

notes2bib — Integrating notes into the bibliography*

Joseph Wright[†]

Released 2008/02/26

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 REV\TeX approach	6
2 Basic use	2	6.3 The <code>notes2bib</code> approach	7
2.1 Package control	2	7 The package code	7
2.2 Output of notes	4	7.1 Package setup code . . .	7
2.3 Cross-referencing notes	4	7.2 Logging	8
2.4 Interaction with other packages	5	7.3 Package options	9
3 Special effects	5	7.4 Footnote and endnote handling	11
4 Package requirements	5	7.5 User macros	12
5 Known issues	6	7.6 Internal macros	14
6 The mechanism	6	7.7 Finalisation	18
6.1 The <code>endnotes</code> approach	6	8 Change History	20
		9 Index	21
		10 Notes	23

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

*This file describes version v1.3c, last revised 2008/02/26.

[†]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]. `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 note is not “fragile,” and so it should not be necessary to use `\bibnotemark` directly. `\bibnotetext` 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. `It is hard to write a good example for this \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

Almost all of the package options take literal text; those which do not are true/false switches.

- `cite`: The csname of the macro used to cite bibnotes; by default “cite”.
- `endnotes`: Whether to convert endnotes into bibnotes; takes a Boolean value.
- `field`: The BibTeX database field name for notes.
- `footnotes`: Whether to convert footnotes into bibnotes; takes a Boolean value, and does not affect the `\thanks` macro.
- `keyhead`, `keynone`, `keytail`: Sorted BibTeX styles can use the `key` field to sort citations; by setting a prefix to the bibnote name, extra control over sorting can be obtained.
- `log`: The amount of detail to add to the log; expects a value from the list `debug` (very detailed information), `verbose` (the same as `debug`), `normal`, `errors` (errors only), `none`.
- `name`: The name given to bibnote citations; by default, this is followed by an automatically-generated number.
- `prefix`: The file-name prefix used for the BibTeX database holding the notes.
- `presorthead`, `presortnone`, `presorttail`: For biblatex users, the `pre-sort` field can be used to control sorting of the bibliography; these keys control the value used depending on the setting of the `sort` key.
- `record`: The name of the BibTeX record type used to store notes.
- `sort`: Controls the placement of notes relative to real citations in the bibliography; expected a value `none` (no control of sorting, intended for unsorted bibliographies and interspersed citations and notes), `head` (notes appear before real citations) and `tail` (notes appear after real citations).

The default options are:

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

```

\niibsetup{%
  cite=cite,
  endnotes=false,
  field=note,
  footnotes=false,
  keyhead=aaa,
  keynone={},
  keytail=zzz,
  log=normal,
  name=Bibnote,
  prefix=niib-,
  presorthead=ml,
  presortnone=mm,
  presorttail=mn,
  record=Misc,
  sort=none}

```

The `sort` option requires some comment. By default, `notes2bib` places notes where cited into the `.aux` file, which means that the order of citations and notes depends on the `.bst` file in use. With an unsorted style, citations and notes will be mixed in the order they appear in the \LaTeX source. The `sort=head` option will cause `notes2bib` to place notes before real citations. This should work with sorted and unsorted `.bst` files, but requires two \LaTeX runs *before* a \BibTeX run in order to work. The package warns if extra runs are needed. The `sort=tail` option places notes after citations; here, only one \LaTeX run is needed.

The options `head` and `tail` are provided as shortcuts for `sort=head` and `sort=tail`, respectively. The `debug` option is a shortcut for `log=debug`. As the `endnotes` and `footnotes` options take Boolean values, giving the option name alone is the same as giving `\option=true`.

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 `\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.

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

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

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
```

4 Package requirements

`notes2bib` has certain requirements to run successfully; if these are not met the package will abort loading.

- ϵ -TeX: The package uses the ϵ -TeX `\unexpanded` primitive, and so the extensions must be available.
- `xkeyval`, v2.5 or later: Option handling uses `xkeyval`, and the features used here are only available from v2.5 of that package.

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

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

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.

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.

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>

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 BibT_EX 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 BibT_EX. After the BibT_EX run, the note text will appear in the .bbl file, in the same way as any other citation. Using an unsorted BibT_EX 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 BibT_EX 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

The package starts with the usual identification code.

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{notes2bib}
3 [2008/02/26 v1.3c Integrating notes into the bibliography]
```

The package requires ε-T_EX, so before going any further, this is tested. This code is taken more-or-less verbatim from biblatex.

```
4 \begingroup
5   \@ifundefined{eTeXversion}
6   {\PackageError{notes2bib}
7     {Not running under e-TeX}
8     {This package requires e-TeX. Try compiling the document
9       with\MessageBreak 'elatex' instead of 'latex'. When using
10      pdfTeX, try 'pdfelatex'\MessageBreak instead of 'pdflatex'}}
11   \endgroup\endinput}
12 {\endgroup}
```

The necessary support packages are loaded. The method used in biblatex is used to check the date of the packages: notes2bib bails out if the support packages are too old.

