

notes2bib — Integrating notes into the bibliography*

Joseph Wright[†]

Released 2008/07/22

Abstract

The `notes2bib` package defines a new type of note, `\bibnote`, which will always be added to the bibliography. The package allows footnotes and endnotes to be moved into the bibliography in the same way. The package can be used with `natbib` and `biblatex` as well as plain \LaTeX citations. Both sorted and unsorted bibliography styles are supported.

Contents

1	Background	1	6.2	The $\text{\texttt{REVTeX}}$ approach	5
2	Basic use	2	6.3	The <code>notes2bib</code> approach	6
2.1	Package control	2	7	The package code	6
2.2	Output of notes	3	7.1	Package setup code . . .	6
2.3	Cross-referencing notes	4	7.2	Logging	7
2.4	Interaction with other packages	4	7.3	Package options	7
3	Special effects	4	7.4	Footnote and endnote handling	10
4	Package requirements	5	7.5	User macros	12
5	Known issues	5	7.6	Internal macros	13
6	The mechanism	5	7.7	Finalisation	18
6.1	The <code>endnotes</code> approach	5	8	Notes	20
			9	Change History	20
			10	Index	22

1 Background

In most subject areas, bibliographic citations and notes are separate entities. However, in some parts of the physical sciences (chemistry and physics) it is usual for references to the literature and notes to be given together in a “References and Notes” section. By default, this requires that $\text{\texttt{BIBTEX}}$ users create a notes database for each document that they write.

*This file describes version v1.4a, last revised 2008/07/22.

[†]E-mail: joseph.wright@morningstar2.co.uk

The `endnotes` package allows the user to create endnotes rather than footnotes. However, this does not place the notes in the bibliography. The `APS` have developed the `REVTeX` document class, which allows footnotes and endnotes to be added to the bibliography. Notes can only be placed at the end of the bibliography using this system. Furthermore, the code to achieve this effect is not available as a package separate from `REVTeX`.

The aim of the `notes2bib` package is to make integration of notes into the bibliography easy. Notes can be written as normal in the `LaTeX` source, and are automatically moved to the bibliography. The package is compatible with sorted and unsorted bibliography styles. The package has been designed for use with numerical citations, although it will work with other systems.

2 Basic use

`\bibnote` In the most basic form, the package can be used simply by loading it in the preamble as normal. This adds a new type of note to the existing `\footnote` type: `\bibnote{<text>}`. This can be used in exactly the same way as a footnote, taking one mandatory argument `<text>`. The `<text>` will be made available as to the bibliography as a note (henceforth referred to as a `bibnote`).

A very simple example of a bibliography note [1]. `\bibnote{Note for the first example}.`

By default, each `bibnote` is given an automatically-generated label. However, `\bibnote` accepts an optional argument `<label>`, which can be used to over-ride this. This is particularly useful when a note will be referenced several times (The use of the `\citenote` command is covered in Section 2.3).

An example of a named note [2]. The text can then continue and reference the note again later [2]. `\bibnote[labelled-note]{Note for the second example}.` The text can then continue and reference the note again later `\citenote{labelled-note}.`

`\bibnotemark` In common with `\footnote`, the basic `\bibnote` macro has companion macros `\bibnotemark` and `\bibnotetext`. The text provided for each not is not “fragile,” and so it should not be necessary to use `\bibnotemark` directly. It is needed when replacing footnotes by `bibnotes`. Notice that there *are* places where `bibnotes` will be problematic, for example in section headings which also appear in the Table of Contents. In these contexts, use `\citenote` to reference the note, or use an optional argument to the `\section`, *etc*.

It is hard to write a good example for this [3]! The text continues here. `\bibnotemark!` The text continues here `\bibnotetext{Note for the third example}.`

2.1 Package control

`\niibsetup` The `notes2bib` package can be controlled using package options, and also dynamically using the `\niibsetup` macro. In both cases the same list of `keyval` options are recognised, in a similar manner to the `graphicx` or `hyperref` packages. Several

	of the package options are aimed at controlling the package internally, but by providing a single macro to control this, use is made easier. ¹
footnote	Some options control the general behaviour of <code>notes2bib</code> in the body of the \LaTeX source. The <code>footnote</code> and <code>endnotes</code> options control whether <code>\footnote</code> and <code>\endnote</code> macros are converted into bibnotes. Both are Boolean options, and are false by default. The citation command used by <code>notes2bib</code> to insert bibnotes. It should be the name of a \LaTeX command (a “ <code>csname</code> ”), and is set to cite on loading <code>notes2bib</code> ; this means that <code>\cite</code> will be used as the citation command.
endnotes	
cite	
field	A number of options control the data added to the \BibTeX database. The <code>field</code> and <code>record</code> options control the type of \BibTeX entry created by <code>notes2bib</code> . On loading, <code>record</code> is set to Misc and <code>field</code> is set as note . Depending on the \BibTeX style in use, better choices may exist for these settings. The <code>name</code> option is used to automatically generate citation names. The option starts with the value Bibnote , which may need to change for author–year styles in particular. The name of the database itself is controlled by the <code>prefix</code> option. This contains the “marker” used by <code>notes2bib</code> to attach to the job name when creating the storage database. The default is niib- .
name	
record	
prefix	Bibnotes can be created so that they will be sorted before or after normal citations. A list of values are recognised: none (no control of sorting), <code>head</code> (notes appear before real citations) and <code>tail</code> (notes appear after real citations). The shortcut options <code>head</code> and <code>tail</code> are also available. A number of mechanisms are used to ensure correct sorting of bibnotes. For normal \BibTeX users, the options <code>keyhead</code> , <code>keynone</code> and <code>keytail</code> are used to control sorting. These values are added to the start of the citation name in the <code>key</code> field, which controls sorting. The default values are aaa , nothing and zzz , respectively. For <code>biblatex</code> users, control is made using the <code>presort</code> system made available there. The <code>notes2bib</code> options <code>presorthead</code> , <code>presortnone</code> and <code>presorttail</code> set up the appropriate values; default values are m1 , mm and m1 , respectively.
sort	
head	
tail	The amount of detail to add to the log; expects a value from the list <code>debug</code> (very detailed information), <code>verbose</code> (the same as <code>debug</code>), normal , <code>errors</code> (errors only), <code>none</code> (what it says). As a shortcut, the <code>debug</code> option is provided as an alias to <code>log=debug</code> . The package has a single load-time only option, <code>etex</code> . This is a Boolean switch, and determines whether $\epsilon\text{-TeX}$ extensions are used if available. This is true by default; the intension here is for testing when sending to publishers, <i>etc.</i> , where $\epsilon\text{-TeX}$ may be an issue.
keyhead	
keynone	
keytail	
presorthead	
presortnone	
presorttail	
log	
debug	
etex	

2.2 Output of notes

Bibnotes are only printed when a bibliography is created. This means that at the very least a `\bibliographystyle` command must appear in the source.² Under most circumstances, the user will be citing literature, and so will also include a `\bibliography` command in their source. Bibliography notes are automatically added to the citations to be printed.

