

# **glossaries-extra.sty v1.47: documented code**

Nicola L.C. Talbot

Dickimaw Books

<http://www.dickimaw-books.com/>

2021-11-04

## Abstract

This is the documented code for the glossaries-extra package. See [glossaries-extra-manual.pdf](#) for the user manual.

This package is experimental and not stable. It's provided for testing purposes only. Future versions may not be compatible with this version. Once it has stabilised I'll add it to CTAN, at which point compatibility with the first stable version will be maintained.

# Contents

<b>1</b>	<b>Main Package Code (glossaries-extra.sty)</b>	<b>5</b>
1.1	Package Initialisation and Options	5
1.2	Extra Utilities	32
1.3	Modifications to Commands Provided by glossaries	50
1.3.1	Existence Checks	55
1.3.2	Document Definitions	65
1.3.3	Existing Glossary Style Modifications	71
1.3.4	Entry Formatting, Hyperlinks and Indexing	75
1.3.5	Entry Counting	115
1.3.6	Acronym Modifications	130
1.3.7	Indexing and Displaying Glossaries	135
1.4	Link Counting	178
1.5	Integration with glossaries-accsupp	180
1.6	Categories	197
1.7	Abbreviations	225
1.7.1	Abbreviation Styles Setup	246
1.7.2	Predefined Styles (Default Font)	249
1.7.3	Predefined Styles (Small Capitals)	268
1.7.4	Predefined Styles (Fake Small Capitals)	285
1.7.5	Predefined Styles (Emphasized)	302
1.7.6	Predefined Styles (User Parentheses Hook)	326
1.7.7	Predefined Styles (Hyphen)	337
1.7.8	Predefined Styles (No Short on First Use)	352
1.8	Using Entries in Headings	355
1.9	Multi-Lingual Support	378
1.10	glossaries-extra-bib2gls.sty	379
<b>2</b>	<b>Style Adjustments (glossaries-extra-stylemods.sty)</b>	<b>425</b>
2.1	Package Initialisation	425
2.2	List-Like Styles	426
2.3	Longtable Styles	430
2.4	Long Ragged Styles	432
2.5	Supertabular Styles	434
2.6	Super Ragged Styles	436
2.7	Inline Style	438
2.8	Tree Styles	438
2.9	Multicolumn Styles	458

<b>3</b>	<b>bookindex style (glossary-bookindex.sty)</b>	<b>465</b>
3.1	Package Initialisation and Options . . . . .	465
<b>4</b>	<b>longextra styles (glossary-longextra.sty)</b>	<b>472</b>
4.1	Package Initialisation and Options . . . . .	472
<b>5</b>	<b>topic styles (glossary-topic.sty)</b>	<b>495</b>
5.1	Package Initialisation and Options . . . . .	495
	<b>Glossary</b>	<b>501</b>
	<b>Change History</b>	<b>502</b>
	<b>Index</b>	<b>529</b>

# 1 Main Package Code (glossaries-extra.sty)

## 1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2021/11/04 v1.47 (NLCT)]

Requires xkeyval to define package options.
3 \RequirePackage{xkeyval}

Requires etoolbox package.
4 \RequirePackage{etoolbox}

Has glossaries already been loaded?
5 \@ifpackageloaded{glossaries}
6 {%

Already loaded so pass any options to \setupglossaries. This means that the options that
can only be set when glossaries is loaded can't be used.
7   \newcommand{\glstr@dooption}[1]{\setupglossaries{#1}}%
8   \let\@glstr@declareoption\@gls@declareoption
9 }
10 {%

Not already loaded, so pass options to glossaries.
11   \newcommand{\glstr@dooption}[1]{%
12     \PassOptionsToPackage{#1}{glossaries}%
13   }%

Set the defaults.
14   \PassOptionsToPackage{toc}{glossaries}
15   \PassOptionsToPackage{nopostdot}{glossaries}
16   \PassOptionsToPackage{noredefwarn}{glossaries}
17   \@ifpackageloaded{polyglossia}%
18   {%
19   {%
20     \@ifpackageloaded{babel}%
21     {\PassOptionsToPackage{translate=babel}{glossaries}}%
22     {}%
23   }%
24   \newcommand*{\@glstr@declareoption}[2]{%
25     \DeclareOptionX{#1}{#2}%
26     \DeclareOption{#1}{#2}%
27   }
28 }
```

Declare package options.

`\glstrundefaction` Determines what to do if an entry hasn't been defined. The two arguments are the error or warning message and the help message if an error should be produced.

```

29 \newcommand*{\glstrundefaction}[2]{%
30   \glstrundeftag\PackageError{glossaries-extra}{#1}{#2}%
31 }

```

`\warnonexistsordo` If user wants undefaction=warn, then glossaries v4.19 is required.

```

32 \newcommand*{\glstr@warnonexistsordo}[1]{%

```

`\glstrundeftag` Text to display when an entry doesn't exist.

```

33 \newcommand*{\glstrundeftag}{??}
34 \newcommand*{\@glstrundeftag}{}

```

This text is switched on at the start of the document to prevent unwanted text inserted into the preamble if any tests are made before the start of the document.

`\warn@undefaction` This is how `\glstrundefaction` should behave if `undefaction=warn` is set.

```

35 \newcommand*{\@glstr@warn@undefaction}[2]{%
36   \glstrundeftag\GlossariesExtraWarning{#1}%
37 }

```

`\err@undefaction` This is how `\glstrundefaction` should behave if `undefaction=error` is set.

```

38 \newcommand*{\@glstr@err@undefaction}[2]{%
39   \glstrundeftag\PackageError{glossaries-extra}{#1}{#2}%
40 }

```

`\warn@onexistsordo` This is how `\glstr@warnonexistsordo` should behave if `undefaction=warn` is set.

```

41 \newcommand*{\@glstr@warn@onexistsordo}[1]{%
42   \GlossariesExtraWarning{\string#1\space hasn't been defined, so
43     some errors won't be converted to warnings.
44     (This most likely means your version of
45     glossaries.sty is below version 4.19.)}%
46 }

```

`\f@for@gl@sentries`

```

47 \newcommand*{\@glstr@redef@for@gl@sentries}{}

```

`\f@for@gl@sentries`

```

48 \newcommand*{\@glstr@do@redef@for@gl@sentries}{%
49   \renewcommand*{\for@gl@sentries}[3][\gl@defaulttype]{%
50     \protected@edef\@glo@list{\csname glolist@##1\endcsname}%
51     \ifdefstring{\@glo@list}{,}%
52     {%
53       \GlossariesExtraWarning{No entries defined in glossary '#1'}%
54     }%
55     {%
56       \@for##2:=\@glo@list\do

```

```

57      {%
58      \ifdefempty{##2}{-}{##3}%
59      }%
60      }%
61      }%
62 }%

```

undefaction

```

63 \define@choicekey{glossaries-extra.sty}{undefaction}%
64 [\glstr@undefaction@val\glstr@undefaction@nr]%
65 {warn,error}%
66 {%
67   \ifcase\glstr@undefaction@nr\relax
68     \let\glstr@undefaction\glstr@warn@undefaction
69     \let\glstr@warnonexistsordo\glstr@warn@onexistsordo
70     \let\glstr@redef@for@glstr@sentries\glstr@do@redef@for@glstr@sentries
71   \or
72     \let\glstr@undefaction\glstr@err@undefaction
73     \let\glstr@warnonexistsordo\glstr@gobble
74     \let\glstr@redef@for@glstr@sentries\relax
75   \fi
76 }

```

To assist bib2gls, v1.08 introduces the record option, which will write information to the aux file whenever an entry needs to be indexed.

\glstr@record Does nothing by default.

```
77 \newcommand*{\glstr@record}[3]{}

```

\glstr@recordsee Does nothing by default.

```
78 \newcommand*{\glstr@recordsee}[2]{}

```

\glstr@defaultnumberformat

```
79 \newcommand*{\glstr@defaultnumberformat}{\glstr@numberformat}%

```

\GlsXtrSetDefaultNumberFormat

```

80 \newcommand*{\GlsXtrSetDefaultNumberFormat}[1]{%
81   \renewcommand*{\glstr@defaultnumberformat}{#1}%
82 }%

```

The record option is somewhat problematic. On the first  $\LaTeX$  run the entries aren't defined. This isn't as straight-forward as commands like `\cite` since attributes associated with the entry's category may switch off the indexing or the entry's glossary type might require a particular counter. This kind of information can't be determined until the entry has been defined. So there are two different commands here. One that's used if the entry hasn't been defined, which tries to use sensible defaults, and one which is used when the entry has been defined.

cord@wrglossary The record=only option sets \@do@wrglossary to this command, which means it's done within \glsadd and \@gls@link, and so is only done if the entry exists.

```

83 \newcommand*{\@glsxtr@do@record@wrglossary}[1]{%
84   \begingroup
85     \ifKV@glslink@noindex
86     \else

87     \protected@edef\@gls@label{\glsdetoklabel{#1}}%
88     \let\glslabel\@gls@label
89     \glswriteentry{#1}%
90     {%
91       \ifdefempty{\@glsxtr@thevalue}%
92       {%
93         \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
94         \else
95           \let\theHglsentrycounter\@glsxtr@theHvalue
96           \fi
97           \glsxtr@saveentrycounter
98           \let\@do@wrglossary\@glsxtr@dorecord
99       }%
100     {%
101       \let\theHglsentrycounter\@glsxtr@thevalue
102       \let\theHglentrycounter\@glsxtr@theHvalue
103       \let\@do@wrglossary\@glsxtr@dorecordnodefer
104     }%
105     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
106     \glsxtr@do@wrglossary{#1}%
107     \else
108     \@glsxtrwrglossmark

```

Increment associated counter.

```

109     \glsxtr@inc@wrglossaryctr{#1}%
110     \@do@wrglossary
111   \fi
112 }%
113 \fi
114 \endgroup
115 }

```

index@wrglossary The record=alsoindex option needs to both record and index.

```

116 \newcommand*{\@glsxtr@do@alsoindex@wrglossary}[1]{%
117   \glsxtr@do@wrglossary{#1}%
118   \@glsxtr@do@record@wrglossary{#1}%
119 }

```

@glsxtr@record The record=only option sets \@glsxtr@record to this. This performs the recording if the entry *doesn't exist* and is done at the start of \@gls@field@link and commands like \@gls@ (before the existence test). This means that it disregards the wrgloss key.



The first argument is the option list (as passed in the first optional argument to commands like `\gls`). This allows the `noindex` setting to be picked up. The second argument is the entry's label. The third argument is the key family (`glslink` in most cases, `glossadd` for `\glsadd`).

```
120 \newcommand*{\@@glsxtr@record}[3]{%
```

Save the label in case it's needed. This needs to be outside the existence check to allow the post-link hook to reference it.

```
121 \protected@edef\@gls@label{\glsdetoklabel{#2}}%
122 \let\glslabel\@gls@label
123 \ifglsentryexists{#2}{}%
124 {%
125   \@@glsxtrwrglossmark
126   \beginngroup
127     \let\@glsnumberformat\@glsxtr@defaultnumberformat
128     \def\@glsxtr@thevalue{}%
129     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
130     \let\@glsxtr@org@theHvalue\@glsxtr@theHvalue
```

Entry hasn't been defined, so we'll have to assume it's `\glscounter` by default.

```
131 \let\@gls@counter\glscounter
```

Unless the equations option is on and this is inside a numbered maths environment.

```
132 \ifglsxtr@equations
133   \@glsxtr@use@equation@counter
134 \fi
```

Check for default options (which may switch off indexing).

```
135 \@gls@setdefault@glslink@opts
```

Implement any pre-key settings.

```
136 \csuse{\@glsxtr@#3@prekeys}%
```

Assign keys.

```
137 \setkeys{#3}{#1}%
```

Implement any post-key settings. Is the auto-add on?

```
138 \glsxtr@do@autoadd{#3}%
```

Check post-key hook.

```
139 \csuse{\@glsxtr@#3@postkeys}%
```

Increment associated counter.

```
140 \glsxtr@inc@wrglossaryctr{#2}%
```

Check if `noindex` option has been used.

```
141 \ifKV@glslink@noindex
142 \else
143   \glswriteentry{#2}%
144   {%
```

Check if `thevalue` has been set.

```
145   \ifdefempty{\@glsxtr@thevalue}%
146   {%
```

Key thevalue hasn't been set, but check if theHvalue has been set. (Not particularly likely, but allow for it.)

```

147         \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
148         \else
149         \let\theHglentrycounter\@glxtr@theHvalue
150         \fi

```

Save the entry counter.

```

151         \glxtr@saveentrycounter

```

Temporarily redefine \@@do@@wrglossary for use with \glxtr@@do@@wrglossary.

```

152         \let\@@do@@wrglossary\@glxtr@dorecord
153     }%
154     {%

```

thevalue has been set, so there's no need to defer writing the location value. (If it's dependent on the page counter, the counter key should be set instead.)

```

155         \let\theHglentrycounter\@glxtr@thevalue
156         \let\theHglentrycounter\@glxtr@theHvalue
157         \let\@@do@@wrglossary\@glxtr@dorecordnodefer
158     }%
159     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
160     \glxtr@@do@@wrglossary{#2}%
161     \else

```

No need to escape special characters.

```

162         \@@do@@wrglossary
163     \fi
164 }%
165 \fi
166 \endgroup
167 }%
168 }

```

glslink@prekeys

```

169 \newcommand{\@glxtr@glslink@prekeys}{\glslinkpresetkeys}

```

lslink@postkeys

```

170 \newcommand{\@glxtr@glslink@postkeys}{\glslinkpostsetkeys}

```

lossadd@prekeys

```

171 \newcommand{\@glxtr@glossadd@prekeys}{\glsaddpresetkeys}

```

ossadd@postkeys

```

172 \newcommand{\@glxtr@glossadd@postkeys}{\glsaddpostsetkeys}

```

glxtr@dorecord If record=alsoindex or record=hybrid is used, then \@glslocref may have been escaped, but this isn't appropriate here.

```

173 \newcommand*\@glxtr@dorecord{%

```

```

174 \global\let\@glsrecordlocref\theglsentrycounter
175 \let\@glsxtr@orgprefix\@glo@counterprefix
176 \ifx\theglsentrycounter\theHglentrycounter
177   \def\@glo@counterprefix{}%
178 \else
  Protect against non-expandable commands occurring in the location.
179   \protected@edef\@glsxtr@theentrycounter{\theglsentrycounter}%
180   \protected@edef\@glsxtr@theHentrycounter{\theHglentrycounter}%
181   \@onelevel@sanitize\@glsxtr@theentrycounter
182   \@onelevel@sanitize\@glsxtr@theHentrycounter
183   \protected@edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
184     {\@glsxtr@theentrycounter}{\@glsxtr@theHentrycounter}}%
185   }%
186   \@do@gls@getcounterprefix
187 \fi

```

Don't protect the \@glsrecordlocref from premature expansion. If the counter isn't

page then it needs expanding. If the location includes \thepage then \protected@write will automatically deal with it.

```

188 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
189   \@glsxtr@do@nameref@record
190   {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
191   {\@glsrecordlocref}%
192 \else
193   \protected@write\@auxout{}{\string\@glsxtr@record
194     {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
195     {\@glsrecordlocref}}%
196 \fi
197 \@glsxtr@counterrecordhook
198 \let\@glo@counterprefix\@glsxtr@orgprefix
199 }

```

dorecordnodefer As above, but don't defer expansion of location. This uses \theglsentrycounter directly for the location rather than \@glslocref since there's no need to guard against premature expansion of the page counter.

```

200 \newcommand*\@glsxtr@dorecordnodefer{%
201   \ifx\theglsentrycounter\theHglentrycounter
202     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
203       \@glsxtr@do@nameref@record
204       {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
205       {\theglsentrycounter}%
206     \else
207       \protected@write\@auxout{}{\string\@glsxtr@record
208         {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
209         {\theglsentrycounter}}%
210     \fi
211   \else
212     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix

```

```

213     {\theglentrycounter}{\theHglentrycounter}%
214 }%
215 \@do@gl@getcounterprefix
216 \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
217   \@glxtr@do@nameref@record
218     {\@gl@label}{\@glo@counterprefix}{\@gl@counter}%
219     {\@gl@numberformat}{\theglentrycounter}%
220 \else
221   \protected@write\@auxout{}\string\glxtr@record
222     {\@gl@label}{\@glo@counterprefix}{\@gl@counter}{\@gl@numberformat}%
223     {\theglentrycounter}}%
224 \fi
225 \fi
226 \@glxtr@counterrecordhook
227 }

```

`\glxtr@ifnum@mmode` Check if in a numbered maths environment. The `amsmath` package is automatically loaded by `datatool-base`, which is required by `glossaries`, so `\ifst@rred` and `\if@display` should both be defined.

```

228 \newcommand{\@glxtr@ifnum@mmode}[2]{%
229   \ifmmode
230     \ifst@rred
231       #2%
232     \else

```

Non-`amsmath` environments and regular inline math mode isn't flagged as starred by `amsmath`, but we can't use `\mathchoice` in this case as it's not the current style that's relevant. Instead we can use `amsmath's \if@display`. This may not work for environments that aren't provided by `amsmath`.

