `pages` → cite key<br>• `doi` → `note` |
| `@techreport` | `journal` is `techreport` | • Type set to `@techreport`<br>• `affiliation` → `institution`<br>• `volume` → `type`<br>• `pages` → `address` |
| `@incollection` | `affiliation` is `incollection` | • `journal` → `publisher`<br>• `volume` → `booktitle`<br>• `number` → `address`<br>• `doi` → `editor` |
| `@inproceedings` | `affiliation` is `inproceedings` | • `journal` → `booktitle`<br>• `number` → `address` |

### 7.2.2 User-Defined Type Corrections

The type corrections can be defined in the configuration file. While the latter is usually parsed line by line and evaluated using Perl's `eval` command, the type corrections are parsed individually and require a special format. Spaces and tabs can be used freely to format the block in a nice way. The general rules for such a configuration block are:

- The block starts with `-> TypeFix:` (add a separate `TypeFix` block for each type replacement);
- a series of commands using `IF`, `IF NOT`, `DELETE`, or `REPLACE` keywords, one per line;
- the block is terminated with the `END` keyword.

A fictional example would be

```
-> TypeFix:
   IF      journal, ^Misc$
   IF      type, ^Misc$
   IF      RefType, ^Misc$
   IF NOT  author
   DELETE  journal, title, author
   REPLACE RefType, @misc
   REPLACE url, howpublished
   REPLACE pages, cite-key
   REPLACE doi, note
END
```

Most of the above example is self-explanatory. The `^` and `$` are regular expressions referring to the beginning and end of the field content, respectively, thus matching the complete field. Mind the commas separating parameters of the instructions. If one of the given `IF` conditions is satisfied, the journal, author, and title fields are deleted, the reference type is replaced with `@misc`, the `url` field with `howpublished`, and so on. The `@` in a reference type is optional.

Three commands are available to configure a type fix:

- `IF`-conditions can be used to test for the existence of field or for particular values of fields and the reference type.
  - `IF` *field*, *regular expression*
    `IF NOT` *field*, *regular expression*
    Defines a condition that triggers the replacements. Multiple conditions can be given and the fix is performed if either one of them is satisfied. The regular expression is matched against the content of the given BibTeX field.
  - `IF` *field*
    `IF NOT` *field*
    Defines a condition to test whether or not a particular field exists in an entry.

    Using the `infile` keyword, it is possible toe provide multiple strings or regular expressions for a particular `TypeFix` block:
  - `IF` *field*, `infile(`*filename*`)`
    `IF NOT` *field*, `infile(`*filename*`)`
    The syntax of the file is the same as for the abbreviation files – one or more strings on each line, separated by tabs. In other words, the file is split into strings at line breaks and tabs.

The filename can be given relative or absolute, and the variable `$ScriptLocation` may be used to refer to the location of all other configuration files (and usually the script itself), for example

```
-> TypeFix:
   IF journal, infile($ScriptLocation/ConferenceJournals.cfg)
   REPLACE RefType, @inproceedings
END

-> TypeFix:
   IF journal, infile($ScriptLocation/booktitles.cfg)
   REPLACE RefType, @book
END
```

- `DELETE` *comma-separated field names*
  Defines one or a list of fields that are to be deleted from the item. In the above example, the command could have been split into three individual `DELETE` instructions with the same result.

- `REPLACE` *old field name*, *new field name*
  Replaces the name of a field with another one. If the old field exists already, then its value is replaced by the new field's content, thus deleting the old one – thou shalt not disable backup creation.

The keyword linked to the reference type is `RefType`, and, as can be seen in the example, this can be used in conditions as well as in replacement instructions. If a user-defined typefix for a particular reference type (e.g. `@book`) is found, a built-in fix for the same type is ignored.

Another important keyword is the `cite-key` field name. This is non-standard and only used by `bibtexformat`. If this field is present, the routine that creates the cite keys will use its value as new label for the entry. Non-standard fields are ignored by BibTeX.

