

xnotes2bib — Integrating notes into the bibliography*

Joseph Wright[†]

Released 2008/09/08

Abstract

The `xnotes2bib` package is an *experimental* re-implementation of `notes2bib` using $\text{\LaTeX}3$ commands internally (taken from the `expl3` bundle). The user interface is almost identical to `notes2bib`.

The `xnotes2bib` 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	5	The code	4
1 Introduction	1	5.1 Preliminaries	4
2 Using the package	2	5.2 User options	5
2.1 Macros	2	5.3 User macros	8
2.2 Control values	2	5.4 Macros for the .aux file . . .	10
3 Testing the package	3	5.5 Internal macros	11
4 Installation	4	5.6 Finalising	13

1 Introduction

The new code syntax introduced by $\text{\LaTeX}3$ is very different from plain \TeX or $\text{\LaTeX}2_{\epsilon}$. This means that converting packages to $\text{\LaTeX}3$ requires working from the ground up if

*This file describes version v0.0, last revised 2008/09/08.

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

the new ideas are to be exploited fully. This package re-implements `notes2bib` using the experimental syntax for internal commands.

The internal logic of the package remains broadly similar to that in `notes2bib`. However, some opportunities to simplify the code or make the flow clearer have been taken. There is therefore no one-to-one correspondence between the internals of the two packages. In particular, this package does not guarantee backward-compatibility with `notes2bib` under all circumstances. \LaTeX 3 will require significant changes to documents, and so alterations to the logic of packages in order to make them more useful in the long term seem sensible.

2 Using the package

2.1 Macros

<code>\bibnote</code>	The user macros of <code>xnotes2bib</code> are intended to work in the same way as those in <code>notes2bib</code> . Please see <code>notes2bib.pdf</code> for details of:
<code>\bibnotemark</code>	
<code>\bibnotetext</code>	
<code>\flushnotestack</code>	
	<ul style="list-style-type: none"> • <code>\bibnote</code> • <code>\bibnotemark</code> • <code>\bibnotetext</code> • <code>\flushnotestack</code>

The `\citenote` macro can be replaced by using `\bibnotemark` to cite the note for a second time.

2.2 Control values

<code>\niibsetup</code>	The main user changes between <code>notes2bib</code> and <code>xnotes2bib</code> are in the user options. These have been completely re-written here, to improve the logic of the interface and to remove unnecessary options. All package control is currently carried out using the <code>\niibsetup</code> macro, as the \LaTeX 2 ϵ kernel expands options and removes spaces. As in <code>notes2bib</code> , the <code>\niibsetup</code> macro accepts a key-value list as input.
-------------------------	---

There are four types of control key. Boolean keys take the values `true` and `false`. If the key name alone is given, this is the same as giving the `true` value. String keys save their value as a string for internal use: they can therefore take any input. Choice keys take one of a range of pre-determined values, in a similar way to Boolean options. However, the list of possible values varies from key to key. Finally, macro keys require a macro as the value.

<code>convert endnotes</code>	To convert endnotes and footnotes into bibnotes, the options <code>convert endnotes</code> and <code>convert footnotes</code>
<code>convert footnotes</code>	

	convert	footnotes are available. Both taken Boolean values, and are set to false when loading <code>xnotes2bib</code> . If endnotes are not defined elsewhere, the <code>convert endnotes</code> only defines them when set to true.
placement		The package is able to position notes in three ways in the bibliography. The standard method is to create the notes interspersed with “real” citations. The order in the source then determines the position in the bibliography. The package can also create notes such that they appear either before or after normal citations. The behaviour here is controlled by the <code>placement</code> option, which takes a choice from the list <code>after</code> , <code>before</code> and <code>mixed</code> . The <code>after</code> option places notes after normal citations, <code>before</code> places notes first and <code>mixed</code> mixes notes and citations based on the order in the source.
key after		Most sorted bibliography styles provide methods for controlling the sort order of entries. With standard \LaTeX styles, the <code>key</code> field is used for this purpose. The options <code>key after</code> , <code>key before</code> and <code>key mixed</code> are used to set this up. They contain an item to be added before the note label when creating the key field. The standard values are <code>aaa</code> for <code>key before</code> and <code>zzz</code> for <code>key after</code> : <code>key mixed</code> is empty. When using <code>biblatex</code> , <code>presort</code> field is available. The values used by <code>xnotes2bib</code> for this field are stored as <code>presort after</code> , <code>presort before</code> and <code>presort mixed</code> , with default values <code>mn</code> , <code>ml</code> and <code>mm</code> , respectively. The <code>presort</code> system is described in the <code>biblatex</code> manual.
key before		
key mixed		
presort after		
presort before		
presort mixed		
cite		As <code>xnotes2bib</code> uses \LaTeX to create notes, it uses a citation command to mark the notes in the body of the text. The command used is controlled by the macro <code>key cite</code> . The standard value is <code>\cite</code> .
field		Four control values are available to affect low-level aspects of the creation of the notes database. The <code>prefix</code> option sets the string used by <code>xnotes2bib</code> to name the database, and is set to <code>niib-</code> initially. The <code>field</code> and <code>record</code> options contain the name of the field used for the note, and the \LaTeX record type used. The standard values are <code>note</code> and <code>Misc</code> , respectively. Finally, the <code>notename</code> option sets the text used in addition to a number when automatically labelling notes. This is set to <code>Bibnote</code> : there is no space between this text and the number as this will confuse \LaTeX .
notename		
prefix		
record		
debug		The <code>debug</code> Boolean option is available for understanding what the package is doing. When set to true, the package writes additional information to the log to aid problem solving.