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

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

```

13 \RequirePackage{xkeyval}
14 \@ifpackagelater{xkeyval}{2005/05/07}
15 {}
16 {\PackageError{notes2bib}
17   {xkeyval >= 2.5 required}
18   {notes2bib requires the 'xkeyval' package, version 2.5 or
19     later.\MessageBreak The version loaded is:
20     '\@nameuse{ver@etoolbox.sty}'.\MessageBreak
21     This is a fatal error; the package will abort.}%
22   \endinput}

```

\niib@tempa Some private temporary macros are declared.

```

\niib@tempb 23 \newcommand*{\niib@tempa}{}
24 \newcommand*{\niib@tempb}{}

```

7.2 Logging

\ifniib@debug To control logging, some new switches are declared.

```

\niib@logmin 25 \newif\ifniib@debug
\niib@lognone 26 \newif\ifniib@logmin
27 \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.

```

28 \newcommand*{\niib@log@err}[2]{%
29   \ifniib@lognone\else
30     \ifniib@logmin
31       \PackageWarning{notes2bib}{#1}%
32     \else
33       \PackageError{notes2bib}{#1}{#2}%
34     \fi
35   \fi}
36 \newcommand*{\niib@log@warn}[1]{%
37   \ifniib@lognone\else
38     \ifniib@logmin\else
39       \PackageWarning{notes2bib}{#1}%
40     \fi
41   \fi}
42 \newcommand*{\niib@log@inf}[1]{%
43   \ifniib@lognone\else
44     \ifniib@logmin\else
45       \PackageInfo{notes2bib}{#1}%
46     \fi
47   \fi}

```

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

```

48 \newcommand*{\niib@log@debug}[1]{%
49   \ifniib@lognone\else
50     \ifniib@debug
51       \PackageInfo{notes2bib}{#1}%
52     \fi
53   \fi}

```

7.3 Package options

<code>\niib@opt@boolkey</code>	<p>To aid maintenance, some shortcuts are defined for generating keys. These also allow the debugging messages to be added automatically to every key.</p> <pre> 54 \newcommand*{\niib@opt@boolkey}[2][\% 55 \define@boolkey[niib]{opt}[niib@]{#2}[true] 56 {#1\niib@log@debug{Option #2 set to ##1}} </pre>
<code>\niib@opt@choicekey</code>	<p>A “fill in the blanks” choice key. In all cases, <code>\niib@tempa</code> is used to hold the value given to the key, so that <code>\ifx</code> testing can occur.</p> <pre> 57 \newcommand*{\niib@opt@choicekey}[5][\% 58 \define@choicekey*+[niib]{opt}{#2}{\niib@tempa}{#3}[#1] 59 {#4\niib@log@debug{Option #2 set to ##1}} 60 {#5\niib@log@debug{Option #2 set to ##1}} </pre>
<code>\niib@opt@cmdkeys</code>	<p>A shortcut for <code>xkeyval</code> command keys.</p> <pre> 61 \newcommand*{\niib@opt@cmdkeys}[1]{% 62 \define@cmdkeys[niib]{opt}[niib@]{#1}} </pre>
<code>\niibsetup</code>	<p>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.</p> <pre> 63 \newcommand*{\niibsetup}[1]{% 64 \iffalse{\fi\ifnum0=}\fi 65 \setkeys[niib]{opt}{#1}% 66 \ifnum0=}\fi\iffalse}\fi} </pre>
<code>\niib@opt@log</code> <code>\niib@tempa</code> <code>\niib@tempb</code>	<p>The <code>xkeyval</code> package option for logging is declared. This is then processed to set the switches correctly.</p> <pre> 67 \niib@opt@choicekey[normal]{log}{debug,verbose,normal,errors,none} </pre> <p>A series of comparisons are made to assign the logging mode. The <code>normal</code> option is not tested, as executing the option sets the switches appropriately.</p> <pre> 68 {\niib@debugfalse 69 \niib@logminfalse 70 \niib@lognonefalse 71 \renewcommand*{\niib@tempb}{none}% 72 \ifx\niib@tempa\niib@tempb 73 \niib@lognonetrue 74 \fi 75 \renewcommand*{\niib@tempb}{minimal}% 76 \ifx\niib@tempa\niib@tempb 77 \niib@logmintrue 78 \fi 79 \renewcommand*{\niib@tempb}{debug}% 80 \ifx\niib@tempa\niib@tempb 81 \niib@debugtrue 82 \fi 83 \renewcommand*{\niib@tempb}{verbose}% 84 \ifx\niib@tempa\niib@tempb 85 \niib@debugtrue 86 \fi} </pre> <p>The option has not been recognised: give a warning (if appropriate).</p> <pre> 87 {\niib@log@warn{Unrecognised value `#1' for option log}} </pre>

<code>\niib@opt@debug</code>	A quick method to set <code>log=debug</code> . 88 <code>\niib@opt@boolkey{debug}</code>
<code>\niib@opt@footnotes</code>	The footnote and endnote options are declared here.
<code>\niib@opt@endnotes</code>	89 <code>\niib@opt@boolkey[\niib@swapfoot]{footnotes}</code>
<code>\ifniib@footnotes</code>	90 <code>\niib@opt@boolkey[\niib@swapend]{endnotes}</code>
<code>\ifniib@endnotes</code>	Switches are needed for placing notes before and after normal citations.
<code>\ifniib@head</code>	
<code>\ifniib@tail</code>	91 <code>\newif\ifniib@tail</code> 92 <code>\newif\ifniib@head</code>
<code>\niib@opt@sort</code>	This option controls the position of notes <i>versus</i> normal citations. The <code>xkeyval</code> option replaces the earlier <code>head</code> and <code>tail</code> options, which are retained for backward compatibility. ¹⁰
<code>\niib@tempa</code>	
<code>\niib@tempb</code>	93 <code>\niib@opt@choicakey[none]{sort}{none,head,tail}</code> 94 <code>{\niib@headfalse</code> 95 <code>\niib@tailfalse</code> 96 <code>\renewcommand*{\niib@tempb}{head}%</code> 97 <code>\ifx\niib@tempa\niib@tempb</code> 98 <code>\niib@headtrue</code> 99 <code>\fi</code> 100 <code>\renewcommand*{\niib@tempb}{tail}%</code> 101 <code>\ifx\niib@tempa\niib@tempb</code> 102 <code>\niib@tailtrue</code> 103 <code>\fi}</code> 104 <code>{\niib@log@warn{Unrecognised value ‘#1’ for option sort}}</code>
<code>\niib@opt@head</code>	The back-compatibility code; unlike earlier versions, this will take whatever the <code>sort-type</code> key is given.
<code>\niib@opt@tail</code>	105 <code>\niib@opt@boolkey[%</code> 106 <code>\ifniib@head</code> 107 <code>\ifniib@tail</code> 108 <code>\niib@tailfalse</code> 109 <code>\niib@log@inf{Option head cancels existing\MessageBreak</code> 110 <code>tail or sort=tail option}</code> 111 <code>\fi</code> 112 <code>\fi}{head}</code> 113 <code>\niib@opt@boolkey[%</code> 114 <code>\ifniib@tail</code> 115 <code>\ifniib@head</code> 116 <code>\niib@headfalse</code> 117 <code>\niib@log@inf{Option tail cancels existing\MessageBreak</code> 118 <code>head or sort=head option}</code> 119 <code>\fi</code> 120 <code>\fi}{tail}</code>
<code>\niib@opt@cite</code>	The various internal control values are set up as command keys.
<code>\niib@cite</code>	121 <code>\niib@opt@cmdkeys{%</code>
<code>\niib@opt@name</code>	122 <code>cite,</code>
<code>\niib@name</code>	123 <code>name,</code>
<code>\niib@opt@prefix</code>	124 <code>prefix,</code>
<code>\niib@prefix</code>	¹⁰ This may change in a future release; the options <code>head</code> and <code>tail</code> are “deprecated.”
<code>\niib@opt@record</code>	
<code>\niib@record</code>	
<code>\niib@opt@field</code>	
<code>\niib@field</code>	
<code>\niib@opt@presorthhead</code>	
<code>\niib@presorthhead</code>	
<code>\niib@opt@presortnone</code>	
<code>\niib@presortnone</code>	
<code>\niib@opt@presorttail</code>	
<code>\niib@presorttail</code>	
<code>\niib@opt@presorthhead</code>	