```

233     \if@display #1\else #2\fi
234   \fi
235 \else
236   #2%
237 \fi
238 }

```

`\glxtr@nameref@record` With `record=nameref`, the current label information is included in the record, but this may not have been defined, so `\csuse` will prevent an undefined control sequence error and just leave the last two arguments blank if there's no information. In the event that a record is in `amsmath's align` environment `\@currentHref` will be out. There may be other instances where `\@currentHref` is out, so this also saves `\theHglentrycounter`, which is useful if it can't be obtained by prefixing `\theglentrycounter`.

```

239 \newcommand*{\@glxtr@do@nameref@record}[5]{%
240   \gl@ifnotmeasuring
241   {%
242     \protected@write\@auxout{}\string\glxtr@record@nameref
243       {#1}{#2}{#3}{#4}{#5}%
244     {\csuse{@currentlabelname}}{\csuse{@currentHref}}}%

```

```

245     {\theHglentrycounter}}%
246   }%
247 }

r@recordcounter
248 \newcommand*{\@glxtr@recordcounter}{%
249   \@glxtr@noop@recordcounter
250 }

p@recordcounter
251 \newcommand*{\@glxtr@noop@recordcounter}[1]{%
252   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
253     requires record=only or record=hybrid package option}{}%
254 }

p@recordcounter
255 \newcommand*{\@glxtr@op@recordcounter}[1]{%
256   \protected@eappto\@glxtr@counterrecordhook{\noexpand\@glxtr@docounterrecord{#1}}}%
257 }

lsxtr@recordsee Deal with \glssee in record mode. (This doesn't increment the associated counter.)
258 \newcommand*{\@glxtr@recordsee}[2]{%
259   \@glxtrwrglossmark
260   \def\@gls@xref{#2}%
261   \@onelevel@sanitize\@gls@xref
262   \protected@write\@auxout{}\{\string\glxtr@recordsee{#1}-{\@gls@xref}}%
263 }

srtglossaryunit
264 \newcommand{\printunsrtglossaryunit}{%
265   \print@noop@unsrtglossaryunit
266 }

tr@setup@record Initialise.
267 \newcommand*{\glxtr@setup@record}{\let\@do@wrglossary\glxtr@do@wrglossary}

saveentrycounter Only store the entry counter information if the indexing is on.
268 \newcommand*{\glxtr@indexonly@saveentrycounter}{%
269   \ifKV@glslink@noindex
270   \else
271     \glxtr@saveentrycounter
272   \fi
273 }

addloclistfield
274 \newcommand*{\glxtr@addloclistfield}{%
275   \key@ifundefined{glossentry}{loclist}%
276   {%

```

```

277 \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
278 \appto\@gls@keymap{,{loclist}{loclist}}%
279 \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
280 \appto\@newglossaryentryposthook{%
281   \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
282 }%
283 \glssetnoexpandfield{loclist}%
284 }%
285 {}%

```

The loclist field is just a comma-separated list. The location field is the formatted list.

```

286 \key@ifundefined{glossentry}{location}%
287 {%
288   \define@key{glossentry}{location}{\def\@glo@location{##1}}%
289   \appto\@gls@keymap{,{location}{location}}%
290   \appto\@newglossaryentryprehook{\def\@glo@location{}}%
291   \appto\@newglossaryentryposthook{%
292     \gls@assign@field{\@glo@label}{location}{\@glo@location}%
293   }%
294   \glssetnoexpandfield{location}%
295 }%
296 {}%

```

Add a key to store the group heading.

```

297 \key@ifundefined{glossentry}{group}%
298 {%
299   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
300   \appto\@gls@keymap{,{group}{group}}%
301   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
302   \appto\@newglossaryentryposthook{%
303     \gls@assign@field{\@glo@label}{group}{\@glo@group}%
304   }%
305   \glssetnoexpandfield{group}%
306 }%
307 {}%
308 }

```

@record@setting Keep track of the record package option.

```

309 \newcommand*{\@glsxtr@record@setting}{off}

```

etting@alsoindex As from v1.46, the record=alsoindex is renamed to record=hybrid with record=alsoindex as a deprecated synonym to avoid confusion. The internal commands that include alsoindex in the name will remain unchanged to avoid breaking things, but this command will need to be redefined by record=hybrid.

```

310 \newcommand*{\@glsxtr@record@setting@alsoindex}{alsoindex}

```

rd@setting@only

```

311 \newcommand*{\@glsxtr@record@setting@only}{only}

```

setting@nameref

```
312 \newcommand*{\@glxtr@record@setting@nameref}{nameref}
```

@if@record@only

```
313 \newcommand*{\@glxtr@if@record@only}[2]{%
314   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
315     #1%
316   \else
317     \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
318       #1%
319     \else
320       #2%
321     \fi
322   \fi
323 }
```

ord@setting@off

```
324 \newcommand*{\@glxtr@record@setting@off}{off}
```

id@noprintgloss    Used by hybrid method if \printglossary isn't used.

```
325 \newcommand\@glxtr@warn@hybrid@noprintgloss{%
326   \ifdefstring{\@glo@types}{,}%
327   {%
328     \GlossariesExtraWarningNoLine{No glossaries have been defined}%
329   }%
330   {%
331     \GlossariesExtraWarningNoLine{No \string\printglossary\space
332     or \string\printglossaries\space
333     found. ^^JYou have requested the hybrid setting
334     record=\@glxtr@record@setting\space which requires a
335     combination of bib2gls (to fetch entries) and makeindex/xindy
336     (to sort and collate the entries). If you only want to use
337     bib2gls then change the option to record=only or record=nameref}%
338   }%
339 }
```

cord@only@setup    Initialisation code for record=only and record=nameref

```
340 \newcommand*{\@glxtr@record@only@setup}{%
341   \def\glxtr@setup@record{%
342     \@glxtr@autoseeindexfalse
343     \let\@do@seeglossary\@glxtr@recordsee
344     \let\@glxtr@record\@glxtr@record
345     \let\@do@wrglossary\@glxtr@do@record@wrglossary
346     \let\@gls@saveentrycounter\relax
347     \let\glxtrundefaction\@glxtr@warn@undefaction
348     \let\glxtr@warnonexistsordo\@glxtr@warn@onexistsordo
349     \glxtr@addloclistfield
350     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
351     \let\@glxtr@recordcounter\@glxtr@op@recordcounter
```

```

352 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
Switch off the index suppression for aliased entries. (bib2gls will deal with them.)
353 \def\glstrsetaliasnoindex{}%
\@glstrsetupsort@none was only introduced to glossaries v4.30, so it may not be available.
If it's defined, use it to remove the unnecessary overhead of escaping and sanitizing the sort
value.
354 \ifdef\@glstrsetupsort@none{\@glstrsetupsort@none}{}%
Warn about using \printglossary:
355 \def\glstrNoGlossaryWarning{\@glstr@record@noglossarywarning}%
Load glossaries-extra-bib2gls:
356 \RequirePackage{glossaries-extra-bib2gls}%
357}%
358}

```

**record** Now define the record package option. As from v1.46, record=alsoindex is a deprecated synonym of record=hybrid to avoid confusion.

```

359 \define@choicekey{glossaries-extra.sty}{record}
360 [ \@glstr@record@setting\glstr@record@nr ]%
361 { off, only, alsoindex, nameref, hybrid }%
362 [ only ]%
363 {%
364 \ifcase\glstr@record@nr\relax
Don't record.
365 \def\glstr@setup@record{%
366 \renewcommand*{\@do@seeglossary}{\@glstr@doseeglossary}%
367 \renewcommand*{\@glstr@record}[3]{}%
368 \let\@do@wrglossary\glstr@do@wrglossary
369 \let\@glstr@saveentrycounter\glstr@indexonly@saveentrycounter
370 \let\glstrundefaction\@glstr@err@undefaction
371 \let\glstr@warnonexistsordo\@gobble
372 \let\@glstr@recordcounter\@glstr@noop@recordcounter
373 \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
374 \undef\glstrsetaliasnoindex
375}%
376 \or

```

Only record (don't index).

```

377 \@glstr@record@only@setup
378 \or

```

Record and index. This option doesn't load glossaries-extra-bib2gls as the sorting is performed by xindy or makeindex. Index in this sense refers to the indexing mechanism used with indexing applications such as makeindex and xindy, but this could be confused with recording locations so "alsoindex" is now deprecated in favour of "hybrid", which is more obvious.

```

379 \def\glstr@setup@record{%
380 \renewcommand*{\@glstr@record@setting@alsoindex}{alsoindex}%

```



```

381 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
382 \let\@glxtr@record\@glxtr@record
383 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
384 \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
385 \let\glxtrundefaction\glxtr@warn@undefaction
386 \let\glxtr@warnonexistssordo\glxtr@warn@onexistssordo
387 \glxtr@addloclistfield
388 \let\@glxtr@recordcounter\glxtr@op@recordcounter
389 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
390 \undef\glxtrsetaliasnoindex
391 }%
392 \or

```

Only record (don't index) but also include nameref information.

```

393 \glxtr@record@only@setup
394 \ifundef\hyperlink
395 {\GlossariesExtraWarning{You have requested record=nameref but
396 the document doesn't support hyperlinks}}%
397 {}%
398 \or
399 % \end{macrocode}
400 % Hybrid record (use bib2gls to fetch definitions) and index (use
401 % makeindex/xindy to sort and collate).
402 % \begin{macrocode}
403 \def\glxtr@setup@record{%
404 \renewcommand*{\@glxtr@record@setting@alsoindex}{hybrid}%
405 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
406 \let\@glxtr@record\@glxtr@record
407 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
408 \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
409 \let\glxtrundefaction\glxtr@warn@undefaction
410 \let\glxtr@warnonexistssordo\glxtr@warn@onexistssordo
411 \glxtr@addloclistfield
412 \let\@glxtr@recordcounter\glxtr@op@recordcounter
413 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
414 \undef\glxtrsetaliasnoindex
415 }%
416 \fi
417 }

```

Version 1.06 changes the docdef option to a choice rather than boolean setting. The available values are: false, true or restricted. The restricted option permits document definitions as long as they occur before the first glossary is displayed.

`\glxtr@docdefval` The docdef value is stored as an integer: 0 (false), 1 (true) and 2 (restricted).

```

418 \newcommand*{\@glxtr@docdefval}{0}

```

Need to provide conditional commands that are backward compatible:

`\if@glxtrdocdef`

```

419 \newcommand*{\if@glxtrdocdef}{\ifnum\@glxtr@docdefval>0 }

lsxtrdocdeftrue
420 \newcommand*{\@glxtrdocdeftrue}{\def\@glxtr@docdefval{1}}

sxtrdocdeffalse
421 \newcommand*{\@glxtrdocdeffalse}{\def\@glxtr@docdefval{0}}

docdef By default don't allow entries to be defined in the document to encourage the user to define
them in the preamble, but if the user is really determined to define them in the document
allow them to request this.
422 \define@choicekey{glossaries-extra.sty}{docdef}
423 [\@glxtr@docdefsetting\@glxtr@docdefval]%
424 {false,true,restricted,atom}[true]%
425 {%
426 \ifnum\@glxtr@docdefval>1\relax
427 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
428 \else
429 \renewcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}%
430 \fi
431 }

ocdefrestricted
432 \newcommand*{\if@glxtrdocdefrestricted}{\ifnum\@glxtr@docdefval>1 }

oifexistsorwarn Need an error to notify user if an undefined entry is being referenced in the glossary for the
docdef=restricted option. This is used by \glossentryname (but not by \glossentrydesc
etc as one error per entry is sufficient).
433 \newcommand*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}

indexcrossrefs Automatically index cross references at the end of the document
434 \define@boolkey{glossaries-extra.sty}{\@glxtr}{indexcrossrefs}[true]{%
435 \if@glxtrindexcrossrefs
436 \else
437 \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
438 \fi
439 }

Switch off since this can increase the build time.
440 \@glxtrindexcrossrefsfalse

But allow see key to switch it on automatically.

oindexcrossrefs
441 \newcommand*{\@glxtr@autoindexcrossrefs}{\@glxtrindexcrossrefstrue}

```

autoseeindex Provide a boolean option to allow the user to prevent the automatic indexing of the cross-referencing keys see, seealso and alias.

```

442 \define@boolkey{glossaries-extra.sty}[@glxtr@]{autoseeindex}[true]{%
443 }
444 \@glxtr@autoseeindextrue

```

equations Provide a boolean option to automatically switch to the equation counter when in a numbered maths environment.

```

445 \define@boolkey{glossaries-extra.sty}[@glxtr@]{equations}[true]{%
446 }
447 \@glxtr@equationsfalse

```

\glxtr@float

```

448 \let\glxtr@float\@float

```

glxtr@dblfloat

```

449 \let\glxtr@dblfloat\@dblfloat

```

floats Provide a boolean option to automatically switch to the the corresponding counter when in a float.

```

450 \define@boolkey{glossaries-extra.sty}[@glxtr@]{floats}[true]{%
451   \if@glxtr@floats
452     \renewcommand*{\@float}[1]{\renewcommand{\glscounter}{##1}\glxtr@float{##1}}%
453     \renewcommand*{\@dblfloat}[1]{\renewcommand{\glscounter}{##1}\glxtr@dblfloat{##1}}%
454   \else
455     \let\@float\glxtr@float
456     \let\@dblfloat\glxtr@dblfloat
457   \fi
458 }
459 \@glxtr@floatsfalse

```

iesExtraWarning Allow users to suppress warnings.

```

460 \newcommand*{\GlossariesExtraWarning}[1]{\PackageWarning{glossaries-extra}{#1}}

```

raWarningNoLine Allow users to suppress warnings.

```

461 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
462   \PackageWarningNoLine{glossaries-extra}{#1}}

463 \@glxtr@declareoption{nowarn}{%
464   \let\GlossariesExtraWarning\@gobble
465   \let\GlossariesExtraWarningNoLine\@gobble
466   \glxtr@doption{nowarn}%
467 }

```

xtr@defpostpunc Redefines \glspostdescription. The postdot and nopostdot options will have to redefine this.

```

468 \newcommand*{\@glxtr@defpostpunc}{%

```

postdot    Shortcut for nopostdot=false

```
469 \@glsxtr@declareoption{postdot}{}%
470   \glsxtr@doption{nopostdot=false}%
471   \renewcommand*{\@glsxtr@defpostpunc}{%
472     \renewcommand*{\glspostdescription}{%
473       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
474   }%
475 }
```

nopostdot    Needs to redefine \@glsxtr@defpostpunc

```
476 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
477   \glsxtr@doption{nopostdot=#1}%
478   \renewcommand*{\@glsxtr@defpostpunc}{%
479     \renewcommand*{\glspostdescription}{%
480       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
481   }%
482 }
```

postpunc    Set the post-description punctuation. This also sets the \ifglsnopostdot conditional, which now indicates if the post-description punctuation has been suppressed.

```
483 \define@key{glossaries-extra.sty}{postpunc}{%
484   \glsxtr@doption{nopostdot=false}%
485   \ifstrequal{#1}{dot}%
486   {%
487     \renewcommand*{\@glsxtr@defpostpunc}{%
488       \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
489     }%
490   }%
491   {%
492     \ifstrequal{#1}{comma}%
493     {%
494       \renewcommand*{\@glsxtr@defpostpunc}{%
495         \renewcommand*{\glspostdescription}{,}%
496       }%
497     }%
498     {%
499       \ifstrequal{#1}{none}%
500       {%
501         \glsxtr@doption{nopostdot=true}%
502         \renewcommand*{\@glsxtr@defpostpunc}{%
503           \renewcommand*{\glspostdescription}{}%
504         }%
505       }%
506       {%
507         \renewcommand*{\@glsxtr@defpostpunc}{%
508           \renewcommand*{\glspostdescription}{#1}%
509         }%
510       }%
511     }%
512   }%
513 }
```

```

512 }%
513 }

glsxtrabbrvtype Glossary type for abbreviations.
514 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}

abbreviationsdef Set by abbreviations option.
515 \newcommand*{\@glsxtr@abbreviationsdef}{}

abbreviationsdef
516 \newcommand*{\@glsxtr@doabbreviationsdef}{%
517   \ifpackageloaded{babel}%
518   {\providecommand{\abbreviationsname}{\acronymname}}%
519   {\providecommand{\abbreviationsname}{Abbreviations}}%
520   \newglossary[glg-abr]{abbreviations}{gls-abr}{glo-abr}{\abbreviationsname}%
521   \renewcommand*{\glsxtrabbrvtype}{abbreviations}%
522   \newcommand*{\printabbreviations}[1][1]{%
523     \printglossary[type=\glsxtrabbrvtype,##1]%
524   }%
525   \disable@keys{glossaries-extra.sty}{abbreviations}%

   If the acronym option hasn't been used, change \acronymtype to \glsxtrabbrvtype.
526   \ifglsacronym
527   \else
528     \renewcommand*{\acronymtype}{\glsxtrabbrvtype}%
529   \fi
530 }%

abbreviations If abbreviations, create a new glossary type for abbreviations.
531 \@glsxtr@declareoption{abbreviations}{%
532   \let\@glsxtr@abbreviationsdef\@glsxtr@doabbreviationsdef
533 }

AbbreviationShortcuts Enable shortcut commands for the abbreviations. Unlike the analogous command provided
by glossaries, this uses \newcommand instead of \let as a safety feature (except for \newabbr
which is also provided with \GlsXtrDefineAcShortcuts).
534 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
535   \newcommand*{\ab}{\cgl}%
536   \newcommand*{\abp}{\cglsp}%
537   \newcommand*{\as}{\glsxtrshort}%
538   \newcommand*{\asp}{\glsxtrshortpl}%
539   \newcommand*{\al}{\glsxtrlong}%
540   \newcommand*{\alp}{\glsxtrlongpl}%
541   \newcommand*{\af}{\glsxtrfull}%
542   \newcommand*{\afp}{\glsxtrfullpl}%
543   \newcommand*{\Ab}{\cGls}%
544   \newcommand*{\Abp}{\cGlspl}%
545   \newcommand*{\As}{\Glsxtrshort}%
546   \newcommand*{\Asp}{\Glsxtrshortpl}%

```

```

547 \newcommand*{\Al}{\Glsxtrlong}%
548 \newcommand*{\Alp}{\Glsxtrlongpl}%
549 \newcommand*{\Af}{\Glsxtrfull}%
550 \newcommand*{\Afp}{\Glsxtrfullpl}%
551 \newcommand*{\AB}{\cGLS}%
552 \newcommand*{\ABP}{\cGLSpl}%
553 \newcommand*{\AS}{\GLSxtrshort}%
554 \newcommand*{\ASP}{\GLSxtrshortpl}%
555 \newcommand*{\AL}{\GLSxtrlong}%
556 \newcommand*{\ALP}{\GLSxtrlongpl}%
557 \newcommand*{\AF}{\GLSxtrfull}%
558 \newcommand*{\AFP}{\GLSxtrfullpl}%

559 \providecommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

560 \let\GlsXtrDefineAbbreviationShortcuts\relax
561 }

```

`\defineAcShortcuts` Enable shortcut commands for the abbreviations, but uses the analogous commands provided by glossaries.