The built-in type conversions given in Table 1, page 27 could be replaced (and changed to one's liking) with the following entries in the configuration file:

```
-> TypeFix:
   IF          affiliation, ^book$
   IF          type,        ^book$
   IF          RefType,     ^book$
   REPLACE     RefType,     @book
   REPLACE     volume,      address
   REPLACE     journal,     publisher
   REPLACE     number,      edition
END
```

```
-> TypeFix:
   IF          affiliation, ^inbook$
   IF          type,        ^inbook$
   IF          RefType,     ^inbook$
   REPLACE     RefType,     @inbook
   REPLACE     volume,      address
   REPLACE     journal,     publisher
   REPLACE     number,      edition
END
```

```
-> TypeFix:
   IF          journal, ^misc$
   IF          type,    ^misc$
   IF          RefType, ^misc$
   DELETE      journal
   REPLACE     RefType, @misc
   REPLACE     url,     howpublished
   REPLACE     pages,   cite-key
   REPLACE     doi,     note
END
```

```
-> TypeFix:
   IF          journal,      ^(Dissertation|Diplomarbeit|Thesis)
   REPLACE     RefType,      @phdthesis
   REPLACE     affiliation,  school
   REPLACE     journal,      type
END
```

```
-> TypeFix:
   IF           journal,     ^MastersThesis
   DELETE       journal
   REPLACE      RefType,     @mastersthesis
   REPLACE      affiliation, school
END
```

```
-> TypeFix:
   IF           journal,      ^TechReport
   DELETE       journal
   REPLACE      affiliation,  institution
   REPLACE      volume,       type
   REPLACE      pages,        address
END
```

```
-> TypeFix:
   IF           affiliation,  ^incollection
   DELETE       affiliation
   REPLACE      RefType,      @incollection
   REPLACE      volume,       booktitle
   REPLACE      journal,      publisher
   REPLACE      number,       address
   REPLACE      doi,          editor
END
```

```
-> TypeFix:
   IF           affiliation,  ^inproceedings
   DELETE       affiliation
   REPLACE      RefType,      @inproceedings
   REPLACE      journal,      booktitle
   REPLACE      number,       address
END
```

### 7.2.3 Check of Publication Types

Reference types generally have a defined set of allowed fields. Some are mandatory (e.g. `author` in an `@article`) and some are optional (e.g. `note` or `abstract`). The problem is that BibTeX does not necessarily complain if a certain field is missing, the information will just be left out of the bibliography, in the worst case with two commas and no text in between. If completeness is crucial, the `-typecheck` option can be used to perform a check on all entries for the required fields:

```
=> Checking reference type fields
   ------------------------------
    -> Required field is missing

       Falk:92:7080
       J. Am. Chem. Soc.
       Falk, Some Fake Title (year)
```

In the above example the `year` field is missing in the database entry.

## 7.3 Expanding Abbreviated Page Ranges

Some journals like to abbreviate page ranges, for example, 712 – 8 instead of 712 – 718. If the option `-rangefix` is given, the script automatically detects and corrects this. The user is informed about that and all changes are printed to be able to easily check the corrections:

```
=> Correcting page ranges
   ----------------------
   The following page rages have been expanded, please check the
   corrections:

      2119-34  =>    2119-2134    Allingham, 2006
       235-42  =>      235-242    Berman, 2000
       325-32  =>      325-332    Cole, 2005
      1481-92  =>    1481-1492    Forli, 2007
      2308-18  =>    2308-2318    de Graaf, 2005
```

## 7.4 Fixing Ahead-of-Print Publications

If journal articles are first published online they are usually not assigned the final page number for the full citation. The DOI (Digital Object Identifier)[10] has to be available though and will never change for this particular publication. To be able to cite the publication, it is possible to replace any field value with the DOI.

There is no field that is generally suited best for this replacement, this depends on the used BibTeX style file. Therefore, there may be some experimenting involved to find a suitable field.

The checks and replacements are enabled with the `-aopfix` option. If this is given, the configuration file is parsed for `-> AOPFix` blocks. Each block can contain one or more conditions that have to be all satisfied for the changes to be carried out. The same instructions can be used that have been introduced for the reference type corrections (see Section 7.2.2, page 28).

Two sets of instructions are predefined in the default configuration file. The first one checks for `pages` fields that contain *no* or *no-no* (this 'format' is used, for example, by Wiley for AOP articles):

```
-> AOPFix:
   # if the reference type is @article or @inproceedings
   IF      RefType, @article|@inproceedings
   # and the pages field is no, no - no, no-no, no--no, etc.
   IF      pages, no(\s*-+\s*no)?
   # and a doi field is present
   IF      doi
   # replace the field name "doi" with "pages" (the old pages content is deleted)
   REPLACE doi, pages
END
```

The second set tests whether a `pages` field is given at all. If a `doi` field is found, its field name is renamed:

```
-> AOPFix:
   IF      RefType, @article
   IF      doi
   IF NOT  pages
   REPLACE doi, pages
END
```

Note that *all* conditions of a block have to be met in order to effect the changes ('and' combination) – in the type replacements, only one condition needs to be satisfied ('or' combination). The two instruction blocks shown above facilitate the fixing of articles with different sets of conditions. There is no maximum number of `AOPFix` instructions; by including a condition for the reference type it is even possible to define a set for each type individually.

