

notes2bib — Integrating notes into the bibliography*

Joseph Wright[†]

Released 2008/06/21

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 L^AT_EX citations. Both sorted and unsorted bibliography styles are supported.

Contents

1	Background	1	6.2	The REVTeX approach	5
2	Basic use	2	6.3	The notes2bib 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 endnotes approach	5	8	Notes	20
			9	Change History	20
			10	Index	21

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 B_BT_EX users create a notes database for each document that they write.

*This file describes version v1.4, last revised 2008/06/21.

[†]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 L^AT_EX 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 `\citenote{labelled-note}`.
An example of a named note `\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@filesw` 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 v1.4\space}
4 \ProvidesPackage{notes2bib}
5 [\niib@id$Id: notes2bib.dtx 11 2008-06-21 16:00:05Z 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 -T_EX `\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  $\epsilon$ -TeX needs to ensure that the user does not ask for  $\epsilon$ -TeX
if it is not there. On the other hand, the default is to use  $\epsilon$ -TeX if it is available.

128 \niib@opt@boolkey{etex}
129 \begingroup
130 \ifundefined{eTeXversion}
131 {}
132 {\global\niib@etexttrue}
133 \endgroup
134 \ProcessOptionsX[niib]<opt>
135 \AtEndOfPackage{
136 \define@key[niib]{opt}{etex}
137 {\niib@log@warn{Option 'etex' only available
138 \MessageBreak when loading notes2bib}}}
139 \begingroup
140 \ifundefined{eTeXversion}
141 {\ifniib@etex
142 \niib@log@warn{e-TeX unavailable}%
143 \global\niib@etextfalse
144 \fi}
145 {\ifniib@etex\else
146 \niib@log@inf{Use of e-TeX disabled}%
147 \fi}
148 \endgroup

```

7.4 Footnote and endnote handling

```

\niib@org@footnote To allow dynamic handling of footnotes and endnotes, the original definitions
\niib@org@footnotemark are backed up.
\niib@org@footnotetext
149 \let\niib@org@footnote\footnote
150 \let\niib@org@footnotemark\footnotemark
151 \let\niib@org@footnotetext\footnotetext

\niib@org@endnote If endnotes is loaded, then \endnote and friends have to be saved.
\niib@org@endnotemark
\niib@org@endnotetext
\niib@org@theendnotes
152 \AtBeginDocument{
153 \ifpackageloaded{endnotes}
154 {\let\niib@org@endnote\endnote
155 \let\niib@org@endnotemark\endnotemark
156 \let\niib@org@endnotetext\endnotetext
157 \let\niib@org@theendnotes\theendnotes}{}}

\thanks The \thanks macro is redefined.
158 \ifundefined{thanks}{}
159 {\renewcommand*{\thanks}[1]{%
160 \niib@org@footnotemark
161 \protected@xdef\@thanks{\@thanks
162 \protect\niib@org@footnotetext[\the\c@footnote]{#1}}}}

\niib@swapfoot The swapping code can now be implemented.
\footnote
\footnotemark
\footnotetext
163 \newcommand*{\niib@swapfoot}{%
164 \ifniib@footnotes

```

```

165 \let\footnote\bibnote
166 \let\footnotemark\bibnotetext
167 \let\footnotetext\bibnotemark
168 \niib@log@debug{Converting footnotes to bibnotes}%
169 \else
170 \let\footnote\niib@org@footnote
171 \let\footnotemark\niib@org@footnotemark
172 \let\footnotetext\niib@org@footnotetext
173 \niib@log@debug{Using kernel definition of footnotes}%
174 \fi}
175 \AtBeginDocument{\niib@swapfoot}

\niib@swapend For endnotes, the code needed depends on whether the endnotes package is
\endnote available or not. If it is, then swapping the two definitions is set up.
\endnotemark 176 \newcommand*{\niib@swapend}{}
\endnotetext 177 \AtBeginDocument{
\theendnotes 178 \ifpackageloaded{endnotes}
179 {\renewcommand*{\niib@swapend}{%
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 \else
187 \let\endnote\niib@org@endnote
188 \let\endnotemark\niib@org@endnotemark
189 \let\endnotetext\niib@org@endnotetext
190 \let\theendnotes\niib@org@theendnotes
191 \niib@log@debug{Using endnotes package to handle
192 endnotes}%
193 \fi}
194 \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.