```

125 record,
126 field,
127 presorthhead,
128 presortnone,
129 presorttail,
130 keyhead,
131 keynone,
132 keytail}
133 \niibsetup{%
134 cite=cite,
135 name=Bibnote,
136 prefix=niib-,
137 record=Misc,
138 field=note,
139 presorthhead=ml,
140 presortnone=mm,
141 presorttail=mn,
142 keyhead=aaa,
143 keynone={},
144 keytail=zzz}

```

7.4 Footnote and endnote handling

`\niib@org@footnote` To allow dynamic handling of footnotes and endnotes, the original definitions are backed up.

```

\niib@org@footnotemark
\niib@org@footnotetext 145 \let\niib@org@footnote\footnote
146 \let\niib@org@footnotemark\footnotemark
147 \let\niib@org@footnotetext\footnotetext

```

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

```

\niib@org@endnotemark 148 \AtBeginDocument{%
\niib@org@endnotetext 149 \@ifpackageloaded{endnotes}
\niib@org@theendnotes 150 {\let\niib@org@endnote\endnote
151 \let\niib@org@endnotemark\endnotemark
152 \let\niib@org@endnotetext\endnotetext
153 \let\niib@org@theendnotes\theendnotes}}

```

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

```

154 \@ifundefined{thanks}{}
155 {\renewcommand*{\thanks}[1]{%
156 \niib@org@footnotemark
157 \protected@xdef\@thanks{\@thanks
158 \protect\niib@org@footnotetext[\the\c@footnote]{#1}}}}

```

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

```

\footnote 159 \newcommand*{\niib@swapfoot}{%
\footnotemark 160 \ifniib@footnotes
\footnotetext 161 \let\footnote\bibnote
162 \let\footnotemark\bibnotetext
163 \let\footnotetext\bibnotemark
164 \niib@log@debug{Converting footnotes to bibnotes}%
165 \else
166 \let\footnote\niib@org@footnote

```