3 Testing the package

This release of `xnotes2bib` has several aims, all related to testing. Firstly, it provides a method to examine new ideas for `notes2bib`: feedback on the altered options is welcome. Second, it is a test-bed for using `expl3` to create packages. People familiar with the current \LaTeX ideas are encouraged to read the source and make suggestions. The new coding environment requires a more structured approach than $\text{\LaTeX} 2_{\epsilon}$, and so it is possible ideas have been missed.

In order to use `xnotes2bib`, you will need an up-to-date \TeX system and recent copies of `expl3` and `xparse`. The package is tested with the current SVN release of the \LaTeX

packages, but these may change on a day-to-day basis. So testers should be prepared for problems such as undefined macros from the new kernel! The ε -TeX extensions to T_EX are required by xnotes2bib.

Due to the experimental nature of the package, it currently has no version number. The package date should be used to check if versions are identical. That said, the version of notes2bib will be stepped when new copies of xnotes2bib are created.

While the package is experimental, it should work properly as a replacement for notes2bib, with the proviso that the options are very different. Any problems of function should be reported as normal.

4 Installation

The entire bundle is supplied with the TDS-ready .zip file notes2bib.tds.zip. Simply unzip this into your local texmf tree and run your hash program (texhash for T_EXLive or initexmf -u for MiK_TE_X)

To extract the package xnotes2bib.sty from xnotes2bib.dtx, run (pdf)L^AT_EX on the later. This will also create the documentation and README.txt file. Three (pdf)L^AT_EX runs with \write18 enabled will also build the index and table of contents in the .pdf.

5 The code

5.1 Preliminaries

The package starts by loading the basics of expl3. To keep things clear, first l3names is loaded to do the basic identification, and then everything else is started up. This is based on Morten Høgholm's xfrac package lead-off.

```

1 \NeedsTeXFormat{LaTeX2e}
2 \begingroup
3   \@ifundefined{eTeXversion}{%
4     \PackageError{xnotes2bib}
5       {Not running under e-TeX}
6       {This package requires e-TeX: enable the extensions!}%
7   \endgroup\endinput
8 }{%
9   \endgroup
10 }
11 \RequirePackage{l3names}
12 \ProvidesExplPackage
13   {xnotes2bib}{2008/09/08}{0.0}
14   {Integrating notes into the bibliography (expl3 version)}
15 \RequirePackage{expl3,xparse}

```

As the `expl3` syntax is in use, there is no need to worry about white space in the code. So there are very few `%` characters in the following. As far as possible, `expl3` or `xparse` macros are used here; however, a lot of the user-space $\text{\LaTeX}2_\epsilon$ macros should still be valid in $\text{\LaTeX}3$. There are also a few `@` macros where things are tidied closely to the $\text{\LaTeX}2_\epsilon$ kernel at present (for example `\@files`).

As the $\text{\LaTeX}2_\epsilon$ kernel expands package options, for the moment `xnotes2bib` does not accept any load-time options.

```

16 \DeclareOption*{
17   \PackageWarning{xnotes2bib}{
18     All~package~control~should~be~given~using~the~\MessageBreak
19     \token_to_string:N\niibsetup\space macro
20   }
21 }
22 \ProcessOptions\scan_stop:

```

`\g_niib_note_int` A counter for the automatically-created notes is needed. This is a global value (life will get very complicated if not).

```

23 \int_new:N \g_niib_note_int

```

`\niib_thenote:` The counter needs to be expressed as a usable name. A check is made so that if there are more than ten notes then the number is padded.

```

24 \def_new:NNn \niib_thenote: 0 {
25   \niib_notename:
26   \num_compare:nNnT {\g_niib_totalnotes_int} > {\c_nine}{
27     \num_compare:nNnF {\g_niib_note_int} > {\c_nine}{
28       0
29     }
30   }
31   \int_to_arabic:n \g_niib_note_int
32 }

```

5.2 User options

`\g_niib_KV_bool_clist` To be going on with, the key-defining macros are created using `keyval`.

```

\niib_KV_bool:nT
\niib_KV_choice:nnT
\niib_KV_cs_n:n
\niib_KV_string:n
33 \RequirePackage{keyval}
34 \clist_new:N\g_niib_KV_bool_clist
35 \clist_put_left:Nn\g_niib_KV_bool_clist{true,false}
36 \def_new:NNn\niib_KV_bool:nT 2 {
37   \define@key{niib}{#1}[true]{
38     \clist_if_in:cnTF{\g_niib_KV_bool_clist}{##1}{
39       #2
40     }{
41       \PackageError{xnotes2bib}{