195 {\ifniib@endnotes
196 \let\endnote\bibnote
197 \let\endnotemark\bibnotemark
198 \let\endnotetext\bibnotetext
199 \let\theendnotes\printbibnotes
200 \niib@log@debug{Converting endnotes to bibnotes}%
201 \fi
202 \renewcommand*{\niib@swapend}{%
203 \ifniib@endnotes
204 \let\endnote\bibnote
205 \let\endnotemark\bibnotemark
206 \let\endnotetext\bibnotetext
207 \let\theendnotes\printbibnotes
208 \niib@log@debug{Converting endnotes to bibnotes}%
209 \else
210 \niib@log@inf{endnotes package not loaded\MessageBreak
211 endnotes=false ignored}%

```

212 \fi}}}

7.5 User macros

`\thebibnote` A counter is needed for the notes created. In analogy to other counters in \LaTeX , 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 \LaTeX `\newcounter` macro is used (rather than the \TeX `\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 \LaTeX 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 `biblatex` 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@presorthead
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 Bib_T_EX 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 L_AT_EX is needed, tracking is needed of any “head” citations.

```

294 \newcommand*\niib@headlist{}

```

`\niib@mark` Adding a citation to the L_AT_EX 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@filesw A switch is used to back up \if@filesw.
314 \newif\ifniib@filesw

\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@filesw 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{\@fileswfalse}%
320       \niib@normcite{#1}}}
321     {\newcommand{\niib@tailcite}[1]{%
322       \niib@addtostack{#1}%
323       \let\ifniib@filesw\if@filesw
324       \@fileswfalse
325       \let\niib@auxhook\niib@restorefilesw
326       \niib@tcite{#1}}}}

```

\niib@restorefilesw 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@restorefilesw}{%
328   \let\if@filesw\ifniib@filesw
329   \let\niib@auxhook\relax}

```

\niib@tcite Actually carrying out the citation, and restoring the value of \if@filesw 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@restorefilesw}}}

```

\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 absorb the text of note, as without $\varepsilon\text{-TeX}$ this is bad news.

`\niib@out`

`\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 $\varepsilon\text{-TeX}$ 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 $\varepsilon\text{-TeX}$ 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 $\varepsilon\text{-TeX}$ 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 \niib@dochook \niib@headcite 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, \niib@headcite 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*\niib@headcite[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@filesw
408 \niib@log@debug{Adding citation #1 to start of .aux file}%
409 \fi
410 \g@addto@macro{\niib@dochook}{\nocite{#1}}}

```

\niib@tailcitelist \niib@tailcite 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*\niib@tailcite[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_TE_X 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#1list\endcsname\@empty