## 8 Journal Abbreviations

The possibility to replace the full journal title with the abbreviated one was implemented using EndNote's *term lists*. These are tab-separated text files, widely available on the web for every subject and regularly updated. They contain the full journal title, the standard abbreviation using periods and usually a second abbreviation without periods.

```
Full Journal Title      Abbrev. 1      Abbrev 2
```

The first abbreviation is taken by default but the command line parameters `-full`, `-abb1` and `-abb2` allow to explicitly select one of the versions. In general, the syntax to request abbreviating the journal title is

```
bibtexformat Library.bib -abb abbreviations.txt
```

where the file `abbreviations.txt` contains the term list as shown before and abbreviation 1 is then taken by default.

The term list files can be given to the script via the option that also enables the abbreviation and takes one or more file names as parameter: `-abb abbrev1.txt abbrev2.txt`. Since often journals from different subjects appear in a bibliography (e.g. chemistry and biology), more than one term list can be given. As the term lists do not tend to change often for a particular user, they can be defined in the configuration file and do not need to be entered any more via the command line:

```
my @AbbFiles = ("$ENV{HOME}/bin/abbreviations-1.txt",
                "$ENV{HOME}/bin/abbreviations-2.txt");
```

Note that in order to request the title abbreviation, one of the abbreviation options (e.g. `-abb1`) has to be set to trigger the processing.

The given term lists are processed in the order given to the script starting with the command line files and then the ones defined via `@AbbFiles`. The processing of a reference is stopped when a match is found, meaning that the command line has a higher precedence than `@AbbFiles`.

## 8.1 Multiple Abbreviation Possibilities

As indicated before, the EndNote term lists may contain two different abbreviations for each title (usually with and without dots). By default `bibtexformat` uses the first abbreviation. In order to use the second possibility instead, the `-abb2` options has to be given:

```
bibtexformat Library.bib -abb abbreviations.txt -abb2
```

The option `-abb1` exists as well for consistency, however, it is not really needed because it is the default. In order to use the full title instead and get rid of all abbreviated titles, the `-full` options can be used.

## 8.2 Unknown Abbreviations and Warnings

The first warning that may be issued concerns references, which do not contain a journal entry at all. If no abbreviation is found for a journal, a warning is issued and the reference in question displayed. Most of these "errors" are due to typos or different spelling (e.g. "and" vs. "&") and this function, therefore, helps to be consistent throughout the whole library.

The other journal titles, which are not due to unknown typos, are usually rather uncommon journals (e.g. very old) or from a different subject and therefore not contained in the "loaded" term lists. The best thing to do is to create a custom term list with all the titles not contained in the other lists. This way it is possible to update the term lists without losing custom abbreviations.

If the second abbreviation was selected via the `-abb2` option but no second possibility was found in the term list, the script does *not* automatically take the first one since this would lead to inconsistent citations. Instead, a warning is given and all journals with no second abbreviation listed.

In summary, the following warnings and lists are produced:

- entries not containing a journal title;
- references, for which no abbreviation was found;
- a summary of the journal titles, for which no abbreviation was found;
- a summary of all journal titles, for which no second abbreviation was found (only if the second possibility was requested via `-abb2`).

```
Summary of all journals with unknown abbreviation:
--------------------------------------------------

Aaps J
Am J Physiol-Heart C
Bioorganic {&} Medicinal Chemistry
Biosensors and Bioelectronics
Chem. Asian J.
Chemometr Intell Lab
Curr Opin Biotech
Mol Divers
Proc Natl Acad Sci USA
R{&}D Mag
Tetrahedron-Asymmetr
The AAPS journal
```

## 9 Links to External Files

Two BibTeX fields are used by programs to link to external files (for example one or more PDFs):

- `local-url`
  This field allows a single link in valid URL format, that is, all spaces replaced with `%20` and starting with `file://localhost/`. `local-url` is used by Papers.