```

562 \newcommand*{\GlsXtrDefineAcShortcuts}{%
563 \newcommand*{\ac}{\cglS}%
564 \newcommand*{\acp}{\cglSpl}%
565 \newcommand*{\acs}{\glSxtrshort}%
566 \newcommand*{\acsp}{\glSxtrshortpl}%
567 \newcommand*{\acl}{\glSxtrlong}%
568 \newcommand*{\aclp}{\glSxtrlongpl}%
569 \newcommand*{\acf}{\glSxtrfull}%
570 \newcommand*{\acfp}{\glSxtrfullpl}%
571 \newcommand*{\Ac}{\cGls}%
572 \newcommand*{\Acp}{\cGlspl}%
573 \newcommand*{\Acs}{\Glsxtrshort}%
574 \newcommand*{\Acsp}{\Glsxtrshortpl}%
575 \newcommand*{\Acl}{\Glsxtrlong}%
576 \newcommand*{\Aclp}{\Glsxtrlongpl}%
577 \newcommand*{\Acf}{\Glsxtrfull}%
578 \newcommand*{\Acfp}{\Glsxtrfullpl}%
579 \newcommand*{\AC}{\cGLS}%
580 \newcommand*{\ACP}{\cGLSpl}%
581 \newcommand*{\ACS}{\GLSxtrshort}%
582 \newcommand*{\ACSP}{\GLSxtrshortpl}%
583 \newcommand*{\ACL}{\GLSxtrlong}%
584 \newcommand*{\ACLP}{\GLSxtrlongpl}%
585 \newcommand*{\ACF}{\GLSxtrfull}%
586 \newcommand*{\ACFP}{\GLSxtrfullpl}%

587 \providecommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

588 \let\GlsXtrDefineAcShortcuts\relax
589 }

```

**eOtherShortcuts** Similarly provide shortcut versions for the commands provided by the symbols and numbers options.

```

590 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
591 \newcommand*{\newentry}{\newglossaryentry}%
592 \ifdef\printsymbols
593 {%
594 \newcommand*{\newsym}{\glstrnewsymbol}%
595 }{}%
596 \ifdef\printnumbers
597 {%
598 \newcommand*{\newnum}{\glstrnewnumber}%
599 }{}%
600 \let\GlsXtrDefineOtherShortcuts\relax
601 }

```

Always use the long forms, not the shortcuts, where portability is an issue. (For example, when defining entries in a file that may be input by multiple documents.)

**@setupshortcuts** Command used to set the shortcuts option.

```

602 \newcommand*{\@glstr@setupshortcuts}{}

```

**tr@shortcutsval** Store the value of the shortcuts option. (Needed by bib2gls.)

```

603 \newcommand*{\@glstr@shortcutsval}{\ifglstrshortcuts acro\else none\fi}%

```

**shortcuts** Provide shortcuts option. Unlike the glossaries version, this is a choice rather than a boolean key but it also provides shortcuts=true and shortcuts=false, which are equivalent to shortcuts=all and shortcuts=none. Multiple use of this option in the *same* option list will override each other. New to v1.17: shortcuts=ac which implements \GlsXtrDefineAcShortcuts (not included in shortcuts=all as it conflicts with other shortcuts).

```

604 \define@choicekey{glossaries-extra.sty}{shortcuts}%
605 [ \@glstr@shortcutsval \@glstr@shortcutsnr ]%
606 {acronyms,acro,abbreviations,abbr,other,all,true,ac,none,false}[true]{%
607 \ifcase \@glstr@shortcutsnr\relax % acronyms
608 \renewcommand*{\@glstr@setupshortcuts}{%
609 \glstrshortcuts true
610 \DefineAcronymSynonyms
611 }%
612 \or % acro
613 \renewcommand*{\@glstr@setupshortcuts}{%
614 \glstrshortcuts true
615 \DefineAcronymSynonyms
616 }%
617 \or % abbreviations
618 \renewcommand*{\@glstr@setupshortcuts}{%
619 \GlsXtrDefineAbbreviationShortcuts

```

```

620     }%
621 \or % abbr
622   \renewcommand*{\@glxtr@setupshortcuts}{%
623     \GlsXtrDefineAbbreviationShortcuts
624   }%
625 \or % other
626   \renewcommand*{\@glxtr@setupshortcuts}{%
627     \GlsXtrDefineOtherShortcuts
628   }%
629 \or % all
630   \renewcommand*{\@glxtr@setupshortcuts}{%
631     \glsacrshortcutstrue
632     \GlsXtrDefineAcShortcuts
633     \GlsXtrDefineAbbreviationShortcuts
634     \GlsXtrDefineOtherShortcuts
635   }%
636 \or % true
637   \renewcommand*{\@glxtr@setupshortcuts}{%
638     \glsacrshortcutstrue
639     \GlsXtrDefineAcShortcuts
640     \GlsXtrDefineAbbreviationShortcuts
641     \GlsXtrDefineOtherShortcuts
642   }%
643 \or % ac
644   \renewcommand*{\@glxtr@setupshortcuts}{%
645     \glsacrshortcutstrue
646     \GlsXtrDefineAcShortcuts
647   }%
648 \else % none, false
649   \renewcommand*{\@glxtr@setupshortcuts}{}%
650 \fi
651 }

```

Leave none and false as last option.

lsxtr@doaccsupp

```
652 \newcommand*{\@glxtr@doaccsupp}{}
```

glossaries-accsupp can't be loaded after glossaries-extra. glossaries-accsupp v4.29+ checks \@glxtr@doaccsupp to determine if it's been loaded too late.

accsupp If accsupp, load glossaries-accsupp package.

```

653 \@glxtr@declareoption{accsupp}{%
654 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

tr@doloadprefix

```
655 \newcommand*{\@glxtr@doloadprefix}{}
```



prefix If prefix, load glossaries-prefix package.

```
656 \@glsxtr@declareoption{prefix}{%
657 \renewcommand*{\@glsxtr@doloadprefix}{\RequirePackage{glossaries-prefix}}}
```

GlossaryWarning Warning text displayed in document if the external glossary file given by the argument is missing.

```
658 \newcommand{\glsxtrNoGlossaryWarning}[1]{%
659 \GlossariesExtraWarning{Glossary ‘#1’ is missing}%
660 \@glsxtr@defaultnoglossarywarning{#1}%
661 }
```

omissingglstext If true, suppress the text and warning produced if the external glossary file is missing.

```
662 \define@choicekey{glossaries-extra.sty}{nomissingglstext}
663 [ \@glsxtr@nomissingglstextval \@glsxtr@nomissingglstextnr ]%
664 {true,false}[true]{%
665 \ifcase \@glsxtr@nomissingglstextnr \relax % true
666 \renewcommand{\glsxtrNoGlossaryWarning}[1]{\null}%
667 \else % false
668 \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
669 \@glsxtr@defaultnoglossarywarning{#1}%
670 }%
671 \fi
672 }
```

Provide option to load glossaries-extra-stylemods (Deferred to the end.)

xtr@redefstyles

```
673 \newcommand*{\@glsxtr@redefstyles}{}
```

stylemods

```
674 \define@key{glossaries-extra.sty}{stylemods}[default]{%
675 \ifstrequal{#1}{default}%
676 {%
677 \renewcommand*{\@glsxtr@redefstyles}{%
678 \RequirePackage{glossaries-extra-stylemods}}%
679 }%
680 {%
681 \ifstrequal{#1}{all}%
682 {%
683 \renewcommand*{\@glsxtr@redefstyles}{%
684 \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
685 \RequirePackage{glossaries-extra-stylemods}%
686 }%
687 }%
688 {%
689 \renewcommand*{\@glsxtr@redefstyles}{}%
690 \@for \@glsxtr@tmp:=#1\do{%
691 \IfFileExists{glossary-\@glsxtr@tmp.sty}%
692 {%
```

```

693      \eappto\@glsxtr@redefstyles{%
694      \noexpand\RequirePackage{glossary-\@glsxtr@tmp}}}%
695  }%
696  {%
697      \PackageError{glossaries-extra}%
698      {Glossaries style package ‘glossary-\@glsxtr@tmp.sty’
699      doesn’t exist (did you mean to use the ‘style’ key?)}%
700      {The list of values (#1) in the ‘stylemods’ key should
701      match the glossary-xxx.sty files provided with
702      glossaries.sty}%
703  }%
704  }%
705  \appto\@glsxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}}%
706  }
707  }%
708 }

```

glsxtr@do@style

```

709 \newcommand*{\@glsxtr@do@style}{}

```

**style** Since the stylemods option can automatically load extra style packages, deal with the style option after those packages have been loaded.

```

710 \define@key{glossaries-extra.sty}{style}{%

```

Defer actual style change:

```

711 \renewcommand*{\@glsxtr@do@style}{%

```

Set this as the default style:

```

712 \setkeys{glossaries.sty}{style={#1}}%

```

Set this style:

```

713 \setglossarystyle{#1}%

```

```

714 }%

```

```

715 }

```

**c@wrglossaryctr** Increments the associated counter if enabled. Does nothing by default. The optional argument is the entry label in case it’s required, but the wrglossary counter is globally used by all entries.

```

716 \newcommand*{\glsxtr@inc@wrglossaryctr}[1]{}

```

cationHyperlink

`\glsxtrinternallocationhyperlink{<counter>}{<prefix>}{<location>}`

The first two arguments are always control sequences.

```

717 \newcommand*{\GlsXtrInternalLocationHyperlink}[3]{%

```

```

718 \glsxtrhyperlink{#1#2#3}{#3}%

```

```

719 }

```

cationhyperlink

```
720 \newcommand*{\@glxstr@wrglossary@locationhyperlink}[3]{%
721   \pageref{wrglossary.#3}%
722 }
```

**indexcounter** Define the wrglossary counter that's incremented every time an entry is indexed, except for cross-references. This is designed for use with bib2gls v1.4+. It can work with the other indexing methods but it will interfere with the number list collation. This option automatically implements counter=wrglossary.

Since glossaries automatically loads amsmath, there may be a problem if the indexing occurs in the equation environment, because only one \label is allowed in each instance of that environment. It's best to change the counter when in maths mode.

```
723 \@glxstr@declareoption{indexcounter}{%
724   \glxstr@doption{counter=wrglossary}%
725   \ifundef\c@wrglossary
726   {%
727     \newcounter{wrglossary}%
728     \renewcommand{\thewrglossary}{\arabic{wrglossary}}%
729   }%
730   {}%
731   \renewcommand*{\glxstr@inc@wrglossaryctr}[1]{%
```

Only increment if the current counter is wrglossary.

```
732   \ifdefstring\@glsc@counter{wrglossary}%
733   {%
734     \refstepcounter{wrglossary}%
735     \label{wrglossary.\thewrglossary}%
736   }%
737   {}%
738 }%
739 \renewcommand*{\GlsXtrInternalLocationHyperlink}[3]{%
740   \ifdefstring\glsc@entrycounter{wrglossary}%
741   {%
742     \@glxstr@wrglossary@locationhyperlink{##1}{##2}{##3}%
743   }%
744   {\glxstr@hyperlink{##1##2##3}{##3}}%
745   }%
746 }
```

**glxstrwrglossmark** Marks the place where indexing occurs. Does nothing by default.

```
747 \newcommand*{\@glxstrwrglossmark}{}
```

**glxstrwrglossmark** Since \glscadd can be used in the preamble, this action needs to be disabled until the start of the document.

```
748 \newcommand*{\@glxstrwrglossmark}{}
749 \AtBeginDocument{\renewcommand*{\@glxstrwrglossmark}{\@glxstrwrglossmark}}
```

**glxstrwrglossmark** Does nothing by default.

```
750 \newcommand*{\glxstrwrglossmark}{\ensuremath{\cdot}}
```

debug Provide extra debug options.

```
751 \define@choicekey{glossaries-extra.sty}{debug}  
752 [ \@glstr@debugval \@glstr@debugnr ] %  
753 {true,false,showtargets,showwrgloss,all,showaccsupp}[true]{%  
754 \ifcase\@glstr@debugnr\relax % true  
755 \glstr@doption{debug=true}%  
756 \renewcommand*{\@glstrwrglossmark}{}%  
757 \or % false  
758 \glstr@doption{debug=false}%  
759 \renewcommand*{\@glstrwrglossmark}{}%  
760 \or % showtargets  
761 \glstr@doption{debug=showtargets}%  
762 \or % showwrgloss  
763 \glstr@doption{debug=true}%  
764 \renewcommand*{\@glstrwrglossmark}{\glstrwrglossmark}%  
765 \or % all  
766 \glstr@doption{debug=showtargets,debug=showaccsupp}%  
767 \renewcommand*{\@glstrwrglossmark}{\glstrwrglossmark}%  
768 \or % showaccsupp  
769 \glstr@doption{debug=showaccsupp}%  
770 \fi  
771 }
```

Pass all other options to glossaries.

```
772 \DeclareOptionX*{%  
773 \expandafter\glstr@doption\expandafter{\CurrentOption}}
```

Process options.

```
774 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
775 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
776 \@glstr@doaccsupp
```

Load the glossaries-prefix package if required.

```
777 \@glstr@doloadprefix
```

Redefine \glspostdescription if required.

```
778 \@glstr@defpostpunc
```

\glsshowtarget This command was introduced to glossaries v4.32 so it may not be defined. Therefore it's defined here using \def. \glsshowtargetouter was introduced in glossaries v4.45, so that also may not be defined.

```
779 \ifdef\glsshowtargetouter  
780 {  
781 \renewcommand*{\glsshowtarget}[1]{%  
782 \glstr@titleorpdforheading  
783 {%  
784 \ifmmode
```

```

785     \nfss@text{\glsshowtargetfont [#1]}%
786   \else
787     \ifinner
788       {\glsshowtargetfont [#1]}%
789     \else
790       \glsshowtargetouter{#1}%
791     \fi
792   \fi
793 }%
794 {[#1]}%
795 {\protect\glsshowtargetfont [#1]}%
796 }
797 }
798 {

```

Old definition.

```

799 \def\glsshowtarget#1{%
800   \glstrtitleorpdforheading
801   {%
802     \ifmmode
803       \texttt{\small [#1]}%
804     \else
805       \ifinner
806         \texttt{\small [#1]}%
807       \else
808         \marginpar{\texttt{\small #1}}%
809       \fi
810     \fi
811   }%
812   {[#1]}%
813   {\texttt{\small [#1]}%
814 }
815 }

```

`g@doseeglossary` Save original definition of `\@do@seeglossary`

```

816 \let\@glxtr@org@doseeglossary\@do@seeglossary

```

`r@doseeglossary` This doesn't increment the associated counter.

```

817 \newcommand*{\@glxtr@doseeglossary}[2]{%
818   \glsdofexists{#1}%
819   {%
820     \@glxtrwrglossmark
821     \@glxtr@org@doseeglossary{#1}{#2}%
822   }%
823 }

```

`oindex@glossary`

```

824 \newcommand*{\@glxtr@dosee@alsoindex@glossary}[2]{%
825   \@glxtr@recordsee{#1}{#2}%