```

```

42     Unknown~choice~'##1'
43   }{
44     The~Boolean~option~'##1'~accepts~only~the~\MessageBreak
45     values~true~and~false
46   }
47 }
48 }
49 }
50 \def_new:NNn\niib_KV_choice:nnT 3 {
51   \cs_if_free:cTF{g_niib_KV_#1_clist}{
52     \clist_new:c{g_niib_KV_#1_clist}
53   }{
54     \clist_gclear:c{g_niib_KV_#1_clist}
55   }
56   \clist_put_left:cn{g_niib_KV_#1_clist}{#2}
57   \define@key{niib}{#1}{
58     \clist_if_in:cnTF{g_niib_KV_#1_clist}{##1}{
59       #3
60     }{
61       \PackageError{xnotes2bib}{
62         Unknown~choice~'##1'
63       }{
64         The~option~'##1'~accepts~only~the~values:\MessageBreak
65         #2
66       }
67     }
68   }
69 }
70 \def_new:NNn\niib_KV_cs_n:n 1 {
71   \define@key{niib}{#1}{
72     \def:cNn{niib_#1:n} 0 {##1}
73   }
74 }
75 \def_new:NNn\niib_KV_string:n 1 {
76   \define@key{niib}{#1}{
77     \def:cNn{niib_#1:} 0 {##1}
78   }
79 }

```

`\niib_cite:n` Options for storing macro names

```
80 \niib_KV_cs_n:n{n{cite}}
```

`\niib_field:` A series of value-storing are created.

`\niib_key after:`

`\niib_key before:`

`\niib_key mixed:`

`\niib_notename:`

`\niib_prefix:`

`\niib_presort after:`

`\niib_presort before:`

`\niib_presort mixed:`

`\niib_record:`

```

81 \niib_KV_string:n{n{field}}
82 \niib_KV_string:n{n{key~after}}
83 \niib_KV_string:n{n{key~before}}
84 \niib_KV_string:n{n{key~mixed}}

```

```

85 \niib_KV_string:n{notename}
86 \niib_KV_string:n{prefix}
87 \niib_KV_string:n{presort~after}
88 \niib_KV_string:n{presort~before}
89 \niib_KV_string:n{presort~mixed}
90 \niib_KV_string:n{record}

```

\niib_convert_endnotes: None of the Boolean keys actually work quite as simply as creating a switch. For footnote and endnote manipulation, the keys created here have to be altered later in any case.

```

91 \let_new:NN\niib_convert_endnotes:\scan_stop:
92 \let_new:NN\niib_convert_footnotes:\scan_stop:
93 \niib_KV_bool:nT{convert~endnotes}{
94   \let:Nc\niib_convert_endnotes:{niib_convert_endnotes_#1:}
95 }
96 \niib_KV_bool:nT{convert~footnotes}{
97   \let:Nc\niib_convert_footnotes:{niib_convert_footnotes_#1:}
98 }

```

\niib_debug:n The debug option is implemented as a choice, to save on some complexity.

```

\niib_debug_true:n
\niib_debug_false:n
99 \niib_KV_bool:nT{debug}{
100   \let:Nc\niib_debug:n{niib_debug_#1:n}
101 }
102 \def_new:NN\niib_debug_true:n 1 {
103   \PackageInfo{xnotes2bib}{#1}
104 }
105 \let_new:NN\niib_debug_false:n\use_none:n

```

\l_niib_after_bool The placement option sets one or other flag if needed.

```

\niib_before_bool
106 \bool_new:N\l_niib_after_bool
107 \bool_new:N\l_niib_before_bool
108 \niib_KV_choice:nnT{placement}{after,before,mixed}{
109   \bool_set_false:N\l_niib_before_bool
110   \bool_set_false:N\l_niib_after_bool
111   \cs_if_really_exist:cT{l_niib_#1_bool}{
112     \bool_set_true:c{l_niib_#1_bool}
113   }
114 }

```

All of the default values are set.

```

115 \setkeys{niib}{
116   cite          = \cite,
117   debug         = false,
118   field         = note,
119   key~after     = zzz,

```

```

120 key-before      = aaa,
121 key-mixed       = {},
122 notename        = Bibnote,
123 placement       = mixed,
124 prefix          = niib-,
125 presort-after    = mn,
126 presort-before   = ml,
127 presort-mixed    = mm,
128 record          = Misc}

```

5.3 User macros

`\bibnote` [#1]: label

#2: text

The main user macro has to check if the automatic note number is needed. If it is, the optional argument is note used to `\bibnotetext` as this automatically assumes the current value of `\niib_thenote:`. The note text must be sorted out before the mark is made, as otherwise packages such as `cite` do not work correctly with punctuation. As the package is designed for \LaTeX 3, it is assumed \LaTeX is available and so `\scantokens` can be used on the mandatory argument. Hence everything is absorbed here without worrying about category codes.

```

129 \NewDocumentCommand{\bibnote}{o>{P}m}{
130   \IfNoValueTF{#1}{
131     \int_gincr:N \g_niib_note_int
132     \bibnotetext{#2}
133     \bibnotemark[\niib_thenote:]
134   }{
135     \bibnotetext[#1]{#2}
136     \bibnotemark[#1]
137   }
138 }

```

`\bibnotemark` [#1]: label

As with `\bibnote`, if no optional argument is given the global note tracking counter is incremented before carrying out the citation.

```

139 \NewDocumentCommand{\bibnotemark}{o}{
140   \IfNoValueTF{#1}{
141     \int_gincr:N \g_niib_note_int
142     \niib_notemark:n{\niib_thenote:}
143   }{
144     \niib_notemark:n{#1}
145   }
146 }