`\printbibnotes` If bibnotes are being used without any other citations, then the user cannot place `\bibliography` in the source.³ The package therefore provides the macro

¹Users upgrading from earlier versions of `notes2bib` will note that the large number of control macros have all been removed from v1.3.

²For `biblatex` users, the package must be loaded!

³ \LaTeX will complain if the user puts `\bibliography{}`.

`\printbibnotes`, which will output only the notes. If the `endnotes` package has been loaded, the `\theendnotes` macro is redefined to achieve the same effect.

2.3 Cross-referencing notes

`\citenote` As explained above, each note is automatically assigned a label, or the user can provide one as an optional argument to the note. In either case, notes may then be cross-referenced. Notes are available to be cited directly using the `\cite` command. However, this can cause problems when using the `sort=tail` option. The `\citenote` command is therefore provided. This is aware of the options, and will act correctly in all circumstances.

Cross-references to the note labelled earlier using `[2]` and using `[2]`.

Cross-references to the note labelled earlier using `\cite{labelled-note}` and using `\citenote{labelled-note}`.

2.4 Interaction with other packages

`notes2bib` is designed to work well with as many other packages as possible. It has been tested with `cite`, `natbib`, `hyperref` and `mciteplus` with no problems. The `notes2bib` package is compatible with the current release of `biblatex` (v0.7); older versions of `biblatex` may or may not work.⁴

3 Special effects

`\flushnotestack` When using the `sort=tail` option, citations are added to a stack as they are made. This stack is then flushed to the `.aux` file at the end of the document. If references are given by chapter (or other unit), this may not give the desired effect. The `\flushnotestack` macro will cause all saved citations to be written at that point, and will reset the stack for continued use. This can therefore be used to control when citation occurs.⁵

`\thebibnote` If a sorted bibliography style is in use, and more than nine notes are created, the sort order will be incorrect. This is because by default `notes2bib` does not pad the automatically-created labels with zeros. To get the correct sort order, `\thebibnote` should be redefined.

```
\makeatletter
\renewcommand*{\thebibnote}{%
  \niib@name%
  \ifnum\value{bibnote} < 9 0\fi%
  \the\value{bibnote}}
\makeatother
```

⁴As `biblatex` is experimental and is not currently added to T_EX distributions, users have little excuse for not using the latest release.

⁵This macro was called `\flushcitestack` prior to v1.3.

4 Package requirements

The package has only one requirement, `xkeyval` version 2.5 or later. ϵ -TeX is used if available, but is not a requirement.

5 Known issues

From v1.1, the method for writing notes to the BibTeX database has been modified. This means that bibnotes cannot contain verbatim text.⁶ This is the same as for normal footnotes, and so the usual work-arounds are applicable.

The next note contains some awkward text [4].

```
The next note contains some awkward text
\bibnote{Some \texttt{\textbackslash verb}-like output}.
```

The package relies on BibTeX being able to open and process the temporary database containing the note text. The name of this file contains `\jobname`, the name of the main L^AT_EX file being processed. This must consist only of characters that BibTeX can handle. In particular, spaces in the file name will lead to problems.

6 The mechanism

The mechanism for positioning notes in the bibliography is somewhat involved. Rather than expect interested users to read all of the code that follows, a condensed overview is given here. The thinking behind the system used is explained first, by considering the `endnotes` package and REVTeX class. Both of these provided inspiration for this package.

6.1 The endnotes approach

The `endnotes` package⁷ allows the user to generate endnotes in the same way as footnotes. In `endnotes`, the text of the note is written to a `.ent` file. This is achieved in an unexpanded form using the `\meaning` T_EX primitive. To produce the list of endnotes, this file is read back into L^AT_EX, with the extra information `\meaning` also writes being stripped off in the process.

This method is relatively simple in concept, but obviously does not integrate with BibTeX. The use of `\meaning` for unexpanded output also means that information requires further processing before it can be included in the bibliography.

6.2 The REVTeX approach

REVTeX takes a similar approach to creating endnotes, but also allows footnotes to be converted into endnotes. This results in a file containing all of the non-literature citations in a document in a single external file (in this case a `.end` file). REVTeX also uses a different method to achieve unexpanded output, meaning that several macros are not “active” in notes.

⁶Actually, they can, but the spacing will go wrong. L^AT_EX will only complain if a note ends with verbatim text. However, verbatim text is not supported in bibnotes: don’t do it!

⁷<http://tug.ctan.org/macros/latex/contrib/misc/endnotes.sty>

The second part of the REVTeX approach is to (optionally) read the notes back into the document. This is achieved by modifying the `\bibliography` environment to output each note in the bibliography. This takes place *en masse*, after the normal citations.

REVTeX makes a number of modifications to L^AT_EX, and is dependent on using natbib. The method used is also not compatible with interspersing normal citations and notes.

6.3 The notes2bib approach

In notes2bib, notes are again written to an external file. However, in contrast to the methods those outlined above, notes2bib writes its output in the well-known BibTeX database format. All of the note text is written almost completely unexpanded to the file, the only requirement being that the braces match within the argument.⁸

Each note results in a citation being placed by notes2bib in the `.aux` file. The `\bibliography` command is also modified so that the new database will be used by BibTeX. After the BibTeX run, the note text will appear in the `.bbl` file, in the same way as any other citation. Using an unsorted BibTeX style, this results in notes interspaced with the normal citations. For sorted styles, notes2bib allows various methods for controlling the placement of notes, based on writing appropriate fields in the BibTeX database.

`niibheadcite` The `sort=head` option works by adding an additional macro to the `.aux` file: `\niibheadcite`. On the next L^AT_EX run, this causes the relevant notes to be cited right at the beginning of the document, before any real citations.⁹ For `sort=tail`, the value of `\if@files` is temporarily altered to prevent writing of a citation to the `.aux` file. The citation is added to a stack, and is written at the end of the document.

7 The package code

7.1 Package setup code

```
\niib@id The package starts with the usual identification code and support loading.
1 \NeedsTeXFormat{LaTeX2e}
2 \def\niib@id$#1: #2.#3 #4 #5-#6-#7 #8 #9${%
3   #5/#6/#7\space v1.4a\space}
4 \ProvidesPackage{notes2bib}
5 [\niib@id$Id: notes2bib.dtx 13 2008-07-22 20:06:39Z joseph $
6   Integrating notes into the bibliography]
7 \RequirePackage{xkeyval}[2005/05/07]

\niib@tempa Some private temporary macros are declared.
\niib@tempb 8 \newcommand*\niib@tempa{ }
9 \newcommand*\niib@tempb{ }
```

⁸Writing to the file uses the ϵ -TeX `\unexpanded` primitive.

⁹Thanks to Michael Shell for the idea for this method.