- `file`
  This allows links to multiple files. The paths are usually given as absolute system paths, for example, starting with `/Users/user/Documents/`. A comment can be given for each file and the file type be defined, separated by colons and each file definition finished with a semicolon:
  `Comment:/Full/path/.../paper.pdf:PDF;`

Since this feature changes data in the library it is strongly recommended to write to an output file instead of replacing the input file. To minimize data loss such as the comment in the `file` field when converting to `local-url`, the original field is not removed. If the removal is desired, it can be facilitated by filtering out the respective field (section 5.4). Conversion in both ways may be performed by different configuration files, which remove either `local-url` or `file` after the replacement (section 2.3).

### 9.1 Checking External Files for Existence

If the `-filecheck` option is provided, the script will check all files linked to via `file` and `local-url`. If the filenames are given with relative paths, it is possible to provide a directory in which the files are stored. In JabRef, for example, the user may define a main directory for external files and all paths are then saved relatively to that folder. The same can be done using the `-filedir` parameter and the given value will be added before the path read from the library. For example, if the following paths are given in the library together with `-filecheck -filedir /Users/fred/Papers`:

```
Bioinformatics/John and Jane Doe, 2005, 4, 126.pdf
→ /Users/fred/Papers/Bioinformatics/John and Jane Doe, 2005, 4, 126.pdf
```

The setting for `filedir` can be predefined in the configuration file:

```
# Absolute or relative path to the directory containing files linked to
# in the file and/or local-url fields.
# -filecheck adds this value before the paths given in the library
$FileDir = "";
```

### 9.2 Converting `local-url` → `file` (Papers to JabRef)

This option converts
    `local-url = { file.pdf }`
to
    `file = { :file.pdf:PDF }`.

The `local-url` field usually contains an absolute path of the form `file://localhost/Users/fred/article.pdf` with spaces encoded as `%20`. The syntax of the `file` entry is *comment*:*filename*:*type* and may contain multiple entries separated by semicolons. The paths to files given in the `file` field are either absolute (`/Users/fred/article.pdf`) or relative (`article.pdf` or `papers/article.pdf`). In order to convert between both formats, the path has to be adapted. To keep this conversion as flexible as possible, two variables in the configuration file hold the differences between the two formats:

```
$LocalURL = "file://localhost/Volumes/Home/username/";
$FileURL  = "/Users/username/";
```

The URL is converted using the substitutions defined in the configuration file (`s/$LocalURL/$FileURL/`) and all occurrences of `%20` are replaced with blanks. The comment entry (before the first colon) is left empty and the extension of the linked file in uppercase letters is used as the file type. If a `file` field is already present, its contents are taken into account:

- If the file linked to via `local-url` is found, no replacement is carried out (in order to preserve a possible comment in the entry).
- If the file is not yet linked then it is added as first one to the `file` entry.

### 9.3 Converting `file` → `local-url` (JabRef to Papers)

If multiple files are found in the `file` field, then either the first PDF file is used or the first file, if no PDF link is found. The substitutions defined in the configuration file (`s/$FileURL/$LocalURL/`) are performed and all blanks are replaced with `%20`. If a `local-url` field is found and the linked file differs from the one read from the `file` field, it is overwritten and a warning issued, stating the deleted link.

## 10 The Perl Library `ParseBibTeX.pm`

The functions concerning BibTeX libraries in general have been combined in the Perl library `ParseBibTeX.pm`. Using its subroutines, a BibTeX file can be parsed with a single command and the content is returned in a Perl data structure that can be easily used in loops, changed or amended. Another routine can then be used to write the changed structure to a BibTeX file. For each routine there are several examples of its usage in `bibtexformat`, this Section only gives a short summary of all functions.