```


`\bibnotetext` [#1]: label

#2: text

Writing the note text to the database should only occur if files are being written. For the moment, this uses the L^AT_EX 2_ε flag (and so the tradition syntax).

```
147 \NewDocumentCommand{\bibnotetext}{0{\niib_thenote:}>{P}m}{
148   \if@filesw
```

`\g_niib_out_stream` A new output stream may need to be created, if this is the first note to write. The L^AT_EX 2_ε file switch is checked to see if this should occur.

```
149   \cs_if_free:NT \g_niib_out_stream {
150     \iow_new:N \g_niib_out_stream
151     \iow_open:Nn \g_niib_out_stream
152       {\niib_prefix:\c_job_name_tlp.bib}
153     \iow_expanded:Nn \g_niib_out_stream
154       {This-is-an-auxiliary-file-used-by-the-‘xnotes2bib’~
155        package.^~J This-file-may-safely-be-deleted.^~J It~
156        will-be-recreated-as-required.^~J}
157   }
```

The writing macro works in expanded mode, as everything except the note text does need expansion. Notice that `\space` has to be used here after the macro names, as `~` does not work.

```
158   \niib_debug:n{
159     Writing-text:\MessageBreak #2\MessageBreak to-database~
160     for-note-‘#1’
161   }
162   \iow_expanded:Nn \g_niib_out_stream {
163     @\niib_record:{
164       #1,^~J
165       \space\space \niib_field:\space
166       =~{\exp_not:n{#2}},^~J
167       \space\space \niib_keyname:\space
168       =~{#1},^~J
169       \space\space presort~
170       =~{\niib_presort:},^~J
171     }^~J
172   }
173   \fi
174 }
```

`\flushnotestack` In order to delay citations to the end of the bibliography (and thus force others to the start), note citations are held in a clist until being added to the .aux file using `\flushnotestack`. At that point, the stack is cleared so that collection can begin again.

```
175 \NewDocumentCommand{\flushnotestack}{}{
```

```

176 \clist_if_empty:NF\g_niib_beforecite_clist{
177   \if@filesw
178     \iow_expanded:Nn\@auxout{
179       \token_to_string:N\niibbeforecite{
180         \g_niib_beforecite_clist
181       }
182     }
183   \fi
184 }
185 \clist_if_empty:NF\g_niib_aftercite_clist{
186   \nocite{\g_niib_aftercite_clist}
187   \if@filesw
188     \iow_expanded:Nn\@auxout{
189       \token_to_string:N\niibaftercite{\g_niib_aftercite_clist}
190     }
191   \fi
192 }
193 \clist_gclear:N\g_niib_aftercite_clist
194 }

```

\niibsetup

```

195 \NewDocumentCommand{\niibsetup}{m}{\setkeys{niib}{#1}}

```

5.4 Macros for the .aux file

To allow tracking of the need to re-run L^AT_EX, each run writes the current out-of-order citations to the .aux file. This requires two user-space macros, although it is not intended the user ever applies them!

\g_niib_oldbeforecite_clist Two lists are created to track the out-of-order citations written in the previous run to the .aux file.
\g_niib_oldaftercite_clist

```

196 \clist_new:N\g_niib_oldbeforecite_clist
197 \clist_new:N\g_niib_oldaftercite_clist

```

\g_niib_document_tlp A hook into the \document macro is needed by \niibbeforecite; this needs to exist before \document is executed, hence needing a tlp for this.

```

198 \tlp_new:N\g_niib_document_tlp
199 \tlp_put_right:Nn\document{\g_niib_document_tlp}

```

\niibbeforecite #1: citations

\niibaftercite The macros themselves are quite simple. \clist_gset_eq:Nc is not used as the macros could appear more than once in the .aux file.

```

200 \NewDocumentCommand{\niibbeforecite}{m}{
201   \clist_gput_right:Nn\g_niib_oldbeforecite_clist{#1}
202   \tlp_gput_right:Nn\g_niib_document_tlp{\nocite{#1}}
203 }
204 \NewDocumentCommand{\niibaftercite}{m}{
205   \clist_gput_right:Nn\g_niib_oldaftercite_clist{#1}
206 }

```

`\g_niib_totalnotes_int` The total number of notes in the last run is needed

```

207 \int_new:N \g_niib_totalnotes_int

```

`\niibtotalnotes` #1: number

The total number of notes created is stored for the next run.

```

208 \NewDocumentCommand{\niibtotalnotes}{m}{
209   \int_gset:Nn \g_niib_totalnotes_int {#1}
210 }

```

5.5 Internal macros

`\niib_notemark:n` The citation command itself is in an internal macro, as it is called from a few places. Both the before and after modes require some tracking of the citations. For the moment this is using `\clist_put_right:Nx`, but it really should be `\clist_gput_right:Nx` (which does not currently exist).

```

211 \def_new:NNn \niib_notemark:n 1 {
212   \niib_debug:n{Creating~citation~for~note~‘#1’}
213   \bool_if:NT\l_niib_before_bool{
214     \niib_debug:n{Note~‘#1’~before~real~citations}
215     \clist_put_right:Nx\g_niib_beforecite_clist{#1}
216   }
217   \bool_if:NTF\l_niib_after_bool{
218     \niib_debug:n{Note~‘#1’~after~real~citations}
219     \clist_put_right:Nx\g_niib_aftercite_clist{#1}
220     \niib_aftercite:n{#1}
221   }{
222     \niib_cite:n{#1}
223   }
224 }

```

`\g_niib_beforecite_clist` Out-of-order citations in the current run are followed using two clists.