```

167 \let\footnotemark\niib@org@footnotemark
168 \let\footnotetext\niib@org@footnotetext
169 \niib@log@debug{Using kernel definition of footnotes}%
170 \fi}
171 \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 172 \newcommand*{\niib@swapend}{}
\endnotetext 173 \AtBeginDocument{%
\theendnotes 174 \@ifpackageloaded{endnotes}
175 {\renewcommand*{\niib@swapend}{%
176 \ifniib@endnotes
177 \let\endnote\bibnote
178 \let\endnotemark\bibnotemark
179 \let\endnotetext\bibnotetext
180 \let\theendnotes\printbibnotes
181 \niib@log@debug{Converting endnotes to bibnotes}%
182 \else
183 \let\endnote\niib@org@endnote
184 \let\endnotemark\niib@org@endnotemark
185 \let\endnotetext\niib@org@endnotetext
186 \let\theendnotes\niib@org@theendnotes
187 \niib@log@debug{Using endnotes package to handle endnotes}%
188 \fi}
189 \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.

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

```
208 \newcounter{bibnote}
209 \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`.

```
210 \DeclareRobustCommand*{\bibnote}{%
211   \stepcounter{bibnote}%
212   \@ifnextchar[%]
213     {\niib@bibnote}
214     {\niib@bibnote[\thebibnote]}}
```

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

```
215 \DeclareRobustCommand*{\bibnotemark}{%
216   \stepcounter{bibnote}%
217   \@ifnextchar[%]
218     {\niib@mark}
219     {\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.

```
220 \DeclareRobustCommand*{\bibnotetext}[1][\thebibnote]{%
221   \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`.

```
222 \AtBeginDocument{%
223   \@ifpackageloaded{biblatex}
224     {\let\printbibnotes\printbibliography}
225     {\DeclareRobustCommand*{\printbibnotes}
226       {\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.

```
227 \DeclareRobustCommand*{\flushnotestack}{%
228   \let\niib@taillist\niib@stack
229   \ifniib@rerun\else
230     \niib@checkrerun{tail}%
231     \fi
232   \ifx@empty\niib@stack@empty
233     \niib@log@debug{Citation stack empty: nothing for\MessageBreak
234       \noexpand\flushnotestack to do}%
235   \else%
236     \niib@log@debug{Flushing note citations to aux file}%
237     \if@filesw
238       \immediate\write\auxout{\string\niibtailcite\niib@stack}%
239     \fi
```

```

240 \expandafter\nocite\expandafter{\niib@stack}%
241 \gdef\niib@stack{}%
242 \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.

```

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

```

7.6 Internal macros

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

```

244 \AtBeginDocument{%
245   \ifpackageloaded{biblatex}
246     {\niib@log@debug{Using field 'keysort' for sorting key}%
247      \newcommand*\niib@keyname{keysort}}
248     {\niib@log@debug{Using field 'key' for sorting key}%
249      \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
250 \newcommand*\niib@presort{}%
251 \ifniib@head
252   \niib@presorthhead%
253 \else
254   \ifniib@tail
255     \niib@presorttail%
256   \else
257     \niib@presortnone%
258   \fi
259 \fi}
260 \newcommand*\niib@key{}%
261 \ifniib@head
262   \niib@keyhead%
263 \else
264   \ifniib@tail
265     \niib@keytail%
266   \else
267     \niib@keynone%
268   \fi
269 \fi}

```

`\niib@msg` To inform the user, the automatically-created Bib_T_EX database needs to carry suitable information on its source.

```

270 \edef\niib@msg{%
271   This is an auxiliary file used by the 'notes2bib' package.^J%
272   This file may safely be deleted. It will be recreated as
273   required.^J}

```

`\niib@stack` This macro is needed to store any citations at the end of the bibliography. Initially, this is empty.