```

```

826 \@glxtr@doseeglossary{#1}{#2}%
827 }

@org@gloautosee Save and restore original definition of \@glo@autosee. (That command may not be defined
as it was only introduced to glossaries v4.30, in which case the synonym won't be defined
either.)
828 \let\@glxtr@org@gloautosee\@glo@autosee

Check if user tried autoseeindex=false when it can't be supported.
829 \if@glxtr@autoseeindex
830 \else
831 \ifdef\@glxtr@org@gloautosee
832 {}%
833 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
834 option requires at least v4.30 of glossaries.sty}%
835 {You need to update the glossaries.sty package}%
836 }
837 \fi

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the au-
toseeindex option.
838 \ifdef\@glo@autosee
839 {%
840 \renewcommand*{\@glo@autosee}{%
841 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
842 }%
843 {}

checkseeallowed Don't prohibit the use of the see key before the indexing files have been opened if the auto-
matic see indexing has been disabled, since it's no longer an issue.
844 \renewcommand*{\gls@checkseeallowed}{%
845 \if@glxtr@autoseeindex\@gls@see@noindex\fi
846 }

Define abbreviations glossaries if required.
847 \@glxtr@abbreviationsdef
848 \let\@glxtr@abbreviationsdef\relax

Setup shortcuts if required.
849 \@glxtr@setupshortcuts

Redefine \@glxtr@redef@for@gl@sentries if required.
850 \@glxtr@redef@for@gl@sentries

ariesextrasetup Allow user to set options after the package has been loaded. First modify \@glxtr@doooption
so that it now uses \@setupglossaries:
851 \renewcommand{\@glxtr@doooption}[1]{\@setupglossaries{#1}}%

```

Disable options that can only be used when the package is loaded:

```
852 \disable@keys{glossaries-extra.sty}{accsupp}
```

Now define the user command:

```
853 \newcommand*{\glossariesextrasetup}[1]{%
854   \let\glsxtr@setup@record\relax
855   \let\glsxtr@setupshortcuts\relax
856   \let\glsxtr@redef@forglsentries\relax
857   \let\glsxtr@doloadprefix\relax
858   \setkeys{glossaries-extra.sty}{#1}%
859   \@glsxtr@abbreviationsdef
860   \let\glsxtr@abbreviationsdef\relax
861   \@glsxtr@setupshortcuts
862   \glsxtr@setup@record
863   \glsxtr@redef@forglsentries
864   \@glsxtr@doloadprefix
865 }
```

`@@do@wrglossary` Save original definition of `\@@do@wrglossary`.

```
866 \let\glsxtr@org@@do@wrglossary\@@do@wrglossary
```

`@@do@wrglossary` The new version adds code that can show a marker for debugging and increments the associated counter if enabled.

```
867 \newcommand*{\glsxtr@@do@wrglossary}[1]{%
868   \@@glsxtrwrglossmark
869   \glsxtr@inc@wrglossaryctr{#1}%
870   \glsxtr@org@@do@wrglossary{#1}%
871 }
```

`saveentrycounter` Save original definition of `\@gls@saveentrycounter`.

```
872 \let\glsxtr@saveentrycounter\@gls@saveentrycounter
```

`saveentrycounter` Change `\@gls@saveentrycounter` so that it only stores the entry counter information if the indexing is on.

```
873 \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
```

`getcounterprefix` This command is provided by the base glossaries package, but is redefined here. The standard indexing methods don't directly store the hypertarget but instead need to split it into the counter, prefix and location parts, which can be reconstituted in the location list. Unfortunately, not all targets are in this form, so the links fail. With `record=nameref`, the complete target name can be saved, so this modification adjusts the warning.

```
874 \renewcommand*\@gls@getcounterprefix[2]{%
875   \protected@edef\@gls@thisloc{#1}\protected@edef\@gls@thisHloc{#2}%
876   \ifx\@gls@thisloc\@gls@thisHloc
877     \def\@glo@counterprefix{}%
878   \else
879     \def\@gls@get@counterprefix##1.#1##2\end@getprefix{%
880       \def\@glo@tmp{##2}%

```

```

881 \ifx\@glo@tmp\@empty
882 \def\@glo@counterprefix{}%
883 \else
884 \def\@glo@counterprefix{##1}%
885 \fi
886 }%
887 \@gls@get@counterprefix#2.#1\end@getprefix
Warn if no prefix can be formed, unless record=nameref.
888 \ifx\@glo@counterprefix\@empty
889 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@nameref
890 \else
891 \GlossariesExtraWarning{Hyper target ‘#2’ can’t be formed by
892 prefixing^^Jlocation ‘#1’. You need to modify the
893 definition of \string\theH\@gls@counter^^Jotherwise you
894 will get the warning: “name{\@gls@counter.#1}’ has been^^J
895 referenced but does not exist”%
896 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@only
897 . You may want to consider using record=nameref instead%
898 \fi}%
899 \fi
900 \fi
901 \fi
902 }

```

Provide script dialect hook (does nothing unless redefined by glossaries-extra-bib2gls).

sxtrdialecthook

```

903 \newcommand*{\@glsxtrdialecthook}{}

```

Set up record option if required.

```

904 \glsxtr@setup@record

```

Disable preamble-only options and switch on the undefined tag at the start of the document.

```

905 \AtBeginDocument{%
906 \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
907 \def\@glsxtrundeftag{\glsxtrundeftag}%
908 }

```

## 1.2 Extra Utilities

usedOrUndefined

```

\GlsXtrIfUnusedOrUndefined{<label>}{<true>}{<false>}

```

Does *<true>* if the entry given by *<label>* is either undefined or hasn’t been used (or has had the first use flag reset).



```

909 \newcommand*{\GlsXtrIfUnusedOrUndefined}[3]{%
910   \ifglentryexists{#1}%
911   {\ifbool{glo@\glsdetoklabel{#1}@flag}{#3}{#2}}%
912   {#2}}%
913 }

```

Starred form of `\ifglossaryexists` was only introduced to glossaries v4.46 so provide it if it hasn't been defined.

```

914 \ifdef\s@ifglossaryexists
915 {}
916 {

```

`\ifglossaryexists`

```

917   \renewcommand{\ifglossaryexists}{%
918     \@ifstar\s@ifglossaryexists\@ifglossaryexists
919   }

```

`\ifglossaryexists`

```

920   \newcommand{\@ifglossaryexists}[3]{%
921     \ifcsundef{glotype@#1@out}{#3}{#2}%
922   }

```

`\ifglossaryexists`

```

923   \newcommand{\s@ifglossaryexists}[3]{%
924     \ifcsundef{glolist@#1}{#3}{#2}%
925   }
926 }

```

`\ifemptyglossary`

`\glstrifemptyglossary{<type>}{<true>}{<false>}`

Provide command to determine if any entries have been added to the glossary (where the glossary label is provided in the first argument). The entries are stored in the comma-separated list `\glolist@<type>`. If this hasn't been defined, the glossary doesn't exist. If it has been defined and is simply a comma, the glossary exists and is empty. (It's initialised to a comma.)

```

927 \newcommand{\glstrifemptyglossary}[3]{%
928   \ifcsdef{glolist@#1}%
929   {%
930     \ifcsstring{glolist@#1}{,}{#2}{#3}%
931   }%
932   {%
933     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
934     #2%
935   }%
936 }

```

xtrifkeydefined Tests if the key given in the first argument has been defined.

```
937 \newcommand*{\glxtrifkeydefined}[3]{%
938   \key@ifundefined{glossentry}{#1}{#3}{#2}%
939 }
```

providestoragekey Like \gl saddstoragekey but does nothing if the key has already been defined.

```
940 \newcommand*{\glxtrprovidestoragekey}{%
941   \@ifstar\@sglsxtr@provide@storagekey\@glxtr@provide@storagekey
942 }
```

vide@storagekey Unstarred version.

```
943 \newcommand*{\@glxtr@provide@storagekey}[3]{%
944   \key@ifundefined{glossentry}{#1}%
945   {%
946     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
947     \appto\@gls@keymap{,{#1}{#1}}%
948     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
949     \appto\@newglossaryentryposthook{%
950       \letcs{\@glo@tmp}{@glo@#1}%
951       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
952     }%

```

Allow the user to omit the user level command if they only intended fetching the value with  
\glxtrusefield

```
953   \ifblank{#3}
954   {}%
955   {%
956     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
957   }%
958 }%
959 {%

```

Provide the no-link command if not already defined.

```
960   \ifblank{#3}
961   {}%
962   {%
963     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
964   }%
965 }%
966 }
```

vide@storagekey Starred version.

```
967 \newcommand*{\s@glxtr@provide@storagekey}[1]{%
968   \key@ifundefined{glossentry}{#1}%
969   {%
970     \expandafter\newcommand\expandafter*\expandafter
971     {\csname gls@assign@#1@field\endcsname}[2]{%
972       \@gls@expand@field{##1}{#1}{##2}%
973     }%

```

```

974 }%
975 {}%
976 \@glxstr@provide@addstoragekey{#1}%
977 }

```

The name of a text-block control sequence can be stored in a field (given by `\GlsXtrFmtField`). This command can then be used with `\glxstrfmt[⟨options⟩]{⟨label⟩}{⟨text⟩}` which effectively does `\glslink[⟨options⟩]{⟨label⟩}{⟨cs⟩{⟨text⟩}}` If the field hasn't been set for that entry just `⟨text⟩` is done.

`\GlsXtrFmtField`

```

978 \newcommand{\GlsXtrFmtField}{useri}

```

`\tDefaultOptions`

```

979 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

`\glxstrfmt` The post-link hook isn't done. This now has a starred form that checks for a final optional argument.

```

980 \newrobustcmd*{\glxstrfmt}{\@ifstar\s@glxstrfmt\@glxstrfmt}

```

`\@glxstrfmt` Unstarred form.

```

981 \newcommand*{\@glxstrfmt}[3][\@glxstrfmt{#1}{#2}{#3}]{%

```

`\s@glxstrfmt` Starred form.

```

982 \newcommand*{\s@glxstrfmt}[3][\@glxstrfmt{#1}{#2}{#3}]{%
983 \new@ifnextchar[\s@glxstrfmt{#1}{#2}{#3}]{%
984 {\@glxstrfmt{#1}{#2}{#3}]{%
985 }

```

`\s@glxstrfmt` Pick up final optional argument.

```

986 \def\s@glxstrfmt#1#2#3[#4]{\@glxstrfmt{#1}{#2}{#3}{#4}}

```

`\@glxstrfmt` Actual inner working.

```

987 \newcommand*{\@glxstrfmt}[4]{%

```

Since there's no post-link hook to worry about, grouping can be added to provide some protection against nesting (but in general nested link text should be avoided).

```

988 \begingroup
989 \def\glslabel{#2}%
990 \glsdoifexistsordo{#2}%
991 {%
992 \ifglshasfield{\GlsXtrFmtField}{#2}%
993 {%
994 \let\do@gl@link@checkfirsthyper\relax
995 \expandafter\@gl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
996 {\glxstrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
997 }%
998 {\glxstrfmtdisplay{@firstofone}{#3}{#4}}%
999 }%
1000 {%

```

Has the default `noindex` been counteracted? If so, this needs `\glsadd` in case `bib2gls` needs to pick up the record.

```

1001      \begingroup
1002      \@gls@setdefault@glslink@opts
1003      \setkeys{glslink}{\GlsXtrFmtDefaultOptions,#1}%
1004      \ifKV@glslink@noindex\else\glsadd{#2}\fi
1005      \endgroup
1006      \glsxtrfmtdisplay{@firstofone}{#3}{#4}%
1007      }%
1008 \endgroup
1009 }

```

`\glsxtrfmtdisplay` The command used internally by `\glsxtrfmt` to do the actual formatting. The first argument is the control sequence name, the second is the control sequence's argument, the third is the inserted material (if starred form used).

```

1010 \newcommand{\glsxtrfmtdisplay}[3]{\csuse{#1}{#2}#3}

```

`\glsxtrentryfmt` No link or indexing.

```

1011 \ifdef\texorpdfstring
1012 {
1013   \newcommand*{\glsxtrentryfmt}[2]{%
1014     \texorpdfstring{\@glsxtrentryfmt{#1}{#2}}{\glsxtrpdfentryfmt{#1}{#2}}%
1015   }
1016 }
1017 {
1018   \newcommand*{\glsxtrentryfmt}{\@glsxtrentryfmt}
1019 }

```

`\glsxtrpdfentryfmt` Use for the PDF bookmarks.

```

1020 \newcommand*{\glsxtrpdfentryfmt}[2]{#2}

```

`@glsxtrentryfmt`

```

1021 \newrobustcmd*{\@glsxtrentryfmt}[2]{%

```

Locally define `\glslabel` in case the helper command needs to access the label.

```

1022  {%
1023   \protected@edef\glslabel{#1}%
1024   \glsdoifexistsordo{#1}%
1025   {%
1026     \ifglshasfield{\GlsXtrFmtField}{#1}%
1027     {%
1028       \csuse{\glscurrentfieldvalue}{#2}%
1029     }%
1030     {#2}%
1031   }%
1032   {#2}%
1033 }%
1034 }

```

`trfieldlistadd` If a field stores an etoolbox internal list (e.g. `loclist`) then this macro provides a convenient way of adding to the list via etoolbox's `\listcsadd`. The first argument is the entry's label, the second is the field label and the third is the element to add to the list.

```
1035 \newcommand*{\glxtrfieldlistadd}[3]{%
1036   \listcsadd{glo@\glsdetoklabel{#1}@\#2}{#3}%
1037 }
```

`trfieldlistgadd` Similarly but uses `\listcsgadd`.

```
1038 \newcommand*{\glxtrfieldlistgadd}[3]{%
1039   \listcsgadd{glo@\glsdetoklabel{#1}@\#2}{#3}%
1040 }
```

`trfieldlisteadd` Similarly but uses `\listcseadd`.

```
1041 \newcommand*{\glxtrfieldlisteadd}[3]{%
1042   \listcseadd{glo@\glsdetoklabel{#1}@\#2}{#3}%
1043 }
```

`trfieldlistxadd` Similarly but uses `\listcsxadd`.

```
1044 \newcommand*{\glxtrfieldlistxadd}[3]{%
1045   \listcsxadd{glo@\glsdetoklabel{#1}@\#2}{#3}%
1046 }
```

Now provide commands to iterate over these lists.

`fielddolistloop`

```
1047 \newcommand*{\glxtrfielddolistloop}[2]{%
1048   \dolistcsloop{glo@\glsdetoklabel{#1}@\#2}%
1049 }
```

`fieldforlistloop`

```
1050 \newcommand*{\glxtrfieldforlistloop}[3]{%
1051   \forlistcsloop{#3}{glo@\glsdetoklabel{#1}@\#2}%
1052 }
```

`fieldformatlist`

```
1053 \newrobustcmd*{\glxtrfieldformatlist}[2]{%
1054   \begingroup
1055   \def\@dtl@formatlist@itemsep{}%
1056   \def\@dtl@formatlist@lastitem{}%
1057   \def\@dtl@formatlist@prelastitem{}%
1058   \def\@dtl@formatlist@prelastitemsep{}%
1059   \forlistcsloop{\@dtl@formatlist@handler}{glo@\glsdetoklabel{#1}@\#2}%
1060   \@dtl@formatlist@prelastitem\@dtl@formatlist@lastitem
1061   \endgroup
1062 }
```

List element tests:

trfieldifinlist First argument label, second argument field, third argument item, fourth true part and fifth false part.

```
1063 \newcommand*{\glstrfieldifinlist}[5]{%
1064   \ifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
1065 }
```

rfieldxifinlist Expands item.

```
1066 \newcommand*{\glstrfieldxifinlist}[5]{%
1067   \xifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
1068 }
```

sxtrforcsvfield

`\glstrforcsvfield{<label>}{<field>}{<cs handler>}`

```
1069 \newcommand*{\glstrforcsvfield}{%
1070   \ifstar\s@glstrforcsvfield\@glstrforcsvfield
1071 }
```

sxtrforcsvfield Unstarred version.

```
1072 \newcommand*{\@glstrforcsvfield}[3]{%
1073   \@glstrifhasfield{#2}{#1}%
1074   {%
1075     \let\glstrendfor\@endfortrue
1076     \@for\@glstr@label:=\glscurrentfieldvalue\do
1077       {\expandafter#3\expandafter{\@glstr@label}}}%
1078   }%
1079 }
```

sxtrforcsvfield Starred version.

```
1080 \newcommand*{\s@glstrforcsvfield}[3]{%
1081   \s@glstrifhasfield{#2}{#1}%
1082   {%
1083     \let\glstrendfor\@endfortrue
1084     \@for\@glstr@label:=\glscurrentfieldvalue\do
1085       {\expandafter#3\expandafter{\@glstr@label}}}%
1086   }%
1087 }
```

ldformatcsvlist

```
1088 \newrobustcmd*{\glstrfieldformatcsvlist}[2]{%
1089   \@glstrifhasfield{#2}{#1}%
1090   {\@dtlformatlist\glscurrentfieldvalue}%
1091   }%
1092 }
```

dValueInCsvList

`\GlsXtrIfFieldValueInCsvList{<label>}{<field>}{<list>}{<true>}{<false>}`

```
1093 \newcommand*{\GlsXtrIfFieldValueInCsvList}{%
1094   \ifstar\s@GlsXtrIfFieldValueInCsvList\@GlsXtrIfFieldValueInCsvList
1095 }
```

Note `\DTLifinlist` performs one level on the list but not the element.

dValueInCsvList    Unstarred version.

```
1096 \newcommand*{\@GlsXtrIfFieldValueInCsvList}[5]{%
1097   \@glstrifhasfield{#2}{#1}%
1098   {%
1099     \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%
1100     {#3}{#4}{#5}%
1101   }%
1102   {#5}%
1103 }
```

dValueInCsvList    Starred version.