`\g_niib_aftercite_clist`

```

225 \clist_new:N\g_niib_beforecite_clist
226 \clist_new:N\g_niib_aftercite_clist

```

`\niib_key:` The key and presort information for the database needs to be created dynamically from the various keys.

```

227 \def_new:NNn\niib_key: 0 {
228   \bool_if:NTF\l_niib_before_bool{
229     \cs_use:c{niib_key~before:}
230   }{
231     \bool_if:NTF\l_niib_after_bool{
232       \cs_use:c{niib_key~after:}
233     }{
234       \cs_use:c{niib_within~key:}
235     }
236   }
237 }
238 \def_new:NNn\niib_presort: 0 {
239   \bool_if:NTF\l_niib_before_bool{
240     \cs_use:c{niib_presort~before:}
241   }{
242     \bool_if:NTF\l_niib_after_bool{
243       \cs_use:c{niib_presort~after:}
244     }{
245       \cs_use:c{niib_within~presort:}
246     }
247   }
248 }

```

`\niib_convert_endnotes_false:` Conversion of footnotes and endnotes takes place using four dedicated macros, which can then be called by `csname`.

```

\niib_convert_endnotes_true:
\niib_convert_footnotes_false:
\niib_convert_footnotes_true:
249 \def_new:NNn\niib_convert_endnotes_false: 0 {
250   \niib_debug:n{Restoring~normal~endnotes}
251   \let:NN\endnote\niib_saved_endnote:
252   \let:NN\endnotemark\niib_saved_endnotemark:
253   \let:NN\endnotetext\niib_saved_endnotetext:
254 }
255 \def_new:NNn\niib_convert_endnotes_true: 0 {
256   \niib_debug:n{Converting~endnotes~to~bibnotes}
257   \let:NN\endnote\bibnote
258   \let:NN\endnotemark\bibnotemark
259   \let:NN\endnotetext\bibnotetext
260 }
261 \def_new:NNn\niib_convert_footnotes_false: 0 {
262   \niib_debug:n{Restoring~normal~footnotes}
263   \let:NN\footnote\niib_saved_footnote:
264   \let:NN\footnotemark\niib_saved_footnotemark:
265   \let:NN\footnotetext\niib_saved_footnotetext:
266 }
267 \def_new:NNn\niib_convert_footnotes_true: 0 {
268   \niib_debug:n{Converting~footnotes~to~bibnotes}
269   \let:NN\footnote\bibnote

```

```

270 \let:NN\footnotemark\bibnotemark
271 \let:NN\footnotetext\bibnotetext
272 }

```

`\thanks` The `\thanks` macro has to be replaced by one that uses the original `\footnotemark` and `\footnotetext`.

```

273 \def:NNn\thanks 1 {
274   \niib_saved_footnotemark:
275   \protected@xdef\@thanks{\@thanks
276   \protect\niib_saved_footnotetext:[\the\c@footnote]{#1}}
277 }

```

5.6 Finalising

`\niib_keyname:` Various macro definitions depend on the loading of other packages. When `biblatex` is in use, life is rather different to when the standard citation method is operational. The `\niib_bibliography:n` macro is used to control sorting: `biblatex` uses a different macro name for this function than normal \LaTeX does. The `\printbibnotes` user macro always prints the bibliography, using only the notes database if possible. The `\niib_aftercite:n` macro has to keep a track of the appropriate bibnotes and also fiddle with the `\@files` switch to prevent writing at the wrong time.

```

278 \AtBeginDocument{
279   \PackageInfo{xnotes2bib}{
280     Beginning~\token_to_string:N\AtBeginDocument\space tasks
281   }
282   \@ifpackageloaded{biblatex}{
283     \def_new:NNn \niib_keyname: 0 {keysort}
284     \gappto\blx@bibfiles{,\niib_prefix\c_job_name_tlp}
285     \let_new:NN\printbibnotes\printbibliography
286     \def_new:NNn\niib_aftercite:n 1 {
287       \AtNextCite{\@filesfalse}
288       \niib_cite:n{#1}
289     }
290   }{
291     \def_new:NNn \niib_keyname: 0 {key}
292     \let_new:NN\niib_bibliography:n\bibliography
293     \def:NNn\bibliography 1 {
294       \int_compare:nNnTF{\g_niib_note_int}={\c_zero}{
295         \niib_bibliography:n{#1}
296       }{
297         \niib_bibliography:n{#1,\niib_prefix:\c_job_name_tlp}
298       }
299     }
300     \NewDocumentCommand{\printbibnotes}{}{
301       \niib_bibliography:n{\niib_prefix:\c_job_name_tlp}
302     }