7.2 Logging

`\ifniib@debug` To control logging, some new switches are declared.
`\ifniib@logmin` 10 `\newif\ifniib@debug`
`\ifniib@lognone` 11 `\newif\ifniib@logmin`
12 `\newif\ifniib@lognone`

`\niib@log@err` Some handy re-usable macros are defined here. These all take names beginning
`\niib@log@warn` These pop up in various places. First errors, warnings and information are
`\niib@log@inf` handled. Package options are used to control how much output is given.

```
13 \newcommand*{\niib@log@err}[2]{%
14   \ifniib@lognone\else
15     \ifniib@logmin
16       \PackageWarning{notes2bib}{#1}%
17     \else
18       \PackageError{notes2bib}{#1}{#2}%
19     \fi
20   \fi}
21 \newcommand*{\niib@log@warn}[1]{%
22   \ifniib@lognone\else
23     \ifniib@logmin\else
24       \PackageWarning{notes2bib}{#1}%
25     \fi
26   \fi}
27 \newcommand*{\niib@log@inf}[1]{%
28   \ifniib@lognone\else
29     \ifniib@logmin\else
30       \PackageInfo{notes2bib}{#1}%
31     \fi
32   \fi}
```

`\niib@log@debug` The debug macro only gives output if the appropriate package option is set.

```
33 \newcommand*{\niib@log@debug}[1]{%
34   \ifniib@lognone\else
35     \ifniib@debug
36       \PackageInfo{notes2bib}{#1}%
37     \fi
38   \fi}
```

7.3 Package options

`\niib@opt@boolkey` To aid maintenance, some shortcuts are defined for generating keys. These also
allow the debugging messages to be added automatically to every key.

```
39 \newcommand*{\niib@opt@boolkey}[2][true]{%
40   \define@boolkey[niib]{opt}{#2}[true]
41   {#1\niib@log@debug{Option #2 set to ##1}}}
```

`\niib@opt@choicekey` A “fill in the blanks” choice key. In all cases, `\niib@tempa` is used to hold the
value given to the key, so that `\ifx` testing can occur.

```
42 \newcommand*{\niib@opt@choicekey}[5][true]{%
43   \define@choicekey*+[niib]{opt}{#2}[\niib@tempa]{#3}[#1]
44   {#4\niib@log@debug{Option #2 set to ##1}}
45   {#5\niib@log@debug{Option #2 set to ##1}}}
```

`\niib@opt@cmdkeys` A shortcut for `xkeyval` command keys.

```

46 \newcommand*{\niib@opt@cmdkeys}[1]{%
47   \define@cmdkeys[niib]{opt}{niib@}{#1}}

```

`\niibsetup` To allow modification of options at run time, a setup macro is provided. The run of strange tests are to prevent problems in arrays and the like.

```

48 \newcommand*{\niibsetup}{\setkeys[niib]{opt}}

```

The `xkeyval` package option for logging is declared. This is then processed to set the switches correctly.

```

49 \niib@opt@choicekey[normal]{log}
50   {debug,verbose,normal,errors,none}

```

A series of comparisons are made to assign the logging mode. The `normal` option is not tested, as executing the option sets the switches appropriately.

```

51   {\niib@debugfalse
52     \niib@logminfalse
53     \niib@lognonefalse
54     \renewcommand*{\niib@tempb}{none}%
55     \ifx\niib@tempa\niib@tempb
56       \niib@lognonetrue
57     \fi
58     \renewcommand*{\niib@tempb}{minimal}%
59     \ifx\niib@tempa\niib@tempb
60       \niib@logmintrue
61     \fi
62     \renewcommand*{\niib@tempb}{debug}%
63     \ifx\niib@tempa\niib@tempb
64       \niib@debugtrue
65     \fi
66     \renewcommand*{\niib@tempb}{verbose}%
67     \ifx\niib@tempa\niib@tempb
68       \niib@debugtrue
69     \fi}

```

The option has not been recognised: give a warning (if appropriate).

```

70   {\niib@log@warn{Unrecognised value '#1' for option log}}

```

`\niib@opt@debug` A quick method to set `log=debug`.

```

71 \niib@opt@boolkey{debug}

```

`\ifniib@footnotes` The footnote and endnote options are declared here.

`\ifniib@endnotes`

```

72 \niib@opt@boolkey[\niib@swapfoot]{footnotes}
73 \niib@opt@boolkey[\niib@swapend]{endnotes}

```

`\ifniib@head` Switches are needed for placing notes before and after normal citations.

`\ifniib@tail`

```

74 \newif\ifniib@tail
75 \newif\ifniib@head

```

This option controls the position of notes *versus* normal citations. The `xkeyval` option replaces the earlier `head` and `tail` options, which are retained for backward compatibility.

```

76 \niib@opt@choicekey[none]{sort}{none,head,tail}

```



```

77  {\niib@headfalse
78  \niib@tailfalse
79  \renewcommand*{\niib@tempb}{head}%
80  \ifx\niib@tempa\niib@tempb
81    \niib@headtrue
82  \fi
83  \renewcommand*{\niib@tempb}{tail}%
84  \ifx\niib@tempa\niib@tempb
85    \niib@tailtrue
86  \fi}
87  {\niib@log@warn{Unrecognised value '#1' for option sort}}

```

The back-compatibility code; unlike earlier versions, this will take whatever the sort-type key is given.

```

88 \niib@opt@boolkey[%
89   \ifniib@head
90     \ifniib@tail
91       \niib@tailfalse
92       \niib@log@inf{Option head cancels existing\MessageBreak
93         tail or sort=tail option}
94     \fi
95   \fi]{head}
96 \niib@opt@boolkey[%
97   \ifniib@tail
98     \ifniib@head
99       \niib@headfalse
100      \niib@log@inf{Option tail cancels existing\MessageBreak
101        head or sort=head option}
102    \fi
103  \fi]{tail}

```

\niib@cite The various internal control values are set up as command keys.

```

\niib@name 104 \niib@opt@cmdkeys{%
\niib@prefix 105 cite,
\niib@record 106 name,
\niib@field 107 prefix,
\niib@presorthead 108 record,
\niib@presortnone 109 field,
\niib@presorttail 110 presorthead,
\niib@keyhead 111 presortnone,
\niib@keynone 112 presorttail,
\niib@keytail 113 keyhead,
114 keynone,
115 keytail}
116 \niibsetup{
117 cite=cite,
118 name=Bibnote,
119 prefix=niib-,
120 record=Misc,
121 field=note,
122 presorthead=ml,
123 presortnone=mm,
124 presorttail=mn,
125 keyhead=aaa,

```

```

126 keynone={},
127 keytail=zzz}

```

`\ifniib@etex` The option to turn off ε -TeX needs to ensure that the user does not ask for ε -TeX if it is not there. On the other hand, the default is to use ε -TeX if it is available.

```

128 \niib@opt@boolkey{etex}
129 \begingroup
130   \@ifundefined{eTeXversion}
131     {}
132     {\global\niib@etexttrue}
133 \endgroup