```
1104 \newcommand*{\s@GlsXtrIfFieldValueInCsvList}[5]{%
1105   \s@glstrifhasfield{#2}{#1}%
1106   {%
1107     \expandafter\DTLifinlist\expandafter{\glscurrentfieldvalue}%
1108     {#3}{#4}{#5}%
1109   }%
1110   {#5}%
1111 }
```

eInFieldCsvList

`\GlsXtrIfValueInFieldCsvList{<label>}{<field>}{<value>}{<true>}{<false>}`

Essentially the reverse. Tests if the given value is in the given field which should contain a comma-separated list.

```
1112 \newcommand*{\GlsXtrIfValueInFieldCsvList}{%
1113   \@ifstar\s@GlsXtrIfValueInFieldCsvList\@GlsXtrIfValueInFieldCsvList
1114 }
```

eInFieldCsvList    Unstarred version.

```
1115 \newcommand*{\@GlsXtrIfValueInFieldCsvList}[5]{%
1116   \@glstrifhasfield{#2}{#1}%
1117   {%
1118     \DTLifinlist{#3}{\glscurrentfieldvalue}{#4}{#5}%
1119   }%
1120   {#5}%
1121 }
```

eInFieldCsvList    Unstarred version.

```

1122 \newcommand*{\s@GlsXtrIfValueInFieldCsvList}[5]{%
1123   \s@glstrifhasfield{#2}{#1}%
1124   {%
1125     \DTLifinlist{#3}{\glscurrentfieldvalue}{#4}{#5}%
1126   }%
1127   {#5}%
1128 }

```

eInFieldCsvList

`\xGlsXtrIfValueInFieldCsvList{<label>}{<field>}{<value>}{<true>}{<false>}`

As above but fully expand *<value>*.

```

1129 \newcommand*{\xGlsXtrIfValueInFieldCsvList}{%
1130   \@ifstar\s@xGlsXtrIfValueInFieldCsvList\xGlsXtrIfValueInFieldCsvList
1131 }

```

eInFieldCsvList    Unstarred version.

```

1132 \newcommand*{\@xGlsXtrIfValueInFieldCsvList}[5]{%
1133   \@glstrifhasfield{#2}{#1}%
1134   {%
1135     \protected@edef\@glstmp{#3}%
1136     \expandafter\DTLifinlist\expandafter{\@glstmp}{\glscurrentfieldvalue}{#4}{#5}%
1137   }%
1138   {#5}%
1139 }

```

eInFieldCsvList    Unstarred version.

```

1140 \newcommand*{\s@xGlsXtrIfValueInFieldCsvList}[5]{%
1141   \s@glstrifhasfield{#2}{#1}%
1142   {%
1143     \protected@edef\@glstmp{#3}%
1144     \expandafter\DTLifinlist\expandafter{\@glstmp}{\glscurrentfieldvalue}{#4}{#5}%
1145   }%
1146   {#5}%
1147 }

```

lsxtrifhasfield    A simpler alternative to `\ifglshasfield` that doesn't complain if the entry or the field doesn't exist. (No mapping is used.) Grouping is added to the unstarred version allow for nested use.

```

1148 \newrobustcmd{\glstrifhasfield}{%
1149   \@ifstar\s@glstrifhasfield{\@glstrifhasfield}%
1150 }

```

lsxtrifhasfield    Unstarred version adds grouping.



```

1151 \newcommand{\@glxtrifhasfield}[4]{%
1152   {\s@glxtrifhasfield{#1}{#2}{#3}{#4}}%
1153 }

```

lsxtrifhasfield Starred version omits grouping.

```

1154 \newcommand{\s@glxtrifhasfield}[4]{%
1155   \letcs{\glscurrentfieldvalue}{glo@glsetoklabel{#2}@#1}%
1156   \ifundef\glscurrentfieldvalue
1157     {#4}%
1158     {%
1159       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
1160     }%
1161 }

```

rIfFieldNonZero Designed for numeric fields.

```

1162 \newcommand{\GlsXtrIfFieldNonZero}{%
1163   \@ifstar\s@GlsXtrIfFieldNonZero\@GlsXtrIfFieldNonZero
1164 }

```

rIfFieldNonZero

```

1165 \newcommand{\@GlsXtrIfFieldNonZero}[4]{%
1166   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{0}{#4}{#3}%
1167 }

```

rIfFieldNonZero

```

1168 \newcommand{\s@GlsXtrIfFieldNonZero}[4]{%
1169   \s@GlsXtrIfFieldCmpNum{#1}{#2}{=}{0}{#4}{#3}%
1170 }

```

XtrIfFieldEqNum

$\backslash\text{GlsXtrIfFieldEqNum}\{\langle field \rangle\}\{\langle label \rangle\}\{\langle value \rangle\}\{\langle true \rangle\}\{\langle false \rangle\}$
---

Designed for numeric fields.

```

1171 \newcommand{\GlsXtrIfFieldEqNum}{%
1172   \@ifstar\s@GlsXtrIfFieldEqNum\@GlsXtrIfFieldEqNum
1173 }

```

XtrIfFieldEqNum

```

1174 \newcommand{\@GlsXtrIfFieldEqNum}[5]{%
1175   \@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%
1176 }

```

XtrIfFieldEqNum

```

1177 \newcommand{\s@GlsXtrIfFieldEqNum}[5]{%
1178   \s@GlsXtrIfFieldCmpNum{#1}{#2}{=}{#3}{#4}{#5}%
1179 }

```

trIfFieldCmpNum

```
\GlsXtrIfFieldCmpNum{<field>}{<label>}{<comparison>}{<value>}{<true>}
{<false>}
```

Designed for numeric fields.

```
1180 \newcommand{\GlsXtrIfFieldCmpNum}{%
1181   \@ifstar\s@GlsXtrIfFieldCmpNum\@GlsXtrIfFieldCmpNum
1182 }
```

trIfFieldCmpNum

```
1183 \newcommand{\@GlsXtrIfFieldCmpNum}[6]{%
1184   {%
1185     \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1186     \ifundef\glscurrentfieldvalue
1187     {\def\glscurrentfieldvalue{0}}%
1188     {%
1189       \ifdefempty\glscurrentfieldvalue
1190       {\def\glscurrentfieldvalue{0}}%
1191       {%
1192         }%
1193         \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1194       }%
1195     }
```

trIfFieldCmpNum

```
1196 \newcommand{\s@GlsXtrIfFieldCmpNum}[6]{%
1197   \letcs{\glscurrentfieldvalue}{glo@\glsdetoklabel{#2}@#1}%
1198   \ifundef\glscurrentfieldvalue
1199   {\def\glscurrentfieldvalue{0}}%
1200   {%
1201     \ifdefempty\glscurrentfieldvalue
1202     {\def\glscurrentfieldvalue{0}}%
1203     {%
1204       }%
1205       \ifnum\glscurrentfieldvalue#3#4\relax #5\else #6\fi
1206     }
```

XtrIfFieldUndef

```
\GlsXtrIfFieldUndef{<field>}{<label>}{<true>}{<false>}
```

Just uses \ifcsundef.

```
1207 \newcommand{\GlsXtrIfFieldUndef}[2]{%
1208   \ifcsundef{glo@\glsdetoklabel{#2}@#1}%
1209 }
```

`\glxtrusefield` Provide a user-level alternative to `\@gls@entry@field`. The first argument is the entry label. The second argument is the field label.

```
1210 \newcommand*{\glxtrusefield}[2]{%
1211   \@gls@entry@field{#1}{#2}%
1212 }
```

`\Glsxtrusefield` Provide a user-level alternative to `\@Gls@entry@field`.

```
1213 \ifdef\texorpdfstring
1214 {
1215   \newcommand*{\Glsxtrusefield}[2]{%
1216     \texorpdfstring
1217       {\@Gls@entry@field{#1}{#2}}
1218       {\@gls@entry@field{#1}{#2}}%
1219   }
1220 }
1221 {
1222   \newcommand*{\Glsxtrusefield}[2]{%
1223     \@Gls@entry@field{#1}{#2}%
1224   }
1225 }
```

`\GLSxtrusefield` As above but convert to all caps.

```
1226 \ifdef\texorpdfstring
1227 {
1228   \newcommand*{\GLSxtrusefield}[2]{%
1229     \texorpdfstring
1230       {\glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}}%
1231       {\@gls@entry@field{#1}{#2}}%
1232   }
1233 }
1234 {
1235   \newcommand*{\GLSxtrusefield}[2]{%
1236     \glsdoifexists{#1}{\mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}}%
1237   }
1238 }
```

`entryparentname`

```
1239 \newcommand*{\glxtrentryparentname}[1]{%
1240   \ifcsdef{glo@\glsdetoklabel{#1}@parent}%
1241     {\csuse{glo@\csuse{glo@\glsdetoklabel{#1}@parent}@name}}%
1242     {}%
1243 }
```

`\glxtrdeffield` Just use `\csdef` to provide a field value for the given entry.

```
1244 \newcommand*{\glxtrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}
```

`glxstredeffield` Just use `\csdef` to provide a field value for the given entry.

```
1245 \newcommand*{\glxstredeffield}[2]{\protected@csdef{glo@\glsdetoklabel{#1}@#2}}
```

`\trapptocsvfield` Similar to the above but will append value with a leading comma if the field is already defined. This is used by **bib2gls**. There's no check if the entry has been defined. (Because of the way that bib2gls's save-from-alias etc options are implemented, the entry may not have yet been written to the glstex file when this command is used.)

```
1246 \newcommand*{\glstrapptocsvfield}[3]{%
1247   \ifcsdef{glo@\glstetoklabel{#1}@#2}%
1248   {\csappto{glo@\glstetoklabel{#1}@#2}{, #3}}%
1249   {\csdef{glo@\glstetoklabel{#1}@#2}{#3}}%
1250 }
```

`\etfieldifexists`

```
1251 \newcommand*{\glstrsetfieldifexists}[3]{\glstoifexists{#1}{#3}}
```

`\GlsXtrSetField` Allow the user to set a field. First argument entry label, second argument field label, third argument value.

```
1252 \newrobustcmd*{\GlsXtrSetField}[3]{%
1253   \glstrsetfieldifexists{#1}{#2}%
1254   {\csdef{glo@\glstetoklabel{#1}@#2}{#3}}%
1255 }
```

`\GlsXtrLetField` Uses `\cslet` instead. Third argument should be a macro.

```
1256 \newrobustcmd*{\GlsXtrLetField}[3]{%
1257   \glstrsetfieldifexists{#1}{#2}%
1258   {\cslet{glo@\glstetoklabel{#1}@#2}{#3}}%
1259 }
```

`\csGlsXtrLetField` Uses `\csletcs` instead. Third argument should be a control sequence name.

```
1260 \newrobustcmd*{\csGlsXtrLetField}[3]{%
1261   \glstrsetfieldifexists{#1}{#2}%
1262   {\csletcs{glo@\glstetoklabel{#1}@#2}{#3}}%
1263 }
```

`\LetFieldToField` Sets the field for one entry to the field for another entry. Third argument should be the other entry and the fourth argument that other field label.

```
1264 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
1265   \glstrsetfieldifexists{#1}{#2}%
1266   {\csletcs{glo@\glstetoklabel{#1}@#2}{glo@\glstetoklabel{#3}@#4}}%
1267 }
```

`\gGlsXtrSetField` Allow the user to set a field. First argument entry label, second argument field label, third argument value.

```
1268 \newrobustcmd*{\gGlsXtrSetField}[3]{%
1269   \glstrsetfieldifexists{#1}{#2}%
1270   {\csgdef{glo@\glstetoklabel{#1}@#2}{#3}}%
1271 }
```

xGlsXtrSetField

```
1272 \newrobustcmd*{\xGlsXtrSetField}[3]{%
1273   \glstrsetfieldifexists{#1}{#2}%
1274   {\protected@csxdef{glo\xGlsdetoklabel{#1}@#2}{#3}}%
1275 }
```

eGlsXtrSetField

```
1276 \newrobustcmd*{\eGlsXtrSetField}[3]{%
1277   \glstrsetfieldifexists{#1}{#2}%
1278   {\protected@csedef{glo\xGlsdetoklabel{#1}@#2}{#3}}%
1279 }
```

XtrIfFieldEqStr    Starred version uses starred version of \glstrifhasfield (that is, no grouping).

```
1280 \newcommand*{\GlsXtrIfFieldEqStr}{%
1281   \@ifstar\s@GlsXtrIfFieldEqStr\@GlsXtrIfFieldEqStr
1282 }
```

XtrIfFieldEqStr

```
1283 \newrobustcmd*{\@GlsXtrIfFieldEqStr}[5]{%
1284   \@glstrifhasfield{#1}{#2}%
1285   {%
1286     \ifdefstring{glscurrentfieldvalue}{#3}{#4}{#5}%
1287   }%
1288   {#5}%
1289 }
```

XtrIfFieldEqStr

```
1290 \newrobustcmd*{\s@GlsXtrIfFieldEqStr}[5]{%
1291   \s@glstrifhasfield{#1}{#2}%
1292   {%
1293     \ifdefstring{glscurrentfieldvalue}{#3}{#4}{#5}%
1294   }%
1295   {#5}%
1296 }
```

rIfFieldEqXpStr    Like the above but first expands the string. Starred version uses starred version of \glstrifhasfield (that is, no grouping).

```
1297 \newcommand*{\GlsXtrIfFieldEqXpStr}{%
1298   \@ifstar\s@GlsXtrIfFieldEqXpStr\@GlsXtrIfFieldEqXpStr
1299 }
```

rIfFieldEqXpStr

```
1300 \newrobustcmd*{\@GlsXtrIfFieldEqXpStr}[5]{%
1301   \@glstrifhasfield{#1}{#2}%
1302   {%
1303     \protected@edef\@Gls@tmp{#3}%
1304     \ifdefequal{glscurrentfieldvalue}{\@Gls@tmp}{#4}{#5}%
1305   }%
```

```

1306  {#5}%
1307 }

```

rIfFieldEqXpStr

```

1308 \newrobustcmd*{\s@GlsXtrIfFieldEqXpStr}[5]{%
1309   \s@glstrifhasfield{#1}{#2}%
1310   {%
1311     \protected@edef\@glstmp{#3}%
1312     \ifdefequal{\glscurrentfieldvalue}{\@glstmp}{#4}{#5}%
1313   }%
1314   {#5}%
1315 }

```

fXpFieldEqXpStr Like the above but also expands the field value. Starred version uses starred version of \glstrifhasfield (that is, no grouping).

```

1316 \newcommand*{\GlsXtrIfXpFieldEqXpStr}{%
1317   \@ifstar\s@GlsXtrIfXpFieldEqXpStr\@GlsXtrIfXpFieldEqXpStr
1318 }

```

fXpFieldEqXpStr

```

1319 \newrobustcmd*{\@GlsXtrIfXpFieldEqXpStr}[5]{%
1320   \@glstrifhasfield{#1}{#2}%
1321   {%
1322     \protected@edef\@glstmp{\glscurrentfieldvalue}%
1323     \let\glscurrentfieldvalue\@glstmp
1324     \protected@edef\@glstmp{#3}%
1325     \ifdefequal{\glscurrentfieldvalue}{\@glstmp}{#4}{#5}%
1326   }%
1327   {#5}%
1328 }

```

fXpFieldEqXpStr

```

1329 \newrobustcmd*{\s@GlsXtrIfXpFieldEqXpStr}[5]{%
1330   \s@glstrifhasfield{#1}{#2}%
1331   {%
1332     \protected@edef\@glstmp{\glscurrentfieldvalue}%
1333     \let\glscurrentfieldvalue\@glstmp
1334     \protected@edef\@glstmp{#3}%
1335     \ifdefequal{\glscurrentfieldvalue}{\@glstmp}{#4}{#5}%
1336   }%
1337   {#5}%
1338 }

```

sXtrForeignText

<code>\GlsXtrForeignText{&lt;entry label&gt;}{&lt;text&gt;}</code>
--

If a field is used to store a language tag (such as en-GB or de-CH-1996) then this command

uses tracklang's interface to encapsulate  $\langle text \rangle$ . The field identifying the locale is given by `\GlsXtrForeignTextField`.

```
1339 \ifdef\foreignlanguage
1340 {
1341   \ifdef\GetTrackedDialectFromLanguageTag
1342   {
1343     \newcommand{\GlsXtrForeignText}[2]{%
```

In case this is used inside the argument of `\glstrifhasfield`, save and restore `\glscurrentfieldvalue`.

```
1344     \let\@glstr@org@currentfieldvalue\glscurrentfieldvalue
1345     \glstrifhasfield{\GlsXtrForeignTextField}{#1}%
1346     {%
1347       \expandafter\GetTrackedDialectFromLanguageTag\expandafter
1348       {\glscurrentfieldvalue}{\@glstr@dialect}%
1349       \let\@glstr@locale\glscurrentfieldvalue
1350       \let\glscurrentfieldvalue\@glstr@org@currentfieldvalue
1351       \ifdefempty\@glstr@dialect
1352       {%
```

An exact match hasn't been found. A partial match can only be obtained with at least tracklang v1.3.6.

```
1353       \ifundef\TrackedDialectClosestSubMatch
1354       {%
1355         \GlossariesExtraWarning{Can't obtain dialect label
1356         (tracklang v1.3.6+ required)}%
1357       }%
1358       {\let\@glstr@dialect\TrackedDialectClosestSubMatch}%
1359     }%
1360   }%
1361   \ifdefempty\@glstr@dialect
1362   {%
```

No tracked dialect found for the root language.

```
1363   }%
1364   {%
```

Check if there's a caption hook for the given dialect label.