```

`\ifniib@filesw` There is a need to back-up the `\@filesw` switch. As this is a $\text{\LaTeX} 2_{\epsilon}$ switch, the name and creation method used here are not $\text{\LaTeX} 3$ -based.

```
303 \newif\ifniib@filesw
```

`\@restore@auxhandle` If `cite` is loaded, there is further trouble.

```
\niib_auxhandle:
304 \ifpackageloaded{cite}{
305   \def_new:NN\niib_aftercite:n 1 {
306     \let:NN\ifniib@filesw\if@filesw
307     \@fileswfalse
308     \def:NN{\niib_auxhandle:} 0 {
309       \let:NN\if@filesw\ifniib@filesw
310       \let:NN\niib_auxhandle:\scan_stop:
311     }
312     \niib_cite:n{#1}
313   }
314   \cs_if_exist:NTF\@restore@auxhandle{
315     \tlp_put_right:Nn\@restore@auxhandle{\niib_auxhandle:}
316   }{
317     \def_new:NN\@restore@auxhandle 0 {\niib_auxhandle:}
318   }
319   \let_new:NN\niib_auxhandle:\scan_stop:
```

Life is slightly easier without `cite`.

```
320 }{
321   \def_new:NN\niib_aftercite:n 1 {
322     \let\ifniib@filesw\if@filesw
323     \@fileswfalse
324     \niib_cite:n{#1}
325     \let\if@filesw\ifniib@filesw
326   }
327 }
328 }
```

`\niib_saved_footnote:` The second thing to worry about is footnotes and endnotes. The definitions are saved here so that any other packages can have made their changes. No argument specifiers are given as these are saving user space macros.

```
\niib_saved_endnote:
\niib_saved_endnotemark:
\niib_saved_endnotetext:
\niib_saved_endnote:
329 \let_new:NN\niib_saved_footnote:\footnote
\niib_saved_endnotemark:
330 \let_new:NN\niib_saved_footnotemark:\footnotemark
\niib_saved_endnotetext:
331 \let_new:NN\niib_saved_footnotetext:\footnotetext
332 \let_new:NN\niib_saved_endnote:\endnote
333 \let_new:NN\niib_saved_endnotemark:\endnotemark
334 \let_new:NN\niib_saved_endnotetext:\endnotetext
```

Any conversion can now be applied, before the `convert footnotes` and `convert endnotes` options are redefined to work immediately.

```

335 \niib_convert_endnotes:
336 \niib_convert_footnotes:
337 \niib_KV_bool:nT{convert~endnotes}{
338   \cs_use:c{niib_convert_endnotes_#1:}
339 }
340 \niib_KV_bool:nT{convert~footnotes}{
341   \cs_use:c{niib_convert_footnotes_#1:}
342 }
343 \PackageInfo{xnotes2bib}{
344   Finished~\token_to_string:N\AtBeginDocument\space tasks
345 }
346 }

```

\niib_reruncheck:n If either of the out-of-order options are in use, then a comparison of the citations used with those used in the last run is needed. There are a couple of easy checks to make: if only one list is empty, there must be changes. If both lists are occupied, a check of every item must be made.

```

347 \def_new:NNn\niib_rerun_check:n 1 {
348   \clist_if_empty:cTF {g_niib_old#1cite_clist}{
349     \clist_if_empty:cF {g_niib_#1cite_clist}{
350       \niib_rerun_log:n {#1}
351     }
352   }{
353     \clist_if_empty:cTF {g_niib_#1cite_clist}{
354       \niib_rerun_log:n {#1}

```

If both lists are occupied by data, the total contents of each one have to be compared with the other. This needs to be done both ways round.

```

355   }{
356     \bool_set_false:N\l_tmpa_bool
357     \def:NNn\niib_rerun_test:n 1 {
358       \clist_if_in:cnTF{g_niib_old#1cite_clist}{##1}{
359         \bool_set_true:N\l_tmpa_bool
360         \clist_map_break:w
361       }{
362       }
363     }
364     \clist_map_function:cn{g_niib_#1cite_clist}
365     \niib_rerun_test:n
366     \def:NNn\niib_rerun_test:n 1 {
367       \clist_if_in:cnTF{g_niib_#1cite_clist}{##1}{
368         \bool_set_true:N\l_tmpa_bool
369         \clist_map_break:w
370       }{
371       }
372     }
373     \clist_map_function:cn{g_niib_old#1cite_clist}
374     \niib_rerun_test:n

```

```

375     \bool_if:NF\l_tmpa_bool{
376       \niib_rerun_log:n {#1}
377     }
378   }
379 }
380 }

```

\niib_rerun_test:n This is redefined in the above, but is made available here.

```

381 \def:NNn\niib_rerun_test:n 0 {}

```

\niib_rerun_log:n The information message needs to be available in a few different places.

```

382 \def_new:NNn\niib_rerun_log:n 1 {
383   \PackageInfo{xnotes2bib}{
384     To~get~‘#1’~notes~in~the~correct~order,~please\MessageBreak
385     rerun~LaTeX,~(re)run~BibTeX~on~the~file~\c_job_name_tlp.aux
386     \MessageBreak and~rerun~LaTeX~again~afterwards
387   }
388   \io_put_log:x {
389     REQ:3:latex:REQ\iow_newline:
390     REQ:2:bibtex:REQ\iow_newline:
391     REQ:1:latex:REQ\iow_newline:
392   }
393 }