```

7.4 Footnote and endnote handling

To allow dynamic handling of footnotes and endnotes, the original definitions are backed up.

```

\niib@org@footnote
\niib@org@footnotemark
\niib@org@footnotetext
134 \let\niib@org@footnote\footnote
135 \let\niib@org@footnotemark\footnotemark
136 \let\niib@org@footnotetext\footnotetext

```

If `endnotes` is loaded, then `\endnote` and friends have to be saved.

```

\niib@org@endnote
\niib@org@endnotemark
\niib@org@endnotetext
\niib@org@theendnotes
137 \AtBeginDocument{
138   \@ifpackageloaded{endnotes}
139     {\let\niib@org@endnote\endnote
140      \let\niib@org@endnotemark\endnotemark
141      \let\niib@org@endnotetext\endnotetext
142      \let\niib@org@theendnotes\theendnotes}{}}

```

`\thanks` The `\thanks` macro is redefined.

```

143 \@ifundefined{thanks}{}
144   {\renewcommand*{\thanks}[1]{%
145     \niib@org@footnotemark
146     \protected@xdef\@thanks{\@thanks
147       \protect\niib@org@footnotetext[\the\c@footnote]{#1}}}}

```

`\niib@swapfoot` The swapping code can now be implemented.

```

\footnote
\footnotemark
\footnotetext
148 \newcommand*{\niib@swapfoot}{%
149   \ifniib@footnotes
150     \let\footnote\bibnote
151     \let\footnotemark\bibnotemark
152     \let\footnotetext\bibnotetext
153     \niib@log@debug{Converting footnotes to bibnotes}%
154   \else
155     \let\footnote\niib@org@footnote
156     \let\footnotemark\niib@org@footnotemark
157     \let\footnotetext\niib@org@footnotetext
158     \niib@log@debug{Using kernel definition of footnotes}%
159   \fi}
160 \AtBeginDocument{\niib@swapfoot}

```

`\niib@swappend` For endnotes, the code needed depends on whether the `endnotes` package is available or not. If it is, then swapping the two definitions is set up.

```

\endnote
\endnotemark
\endnotetext
\theendnotes

```

```

161 \newcommand*\niib@swapend}{ }
162 \AtBeginDocument{
163   \ifpackageloaded{endnotes}
164     {\renewcommand*\niib@swapend}{%
165       \ifniib@endnotes
166         \let\endnote\bibnote
167         \let\endnotemark\bibnotemark
168         \let\endnotetext\bibnotetext
169         \let\theendnotes\printbibnotes
170         \niib@log@debug{Converting endnotes to bibnotes}%
171       }
172     {\let\endnote\niib@org@endnote
173      \let\endnotemark\niib@org@endnotemark
174      \let\endnotetext\niib@org@endnotetext
175      \let\theendnotes\niib@org@theendnotes
176      \niib@log@debug{Using endnotes package to handle
177        endnotes}%
178    }
179   \niib@swapend}

```

`endnotes` is not loaded; once the `endnotes` option has been given, there is nothing to go back. Of course, if the user does not give the `endnotes` option, `\endnote` is not defined at all.

```

180   {\ifniib@endnotes
181     \let\endnote\bibnote
182     \let\endnotemark\bibnotemark
183     \let\endnotetext\bibnotetext
184     \let\theendnotes\printbibnotes
185     \niib@log@debug{Converting endnotes to bibnotes}%
186   }
187   \renewcommand*\niib@swapend}{%
188     \ifniib@endnotes
189       \let\endnote\bibnote
190       \let\endnotemark\bibnotemark
191       \let\endnotetext\bibnotetext
192       \let\theendnotes\printbibnotes
193       \niib@log@debug{Converting endnotes to bibnotes}%
194     }
195     \niib@log@inf{endnotes package not loaded\MessageBreak
196       endnotes=false ignored}%
197   }

```

Options can now be processed.

```

198 \ProcessOptionsX[niib]<opt>
199 \AtEndOfPackage{
200   \define@key[niib]{opt}{etex}
201     {\niib@log@warn{Option 'etex' only available
202       \MessageBreak when loading notes2bib}}}
203 \begingroup
204   \ifundefined{eTeXversion}
205     {\ifniib@etex
206       \niib@log@warn{e-TeX unavailable}%
207       \global\niib@etexfalse
208     }

```

```

209     {\ifniib@etex\else
210       \niib@log@inf{Use of e-TeX disabled}%
211       \fi}
212 \endgroup

```

7.5 User macros

`\thebibnote` A counter is needed for the notes created. In analogy to other counters in L^AT_EX, this is given a `\the...` name. The user should not really need to use this macro, but convention dictate that it has a user-space name. The L^AT_EX `\newcounter` macro is used (rather than the T_EX `\newcount`) as the automatic system expects the numbers to be globally unique.

```

213 \newcounter{bibnote}
214 \renewcommand*{\thebibnote}{\niib@name\the\value{bibnote}}

```

`\bibnote` Each new `\bibnote` increments the note counter, then checks for an optional label, before handing off to the internal macro `\niib@bibnote`.

```

215 \DeclareRobustCommand*{\bibnote}{%
216   \stepcounter{bibnote}%
217   \@ifnextchar[%]
218     {\niib@bibnote}
219     {\niib@bibnote[\thebibnote]}}

```

`\bibnotemark` The `\bibnotemark` macro works in the same way as `\bibnote`, but calls `\niib@mark` rather than `\niib@bibnote`.

```

220 \DeclareRobustCommand*{\bibnotemark}{%
221   \stepcounter{bibnote}%
222   \@ifnextchar[%]
223     {\niib@mark}
224     {\niib@mark[\thebibnote]}}

```

`\bibnotetext` The text companion to the mark macro above, with no increment of the counter. There is nothing special to do, so the L^AT_EX kernel handling of optional arguments can be used.

```

225 \DeclareRobustCommand*{\bibnotetext}[1][\thebibnote]{%
226   \let\niib@next\niib@gobble
227   \niib@text{#1}}

```

`\printbibnotes` To allow for the possibility of there being no other notes, a command to print only notes is given. In the bibl_{at}ex case, the best that can be done is to issue `\printbibliography`.

```

228 \AtBeginDocument{
229   \@ifpackageloaded{biblatex}
230     {\let\printbibnotes\printbibliography}
231     {\DeclareRobustCommand*{\printbibnotes}
232       {\niib@org@bib{\niib@prefix\jobname}}}}

```

`\flushnotestack` In order to delay citations to the end of the bibliography (and thus force others to the start), a “stack” is created of citations which need to be written to the `.aux` file. This is done here, and the stack is cleared so collection can begin again.

```

233 \DeclareRobustCommand*{\flushnotestack}{%
234   \let\niib@taillist\niib@stack