## 10.1 Routines Concerning the Entire Library

`ParseBibTeX`

**Syntax:** `ParseBibTeX (` `$Library,` (reference to an array) `$Content` (reference to an array or a filename) `);`

This routine parses the content of a BibTeX database. `$Content` can either be a string with the filename (with or without a `.bib` extension or a reference to an array with the content of the file. The routine returns the result in `$Library`. The latter is a reference to an array in which each item represents a BibTeX entry. To process each entry one can use a `foreach` loop:

```
foreach $Item ( @{$Library} ) {  ...  }
```

Each item read from the BibTeX database is a hash reference with two main hash keys:

- `Entry`
  An array that contains the complete entry almost as it was read from the database. The first line contains the type and label (e.g. `@type{label,`) and the last line the final curly brace. If the label was originally in the second line, it is combined with the first one. Similarly, if the last brace was in one line with the final field (e.g. `pages = {12} }`), it is forced into its own line. All other array items contain a complete field of the entry. If a field was spread over multiple lines, these line feeds are preserved (`\n` is added to the strings).
- `Fields`
  This contains basically the same information as the `Entry` array in form of a hash. Each field read from the database is split into the field name (e.g. `author`, used as hash key) and the field content (used as hash value). The field name is always formatted lowercase.

When `bibtexformat` handles `$Library` it takes care to keep the `Entry` and `Fields` items synchronized.

Abbreviated example of one reference:

```
Fields =>{
          'author' => 'Eefei Chen and Robert A Goldbeck and David S Kliger',
          'title' => 'The Earliest Events in Protein Folding: ...',
          'journal' => 'J. Am. Chem. Soc.',
          'year' => '2004'
          'volume' => '126',
          'pages' => '11175--11181',
        };
Entry =>[
          '@article{Chen:04:11175,',
          '   author  = {Eefei Chen and Robert A Goldbeck and David S Kliger},',
          '   title   = {The Earliest Events in Protein Folding: ... },',
          '   journal = {J. Am. Chem. Soc.},',
          '   year    = {2004},',
          '   volume  = {126},',
          '   pages   = {11175--11181},',
          '}'
        ];
RefType     => 'article';
FirstAuthor => 'Chen';
StartPage   => 11175';
EndPage     => '11181';
Label       => 'Chen:04:11175';
```

```
  Year         => '2004';
  ShortYear    => '04';
  AuthorList   => [ 'Eefei Chen',
                    'Robert A Goldbeck',
                    'David S Kliger' ];
```

`WriteBibTeX`

**Syntax:** `WriteBibTeX (` `$Library,` (reference to an array) `$OutFile,` (string) `$NewLines` (integer) `);`

This routine takes an array reference as generated by `ParseBibTeX` and writes it to the file `$OutFile`. The file is overwritten without notification, such user interaction has to be taken care of beforehand. `$NewLines` defines the number of empty lines between the BibTeX items and is set to 2 if it is not given to the routine.

## 10.2 Parsing Individual Fields

The following routines are provided for fields that need further parsing than just the splitting in field name and field content. Usually, the additional information is added to the `$Item` hash.

`ParseAuthor`

**Syntax:** `ParseAuthor (` `$Item,` (reference to a hash) `$Line` (string) `);`

Splits up the individual authors in the `author` field. The authors (in the format as read from the database) are added as array to `$Item->{AuthorList}`. The last name of the first author is removed spaces and BibTeX codes and saved to `$Item->{FirstAuthor}`.

`ParseTitle`

**Syntax:** `ParseTitle (` `$Item,` (reference to a hash) `$Line` (string) `);`

This routine does not add additional entries to `$Item` but is used to detect an additional brace level around the title.

`ParsePages`

**Syntax:** `ParsePages (` `$Item` (reference to a hash) `$Line` (string) `);`

Splits up the `pages` field and adds `$Item->{FirstPage}` and `$Item->{LastPage}`.

`ParseYear`

**Syntax:** `ParseYear (` `$Item,` (reference to a hash) `$Line` (string) `);`

Adds `$Item->{Year}` (the year as read from the database) and `$Item->{ShortYear}` (using only the last two digits).

`ParseLocalURL`

**Syntax:** `ParseLocalURL (` `$Item,` (reference to a hash) `$Line` (string) `);`

This adds the link in the `local-url` field to `$Item->{LocalURL}`.

`ParseFile`

**Syntax:** `ParseFile (` `$Item,` (reference to a hash) `$Line` (string) `);`

Each `file` entry can contain links to multiple files, separated by semicolons. Each link contains a comment, the actual link and the file type, separated by colons. The line is fully parsed and saved to the array `$Item->{File}`:

```
File[0]  =  Comment => {...}
            URL     => {...}
            Type    => {...}
...
```

## 10.3 Helper Routines

Some routines, which are frequently needed when dealing with BibTeX databases, are also exported by `ParseBibTeX.pm`.