```

At the end of the document, the package should tidy up.

```

394 \AtEndDocument{
395   \niib_rerun_check:n{before}
396   \niib_rerun_check:n{after}
397   \flushnotestack
398   \iow_expanded:Nn\@auxout{
399     \token_to_string:N\niibttotalnotes{
400       \int_to_arabic:n \g_niib_note_int
401     }
402   }
403   \num_compare:nNnF{\g_niib_totalnotes_int}={\g_niib_note_int}{
404     \niib_rerun_log:n
405   }
406   \cs_if_free:NF \g_niib_out_stream{
407     \iow_close:N \g_niib_out_stream
408   }
409 }

```


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	<code>\clist_gput_right:Nn .</code>	<code>\document</code>	199
<code>\@auxout</code>	178, 188, 398	<code>.</code>	201, 205
<code>\@filesfalse</code>	287, 307, 323	<code>\clist_if_empty:cF . .</code>	349
<code>\@ifpackageloaded</code>	282, 304	<code>\clist_if_empty:cTF .</code>	348, 353
<code>\@ifundefined</code>	3	<code>.</code>	176, 185
<code>\@restore@auxhandle .</code>	304	<code>\clist_if_in:cnTF . . .</code>	38, 58, 358, 367
<code>\@thanks</code>	275	<code>.</code>	360, 369
A		<code>\clist_map_break:w</code>	360, 369
<code>\AtBeginDocument</code>		<code>\clist_map_function:cN</code>	364, 373
<code>.</code>	278, 280, 344	<code>\clist_new:c</code>	52
<code>\AtEndDocument</code>	394	<code>\clist_new:N</code>	34, 196, 197, 225, 226
<code>\AtNextCite</code>	287	<code>.</code>	56
B		<code>\clist_put_left:cn . . .</code>	35
<code>\begingroup</code>	2	<code>\clist_put_left:Nn . . .</code>	215, 219
<code>\bibliography</code>	292, 293	<code>\clist_put_right:Nx .</code>	215, 219
<code>\bibnote</code>	2, 129, 257, 269	<code>.</code>	215, 219
<code>\bibnotemark</code>	2,	convert endnotes (op-	2
<code>.</code>	133, 136, 139, 258, 270	tion)	2
<code>\bibnotetext</code>	2,	convert footnotes (op-	2
<code>.</code>	132, 135, 147, 259, 271	tion)	2
<code>\blx@bibfiles</code>	284	<code>\cs_if_exist:NTF</code>	314
<code>\bool_if:NF</code>	375	<code>\cs_if_free:cTF</code>	51
<code>\bool_if:NT</code>	213	<code>\cs_if_free:NF</code>	406
<code>\bool_if:NTF</code>		<code>\cs_if_free:NT</code>	149
<code>.</code>	217, 228, 231, 239, 242	<code>\cs_if_really_exist:cT</code>	111
<code>\bool_new:N</code>	106, 107	<code>\cs_use:c</code>	229, 232, 234, 240, 243, 245, 338, 341
<code>\bool_set_false:N . . .</code>		D	
<code>.</code>	109, 110, 356	debug (option)	3
<code>\bool_set_true:c</code>	112	<code>\DeclareOption</code>	16
<code>\bool_set_true:N . . .</code>	359, 368	<code>\def:cNn</code>	72, 77
C		<code>\def:NNn</code>	273, 293, 308, 357, 366, 381
<code>\c@footnote</code>	276	<code>\def_new:NNn</code>	24, 36, 50, 70, 75, 102, 211, 227, 238, 249, 255, 261, 267, 283, 286, 291, 305, 317, 321, 347, 382
<code>\c_job_name_tlp</code>	152, 284, 297, 301, 385	<code>.</code>	37, 57, 71, 76
<code>\c_nine</code>	26, 27	<code>\document</code>	199
<code>\c_zero</code>	294	E	
<code>\cite</code>	116	<code>\endgroup</code>	7, 9
<code>cite (option)</code>	3	<code>\endinput</code>	7
<code>\clist_gclear:c</code>	54	<code>\endnote</code>	251, 257, 332
<code>\clist_gclear:N</code>	193	<code>\endnotemark</code>	252, 258, 333
		<code>\endnotetext</code>	253, 259, 334
		<code>\exp_not:n</code>	166
		F	
		<code>\fi</code>	173, 183, 191
		<code>field (option)</code>	3
		<code>\flushnotestack . . .</code>	2, 175, 397
		<code>\footnote</code>	263, 269, 329
		<code>\footnotemark</code>	264, 270, 330
		<code>\footnotetext</code>	265, 271, 331
		G	
		<code>\g_niib_aftercite_clist</code>	185, 186, 189, 193, 219, 225
		<code>.</code>	176, 180, 215, 225
		<code>\g_niib_beforecite_clist</code>	176, 180, 215, 225
		<code>.</code>	198, 202
		<code>\g_niib_document_tlp .</code>	33
		<code>.</code>	23, 27, 31, 131, 141, 294, 400, 403
		<code>\g_niib_KV_bool_clist .</code>	196, 205
		<code>.</code>	196, 201
		<code>\g_niib_note_int</code>	149, 406, 407
		<code>.</code>	26, 207, 209, 403
		<code>\g_niib_oldaftercite_clist</code>	284
		<code>.</code>	
		<code>\g_niib_oldbeforecite_clist</code>	
		<code>.</code>	
		<code>\g_niib_out_stream . .</code>	
		<code>.</code>	
		<code>\g_niib_totalnotes_int</code>	
		<code>.</code>	
		<code>\gappto</code>	
		I	
		<code>\if@files</code>	148, 177, 187, 306, 309, 322, 325