```

```

235 \ifniib@rerun\else
236   \niib@checkrerun{tail}%
237 \fi
238 \ifx\@empty\niib@stack\@empty
239   \niib@log@debug{Citation stack empty: nothing for
240     \MessageBreak\noexpand\flushnotestack to do}%
241 \else%
242   \niib@log@debug{Flushing note citations to aux file}%
243   \if@filesw
244     \immediate\write\@auxout{%
245       \string\niibtailcite{\niib@stack}}%
246   \fi
247   \expandafter\nocite\expandafter{\niib@stack}%
248   \gdef\niib@stack{}%
249 \fi}

```

`\citenote` Problems arise with `\cite` and the `sort=tail` option. Rather than overload `\cite` with all of the problems that can bring, a new command is provided that can be guaranteed to work.

```

250 \DeclareRobustCommand*\citenote[1]{\niib@mark[#1]}

```

7.6 Internal macros

`\niib@keyname` If biblatex is in use, the key field in the BibTeX database should be called “keysort,” whereas otherwise it should be “key.”

```

251 \AtBeginDocument{
252   \@ifpackageloaded{biblatex}
253     {\niib@log@debug{Using field 'keysort' for sorting key}%
254       \newcommand*\niib@keyname{keysort}}
255     {\niib@log@debug{Using field 'key' for sorting key}%
256       \newcommand*\niib@keyname{key}}

```

`\niib@presort` The values taken by `\niib@presort` and `\niib@key` depend on the desired positioning of notes in the bibliography.

```

\niib@key
257 \newcommand*\niib@presort{%
258   \ifniib@head
259     \niib@presorthhead
260   \else
261     \ifniib@tail
262       \niib@presorttail
263     \else
264       \niib@presortnone
265   \fi
266 \fi}
267 \newcommand*\niib@key{%
268   \ifniib@head
269     \niib@keyhead
270   \else
271     \ifniib@tail
272       \niib@keytail
273     \else
274       \niib@keynone
275   \fi

```

```

276 \fi}

\niib@msg To inform the user, the automatically-created BibTeX database needs to carry
suitable information on its source.
277 \edef\niib@msg{%
278   This is an auxiliary file used by the 'notes2bib'
279   package.^J This file may safely be deleted. It
280   will be recreated as required.^J}

\niib@stack This macro is needed to store any citations at the end of the bibliography. Initially,
this is empty.
281 \newcommand*\niib@stack{}

\niib@addtostack The various optional argument tricks above all use the same core code, which
adds the mandatory argument of the citation to the stack. The stack is global (see
also \flushnotestack).
282 \newcommand*\niib@addtostack[1]{%
283   \niib@log@debug{Adding citation #1\MessageBreak to 'tail'
284     stack}%
285   \edef\niib@tempa{#1}%
286   \ifx\@empty\niib@stack\@empty
287     \xdef\niib@stack{\niib@tempa}%
288   \else
289     \xdef\niib@stack{\niib@stack,\niib@tempa}%
290   \fi}

\niib@bibnote Two steps are needed here, writing the text of the note to file (handled by
\niib@text, and marking the citation (using \niib@cite).
291 \long\def\niib@bibnote[#1]#2{%
292   \let\niib@next\niib@mark
293   \niib@text{#1}{#2}}

\niib@headlist To inform the user that a re-run of LATEX is needed, tracking is needed of any
“head” citations.
294 \newcommand*\niib@headlist{}

\niib@mark Adding a citation to the LATEX file is handled here. When using the sort=head
option, the citation is written to the .aux file for sorting control. The normal
citation command is then called.
295 \def\niib@mark[#1]{%
296   \ifniib@head
297     \edef\niib@tempa{#1}%
298     \ifx\@empty\niib@headlist\@empty
299       \xdef\niib@headlist{\niib@tempa}%
300     \else
301       \xdef\niib@headlist{\niib@headlist,\niib@tempa}%
302     \fi
303     \if@filesw
304       \niib@log@debug{Adding citation #1 to list for next run}%
305       \immediate\write\@auxout{\string\niibheadcite{#1}}%
306     \fi
307   \fi

```

When the `sort=tail` option is active, citation is handled by another macro, so a switch is needed.

```
308 \ifniib@tail
309   \expandafter\niib@tailcite%
310 \else
311   \expandafter\niib@normcite%
312 \fi
313 {#1}}
```

`\ifniib@files` A switch is used to back up `\if@files`.

```
314 \newif\ifniib@files
```

`\niib@tailcite` When using the `sort=tail` option, bibnote citation need to be stored for later. With `biblatex`, the `\AtEndCite` macro is available to provide a hook for the necessary switch. In other cases, the current value of `\if@files` is then saved, before turning it off and setting up the restore system.

```
315 \AtBeginDocument{
316   \@ifpackageloaded{biblatex}
317     {\newcommand{\niib@tailcite}[1]{%
318       \niib@addtostack{#1}%
319       \AtNextCite{\@filesfalse}%
320       \niib@normcite{#1}}}
321   {\newcommand{\niib@tailcite}[1]{%
322     \niib@addtostack{#1}%
323     \let\ifniib@files\if@files
324     \@filesfalse
325     \let\niib@auxhook\niib@restorefiles
326     \niib@tcite{#1}}}}
```

`\niib@restorefiles` Restoring the switch is set up here. The reference to `\niib@auxhook` ensures that the mechanism is turned off for the next real citation.

```
327 \newcommand*\niib@restorefiles{%
328   \let\if@files\ifniib@files
329   \let\niib@auxhook\relax}
```

`\niib@tcite` Actually carrying out the citation, and restoring the value of `\if@files` depends on whether `cite` is loaded.

```
330 \AtBeginDocument{%
331   \@ifpackageloaded{cite}
332     {\newcommand*\niib@tcite[1]{\niib@normcite{#1}}}
333     {\newcommand*\niib@tcite[1]{%
334       \niib@normcite{#1}%
335       \niib@restorefiles}}}
```

`\niib@normcite` The normal citation command.

```
336 \newcommand*\niib@normcite{\@nameuse{\niib@cite}}
```

`\niib@next` To allow correct detection of following punctuation by `\niib@mark`, it has to follow `\niib@text`. However, to prevent category code issues, `\niib@text` can only take *one* argument. Hence, a “follow-on” macro is needed; this is provided here. The no-op here has to gobble a square-bracketed argument.