```

274 \newcommand*\niib@stack{}

```

<code>\niib@addtostack</code> <code>\niib@tempa</code>	<p>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 <code>\flushnotestack</code>).</p> <pre> 275 \newcommand*{\niib@addtostack}[1]{% 276 \niib@log@debug{Adding citation #1\MessageBreak to 'tail' stack}% 277 \edef\niib@tempa{#1}% 278 \ifx\@empty\niib@stack\@empty 279 \xdef\niib@stack{\niib@tempa}% 280 \else 281 \xdef\niib@stack{\niib@stack,\niib@tempa}% 282 \fi}</pre>
<code>\niib@bibnote</code>	<p>Two steps are needed here, writing the text of the note to file (handled by <code>\niib@text</code>, and marking the citation (using <code>\niib@cite</code>).</p> <pre> 283 \long\def\niib@bibnote[#1]#2{% 284 \niib@text{#1}{#2}% 285 \niib@mark{#1}}</pre>
<code>\niib@headlist</code>	<p>To inform the user that a re-run of \LaTeX is needed, tracking is needed of any “head” citations.</p> <pre> 286 \newcommand*{\niib@headlist}{{}</pre>
<code>\niib@mark</code> <code>\niib@tempa</code>	<p>Adding a citation to the \LaTeX file is handled here. When using the <code>sort=head</code> option, the citation is written to the <code>.aux</code> file for sorting control. The normal citation command is then called.</p> <pre> 287 \def\niib@mark[#1]{% 288 \ifniib@head 289 \edef\niib@tempa{#1}% 290 \ifx\@empty\niib@headlist\@empty 291 \xdef\niib@headlist{\niib@tempa}% 292 \else 293 \xdef\niib@headlist{\niib@headlist,\niib@tempa}% 294 \fi 295 \if@filesw 296 \niib@log@debug{Adding citation #1 to list for next run}% 297 \immediate\write\@auxout{\string\niibheadcite{#1}}% 298 \fi 299 \fi}</pre> <p>When the <code>sort=tail</code> option is active, citation is handled by another macro, so a switch is needed.</p> <pre> 300 \ifniib@tail 301 \expandafter\niib@tailcite% 302 \else 303 \expandafter\niib@normcite% 304 \fi 305 {#1}}</pre>
<code>\ifniib@filesw</code>	<p>A switch is used to back up <code>\if@filesw</code>.</p> <pre> 306 \newif\ifniib@filesw</pre>
<code>\niib@tailcite</code>	<p>When using the <code>sort=tail</code> option, bibnote citation need to be stored for later. With <code>biblatex</code>, the <code>\AtEndCite</code> macro is available to provide a hook for the</p>

necessary switch. In other cases, the current value of `\if@files` is then saved, before turning it off and setting up the restore system.

```

307 \AtBeginDocument{%
308   \ifpackageloaded{biblatex}
309     {\newcommand{\niib@tailcite}[1]{%
310       \niib@addtostack{#1}%
311       \AtNextCite{\@filesfalse}%
312       \niib@normcite{#1}}}
313     {\newcommand{\niib@tailcite}[1]{%
314       \niib@addtostack{#1}%
315       \let\ifniib@files\if@files
316       \@filesfalse
317       \let\niib@auxhook\niib@restorefiles
318       \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.

```

319 \newcommand*{\niib@restorefiles}{%
320   \let\if@files\ifniib@files
321   \let\niib@auxhook\relax}
```

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

```

322 \AtBeginDocument{%
323   \ifpackageloaded{cite}
324     {\newcommand*{\niib@tcite}[1]{\niib@normcite{#1}}}
325     {\newcommand*{\niib@tcite}[1]{%
326       \niib@normcite{#1}%
327       \niib@restorefiles}}
```

`\niib@normcite` The normal citation command.

```

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

`\niib@text` The “business end” of writing the notes to file. This is a `\long` macro, so no star is used for `\newcommand`.

```

329 \newcommand{\niib@text}[2]{%
```

`\niib@out` If this is the first note, then a new output stream is needed, otherwise it will
`\niib@stream` already be open.

```

330   \ifundefined{niib@out}{%
331     \if@files
332       \newwrite\niib@out%
333       \gdef\niib@stream{\niib@prefix\jobname.bib}%
334       \niib@log@debug{Creating BibTeX database file \MessageBreak
335         \niib@stream\space to contain bibnotes}%
336       \immediate\openout\niib@out\niib@stream\relax
```

The new file starts with the message that it has been automatically generated by `notes2bib`.

```

337     \immediate\write\niib@out{\niib@msg}%
338     \fi}{%}
```

The new record is now written to file. The `\unexpanded` ϵ -TeX primitive is used to avoid expansion of macros in the note text. The only issue with this is the addition of spaces after command names; this is the reason verbatim text cannot be used in bibnotes.

```

339 \if@filesw
340   \niib@log@debug{Writing bibnote #1 contents
341     \MessageBreak---\MessageBreak#2\MessageBreak---\MessageBreak
342     to BibTeX database}%
343   \immediate\write\niib@out{%
344     @\niib@record\string{#1,^^J%
345     presort = \string{\niib@presort\string},^^J%
346     \niib@keyname\space= \string{\niib@key#1\string},^^J%
347     \niib@field\space= \string{\unexpanded{#2}\string}^^J%
348     \string}^^J}%
349 \fi}

```

`\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.

```

350 \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.

```

351 \g@addto@macro{\document}{\niib@dochook}
352 \newcommand*{\niibheadcite}[1]{%
353   \edef\niib@tempa{#1}%
354   \ifx\@empty\niib@headcitelist\@empty
355     \xdef\niib@headcitelist{\niib@tempa}%
356   \else
357     \xdef\niib@headcitelist{\niib@headcitelist,\niib@tempa}%
358   \fi
359 \if@filesw
360   \niib@log@debug{Adding citation #1 to start of .aux file}%
361 \fi
362 \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.