```
1365     \ifcsundef{captions\@glstr@dialect}{}%
1366     {%
```

Dialect label not recognised. Check if there's a known mapping.

```
1367     \IfTrackedDialectHasMapping{\@glstr@dialect}%
1368     {%
1369       \edef\@glstr@dialect{%
1370         \GetTrackedDialectToMapping{\@glstr@dialect}}%
```

Does a caption hook exist for this?

```
1371     \ifcsundef{captions\@glstr@dialect}{}%
1372     {%
```

No mapping. Try root language label instead.

```
1373     \ifcsundef{captions\@tracklang@lang}{}%
```

```

1374         {%
1375         \let\@glsxtr@dialect\@tracklang@lang
1376         }%
1377     }%
1378 }%
1379 {%

No mapping. Try root language label instead.

1380     \ifcsundef{captions\@tracklang@lang}{}%
1381     {%
1382     \let\@glsxtr@dialect\@tracklang@lang
1383     }%
1384 }%
1385 }%
1386 }%
1387 \ifdefempty\@glsxtr@dialect
1388 {%
1389     \GlsXtrUnknownDialectWarning{\@glsxtr@locale}{\@tracklang@lang}%
1390     #2%
1391 }%
1392 {\foreignlanguage{\@glsxtr@dialect}{#2}}%
1393 }%
1394 {#2}% key not set
1395 }
1396 }
1397 {
1398     \newcommand{\GlsXtrForeignText}[2]{%
1399         \GlossariesExtraWarning{Can't encapsulate foreign text:
1400             tracklang v1.3.6+ required}%
1401         #2%
1402     }
1403 }
1404 }
1405 {

\foreignlanguage isn't defined so just do text.
1406 \newcommand{\GlsXtrForeignText}[2]{#2}
1407 }

```

foreignTextField This is the user2 field by default but may be redefined as required.

```

1408 \newcommand*{\GlsXtrForeignTextField}{userii}

```

nDialectWarning

```

1409 \newcommand*{\GlsXtrUnknownDialectWarning}[2]{%
1410     \GlossariesExtraWarning{Can't determine valid dialect label
1411         for locale '#1' (root language: #2)}%
1412 }

```



`\glstrpageref` Like `\glsrefentry` but references the page number instead (if entry counting is on). The base glossaries package only introduced `\GlsEntryCounterLabelPrefix` in version 4.38, so it may not be defined.

```

1413 \ifdef\GlsEntryCounterLabelPrefix
1414 {%
1415   \newcommand*\glstrpageref}[1]{%
1416     \ifglentrycounter
1417       \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1418     \else
1419       \ifglssubentrycounter
1420         \pageref{\GlsEntryCounterLabelPrefix\glsdetoklabel{#1}}%
1421       \else
1422         \gls{#1}%
1423       \fi
1424     \fi
1425   }
1426 }%
1427 {%
1428   \newcommand*\glstrpageref}[1]{%
1429     \ifglentrycounter
1430       \pageref{glsentry-\glsdetoklabel{#1}}%
1431     \else
1432       \ifglssubentrycounter
1433         \pageref{glsentry-\glsdetoklabel{#1}}%
1434       \else
1435         \gls{#1}%
1436       \fi
1437     \fi
1438   }
1439 }%

```

`lossarypreamble`

```

1440 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
1441   \ifcsdef{glolist@#1}%
1442   {%
1443     \ifcsundef{@glossarypreamble@#1}%
1444     {\csdef{@glossarypreamble@#1}{}}%
1445     {}%
1446     \csappto{@glossarypreamble@#1}{#2}%
1447   }%
1448   {%
1449     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1450   }%
1451 }

```

`lossarypreamble`

```

1452 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
1453   \ifcsdef{glolist@#1}%
1454   {%

```

```

1455 \ifcsundef{@glossarypreamble@#1}%
1456 {\csdef{@glossarypreamble@#1}{}}%
1457 {}%
1458 \cspretto{@glossarypreamble@#1}{#2}%
1459 }%
1460 {%
1461 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
1462 }%
1463 }

```

### 1.3 Modifications to Commands Provided by glossaries

Some of the commands provided by glossaries are modified to take into account new options or to change default behaviour.

The original `\gls@entry@field` causes a problem for undefined entries when used in section headings or captions. Since entries must be defined with just the base package this isn't a significant issue, but it will cause a problem with `bib2gls` where no entries are defined on the first  $\LaTeX$  call, so redefine `\gls@entry@field` to use `\csuse` instead of `\csname`.

`\gls@entry@field`

```
\gls@entry@field{<label>}{<field>}
```

This command was introduced to glossaries version 4.03 but older versions are likely to be incompatible with glossaries-extra.

```

1464 \ifdef\gls@entry@field
1465 {
1466 \renewcommand*{\gls@entry@field}[2]{\csuse{gls@entry@field}{#1}{#2}}
1467 }
1468 {}

```

`\ifglsused`

```
\ifglsused{<label>}{<true part>}{<false part>}
```

In the event that undefined entries should trigger a warning rather than an error, `\ifglsused` needs to be modified to check for existence. If the boolean variable is undefined, then its state is indeterminate and is neither true nor false, so neither *<true part>* nor *<false part>* will be performed if *<label>* is undefined. See also `\GlsXtrIfUnusedOrUndefined`.

```

1469 \renewcommand*{\ifglsused}[3]{%
1470 \glsdoifexists{#1}{\ifbool{gls@entry@field}{#2}{#3}}%
1471 }

```

Provide a starred version of `\longnewglossaryentry` that doesn't automatically insert `\leavevmode\unskip\nopostdesc` at the end of the description. The unstarred version is modified to use `\glstrpostlongdescription` instead.

`ewglossaryentry`

```
1472 \renewcommand*{\longnewglossaryentry}{%
1473   \@ifstar\@glstr@s@longnewglossaryentry\@glstr@longnewglossaryentry
1474 }
```

`ewglossaryentry` Starred version.

```
1475 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
1476   \glstoifnoexists{#1}%
1477   {%
1478     \bgroup
1479     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1480     \long\def\@newglossaryentryprehook{%
1481       \long\def\@glo@desc{#3}%
1482       \@org@newglossaryentryprehook
1483     }%
1484     \renewcommand*{\gls@assign@desc}[1]{%
1485       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1486       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1487     }
1488     \gls@defglossaryentry{#1}{#2}%
1489   \egroup
1490   }%
1491 }
```

`ewglossaryentry` Unstarred version.

```
1492 \newcommand{\@glstr@longnewglossaryentry}[3]{%
1493   \glstoifnoexists{#1}%
1494   {%
1495     \bgroup
1496     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
1497     \long\def\@newglossaryentryprehook{%
1498       \long\def\@glo@desc{#3\glstrpostlongdescription}%
1499       \@org@newglossaryentryprehook
1500     }%
1501     \renewcommand*{\gls@assign@desc}[1]{%
1502       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
1503       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1504     }
1505     \gls@defglossaryentry{#1}{#2}%
1506   \egroup
1507   }%
1508 }
```

The following is different from the base glossaries.sty:

```
1503   \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
1504   }
1505   \gls@defglossaryentry{#1}{#2}%
1506 \egroup
1507 }%
1508 }
```

longdescription Hook at the end of the description when using the unstarred \longnewglossaryentry.

```
1509 \newcommand*{\glstrpostlongdescription}{\leavevmode\unskip\nopostdesc}
```

Provide a starred version of \newignoredglossary that doesn't add the glossary to the nohyperlist list.

ignoredglossary Redefine to check for star.

```
1510 \renewcommand{\newignoredglossary}{%
1511 \ifstar\glstr@s@newignoredglossary\glstr@org@newignoredglossary
1512 }
```

ignoredglossary The original definition is patched to check for existence.

```
1513 \newcommand*{\glstr@org@newignoredglossary}[1]{%
1514 \ifcsdef{glolist@#1}
1515 {%
1516 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1517 }%
1518 {%
1519 \ifdefempty\@ignored@glossaries
1520 {%
1521 \protected@edef\@ignored@glossaries{#1}%
1522 }%
1523 {%
1524 \protected@eappto\@ignored@glossaries{,#1}%
1525 }%
1526 \csgdef{glolist@#1}{,}%
1527 \ifcsundef{gls@#1@entryfmt}%
1528 {%
1529 \defglsentryfmt[#1]{\glsentryfmt}%
1530 }%
1531 }%
1532 \ifdefempty\@gls@nohyperlist
1533 {%
1534 \renewcommand*{\@gls@nohyperlist}{#1}%
1535 }%
1536 {%
1537 \protected@eappto\@gls@nohyperlist{,#1}%
1538 }%
1539 }%
1540 }
```

ignoredglossary Starred form.

```
1541 \newcommand*{\glstr@s@newignoredglossary}[1]{%
1542 \ifcsdef{glolist@#1}
1543 {%
1544 \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1545 }%
1546 {%
```

```

1547 \ifdefempty\@ignored@glossaries
1548 {%
1549 \protected@edef\@ignored@glossaries{#1}%
1550 }%
1551 {%
1552 \protected@eappto\@ignored@glossaries{,#1}%
1553 }%
1554 \csgdef{glolist@#1}{,}%
1555 \ifcsundef{gls@#1@entryfmt}%
1556 {%
1557 \defglsentryfmt[#1]{\glsentryfmt}%
1558 }%
1559 {}%
1560 }%
1561 }

```

`\glssettoctitle` Ignored glossaries don't have an associated title, so modify `\glssettoctitle` to check for it to prevent an undefined command written to the toc file.

```

1562 \glsifusetranslator
1563 {%
1564 \renewcommand*{\glssettoctitle}[1]{%
1565 \ifcsdef{gls@tr@set@#1@toctitle}%
1566 {%
1567 \csuse{gls@tr@set@#1@toctitle}%
1568 }%
1569 {%
1570 \ifcsdef{@glotype@#1@title}%
1571 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1572 {\def\glossarytoctitle{\glossarytitle}}%
1573 }%
1574 }%
1575 }
1576 {
1577 \renewcommand*{\glssettoctitle}[1]{%
1578 \ifcsdef{@glotype@#1@title}%
1579 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1580 {\def\glossarytoctitle{\glossarytitle}}%
1581 }
1582 }

```

`ignoredglossary` As above but won't do anything if the glossary already exists.

```

1583 \newcommand{\provideignoredglossary}{%
1584 \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1585 }

```

`ignoredglossary` Unstarred version.

```

1586 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1587 \ifcsdef{glolist@#1}
1588 {}%

```

```

1589 {%
1590   \ifdefempty\@ignored@glossaries
1591   {%
1592     \protected@edef\@ignored@glossaries{#1}%
1593   }%
1594   {%
1595     \protected@eappto\@ignored@glossaries{,#1}%
1596   }%
1597   \csgdef{glolist@#1}{,}%
1598   \ifcsundef{gls@#1@entryfmt}%
1599   {%
1600     \defglsentryfmt[#1]{\glsentryfmt}%
1601   }%
1602   {}%
1603   \ifdefempty\@gls@nohyperlist
1604   {%
1605     \renewcommand*{\@gls@nohyperlist}{#1}%
1606   }%
1607   {%
1608     \protected@eappto\@gls@nohyperlist{,#1}%
1609   }%
1610 }%
1611 }

```

`ignoredglossary` Starred form.

```

1612 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1613   \ifcsdef{glolist@#1}
1614   {}%
1615   {%
1616     \ifdefempty\@ignored@glossaries
1617     {%
1618       \protected@edef\@ignored@glossaries{#1}%
1619     }%
1620     {%
1621       \protected@eappto\@ignored@glossaries{,#1}%
1622     }%
1623     \csgdef{glolist@#1}{,}%
1624     \ifcsundef{gls@#1@entryfmt}%
1625     {%
1626       \defglsentryfmt[#1]{\glsentryfmt}%
1627     }%
1628     {}%
1629   }%
1630 }

```

`rcopytoglossary` Adds an entry label to another glossary list. First argument is entry label. Second argument is glossary label.

```

1631 \newcommand*{\glxtrcopytoglossary}[2]{%
1632   \glstoifexists{#1}%
1633   {%
1634     \ifcsdef{glolist@#2}
1635     {%
1636       \protected@csapppto{glolist@#2}{#1,}%
1637     }%
1638     {%
1639       \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1640     }%
1641   }%
1642 }

```

### 1.3.1 Existence Checks

`\glstoifexists` Modify `\glstoifexists` to take account of the undefaction setting.

```

1643 \renewcommand{\glstoifexists}[2]{%
1644   \ifglstryexists{#1}{#2}%
1645   {%

```

Define `\glslabel` in case it’s needed after this command (for example in the post-link hook).

```

1646   \protected@edef\glslabel{\glstetoklabel{#1}}%
1647   \glxtrundefaction{Glossary entry ‘\glslabel’
1648     has not been defined}{You need to define a glossary entry before
1649     you can reference it.}%
1650   }%
1651 }

```

`\glstoifnoexists` Modify `\glstoifnoexists` to take account of the undefaction setting.

```

1652 \renewcommand{\glstoifnoexists}[2]{%
1653   \ifglstryexists{#1}{%
1654     \glxtrundefaction{Glossary entry ‘\glstetoklabel{#1}’
1655       has already been defined}{}}{#2}%
1656 }

```

`\glstoifexistsordo` Modify `\glstoifexistsordo` to take account of the undefaction setting. This command was introduced in glossaries version 4.19, so check if it has been defined first.

```

1657 \ifdef\glstoifexistsordo
1658 {%
1659   \renewcommand{\glstoifexistsordo}[3]{%
1660     \ifglstryexists{#1}{#2}%
1661     {%
1662       \glxtrundefaction{Glossary entry ‘\glstetoklabel{#1}’
1663         has not been defined}{You need to define a glossary entry
1664         before you can use it.}%
1665       #3%
1666     }%

```

```

1667 }%
1668 }
1669 {%
1670 \glstr@warnonexistsordo\glsoifexistsordo
1671 \newcommand{\glsoifexistsordo}[3]{%
1672   \ifglstryexists{#1}{#2}%
1673   {%
1674     \glstrundefaction{Glossary entry '\glstoklabel{#1}'
1675       has not been defined}{You need to define a glossary entry
1676       before you can use it.}%
1677     #3%
1678   }%
1679 }%
1680 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1681 \ifdef\doifglossarynoexistsordo
1682 {%
1683 \renewcommand{\doifglossarynoexistsordo}[3]{%
1684   \ifglossaryexists*{#1}%
1685   {%
1686     \glstrundefaction{Glossary type '#1' already exists}{}%
1687     #3%
1688   }%
1689   {#2}%
1690 }%
1691 }
1692 {%
1693 \glstr@warnonexistsordo\doifglossarynoexistsordo
1694 \newcommand{\doifglossarynoexistsordo}[3]{%
1695   \ifglossaryexists*{#1}%
1696   {%
1697     \glstrundefaction{Glossary type '#1' already exists}{}%
1698     #3%
1699   }%
1700   {#2}%
1701 }%
1702 }
1703

```

There are now three types of cross-references: the see key (as original), the alias key (from glossaries-extra v1.12) and the seealso key (from glossaries-extra v1.16). The original see key needs to have a corresponding field (which it doesn't with the base glossaries package).

ryentryposthook Hook into end of \newglossaryentry to add "see" value as a field.

```

1704 \appto\newglossaryentryposthook{%
1705   \ifdefvoid\@glo@see
1706   {\csxdef{glo@\@glo@label @see}{}}%
1707   {%

```



```

1708 \csxdef{glo@\glo@label @see}{\@glo@see}%
1709 \if@glxtr@autoseeindex
1710 \@glxtr@autoindexcrossrefs
1711 \fi
1712 }%
1713 }
1714 \appto\@gls@keymap{,{see}{see}}

```

`\glxtrusesee` Apply `\glsseeformat` to the see key if not empty.

```

1715 \newcommand*{\glxtrusesee}[1]{%
1716 \glsdoifexists{#1}%
1717 {%
1718 \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1719 \ifdefempty\@glo@see
1720 {}%
1721 {%
1722 \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1723 }%
1724 }%
1725 }

```

`\glxtr@usesee`

```

1726 \newcommand*{\glxtr@usesee}[1][\seename]{%
1727 \@glxtr@usesee{#1}%
1728 }

```

`\@glxtr@usesee`

```

1729 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1730 \glxtruseseeformat{#1}{#2}%
1731 }

```

`xtruseseeformat` The format used by `\glxtrusesee`. The first argument is the tag (such as `\seename`). The second argument is the comma-separated list of cross-referenced labels.

```

1732 \newcommand*{\glxtruseseeformat}[2]{%
1733 \glsseeformat{#1}{#2}{}%
1734 }

```

`lsseeitemformat` glossaries originally defined `\glsseeitemformat` to use `\glsentryname` but in v3.0 this was switched to use `\glsentrytext` due to problems occurring with the name field being sanitized. Since this is no longer a problem, glossaries-extra restored the original definition as it makes more sense to use the name in the cross-reference list. Unfortunately this doesn't take style changes into account, so as from v1.42, this now uses `\glsfmttext` and `\glsfmtname` instead. (The text field is chosen rather than the short field to allow for the “noshort” styles.)