`\niib@gobble`

```
337 \newcommand*\niib@next{}
```

```

338 \def\niib@gobble[#1]{}
339 \let\niib@next\niib@gobble

\niib@text The \niib@text macro is the outer part of the writing system. It does not
\niib@out absorb the text of note, as without  $\epsilon$ -TeX this is bad news.
\niib@stream 340 \newcommand*{\niib@text}{%
341 \@bsphack
342 \@ifundefined{niib@out}
343 {\if@filesw
344 \newwrite\niib@out
345 \gdef\niib@stream{\niib@prefix\jobname.bib}%
346 \niib@log@debug{Creating BibTeX database file
347 \MessageBreak\niib@stream\space to contain bibnotes}%
348 \immediate\openout\niib@out\niib@stream\relax
349 \immediate\write\niib@out{\niib@msg}%
350 \fi}
351 {}%
352 \if@filesw
353 \expandafter\niib@write
354 \fi}

\niib@write The \niib@write macro deals with writing data verbatim to the BibTeX file.
When  $\epsilon$ -TeX is available, this is easy; the macro absorbs the note text and uses
\unexpanded.

355 \ifniib@etex
356 \newcommand*{\niib@write}[2]{%
357 \niib@log@debug{Writing bibnote #1 contents
358 \MessageBreak---\MessageBreak#2\MessageBreak
359 ---\MessageBreak to BibTeX database}%
360 \immediate\write\niib@out{%
361 @\niib@record\string{#1,^^J%
362 presort = \string{\niib@presort\string},^^J%
363 \niib@keyname\space= \string{\niib@key#1\string},^^J%
364 \niib@field\space= \string{\unexpanded{#2}\string}^^J%
365 \string}^^J}%
366 \@esphack
367 \niib@next[#1]}

Life is more awkward when  $\epsilon$ -TeX is unavailable. The code here is based on a
suggestion by Ulrich Diez, with alterations to the specifics needed here.

368 \else
369 \newcommand*{\niib@write}[1]{%
370 \begingroup
371 \let\do\@makeother
372 \dospecials
373 \catcode`\{=1\relax
374 \catcode`\}=2\relax
375 \niib@write@{#1}}
376 \fi

\niib@write@ Two support macros are needed if  $\epsilon$ -TeX is not available. \niib@write@ works
with the category codes altered as above, which means that text is written
(essentially) verbatim to the BibTeX file.

377 \newcommand\niib@write@[1]{%

```



```

378 \long\def\niib@tempa##1{%
379 \def\niib@tempa{##1}%
380 \@onelevel@sanitize\niib@tempa\expandafter\endgroup
381 \expandafter\def\expandafter\niib@tempa\expandafter{%
382 \niib@tempa}%
383 \niib@write@@{#1}}%
384 \catcode\^^M=10\relax
385 \niib@tempa}

```

`\niib@write@@` Writing to the file is a very slightly altered version of the ϵ -TeX version. In this case, the debugging data is given here, after category code changes.

```

386 \newcommand*\niib@write@@[1]{%
387 \niib@log@debug{Writing bibnote #1 contents
388 \MessageBreak---\MessageBreak\niib@tempa\MessageBreak
389 ---\MessageBreak to BibTeX database}%
390 \immediate\write\niib@out{%
391 @\niib@record\string{#1,^^J%
392 presort = \string{\niib@presort\string},^^J%
393 \niib@keyname\space= \string{\niib@key#1\string},^^J%
394 \niib@field\space= \string{\niib@tempa\string}^^J%
395 \string}^^J}%
396 \@esphack
397 \niib@next[#1]}

```

`\niib@headcitelist` To inform the user that a re-run of L^AT_EX is needed, tracking is needed of any citations that have been moved to the start of the .aux file. This needs an initially-empty macro.

```

398 \newcommand*\niib@headcitelist{}

```

`\document` When using the sort=head option, bibnotes need to appear in the .aux file before other citations. Other approaches cause all sorts of problems, so the suggestion of Michael Shell is implemented here. When head is active, `\niibheadcite` is added to the .aux file. At the next L^AT_EX run, this will add a citation to the beginning of the .aux file. To get the `\nocite` to work, a hook has to be added to `\document`. The reason is that `\AtBeginDocument` cannot be used here: it is not available once the old .aux file has been read.

```

399 \g@addto@macro{\document}{\niib@dochook}
400 \newcommand*\niibheadcite[1]{%
401 \edef\niib@tempa{#1}%
402 \ifx\@empty\niib@headcitelist\@empty
403 \xdef\niib@headcitelist{\niib@tempa}%
404 \else
405 \xdef\niib@headcitelist{\niib@headcitelist,\niib@tempa}%
406 \fi
407 \if@files@w
408 \niib@log@debug{Adding citation #1 to start of .aux file}%
409 \fi
410 \g@addto@macro{\niib@dochook}{\nocite{#1}}}

```

`\niib@tailcitelist` To enable proper logging of citations when sort=tail, a similar system to the above is employed without the `\nocite` part.

```

411 \newcommand*\niib@tailcitelist{}
412 \newcommand*\niibtailcite[1]{%

```

```

413 \edef\niib@tempa{#1}%
414 \ifx\@empty\niib@tailcitelist\@empty
415   \xdef\niib@tailcitelist{\niib@tempa}%
416 \else
417   \xdef\niib@tailcitelist{\niib@tailcitelist,\niib@tempa}%
418 \fi}

```

\niib@auxhook To allow the automatic punctuation-searching of cite to work, some code has to
\niib@dochook be added to the hook available there. However, as that is intended for multibib,
\@restore@auxhandle care is needed to get the desired result. \niib@dochook is defined here, even
though it is needed above, as the code here will always be executed.

```

419 \newcommand*{\niib@dochook}{%
420   \@ifundefined{@restore@auxhandle}%
421   {\newcommand*{\@restore@auxhandle}{\niib@auxhook}}%
422   {\ifx\relax\@restore@auxhandle\relax
423     \newcommand*{\@restore@auxhandle}{\niib@auxhook}%
424   \else
425     \g@addto@macro{\@restore@auxhandle}{\niib@auxhook}%
426   \fi}}
427 \newcommand*{\niib@auxhook}{%
428 \let\niib@auxhook\relax

```

\blx@bibfiles The \bibliography macro is patched to ensure that when it is executed the
note file is also processed. biblatex does things very differently, but this actually
makes it much easier to patch for.

```

429 \AtBeginDocument{
430   \@ifpackageloaded{biblatex}%
431   {\expandafter\gappto\expandafter\blx@bibfiles\expandafter
432     {,\niib@prefix\jobname}%
433   \niib@log@debug{Added bibnotes database to biblatex file
434     list}}%

```

\niib@org@bib Without biblatex, the bibliography command is patched so that it will run on the
\bibliography automatically-generated Bib_{TEX} database. If no notes have been added, then the
macro doesn't actually do anything.

```

435   {\let\niib@org@bib\bibliography
436   \renewcommand*{\bibliography}[1]{%
437     \ifnum\the\value{bibnote} > \z@
438       \niib@org@bib{\niib@prefix\jobname,#1}%
439     \else
440       \niib@org@bib{#1}%
441     \fi}
442   \niib@log@debug{Added bibnote database to
443     \noexpand\bibliography}}%

```

7.7 Finalisation

\ifniib@rerun A switch is needed for the re-run test.

```

444 \newif\ifniib@rerun

```

\niib@checkrerun Any “head” notes may mean a second L^AT_EX run is needed.

```

445 \newcommand*{\niib@checkrerun}[1]{%