```

\niibtailcite
\niib@tempa
363 \newcommand*{\niib@tailcitelist}{}
364 \newcommand*{\niibtailcite}[1]{%
365   \edef\niib@tempa{#1}%
366   \ifx\@empty\niib@tailcitelist\@empty
367     \xdef\niib@tailcitelist{\niib@tempa}%
368   \else
369     \xdef\niib@tailcitelist{\niib@tailcitelist,\niib@tempa}%
370   \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.

```

371 \newcommand*{\niib@dochook}{%
372   \@ifundefined{@restore@auxhandle}
373     {\newcommand*{\@restore@auxhandle}{\niib@auxhook}}
374     {\ifx\relax\@restore@auxhandle\relax
375       \newcommand*{\@restore@auxhandle}{\niib@auxhook}%
376     \else
377       \g@addto@macro{\@restore@auxhandle}{\niib@auxhook}%
378     \fi}}
379 \newcommand*{\niib@auxhook}{}
380 \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.

```

381 \AtBeginDocument{%
382   \@ifpackageloaded{biblatex}%
383     {\expandafter\gappto\expandafter\blx@bibfiles\expandafter%
384       {,\niib@prefix\jobname}
385     \niib@log@debug{Added bibnotes database to biblatex file 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.

```

386   {\let\niib@org@bib\bibliography
387    \renewcommand*{\bibliography}[1]{%
388      \ifnum\the\value{bibnote} > \z@
389        \niib@org@bib{\niib@prefix\jobname,#1}%
390      \else
391        \niib@org@bib{#1}%
392      \fi}
393    \niib@log@debug{Added bibnote database to
394    \noexpand\bibliography}}

```

7.7 Finalisation

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

```

395 \newif\ifniib@rerun

```

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

```

396 \newcommand*{\niib@checkrerun}[1]{%
397   \niib@rerunfalse
398   \expandafter\ifx\expandafter\@empty\csname niib@#1list\endcsname%
399   \@empty
400   \expandafter\ifx\expandafter\@empty\csname niib@#1citelist%
401   \endcsname\@empty
402   \niib@log@debug{No ‘#1’ notes detected}%
403   \else
404   \niib@reruntrue

```

```

405     \niib@log@debug{No '#1' notes found this run\MessageBreak
406         but .aux files contained the '#1' requests:\MessageBreak
407         \csname niib@#1citelist\endcsname}%
408     \fi
409 \else
410     \expandafter\ifx\expandafter\@empty\csname niib@#1citelist%
411         \endcsname\@empty
412         \niib@reruntrue
413         \niib@log@debug{No '#1' requests in .aux file\MessageBreak
414             but '#1' notes in this run:\MessageBreak
415             \csname niib@#1list\endcsname}%
416     \else

```

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

```

417     \niib@checklists{#1}%
418 \fi
419 \fi
420 \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.

```

421     \niib@log@warn{Rerun LaTeX to get correct \MessageBreak
422         '#1' notes}%
423     \niib@log@warn{Please (re)run BibTeX on the file(s):
424         \MessageBreak\jobname.aux
425         \MessageBreak and rerun LaTeX afterwards.}%
426 \ifniib@lognone\else
427     \typeout{%
428         REQ:3:latex:REQ^^J%
429         REQ:2:bibtex:REQ^^J%
430         REQ:1:latex:REQ}%
431 \fi
432 \fi}

```

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

`\niib@tempa`
`\niib@tempb`

```
433 \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`

```

434 \expandafter\edef\expandafter\niib@list\expandafter%
435     {\csname niib@#1list\endcsname}%
436 \expandafter\edef\expandafter\niib@citelist\expandafter%
437     {\csname niib@#1citelist\endcsname}%

```

The loops can now being.

```

438 \@for\niib@tempa:=\niib@list\do{%
439     \niib@reruntrue
440     \@for\niib@tempb:=\niib@citelist\do{%
441         \ifx\niib@tempa\niib@tempb
442             \niib@rerunfalse
443         \fi}

```

```

444 \ifniib@rerun
445 \niib@log@debug{Note \niib@tempa\space is a '#1' note
446 \MessageBreak but request not in .aux file}%
447 \fi}
448 \ifniib@rerun\else
449 \@for\niib@tempa:=\niib@citelist\do{%
450 \niib@reruntrue
451 \@for\niib@tempb:=\niib@list\do{%
452 \ifx\niib@tempa\niib@tempb
453 \niib@rerunfalse
454 \fi}
455 \ifniib@rerun
456 \niib@log@debug{Note \niib@tempa\space is set to '#1' in
457 .aux\MessageBreak file but is not a '#1' note}%
458 \fi}
459 \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.

```

460 \AtEndDocument{%
461 \niib@rerunfalse
462 \niib@checkrerun{head}%
463 \flushnotestack%
464 \@ifundefined{niib@out}{}
465 {\immediate\closeout\niib@out%
466 \niib@log@debug{Closed BibTeX database file\MessageBreak
467 \niib@stream}}}

```

Options are processed at the end of the package, to avoid any odd issues arising with definition of macros.