\ifniib@files	\niib_bibliography:n	.. 278	\niib_reruncheck:n	.. 347
303, 306, 309, 322, 325		\niib_cite:n	\niib_saved_endnote:	..
\IfNoValueTF 130, 140	80, 222, 288, 312, 324		251, 329
\int_compare:nNnTF	.. 294	\niib_convert_endnotes:	\niib_saved_endnotemark:
\int_gincr:N 131, 141 91, 335		252, 329
\int_gset:Nn 209	\niib_convert_endnotes_false\niib_saved_endnotetext:	253, 329
\int_new:N 23, 207 249		263, 329
\int_to_arabic:n	.. 31, 400	\niib_convert_endnotes_true\niib_saved_footnote:	264, 274, 329
\io_put_log:x 388 249		265, 276, 329
\iow_close:N 407	\niib_convert_footnotes:	\niib_saved_footnotemark:	
\iow_expanded:Nn 91, 336	
153, 162, 178, 188, 398		\niib_convert_footnotes_false\niib_saved_footnotetext:	
\iow_new:N 150 249		
\iow_newline:	389, 390, 391	\niib_convert_footnotes_true\niib_thenote:	
\iow_open:Nn 151 249	 24, 133, 142, 147	
K					
key after (option) 3	\niib_debug:n	\niibaftercite	... 189, 200
key before (option) 3	99, 158, 212, 214,		\niibbeforecite	.. 179, 200
key mixed (option) 3	218, 250, 256, 262, 268		\niibsetup 2, 19, 195
L					
\l_niib_after_bool	..	\niib_debug_false:n	.. 99	\niibtotallnotes	.. 208, 399
.... 106, 217, 231, 242		\niib_debug_true:n	... 99	\nocite 186, 202
\l_niib_before_bool	..	\niib_field: 81, 165	notename (option) 3
.... 106, 213, 228, 239		\niib_key_after: 81	\num_compare:nNnF	.. 27, 403
\l_tmpa_bool	\niib_key_before: 81	\num_compare:nNnT 26
.... 356, 359, 368, 375		\niib_key_mixed: 81	O	
\let 322, 325	\niib_key: 227	options:	
\let:Nc 94, 97, 100	\niib_keyname:	... 167, 278	cite 3
\let:NN 251, 252,	\niib_KV_bool:nT	convert endnotes	... 2
253, 257, 258, 259,		33, 93, 96, 99, 337, 340		convert footnotes	.. 2
263, 264, 265, 269,		\niib_KV_choice:nnT	33, 108	debug 3
270, 271, 306, 309, 310		\niib_KV_cs_n:n	... 33, 80	field 3
\let_new:NN	91, 92, 105,	\niib_KV_string:n	...	key after 3
285, 292, 319, 329,	 33, 81, 82, 83,		key before 3
330, 331, 332, 333, 334		84, 85, 86, 87, 88, 89, 90		key mixed 3
M					
\MessageBreak	\niib_notemark:n	notename 3
18, 44, 64, 159, 384, 386	 142, 144, 211		placement 3
N					
\NeedsTeXFormat 1	\niib_notename:	... 25, 81	prefix 3
\NewDocumentCommand	..	\niib_prefix 284	presort after 3
129, 139, 147, 175,		\niib_prefix:	presort before 3
195, 200, 204, 208, 300	 81, 152, 297, 301		presort mixed 3
\newif 303	\niib_presort_after:	.. 81	record 3
\niib_aftercite:n	220, 278	\niib_presort_before:	.. 81	P	
\niib_auxhandle: 304	\niib_presort_mixed:	.. 81	\PackageError 4, 41, 61
		\niib_presort:	... 170, 227	\PackageInfo
		\niib_record: 81, 163 103, 279, 343, 383	
		\niib_rerun_check:n	\PackageWarning 17
	 347, 395, 396		placement (option) 3
		\niib_rerun_log:n	...	prefix (option) 3
		350, 354, 376, 382, 404		presort after (option)	.. 3
		\niib_rerun_test:n	..		
		357, 365, 366, 374, 381			

presort before (option) . 3	\RequirePackage . 11, 15, 33	\the 276
presort mixed (option) .. 3		\tlp_gput_right:Nn .. 202
\printbibliography .. 285	S	\tlp_new:N 198
\printbibnotes 278	\scan_stop: 22, 91, 92, 310, 319	\tlp_put_right:Nn 199, 315
\ProcessOptions 22	\setkeys 115, 195	\token_to_string:N 19,
\protect 276	\space 19,	179, 189, 280, 344, 399
\protected@xdef 275	\ProvidesExplPackage .. 12	
	165, 167, 169, 280, 344	
R	T	U
record (option) 3	\thanks 273	\use_none:n 105