```

1735 \renewcommand*{\glsseeitemformat}[1]{%
1736 \ifglshashshort{#1}{\glsfmttext{#1}}{\glsfmtname{#1}}%
1737 }

```

`\glxtrhiername`

`\glxtrhiername{<label>}`

Displays the hierarchical name for the given entry. The cross-reference format `\glseeitemformat` may be redefined to use this command to show the hierarchy, if required. This now uses `\glsfmtext` and `\glsfmtname` instead of `\glaccessshort` and `\glaccessname` to allow for style formatting.

```
1738 \newcommand*{\glxtrhiername}[1]{%
1739   \glsoifexists{#1}%
1740   {%
1741     \glxtrifhasfield{parent}{#1}%
1742     {\glxtrhiername{\glscurrentfieldvalue}\glxtrhiernamesep}%
1743     }%
1744     \ifglshasshort{#1}{\glsfmtext{#1}}{\glsfmtname{#1}}%
1745   }%
1746 }
```

`\Glsxtrhiername`

`\Glsxtrhiername{<label>}`

As above but displays the top-level name with an initial capital.

```
1747 \newcommand*{\Glsxtrhiername}[1]{%
1748   \glsoifexists{#1}%
1749   {%
1750     \glxtrifhasfield{parent}{#1}%
1751     {%
1752       \Glsxtrhiername{\glscurrentfieldvalue}\glxtrhiernamesep
1753       \ifglshasshort{#1}{\glsfmtext{#1}}{\glsfmtname{#1}}%
1754     }%
1755     {\ifglshasshort{#1}{\Glsfmtext{#1}}{\Glsfmtname{#1}}}%
1756   }%
1757 }
```

`\GlsXtrhiername`

`\GlsXtrhiername{<label>}`

As above but converts the first letter of each name to a capital. (Note that this isn't applying title case, just capitalising the start of each hierarchical element.)

```
1758 \newcommand*{\GlsXtrhiername}[1]{%
1759   \glsoifexists{#1}%
1760   {%
1761     \glxtrifhasfield{parent}{#1}%
1762     {\GlsXtrhiername{\glscurrentfieldvalue}\glxtrhiernamesep}%
1763     {\ifglshasshort{#1}{\GlsXfmtext{#1}}{\GlsXfmtname{#1}}}%
1764   }%
1765 }
```

```

1762     {\GlsXtrhiername{\glscurrentfieldvalue}\glxtrhiernamesep}%
1763     {}}%
1764     \ifglshasshort{#1}{\Glsfmttext{#1}}{\Glsfmtname{#1}}%
1765 }%
1766 }

```

\GLSxtrhiername

`\GLSxtrhiername{<label>}`

As above but displays the top-level name in all-caps.

```

1767 \newcommand*{\GLSxtrhiername}[1]{%
1768   \glstoifexists{#1}%
1769   {%
1770     \glxtrifhasfield{parent}{#1}%
1771     {%
1772       \GLSxtrhiername{\glscurrentfieldvalue}\glxtrhiernamesep
1773       \ifglshasshort{#1}{\Glsfmttext{#1}}{\Glsfmtname{#1}}%
1774     }%
1775     {\ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}}%
1776   }%
1777 }

```

\GLSXTRhiername

`\GLSXTRhiername{<label>}`

As above but displays all names in all-caps.

```

1778 \newcommand*{\GLSXTRhiername}[1]{%
1779   \glstoifexists{#1}%
1780   {%
1781     \glxtrifhasfield{parent}{#1}%
1782     {\GLSXTRhiername{\glscurrentfieldvalue}\glxtrhiernamesep}%
1783     {}
1784     \ifglshasshort{#1}{\GLSfmttext{#1}}{\GLSfmtname{#1}}%
1785   }%
1786 }

```

glxtrhiernamesep Separator used in \glxtrhiername and variants.

```

1787 \newcommand*{\glxtrhiernamesep}{\,\{\small$\triangleright$}\,\,}

```

glxtruseseealso Apply \glseeformat to the seealso key if not empty. There's no optional tag to worry about here.

```

1788 \newcommand*{\glxtruseseealso}[1]{%
1789   \glstoifexists{#1}%
1790   {%

```

```

1791 \letcs{\@glo@see}{glo@glstdetoklabel{#1}@seealso}%
1792 \ifdefempty\@glo@see
1793 {}%
1794 {%
1795 \expandafter\glxtruseseealsoformat\expandafter{\@glo@see}%
1796 }%
1797 }%
1798 }

```

`\glxtrusealias` Apply `\glssseeformat` to the alias key if not empty. There's no optional tag to worry about here. The value also isn't a comma-separated list, but use the same interface.

```

1799 \newcommand*{\glxtrusealias}[1]{%
1800 \glstdoifexists{#1}%
1801 {%
1802 \letcs{\@glo@see}{glo@glstdetoklabel{#1}@alias}%
1803 \ifdefempty\@glo@see
1804 {}%
1805 {%

```

Expansion isn't necessary because the value is a single label not a list.

```

1806 \glxtruseseeformat{\seename}{\@glo@see}%
1807 }%
1808 }%
1809 }

```

`\glxtrusealsoformat` The format used by `\glxtruseseealso`. The argument is the comma-separated list of cross-referenced labels.

```

1810 \newcommand*{\glxtrusealsoformat}[1]{%
1811 \glssseeformat[\seealso]{#1}%
1812 }

```

`\glxtrseelist` Fully expands argument before passing to `\glsseelist`. (The argument to `\glsseelist` must be a comma-separated list of entry labels.)

```

1813 \newrobustcmd{\glxtrseelist}[1]{%
1814 \protected@edef\@glo@tmp{\noexpand\glsseelist{#1}}\@glo@tmp
1815 }

```

`\glsseelist` Redefine to make `\glsseelist` more flexible.

```

1816 \renewrobustcmd*{\glsseelist}[1]{%
1817 \let\@gls@dolast\relax
1818 \let\@gls@donext\relax
1819 \let\@glssseeitem\glssseefirstitem
1820 \let\@glssseelastsep\glssseelastsep
1821 \@for\@gls@thislabel:=#1\do{%
1822 \ifx\@xfor@nextelement\@nnil
1823 \@gls@dolast
1824 \else
1825 \@gls@donext
1826 \fi

```

```

1827 \expandafter\@glsseeitem\expandafter{\@gls@thislabel}%
1828 \let\@gls@dolast\@glsseelastsep
1829 \let\@gls@donext\glsseesep
1830 \let\@glsseeitem\glsseeitem
1831 \let\@glsseelastsep\glsseelastoxfordsep
1832 }%
1833 }

glsseefirstitem
1834 \newcommand*{\glsseefirstitem}{\glsseeitem}

eelastoxfordsep
1835 \newcommand*{\glsseelastoxfordsep}{\glsseelastsep}

\seealso In case this command hasn't been defined. Languages packages actually provide \alsoname
so use that if it's defined.
1836 \ifdef\alsoname
1837 {\providecommand{\seealso}{\alsoname}}
1838 {\providecommand{\seealso}{see also}}

xtrindexseealso If \@xdycrossrefhook is defined, provide a seealso crossref class. Otherwise this just does
\glssee with \seealso as the tag. The hook is only defined if both xindy and glossaries
v4.30+ are being used.
1839 \ifdef\@xdycrossrefhook
1840 {
    Add the cross-reference class definition to the hook.
1841 \appto\@xdycrossrefhook{%
1842 \write\glswrite{(define-crossref-class \string"seealso\string"
1843 :unverified )}%
1844 \write\glswrite{(markup-crossref-list
1845 :class \string"seealso\string"^^J\space\space\space
1846 :open \string"\string\glsxtrusealsoformat\glsopenbrace\string"
1847 :close \string"\glsclosebrace\string")}%
1848 }

    Append to class list.
1849 \appto\@xdylocationclassorder{\space\string"seealso\string"}

    This essentially works like \@do@seeglossary but uses the seealso class. This doesn't in-
    crement the associated counter.
1850 \newrobustcmd*{\glsxtrindexseealso}[2]{%
1851 \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
1852 \@glsxtr@recordsee{#1}{#2}%
1853 \fi
1854 \glsdoifexists{#1}%
1855 {%
1856 \@@glsxtrwrglossmark
1857 \def\@gls@xref{#2}%

```

```

1858 \onelevel@sanitize\@gls@xref
1859 \@gls@checkmkidxchars\@gls@xref
1860 \gls@glossary{\csname glo@#1@type\endcsname}{%
1861   (indexentry
1862     :tkey (\csname glo@#1@index\endcsname)
1863     :xref (\string"\@gls@xref\string")
1864     :attr \string"seealso\string"
1865   )
1866 }%
1867 }%
1868 }
1869 }
1870 {
  xindy not in use or glossaries version too old to support this.
1871 \newrobustcmd*{\glsxtrindexseealso}{\glssee[\seealsoname]}
1872 }

```

The alias key should be set to the label of the synonymous entry. The seealso key essentially behaves like `see=[\seealsoname]{\langle x r - l i s t \rangle}`. Neither of these new keys has the optional tag part allowed with `see`.

If `\gls@set@xr@key` has been defined (glossaries v4.30), use that, otherwise just use `\glsaddstoragekey`.

```

1873 \ifdef\gls@set@xr@key
1874 {

```

We have at least glossaries v4.30. This means the new keys can be governed by the same settings as the `see` key.

```

1875 \define@key{glossentry}{alias}{%
1876   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1877 }
1878 \define@key{glossentry}{seealso}{%
1879   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1880 }

```

Add to the key mappings.

```

1881 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1882 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1883 \appto\@newglossaryentryposthook{%
1884   \ifdefvoid\@glo@seealso
1885     {\csxdef{glo@\@glo@label @seealso}{}}%
1886     {%
1887       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1888       \if@glxtr@autoseeindex
1889         \@glxtr@autoindexcrossrefs
1890       \fi
1891     }%

```

The alias field doesn't trigger the automatic cross-reference indexing performed at the end of the document.

```

1892 \ifdefvoid\@glo@alias
1893 {\csxdef{glo@\@glo@label @alias}{}}%
1894 {%
1895 \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1896 }%
1897 }

```

Provide user-level commands to access the values.

`\glxtralias`

```

1898 \newcommand*\glxtralias[1]{\@gls@entry@field{#1}{alias}}

```

`trseealsolabels`

```

1899 \newcommand*\glxtrseealsolabels[1]{\@gls@entry@field{#1}{seealso}}

```

Add to the `\@glo@autosee` hook.

```

1900 \appto\@glo@autoseehook{%
1901 \ifdefvoid\@glo@alias
1902 {%
1903 \ifdefvoid\@glo@seealso
1904 {}%
1905 {%
1906 \protected@edef\@do@glsssee{\noexpand\glxtrindexseealso
1907 {\@glo@label}{\@glo@seealso}}%
1908 \@do@glsssee
1909 }%
1910 }%
1911 {%

```

Add cross-reference if see key hasn't been used.

```

1912 \ifdefvoid\@glo@see
1913 {%
1914 \protected@edef\@do@glsssee{\noexpand\glsssee{\@glo@label}{\@glo@alias}}%
1915 \@do@glsssee
1916 }%
1917 {}%
1918 }%
1919 }%
1920 }
1921 {

```

We have an older version of glossaries, so just use `\glsaddstoragekey`.

`\glxtralias`

```

1922 \glsaddstoragekey*{alias}{}{\glxtralias}

```

`trseealsolabels`

```

1923 \glsaddstoragekey*{seealso}{}{\glxtrseealsolabels}

```

If `\gls@set@xr@key` isn't defined, then `\@glo@autosee` won't be either, so use the post entry definition hook.

`ryentryposthook` Append to the hook to check for the alias and seealso keys.

```

1924 \appto\@newglossaryentryposthook{%
1925   \ifcsvoid{glo@\@glo@label @alias}%
1926   {%
1927     \ifcsvoid{glo@\@glo@label @seealso}%
1928     {}%
1929     {%
1930       \protected@edef\@do@glsssee{\noexpand\glstrindexseealso
1931         {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1932       \@do@glsssee
1933     }%
1934   }%
1935   {%

```

Add cross-reference if see key hasn't been used.

```

1936   \ifdefvoid\@glo@see
1937   {%
1938     \protected@edef\@do@glsssee{\noexpand\glsssee
1939       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1940     \@do@glsssee
1941   }%
1942   {}%
1943 }%
1944 }
1945 }

```

Add all unused cross-references at the end of the document.

```

1946 \AtEndDocument{\if@glstrindexcrossrefs\glstraddallcrossrefs\fi}

```

`addallcrossrefs` Iterate through all used entries and if they have a cross-reference, make sure the cross-reference has been added.

```

1947 \newcommand*\@glstraddallcrossrefs{%
1948   \forallglossaries{\@glo@type}%
1949   {%
1950     \forglssentries[\@glo@type]{\@glo@label}%
1951     {%
1952       \ifglssused{\@glo@label}%
1953       {\expandafter\@glstr@addunusedxrefs\expandafter{\@glo@label}}{}%
1954     }%
1955   }%
1956 }

```

`@addunusedxrefs` If the given entry has a see or seealso field add all unused cross-references. (The alias field isn't checked.)



```

1957 \newcommand*{\@glsxtr@addunusedxrefs}[1]{%
1958   \letcs{\@glo@see}{glo@glstdetoklabel{#1}@see}%
1959   \ifdefvoid\@glo@see
1960   {%
1961   {%
1962     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
1963   }%
1964   \letcs{\@glo@see}{glo@glstdetoklabel{#1}@seealso}%
1965   \ifdefvoid\@glo@see
1966   {%
1967   {%
1968     \expandafter\glsxtr@addunused\@glo@see\@end@glsxtr@addunused
1969   }%
1970 }

```

`\glsxtr@addunused` Adds all the entries if they haven't been used.

```

1971 \newcommand*{\glsxtr@addunused}[1][]{%
1972   \@glsxtr@addunused
1973 }

```

`\glsxtr@addunused` Adds all the entries if they haven't been used.

```

1974 \def\@glsxtr@addunused#1\@end@glsxtr@addunused{%
1975   \@for\@glsxtr@label:=#1\do
1976   {%
1977     \ifglsused{\@glsxtr@label}{}%
1978     {%
1979       \glsadd[format=glsxtrunusedformat]{\@glsxtr@label}%
1980       \glsunset{\@glsxtr@label}%
1981       \expandafter\@glsxtr@addunusedxrefs\expandafter{\@glsxtr@label}%
1982     }%
1983   }%
1984 }

```

`xtrunusedformat`

```

1985 \newcommand*{\glsxtrunusedformat}[1]{\unskip}

```

### 1.3.2 Document Definitions

`\gls@begindocdefs` This command was only introduced to glossaries v4.37, so it may not be defined. If it has been defined, redefine it to check `\@glsxtr@docdefval` so that it only inputs the `.glsdefs` file if `docdef=true`.

```

1986 \ifdef\gls@begindocdefs
1987 {%
1988   \renewcommand*{\gls@begindocdefs}{%
1989     \ifnum\@glsxtr@docdefval=1\relax
1990       \gls@enablesavenonumberlist
1991       \edef\@gls@restoreat{%
1992         \noexpand\catcode'\noexpand\@=\number\catcode'\@ \relax}%

```

```

1993      \makeatletter
1994      \InputIfFileExists{\jobname.glsdefs}{\relax}{\relax}%
1995      \@gls@restoreat
1996      \undef\@gls@restoreat
1997      \gls@defdocnewglossaryentry
1998      \else
1999      \ifnum\@glsxtr@docdefval=3\relax

```

The docdef=atom package option has been set. Create the .glsdefs file for the autocomplete support but don't read it.

```

2000      \@gls@enablesavenonumberlist
2001      \let\gls@checkseeallowed\relax
2002      \let\newglossaryentry\new@atom@glossaryentry
2003      \global\newwrite\@gls@deffile
2004      \immediate\openout\@gls@deffile=\jobname.glsdefs

```

Write all currently defined entries.

```

2005      \forallglsentries{\@glsentry}{\@gls@writedef{\@glsentry}}%
2006      \fi
2007      \fi
2008  }
2009 }
2010 {%
2011  \ifnum\@glsxtr@docdefval=3\relax
2012    \PackageError{glossaries-extra}{Package option
2013      'docdef=\@glsxtr@docdefsetting' requires at least version 4.37
2014      of the base glossaries.sty package}{}
2015  \fi
2016 }

```

m@glossaryentry

```

2017 \newrobustcmd{\new@atom@glossaryentry}[2]{%
2018   \gls@defglossaryentry{#1}{#2}%
2019   \@gls@writedef{#1}%
2020 }

```

noidxglossaries Modify \makenoidxglossaries so that it automatically sets docdef=false (unless the restricted setting is on) and disables the docdef key. This command isn't allowed with the record option.

```

2021 \let\glsxtr@orgmakenoidxglossaries\makenoidxglossaries
2022 \renewcommand{\makenoidxglossaries}{%
2023   \@domakeglossaries
2024   {%
2025     \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
2026     {%
2027       \glsxtr@orgmakenoidxglossaries

```

Add marker to \@do@seeglossary but don't increment associated counter.

```

2028   \renewcommand{\@do@seeglossary}[2]{%
2029     \@glsxtrwrglossmark

```

```

2030     \protected@edef\@gls@label{\glsdetoklabel{##1}}%
2031     \protected@write\@auxout{}\{%
2032         \string\@gls@reference
2033         {\csname glo@\@gls@label @type\endcsname}%
2034         {\@gls@label}%
2035         {%
2036             \string\glsseeformat##2}%
2037         }%
2038     }%
2039 }%

```

Check for docdefs=restricted:

```

2040     \ifglsxtrdocdefrestricted

```

If restricted document definitions allowed, adjust \@gls@reference so that it doesn't test for existence.