```

468 \ProcessOptionsX[niib]<opt>

```

8 Change History

v1.0		<code>\bibnotemark</code> : Macro made ro-
General: Initial public release 1	bust 13
v1.0a		<code>\bibnotetext</code> : Macro made ro-
<code>\citenote</code> : New macro 14	bust 13
v1.1		<code>\citenote</code> : Macro made robust . 14
General: <code>\Percent</code> macro re-		<code>\flushnotestack</code> : Macro made
moved 1	robust 13
Documentation improvements	.. 1	<code>\niib@text</code> : Combined
Improvements to documentation		<code>\niib@text</code> and <code>\niib@@text</code>
and <code>dtx</code> file 1 16
License changed from GPL to		Improved awareness of files
LPPL 1	switch 16
Removed use of <code>xspace</code> 7	<code>\printbibnotes</code> : Macro made ro-
Require ϵ -T _E X extensions 7	bust 13
Several code sections re-factored	1 v1.2	
<code>\bibnote</code> : Macro made robust	.. 13	General: Altered implementation of

head and tail options to allow moving of superscript citations .	<i>1</i>	\niib@tempb: New macro	<i>8</i>
\blx@bibfiles: Fixed bug with biblatex support	<i>18</i>	\niibheadcite: New macro ...	<i>17</i>
v1.3		\niibsetup: New macro	<i>9</i>
General: Added xkeyval option interface	<i>1</i>	v1.3a	
All options now work anywhere in input	<i>1</i>	General: Require at least v1.3 of etoolbox	<i>7</i>
Fixed serious errors with head and tail implementation	<i>1</i>	Require at least v2.5 of xkeyval ..	<i>7</i>
\bibliography: Modification now at beginning of document	<i>18</i>	v1.3b	
\blx@bibfiles: Modification now at beginning of document	<i>18</i>	\flushnotestack: Added check for correct tail citations in .aux file	<i>13</i>
Updated for biblatex v0.7: moved to end of preamble, switch from \bib@gadd to \gappto	<i>18</i>	Writes stack to .aux file	<i>13</i>
\citenote: Now a wrapper for \niib@mark	<i>14</i>	\niib@checkrerun: Added ability to check for both head and tail re-runs	<i>18</i>
\document: Added hook after .aux files has been read but before document begins	<i>17</i>	Bug fix with logging for debug logging	<i>18</i>
\flushnotestack: Fixed bug with empty stack	<i>13</i>	\niib@citelist: New macro ..	<i>19</i>
Renamed from \flushcitestack	<i>13</i>	\niib@list: New macro	<i>19</i>
\niib@addtostack: Macro renamed from \niib@stackup	<i>15</i>	\niib@tailcitelist: New macro	<i>17</i>
\niib@auxhook: New macro ...	<i>18</i>	\niib@taillist: New macro ..	<i>20</i>
\niib@bibnote: Use \niib@mark for citation	<i>15</i>	\niibtailcite: New macro ...	<i>17</i>
\niib@dochook: New macro ...	<i>18</i>	\printbibnotes: Definition moved to beginning of document	<i>13</i>
\niib@key: Dynamic rather than static definition	<i>14</i>	v1.3c	
\niib@keyname: Moved to \AtBeginDocument	<i>14</i>	General: Dropped requirement for etoolbox	<i>7</i>
\niib@presort: Dynamic rather than static definition	<i>14</i>	\niib@org@footnote: Saved in package	<i>11</i>
\niib@tempa: New macro	<i>8</i>	\niib@org@footnotemark: Saved in package	<i>11</i>
		\niib@org@footnotetext: Saved in package	<i>11</i>
		\niib@swapfoot: Simplified code	<i>11</i>
		\thanks: Altered redefinition method	<i>11</i>
		Redefinition in preamble	<i>11</i>

9 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	B	178, 192, 200, <u>215</u>
\@restore@auxhandle	\bibliography ... <u>386</u>	\bibnotetext <i>2</i> , <u>162</u> , 179, 193, 201, <u>220</u>
\@thanks	\bibnote ... <i>2</i> , <u>161</u> , 177, 191, 199, <u>210</u>	\blx@bibfiles ... <u>381</u>
	\bibnotemark <i>2</i> , <u>163</u> ,	

C	\niib@debugfalse . 68	\niib@opt@keyhead <u>121</u>
\c@footnote 158	\niib@debugtrue 81, 85	\niib@opt@keynone <u>121</u>
\citenote 4, <u>243</u>	\niib@dochook <u>351</u> , <u>371</u>	\niib@opt@keytail <u>121</u>
D	\niib@field .. <u>121</u> , 347	\niib@opt@log 67
\document <u>351</u>	\niib@headcitelist	\niib@opt@name .. <u>121</u>