```

```

446 \niib@rerunfalse
447 \expandafter\ifx\expandafter\@empty
448   \csname niib@#l1list\endcsname\@empty
449   \expandafter\ifx\expandafter\@empty
450     \csname niib@#l1citelist\endcsname\@empty
451     \niib@log@debug{No '#1' notes detected}%
452   \else
453     \niib@reruntrue
454     \niib@log@debug{No '#1' notes found this run\MessageBreak
455       but .aux files contained the '#1' requests:
456       \MessageBreak\csname niib@#l1citelist\endcsname}%
457   \fi
458 \else
459   \expandafter\ifx\expandafter\@empty
460     \csname niib@#l1citelist\endcsname\@empty
461     \niib@reruntrue
462     \niib@log@debug{No '#1' requests in .aux file
463       \MessageBreak but '#1' notes in this run:
464       \MessageBreak\csname niib@#l1list\endcsname}%
465   \else

```

If the package gets here, then there are some notes and some requests in the aux file. The two lists are now compared.

```

466     \niib@checklists{#1}%
467   \fi
468 \fi
469 \ifniib@rerun

```

Both human-readable requests for new runs, and biblatex-style automated requests are made. The package does not know if BibTeX8 is in use, so just asks for BibTeX.

```

470 \niib@log@warn{Rerun LaTeX to get correct \MessageBreak
471   '#1' notes}%
472 \niib@log@warn{Please (re)run BibTeX on the file(s):
473   \MessageBreak\jobname.aux
474   \MessageBreak and rerun LaTeX afterwards.}%
475 \ifniib@lognone\else
476   \typeout{%
477     REQ:3:latex:REQ^^J%
478     REQ:2:bibtex:REQ^^J%
479     REQ:1:latex:REQ}%
480   \fi
481 \fi}

```

`\niib@checklists` The business end of comparing the two lists. Two sweeps are made, to check that the lists match entirely.

```
482 \newcommand*{\niib@checklists}[1]{%
```

`\niib@list` To allow `\niib@checklists` to handle both `sort=head` and `sort=tail` citations, the expanded list is needed for the `\@for` loops.

`\niib@citelist`

```

483 \expandafter\edef\expandafter\niib@list\expandafter
484   {\csname niib@#l1list\endcsname}%
485 \expandafter\edef\expandafter\niib@citelist\expandafter
486   {\csname niib@#l1citelist\endcsname}%

```

The loops can now being.

```

487 \@for\niib@tempa:=\niib@list\do{%
488     \niib@reruntrue
489     \@for\niib@tempb:=\niib@citelist\do{%
490         \ifx\niib@tempa\niib@tempb
491             \niib@rerunfalse
492         \fi}
493     \ifniib@rerun
494         \niib@log@debug{Note \niib@tempa\space is a '#1' note
495             \MessageBreak but request not in .aux file}%
496     \fi}
497 \ifniib@rerun\else
498     \@for\niib@tempa:=\niib@citelist\do{%
499         \niib@reruntrue
500         \@for\niib@tempb:=\niib@list\do{%
501             \ifx\niib@tempa\niib@tempb
502                 \niib@rerunfalse
503             \fi}
504         \ifniib@rerun
505             \niib@log@debug{Note \niib@tempa\space is set to '#1'
506                 in .aux\MessageBreak file but is not a '#1' note}%
507         \fi}
508 \fi}

```

\niib@taillist At the end of the document, any delayed citations are written to the .aux file, and the database file is closed cleanly. A check is also made for the need for an additional L^AT_EX run for “head” notes.

```

509 \AtEndDocument{%
510     \niib@rerunfalse
511     \niib@checkrerun{head}%
512     \flushnotestack
513     \@ifundefined{niib@out}{}
514     {\immediate\closeout\niib@out
515         \niib@log@debug{Closed BibTeX database file\MessageBreak
516             \niib@stream}}}

```

8 Notes

- [1] Note for the first example.
- [2] Note for the second example.
- [3] Note for the third example.
- [4] Some \verb-like output.

9 Change History

v1.0

v1.0a

General: Initial public release 1 \citenote: New macro 13

v1.1	General: \Percent macro removed	1	\niib@addtostack: Macro renamed from \niib@stackup ..	14
	Documentation improvements ..	1	\niib@auxhook: New macro ...	18
	Improvements to documentation and dtx file	1	\niib@bibnote: Use \niib@mark for citation	14
	License changed from GPL to LPPL	1	\niib@dochook: New macro ...	18
	Several code sections re-factored	1	\niib@key: Dynamic rather than static definition	13
	\bibnote: Macro made robust ..	12	\niib@keyname: Moved to \AtBeginDocument	13
	\bibnotemark: Macro made robust	12	\niib@presort: Dynamic rather than static definition	13
	\bibnotetext: Macro made robust	12	\niib@headcite: New macro ...	17
	\citenote: Macro made robust ..	13	\niib@setup: New macro	8
	\flushnotestack: Macro made robust	12	v1.3b	
	\niib@id: Removed use of xspace	6	\flushnotestack: Added check for correct tail citations in .aux file	12
	\printbibnotes: Macro made robust	12	Writes stack to .aux file	12
v1.2	General: Altered implementation of head and tail options to allow moving of superscript citations ..	1	\niib@checkrerun: Added ability to check for both head and tail re-runs	18
	\blx@bibfiles: Fixed bug with biblatex support	18	Bug fix with logging for debug logging	18
v1.3	General: Added xkeyval option interface	1	\niib@citelist: New macro ..	19
	All options now work anywhere in input	1	\niib@list: New macro	19
	Fixed serious errors with head and tail implementation	1	\niib@tailcitelist: New macro	17
	\bibliography: Modification now at beginning of document	18	\niib@taillist: New macro ..	20
	\blx@bibfiles: Modification now at beginning of document	18	\niib@tailcite: New macro ...	17
	Updated for biblatex v0.7: moved to end of preamble, switch from \bib@gadd to \gappto	18	\printbibnotes: Definition moved to beginning of document ..	12
	\citenote: Now a wrapper for \niib@mark	13	v1.3c	
	\document: Added hook after .aux files has been read but before document begins	17	\niib@org@footnote: Saved in package	10
	\flushnotestack: Fixed bug with empty stack	12	\niib@org@footnotemark: Saved in package	10
	Renamed from \flushcitestack	12	\niib@org@footnotetext: Saved in package	10
			\niib@swapfoot: Simplified code	10
			\thanks: Altered redefinition method	10
			Redefinition in preamble	10
			v1.4	
			General: ϵ -TeX made optional (again), with control option to turn off use even if available ...	1