```

```

449 \expandafter\ifx\expandafter\@empty
450 \csname niib@#1citelist\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@#1citelist\endcsname}%
457 \fi
458 \else
459 \expandafter\ifx\expandafter\@empty
460 \csname niib@#1citelist\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@#1list\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@#1list\endcsname}%
485 \expandafter\edef\expandafter\niib@citelist\expandafter
486 {\csname niib@#1citelist\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		v1.1
General: Initial public release	1	General: \Percent macro removed 1
v1.0a		Documentation improvements . . 1
\citenote: New macro	13	Improvements to documentation

and dtx file	1	\niib@auxhook: New macro ...	17
License changed from GPL to		\niib@bibnote: Use \niib@mark	
LPPL	1	for citation	14
Several code sections re-factored	1	\niib@dochook: New macro ...	17
\bibnote: Macro made robust ..	12	\niib@key: Dynamic rather than	
\bibnotemark: Macro made ro-		static definition	13
bust	12	\niib@keyname: Moved to	
\bibnotetext: Macro made ro-		\AtBeginDocument	13
bust	12	\niib@presort: Dynamic rather	
\citenote: Macro made robust ..	13	than static definition	13
\flushnotestack: Macro made		\niibheadcite: New macro ...	17
robust	12	\niibsetup: New macro	7
\niib@id: Removed use of xspace	6		
\printbibnotes: Macro made ro-		v1.3b	
bust	12	\flushnotestack: Added check	
		for correct tail citations in	
v1.2		.aux file	12
General: Altered implementation of		Writes stack to .aux file	12
head and tail options to allow		\niib@checkrerun: Added abil-	
moving of superscript citations ..	1	ity to check for both head and	
\blx@bibfiles: Fixed bug with		tail re-runs	18
biblatex support	18	Bug fix with logging for debug	
		logging	18
v1.3		\niib@citelist: New macro ..	19
General: Added xkeyval option inter-		\niib@list: New macro	19
face	1	\niib@tailcitelist: New	
All options now work anywhere		macro	17
in input	1	\niib@taillist: New macro ..	20
Fixed serious errors with head		\niibtailcite: New macro ...	17
and tail implementation	1	\printbibnotes: Definition	
\bibliography: Modification		moved to beginning of docu-	
now at beginning of document	18	ment	12
\blx@bibfiles: Modification			
now at beginning of document	18	v1.3c	
Updated for biblatex v0.7: moved		\niib@org@footnote: Saved in	
to end of preamble, switch from		package	10
\bib@gadd to \gappto	18	\niib@org@footnotemark:	
\citenote: Now a wrapper for		Saved in package	10
\niib@mark	13	\niib@org@footnotetext:	
\document: Added hook after		Saved in package	10
.aux files has been read but be-		\niib@swapfoot: Simplified code	10
fore document begins	17	\thanks: Altered redefinition	
\flushnotestack: Fixed bug		method	10
with empty stack	12	Redefinition in preamble	10
Renamed from \flushcitestack			
.....	12	v1.4	
\niib@addtostack: Macro re-		General: ϵ -TeX made optional	
named from \niib@stackup	14	(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 419₂₁

\{	373	keytail (option)	3
\}	374		
\^	384		
B		L	
\bibliography	435	log (option)	3
\bibnote	2, 165, 181, 196, 204, 215	N	
\bibnotemark	2, 167, 182, 197, 205, 220	name (option)	3
\bibnotetext	2, 166, 183, 198, 206, 225	\niib@addtostack	282, 318, 322
\blx@bibfiles	429	\niib@auxhook	325, 329, 419
C		\niib@bibnote	218, 219, 291
cite (option)	2	\niib@checklists	466, 482
\citenote	3, 250	\niib@checkrerun	236, 445, 511
D		\niib@cite	104, 336
debug (option)	3	\niib@citelist	483, 489, 498
\document	399	\niib@debugfalse	51
E		\niib@debugtrue	64, 68
\endnote	154, 176	\niib@dochook	399, 419
\endnotemark	155, 176	\niib@etexfalse	143
endnotes (option)	2	\niib@etextrue	132
\endnotetext	156, 176	\niib@field	104, 364, 394
etex (option)	3	\niib@gobble	226, 337
F		\niib@headcitelist	398, 402, 403, 405
field (option)	3	\niib@headfalse	77, 99
\flushnotestack	4, 233, 512	\niib@headlist ...	294, 298, 299, 301
\footnote	149, 163	\niib@headtrue	81
footnote (option)	2	\niib@id	1
\footnotemark	150, 163	\niib@key	257, 363, 393
\footnotetext	151, 163	\niib@keyhead	104, 269
H		\niib@keyname	251, 363, 393
head (option)	3	\niib@keynone	104, 274
I		\niib@keytail	104, 272
\ifniib@debug	10, 35	\niib@list	483, 487, 500
\ifniib@endnotes ..	72, 180, 195, 203	\niib@log@debug	
\ifniib@etex	128, 355	33, 41, 44, 45, 168, 173,
\ifniib@files	314, 323, 328	185, 191, 200, 208, 239, 242, 253,
\ifniib@footnotes	72, 164	255, 283, 304, 346, 357, 387, 408,
\ifniib@head .	74, 89, 98, 258, 268, 296	433, 442, 451, 454, 462, 494, 505, 515
\ifniib@logmin	10, 15, 23, 29	\niib@log@err	13
\ifniib@lognone	10, 14, 22, 28, 34, 475	\niib@log@inf ...	13, 92, 100, 146, 210
\ifniib@rerun		\niib@log@warn	
.....	235, 444, 469, 493, 497, 504	13, 70, 87, 137, 142, 470, 472
\ifniib@tail .	74, 90, 97, 261, 271, 308	\niib@logminfalse	52
K		\niib@logmintrue	60
keyhead (option)	3	\niib@lognonefalse	53
keynone (option)	3	\niib@lognonetrue	56
		\niib@mark	223, 224, 250, 292, 295
		\niib@msg	277, 349
		\niib@name	104, 214
		\niib@next	226, 292, 337, 367, 397
		\niib@normcite	311, 320, 332, 334, 336