E	. 350, 354, 355, 357	\niib@opt@endnotes <u>89</u>
\endnote 150, <u>172</u>	\niib@headfalse 94, 116	\niib@opt@prefix <u>121</u>
\endnotemark . 151, <u>172</u>	\niib@headlist ..	\niib@opt@presorthead
\endnotetext . 152, <u>172</u>	. 286, 290, 291, 293 <u>121</u>
F	\niib@headtrue ... 98	\niib@opt@presortnone
\flushnotestack .	\niib@key 250, 346 <u>121</u>
..... 5, <u>227</u> , 463	\niib@keyhead <u>121</u> , 262	\niib@opt@presorttail
\footnote 145, <u>159</u>	\niib@keyname 244, 346 <u>121</u>
\footnotemark 146, <u>159</u>	\niib@keynone <u>121</u> , 267	\niib@opt@record <u>121</u>
\footnotetext 147, <u>159</u>	\niib@keytail <u>121</u> , 265	\niib@opt@sort ... <u>93</u>
I	\niib@list 434, 438, 451	\niib@opt@tail .. <u>105</u>
\ifniib@debug . 25, 50	\niib@log@debug .	\niib@org@bib 226, <u>386</u>
\ifniib@endnotes 48, 56, 59,	\niib@org@endnote
.. 89, 176, 190, 198	60, 164, 169, 181, 148, 183
\ifniib@files ..	187, 195, 203, 233,	\niib@org@endnotemark
..... 306, 315, 320	236, 246, 248, 148, 184
\ifniib@footnotes	276, 296, 334,	\niib@org@endnotetext
..... 89, 160	340, 360, 385, 148, 185
\ifniib@head 91, 106,	393, 402, 405,	\niib@org@footnote
115, 251, 261, 288	413, 445, 456, 466 145, 166
\ifniib@logmin ..	\niib@log@err 28	\niib@org@footnotemark
..... 25, 30, 38, 44	\niib@log@inf 145, 156, 167
\ifniib@lognone 25,	.. 28, 109, 117, 205	\niib@org@footnotetext
29, 37, 43, 49, 426	\niib@log@warn 145, 158, 168
\ifniib@rerun ...	28, 87, 104, 421, 423	\niib@org@theendnotes
..... 229, <u>395</u> ,	\niib@logminfalse 69 148, 186
420, 444, 448, 455	\niib@logmintrue . 77	\niib@out <u>330</u> , 343, 465
\ifniib@tail 91, 107,	\niib@lognonefalse	\niib@prefix . <u>121</u> ,
114, 254, 264, 300 70	226, 333, 384, 389
N	\niib@lognonetrue 73	\niib@presort 250, 345
\niib@addtostack	\niib@mark ... 218,	\niib@presorthead
..... 275, 310, 314	219, 243, 285, 287 121, 252
\niib@auxhook ...	\niib@msg 270, 337	\niib@presortnone
..... 317, 321, <u>371</u>	\niib@name ... 121, 209 121, 257
\niib@bibnote ...	\niib@normcite 303,	\niib@presorttail
..... 213, 214, <u>283</u>	312, 324, 326, <u>328</u> 121, 255
\niib@checklists	\niib@opt@boolkey	\niib@record . <u>121</u> , 344
..... 417, <u>433</u>	. 54, 88–90, 105, 113	\niib@rerunfalse
\niib@checkrerun	\niib@opt@choicekey	.. 397, 442, 453, 461
..... 230, <u>396</u> , 462 57, 67, 93	\niib@reruntrue .
\niib@cite ... <u>121</u> , 328	\niib@opt@cite .. <u>121</u>	. 404, 412, 439, 450
\niib@citelist ..	\niib@opt@cmdkeys	\niib@restorefiles ..
..... 434, 440, 449 61, 121 317, 319, 327
	\niib@opt@debug .. 88	\niib@stack 228, 232,
	\niib@opt@field . <u>121</u>	238, 240, 241,
	\niib@opt@footnotes	274, 278, 279, 281
 89	\niib@stream . <u>330</u> , 467
	\niib@opt@head .. <u>105</u>	\niib@swapend . 90, <u>172</u>

<code>\niib@swapfoot</code>	89, <u>159</u>	58, <u>67</u> , <u>93</u> , <u>275</u> ,	P
<code>\niib@tailcite</code>	..	<u>287</u> , <u>351</u> , <u>363</u> , <u>433</u>	<code>\printbibnotes</code> ..
.	<u>301</u> , <u>307</u> , 309, 313	<code>\niib@tempb</code>	<u>4</u> , 180, 194, 202, <u>222</u>
<code>\niib@tailcitelist</code>	 <u>23</u> , <u>67</u> , <u>93</u> , <u>433</u>	
.....	<u>363</u>	<code>\niib@text</code> <u>221</u> , <u>284</u> , <u>329</u>	T
<code>\niib@tailfalse</code>	95, <u>108</u>	<code>\niibheadcite</code> ...	<code>\thanks</code> <u>154</u>
<code>\niib@taillist</code>	<u>228</u> , <u>460</u> <u>7</u> , <u>297</u> , <u>351</u>	<code>\thebibnote</code>
<code>\niib@tailtrue</code>	.. <u>102</u>	<code>\niibsetup</code> .. <u>2</u> , <u>63</u> , 133	<u>5</u> , <u>208</u> , 214, 219, 220
<code>\niib@tcite</code>	.. <u>318</u> , <u>322</u>	<code>\niibtailcite</code> <u>238</u> , <u>363</u>	<code>\theendnotes</code> . <u>153</u> , <u>172</u>
<code>\niib@tempa</code>	... <u>23</u> ,		

10 Notes

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