10 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\@restore@auxhandle	<u>419</u>
\{	373
\}	374
\^	384
B	
\bibliography	<u>435</u>
\bibnote	<i>2</i> , 150, 166, 181, 189, <u>215</u>
\bibnotemark	<i>2</i> , 152, 167, 182, 190, <u>220</u>
\bibnotetext	<i>2</i> , 151, 168, 183, 191, <u>225</u>
\blx@bibfiles	<u>429</u>
C	
cite (option)	<i>2</i>
\citenote	<i>3</i> , <u>250</u>
D	
debug (option)	3
\document	<u>399</u>
E	
\endnote	139, <u>161</u>
\endnotemark	140, <u>161</u>
endnotes (option)	<i>2</i>
\endnotetext	141, <u>161</u>
etex (option)	3
F	
field (option)	3
\flushnotestack	<i>4</i> , <u>233</u> , 512
\footnote	134, <u>148</u>
footnote (option)	<i>2</i>
\footnotemark	135, <u>148</u>
\footnotetext	136, <u>148</u>
H	
head (option)	3
I	
\ifniib@debug	<u>10</u> , 35
\ifniib@endnotes ..	<u>72</u> , 165, 180, 188
\ifniib@etex	<u>128</u> , 205, 209, 355
\ifniib@fileswh	<u>314</u> , 323, 328
\ifniib@footnotes	<u>72</u> , 149
\ifniib@head ..	<u>74</u> , 89, 98, 258, 268, 296
\ifniib@logmin	<u>10</u> , 15, 23, 29
\ifniib@lognone ..	<u>10</u> , 14, 22, 28, 34, 475
\ifniib@rerun	235, <u>444</u> , 469, 493, 497, 504
\ifniib@tail ..	<u>74</u> , 90, 97, 261, 271, 308
K	
keyhead (option)	3
keynone (option)	3
keytail (option)	3
L	
log (option)	3
N	
name (option)	3
\niib@addtostack	<u>282</u> , 318, 322
\niib@auxhook	325, 329, <u>419</u>
\niib@bibnote	218, 219, <u>291</u>
\niib@checklists	466, <u>482</u>
\niib@checkrerun	236, <u>445</u> , 511
\niib@cite	<u>104</u> , 336
\niib@citelist	<u>483</u> , 489, 498
\niib@debugfalse	51
\niib@debugtrue	64, 68
\niib@dochook	<u>399</u> , <u>419</u>
\niib@etexfalse	207
\niib@etextrue	132
\niib@field	<u>104</u> , 364, 394
\niib@gobble	226, <u>337</u>
\niib@headcitelist ..	<u>398</u> , 402, 403, 405
\niib@headfalse	77, 99
\niib@headlist ...	<u>294</u> , 298, 299, 301
\niib@headtrue	81
\niib@id	<u>1</u>
\niib@key	<u>257</u> , 363, 393
\niib@keyhead	<u>104</u> , 269
\niib@keyname	<u>251</u> , 363, 393
\niib@keynone	<u>104</u> , 274
\niib@keytail	<u>104</u> , 272
\niib@list	<u>483</u> , 487, 500
\niib@log@debug	33, 41, 44, 45, 153, 158,
.....	170, 176, 185, 193, 239, 242, 253,
.....	255, 283, 304, 346, 357, 387, 408,
.....	433, 442, 451, 454, 462, 494, 505, 515
\niib@log@err	<u>13</u>

\niib@log@inf ...	<u>13</u> , 92, 100, 195, 210	385, 388, 394, 401, 403, 405, 413,
\niib@log@warn		415, 417, 487, 490, 494, 498, 501, 505
.....	<u>13</u> , 70, 87, 201, 206, 470, 472	\niib@tempb
\niib@logminfalse	52	8,
\niib@logmintrue	60	54, 55, 58, 59, 62, 63, 66, 67,
\niib@lognonefalse	53	79, 80, 83, 84, 489, 490, 500, 501
\niib@lognonetrue	56	\niib@text
\niib@mark	223, 224, 250, 292, <u>295</u>	227, 293, <u>340</u>
\niib@msg	<u>277</u> , 349	\niib@write
\niib@name	<u>104</u> , 214	353, <u>355</u>
\niib@next	226, 292, <u>337</u> , 367, 397	\niib@write@
\niib@normcite	311, 320, 332, 334, <u>336</u>	375, <u>377</u>
\niib@opt@boolkey		\niib@write@@
.....	<u>39</u> , 71–73, 88, 96, 128	383, <u>386</u>
\niib@opt@choicekey	42, 49, 76	\niibheadcite
\niib@opt@cmdkeys	<u>46</u> , 104	6, 305, <u>399</u>
\niib@opt@debug	<u>71</u>	\niibsetup
\niib@org@bib	232, <u>435</u>	2, <u>48</u> , 116
\niib@org@endnote	<u>137</u> , 172	\niibtailcite
\niib@org@endnotemark ...	<u>137</u> , 173	245, <u>411</u>
\niib@org@endnotetext ...	<u>137</u> , 174	
\niib@org@footnote	<u>134</u> , 155	
\niib@org@footnotemark	<u>134</u> , 145, 156	
\niib@org@footnotetext	<u>134</u> , 147, 157	
\niib@org@theendnotes ...	<u>137</u> , 175	
\niib@out	<u>340</u> , 360, 390, 514	
\niib@prefix ..	<u>104</u> , 232, 345, 432, 438	
\niib@presort	257, 362, 392	
\niib@presorthead	<u>104</u> , 259	
\niib@presortnone	<u>104</u> , 264	
\niib@presorttail	<u>104</u> , 262	
\niib@record	<u>104</u> , 361, 391	
\niib@rerunfalse .	446, 491, 502, 510	
\niib@reruntrue ..	453, 461, 488, 499	
\niib@restorefiles ..	325, <u>327</u> , 335	
\niib@stack	234,	
	238, 245, 247, 248, <u>281</u> , 286, 287, 289	
\niib@stream	<u>340</u> , 516	
\niib@swapend	73, <u>161</u>	
\niib@swapfoot	72, <u>148</u>	
\niib@tailcite ...	309, <u>315</u> , 317, 321	
\niib@tailcitelist	<u>411</u>	
\niib@tailfalse	78, 91	
\niib@taillist	234, <u>509</u>	
\niib@tailtrue	85	
\niib@tcite	326, <u>330</u>	
\niib@tempa	8,	
	43, 55, 59, 63, 67, 80, 84, 285,	
	287, 289, 297, 299, 301, 378–382,	

O

options:	
cite	2
debug	3
endnotes	2
etex	3
field	3
footnote	2
head	3
keyhead	3
keynone	3
keytail	3
log	3
name	3
prefix	3
presorthead	3
presortnone	3
presorttail	3
record	3
sort	3
tail	3

P

prefix (option)	3
presorthead (option)	3
presortnone (option)	3
presorttail (option)	3
\printbibnotes ..	3, 169, 184, 192, <u>228</u>

R

record (option)	3
-----------------------	---

S

sort (option)	3
---------------------	---

T

tail (option)	3
\thanks	<u>143</u>
\thebibnote	4, <u>213</u> , 219, 224, 225
\theendnotes	142, <u>161</u>