`FieldName`

**Syntax:** `$Value =` `FieldName (` `$Line` (string) `);`

Returns the name of the BibTeX field contained in `$Line`. This is everything before the first equal sign, all blanks are removed.

`FieldContent`

**Syntax:** `$Value =` `FieldContent (` `$Line` (string) `);`

Returns the content of a BibTeX field. This is everything after the first equal sign between the following and the last delimiter (curly braces or double quotes). If there is no delimiter, everything between the first equal sign and the last comma of the entry is returned.

`GetDelimiter`

**Syntax:** `$Value =` `GetDelimiter (` `$Line,` (string) `$Side,` (string) `$Code` (integer) `);`

Determines whether curly braces or double quotes are used as delimiters of a field. If double quotes are used, the return value is always `"`, otherwise it depends on the value of `$Side`:

- `$Side` is not given or “left”: Return value is “`{`”
- `$Side` is “right”: Return value is “`}`”

For fields without delimiters an empty string is returned.

`$Code` is a string that is printed with the error message in case the delimiter could not be determined. It should be the name of the routine calling `GetDelimiter` and helps then to trace an error back to the routine it originated from. `$Side` and `$Code` are optional.

## `ReplaceType`

| | | | | | |
|---|---|---|---|---|---|
| **Syntax:** | `ReplaceType (` | `$Item,` | `$Old,` | `$New` | `);` |
| | | reference to a hash | string | string | |

Replaces field names in the `Entry` and `Field` hash items. This is used to convert BibTEX types, for example, to turn an `@article` into a `@book`. To rename the actual type name, `$Old` has to be named `RefType` since the original type name may be unknown. Example:

```
ReplaceType ($Item, "RefType", "\@book");
ReplaceType ($Item, "volume",  "address");
ReplaceType ($Item, "journal", "publisher");
ReplaceType ($Item, "number",  "edition");
```

## `DeleteField`

| | | | | |
|---|---|---|---|---|
| **Syntax:** | `DeleteField (` | `$Item,` | `$Field` | `);` |
| | | reference to a hash | string | |

Deletes the field `$Field` in the `Entry` and `Fields` hashes.

## `GetIndex`

| | | | | |
|---|---|---|---|---|
| **Syntax:** | `GetIndex (` | `$Item`, | `$Field` | `);` |
| | | reference to a hash | string | |

Returns the array index of the field `$Field` in the `Entry` array.

## References

[1] Alexander Griekspoor and Tom Groothuis. Papers.
http://mekentosj.com/papers.

[2] Michael O. McCracken. BibDesk.
http://bibdesk.sourceforge.net.

[3] Morten O. Alver, Nizar N. Batada, Michel Baylac, Kolja Brix, Frederic Darboux, Guillaume Gardey, Cyrille d'Haese, S M Mahbub Murshed, Raik Nagel, Christopher Oezbek, Ellen Reitmayr, Gert Renckens, Andreas Rudert, Michael Spiegel, Ulrik Stervbo, Dominik Wasenhoven, Joerg K. Wegner, Michael Wrighton, Egon Willighagen, and Jorg Zieren. JabRef.
http://jabref.sourceforge.net.

[4] Thomson Reuters. Endnote.
http://www.endnote.com.

[5] ActivePerl.
http://www.activeperl.com.

[6] Nicolas Markey. Tame the BeaST – The B to X of BibTEX.
ftp://ftp.tex.ac.uk/tex-archive/info/bibtex/tamethebeast/ttb_en.pdf.

[7] Michael Shell and David Hoadley. BibTEX Tips and FAQ.
ftp://ftp.tex.ac.uk/tex-archive/biblio/bibtex/contrib/doc/btxFAQ.pdf.

[8] Oren Patashnik. BibTEXing.
http://bibtexml.sourceforge.net/btxdoc.pdf.

[9] Patrick W. Daly. The custom-bib package.
http://www.ctan.org/tex-archive/macros/latex/contrib/custom-bib/.

[10] Digital Object Identifier (DOI).
http://www.doi.org/.

## Glossary

- **BibTEX type, reference type, publication type**
  These descriptors are used synonymously for the type of a reference, for example, `@article` or `@book`.

- **BibTEX field**
  Each reference contains a number of mandatory and optional fields, for example, `author` or `journal`.

- **Label, cite(ation) key**
  These descriptors are used synonymously for the unique key of each reference that is used to cite it: `@article{CiteKey, ...`.

## Index