```

2041     \renewcommand*\@gls@reference}[3]{%
2042         \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
2043         \ifinlistcs{##2}{\glsref@##1}%
2044         {}%
2045         {\listcsgadd{\glsref@##1}{##2}}%
2046         \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
2047         {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
2048         {}%
2049         \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
2050     }%
2051     \else

```

Disable document definitions.

```

2052     \@glsxtrdocdeffalse
2053     \fi
2054     \disable@keys{glossaries-extra}{docdef}%
2055 }%
2056 {%
2057     \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
2058         not permitted\MessageBreak
2059         with record=\@glsxtr@record@setting\space package option}%
2060     {You may only use \string\makenoidxglossaries\space with the
2061         record=off option}%
2062 }%
2063 }%
2064 }

```

`\newglossaryentry` Modify \@gls@defdocnewglossaryentry so that it checks the docdef value.

```

2065 \renewcommand*\@gls@defdocnewglossaryentry{%
2066     \ifcase\@glsxtr@docdefval
2067         docdef=false:
2068     \renewcommand*\@newglossaryentry}[2]{%
2069         \PackageError{glossaries-extra}{Glossary entries must

```

```

2069     be \MessageBreak defined in the preamble with \MessageBreak
2070     package option 'docdef=false'\MessageBreak(consider using
2071     'docdef=restricted')}{Move your glossary definitions to
2072     the preamble. You can also put them in a \MessageBreak separate file
2073     and load them with \string\loadglsentries.}%
2074 }%
2075 \or

```

(docdef=true case.) Since the see value is now saved in a field, it can be used by entries that have been defined in the document.

```

2076     \let\gls@checkseeallowed\relax
2077     \let\newglossaryentry\new@glossaryentry
2078 \else

```

Restricted mode just needs to allow the see value.

```

2079     \let\gls@checkseeallowed\relax
2080 \fi
2081 }%

```

Permit a special form of document definition, but only allow it if the glossaries come at the end of the document. These commands behave a little like a combination of \newterm and \gls. This must be explicitly enabled with the following.

rEnableOnTheFly

```

2082 \newcommand*{\GlsXtrEnableOnTheFly}{%
2083   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
2084 }

```

rEnableOnTheFly

The starred version attempts to allow UTF8 characters in the label, but this may break! (Formatting commands mustn't be used in the label, but the label may be a command whose replacement text is the actual label. This doesn't take into account a command that's defined in terms of another command that may eventually expand to the label text.)

```

2085 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
2086   \renewcommand*{\glsdetoklabel}[1]{%
2087     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
2088     {%
2089       \expandafter\detokenize\expandafter{##1}%
2090     }%
2091     {\detokenize{##1}}}%
2092   }%
2093   \@GlsXtrEnableOnTheFly
2094 }
2095 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
2096   \expandafter\if\glsbackslash#1%
2097     #3%
2098   \else
2099     #4%
2100 \fi
2101 }

```

sxtrstarflywarn

```
2102 \newcommand*{\glxtrstarflywarn}{%
2103   \GlossariesExtraWarning{Experimental starred version of
2104   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
2105   read the warnings in the glossaries-extra user manual)}}%
2106 }
```

rEnableOnTheFly

```
2107 \newcommand*{\@GlsXtrEnableOnTheFly}{%
```

Don't redefine `\glstdetoklabel` if LuaTeX or XeTeX is being used, since it's mainly to allow accented characters in the label.

These definitions are all assigned the category given by:

`\glxtrcat`

```
2108   \newcommand*{\glxtrcat}{general}
```

`\glxtr`

```
2109   \newcommand*{\glxtr}[1] [] {%
2110     \def\glxtr@keylist{##1}%
2111     \@glxtr
2112   }
```

`\@glxtr`

```
2113   \newcommand*{\@glxtr}[2] [] {%
2114     \ifglstryexists{##2}%
2115     {%
2116       \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2117     }%
2118     {%
2119       \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
2120       description={\nopostdesc},##1}%
2121     }%
2122     \expandafter\gls\expandafter[\glxtr@keylist]{##2}%
2123   }
```

`\Glsxtr`

```
2124   \newcommand*{\Glsxtr}[1] [] {%
2125     \def\glxtr@keylist{##1}%
2126     \@Glsxtr
2127   }
```

`\@Glsxtr`

```
2128   \newcommand*{\@Glsxtr}[2] [] {%
2129     \ifglstryexists{##2}%
2130     {%
2131       \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
2132     }%
```

```

2133   {%
2134   \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
2135   description={\nopostdesc},##1}%
2136   }%
2137   \expandafter\Gls\expandafter[\glsxtr@keylist]{##2}%
2138   }

```

\glsxtrpl

```

2139 \newcommand*\glsxtrpl[1][]{%
2140 \def\glsxtr@keylist{##1}%
2141 \@glsxtrpl
2142 }

```

\@glsxtrpl

```

2143 \newcommand*\@glsxtrpl[2][]{%
2144 \ifglsentryexists{##2}%
2145 {%
2146 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
2147 }%
2148 {%
2149 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
2150 description={\nopostdesc},##1}%
2151 }%
2152 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
2153 }

```

\Glsxtrpl

```

2154 \newcommand*\Glsxtrpl[1][]{%
2155 \def\glsxtr@keylist{##1}%
2156 \@Glsxtrpl
2157 }

```

\@Glsxtrpl

```

2158 \newcommand*\@Glsxtrpl[2][]{%
2159 \ifglsentryexists{##2}
2160 {%
2161 \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
2162 }%
2163 {%
2164 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
2165 description={\nopostdesc},##1}%
2166 }%
2167 \expandafter\Glspl\expandafter[\glsxtr@keylist]{##2}%
2168 }

```

\GlsXtrWarning

```

2169 \newcommand*\GlsXtrWarning[2]{%
2170 \def\@glsxtr@optlist{##1}%
2171 \@onelevel@sanitize\@glsxtr@optlist

```

```

2172 \GlossariesExtraWarning{The options ‘\@glxtr@optlist’ have
2173 been ignored for entry ‘##2’ as it has already been defined}%
2174 }

```

Disable commands after the glossary:

```

2175 \renewcommand\@printglossary[2]{%
2176 \def\@glxtr@printglossopts{##1}%
2177 \@glxtr@orgprintglossary{##1}{##2}%
2178 \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
2179 \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
2180 \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
2181 \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
2182 }

```

abledflycommand

```

2183 \newcommand*{\@glxtr@disabledflycommand}[1]{%
2184 \PackageError{glossaries-extra}%
2185 {\string##1\space can’t be used after any of the \MessageBreak
2186 glossaries have been displayed}%
2187 {The on-the-fly commands enabled by
2188 \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
2189 before the glossaries. If you want to use any entries \MessageBreak
2190 after any of the glossaries, you must use the standard \MessageBreak
2191 method of first defining the entry and then using the \MessageBreak
2192 entry with commands like \string\glx}%
2193 \@glxtr@disabledflycommand
2194 }%
2195 \newcommand*{\@glxtr@disabledflycommand}[2][{}]{##2}

```

End of \GlsXtrEnableOnTheFly. Disable since it can only be used once.

```

2196 \let\GlsXtrEnableOnTheFly\relax
2197 }
2198 \@onlypreamble\GlsXtrEnableOnTheFly

```

### 1.3.3 Existing Glossary Style Modifications

Modify \setglossarystyle to keep track of the current style. This allows the \glossaries-extra-stylemods package to reset the current style after the required modifications have been made.

r@current@style Initialise the current style to the default style.

```

2199 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

Modify \setglossarystyle to set \@glxtr@current@style.

etglossarystyle

```

2200 \renewcommand*{\setglossarystyle}[1]{%
2201 \ifcsundef{\@glssstyle@#1}%
2202 {%
2203 \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%

```

```

2204 }%
2205 {%
2206   \csname @glsstyle@#1\endcsname
    Only set the current style if it exists.
2207   \protected@edef\@glstr@current@style{#1}%
2208 }%
2209 \ifx\@glossary@default@style\relax
2210   \protected@edef\@glossary@default@style{#1}%
2211 \fi
2212 }

```

In case we have an old version of glossaries:

```

2213 \ifdef\@glossary@default@style
2214 {}
2215 {%
2216   \let\@glossary@default@style\relax
2217 }

```

`\listdottedwidth` If `\glslistdottedwidth` has been defined and is currently equal to `.5\hsize` then make the modification suggested in [bug report #92](#)

```

2218 \ifdef\glslistdottedwidth
2219 {%
2220   \ifdim\glslistdottedwidth=.5\hsize
2221     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
2222     \AtBeginDocument{%
2223       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
2224         \setlength{\glslistdottedwidth}{.5\columnwidth}%
2225       \fi
2226     }%
2227 \fi
2228 }
2229 {}%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

2230 \ifdef\glsdescwidth
2231 {%
2232   \ifdim\glsdescwidth=.6\hsize
2233     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
2234     \AtBeginDocument{%
2235       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
2236         \setlength{\glsdescwidth}{.6\columnwidth}%
2237       \fi
2238     }%
2239 \fi
2240 }
2241 {}%

```



and for \glspagelistwidth:

lspagelistwidth

```
2242 \ifdef\glspagelistwidth
2243 {%
2244   \ifdim\glspagelistwidth=.1\hsize
2245     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
2246     \AtBeginDocument{%
2247       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
2248         \setlength{\glspagelistwidth}{.1\columnwidth}%
2249       \fi
2250     }%
2251   \fi
2252 }
2253 {}%
```

aryentrynumbers Has the nonumberlist option been used?

```
2254 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
2255 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
2256   \glsnonumberlistfalse
2257   \renewcommand*{\glossaryentrynumbers}[1]{%
2258     \ifglentryexists{\glscurrententrylabel}%
2259     {%
2260       \@glsxtrpreloctag
2261       \GlsXtrFormatLocationList{#1}%
2262       \@glsxtrpostloctag
2263       \gls@save@numberlist{#1}%
2264     }{}%
2265   }%
2266 \else
2267   \glsnonumberlisttrue
2268   \renewcommand*{\glossaryentrynumbers}[1]{%
2269     \ifglentryexists{\glscurrententrylabel}%
2270     {%
2271       \gls@save@numberlist{#1}%
2272     }{}%
2273   }%
2274 \fi
```

matLocationList Provide an easy interface to change the format of the location list without removing the save number list stuff.

```
2275 \newcommand*{\GlsXtrFormatLocationList}[1]{#1}
```

Sometimes users want to prefix the location list with “page”/“pages”. The simplest way to determine if the location list consists of a single location is to check for instances of `\delimN` or `\delimR`, but this isn’t so easy to do as they might be embedded inside the argument of formatting commands. With a bit of trickery we can find out by adjusting `\delimN` and `\delimR` to set a flag and then save information to the auxiliary file for the next run.

ePreLocationTag

```
2276 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
2277   \let\@glxtrpreloctag\@glxtrpreloctag
2278   \let\@glxtrpostloctag\@glxtrpostloctag
2279   \renewcommand*{\@glxtr@pagetag}{#1}%
2280   \renewcommand*{\@glxtr@pagetag}{#2}%
2281   \renewcommand*{\@glxtr@savepreloctag}[2]{%
2282     \csgdef{\@glxtr@preloctag##1}{##2}%
2283   }%
2284   \renewcommand*{\@glxtr@doloctag}{%
2285     \ifcsundef{\@glxtr@preloctag\@glscurrententrylabel}%
2286     {%
2287       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
2288         Rerun required}%
2289     }%
2290     {%
2291       \csuse{\@glxtr@preloctag\@glscurrententrylabel}%
2292     }%
2293   }%
2294 }
2295 \@onlypreamble\GlsXtrEnablePreLocationTag
```

glxtrpreloctag

```
2296 \newcommand*{\@glxtrpreloctag}{%
2297   \let\@glxtr@org@delimN\delimN
2298   \let\@glxtr@org@delimR\delimR
2299   \let\@glxtr@org@glsgignore\glsgignore
2300   \gdef is required as the delimiters may occur inside a scope.
2301   \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2302   \renewcommand*{\delimN}{%
2303     \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2304     \@glxtr@org@delimN}%
2305   \renewcommand*{\delimR}{%
2306     \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
2307     \@glxtr@org@delimR}%
2308   \renewcommand*{\glsgignore}[1]{%
2309     \gdef\@glxtr@thisloctag{\relax}%
2310     \@glxtr@org@glsgignore{##1}}%
2311   \@glxtr@doloctag
2311 }
```

glxtrpreloctag

```
2312 \newcommand*{\@glxtrpreloctag}{}
```

@glxtr@pagetag

```
2313 \newcommand*{\@glxtr@pagetag}{}%
```

glxtr@pagetag

```
2314 \newcommand*{\@glxtr@pagetag}{}%
```

lsxtrpostloctag

```

2315 \newcommand*{\@@glxtrpostloctag}{%
2316   \let\delimN\@glxtr@org@delimN
2317   \let\delimR\@glxtr@org@delimR
2318   \let\glignore\@glxtr@org@glignore
2319   \protected@write\@auxout{}{%
2320     {\string\@glxtr@savepreloctag{\glscurrententrylabel}{\@glxtr@thisloctag}}%
2321 }

```

lsxtrpostloctag

```

2322 \newcommand*{\@glxtrpostloctag}{%

```

lsxtr@preloctag

```

2323 \newcommand*{\@glxtr@savepreloctag}[2]{%
2324 \protected@write\@auxout{}{%
2325   \string\providecommand\string\@glxtr@savepreloctag[2]{%

```

glxtr@doloctag

```

2326 \newcommand*{\@glxtr@doloctag}{%

```

ss@nonumberlist Modify the nonumberlist key to use \GlsXtrFormatLocationList (and also save the number list):

```

2327 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
2328   \XKV@plfalse
2329   \XKV@sttrue
2330   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
2331   {%
2332     \csname glsnonumberlist\XKV@resa\endcsname
2333     \ifglsnonumberlist
2334       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
2335     \else
2336       \def\glossaryentrynumbers##1{%
2337         \@glxtrpreloctag
2338         \GlsXtrFormatLocationList{##1}%
2339         \@glxtrpostloctag
2340         \gls@save@numberlist{##1}}%
2341     \fi
2342   }%
2343 }

```

### 1.3.4 Entry Formatting, Hyperlinks and Indexing

`\glsentryfmt` Change default entry format. Use the generic format for regular terms (that is, entries that have a category with the regular attribute set) or non-regular terms without a short value and use the abbreviation format for non-regular terms that have a short value. If further attributes need to be checked, then `\glsentryfmt` will need redefining as appropriate (or use `\defglsentryfmt`). The abbreviation format is set here for entries that have a short form, even if they are regular entries to ensure the abbreviation fonts are correct.

```

2344 \renewcommand*{\glentryfmt}{%
2345   \ifglshasshort{\glslabel}{\glsetabbrvfmt{\glscategory{\glslabel}}}{}%
2346   \glrifregular{\glslabel}%
2347   {\glxtrregularfont{\glsgenentryfmt}}}%
2348   {%
2349     \ifglshasshort{\glslabel}%
2350     {\glxtrabbreviationfont{\glxtrgenabbrvfmt}}}%
2351     {\glxtrregularfont{\glsgenentryfmt}}}%
2352   }%
2353 }

```

**glxtrregularfont** Font used for regular entries.

```
2354 \newcommand*{\glxtrregularfont}[1]{#1}
```

**glxtrabbreviationfont** Font used for abbreviation entries.

```
2355 \newcommand*{\glxtrabbreviationfont}[1]{#1}
```

Commands like `\glrifplural` are only used by the `\gls`-like commands in the glossaries package, but it might be useful for the postlink hook to know if the user has used, say, `\glfirst` or `\glsplural`. This can provide better consistency with the formatting of the `\gls`-like commands, even though they don't use `\glentryfmt`.

**glfield@link** Redefine `\@glfield@link` so that commands like `\glfirst` can setup `\glxtrifwasfirstuse` etc to allow the postlink hook to work better. This now has an optional argument that sets up the defaults.

```
2356 \renewcommand{\@glfield@link}[4][]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

2357   \@glxtr@record{#2}{#3}{\glslink}%
2358   \glsoifexists{#3}%
2359   {%

```

Save and restore the hyper setting (`\@gl@link` also does this, but that's too late if the optional argument of `\@glfield@link` modifies it).

```

2360     \let\glxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
2361     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2362     \def\glscustomtext{#4}%
2363     \@glxtr@field@linkdefs
2364     #1%
2365     \@gl@link[#2]{#3}{#4}%
2366     \let\ifKV@glslink@hyper\glxtrorg@ifKV@glslink@hyper
2367   }%
2368   \glspostlinkhook
2369 }

```

The commands `\gls`, `\Gls` etc don't use `\@glfield@link`, so they need modifying as well to use `\@glxtr@record`.

\@gls@ Save the original definition and redefine.

```
2370 \let\@glsxtr@org@gls@\@gls@
2371 \def\@gls@#1#2{%
2372   \@glsxtr@record{#1}{#2}{glslink}%
2373   \@glsxtr@org@gls@{#1}{#2}%
2374 }%
```

\@glspl@ Save the original definition and redefine.

```
2375 \let\@glsxtr@org@glspl@\@glspl@
2376 \def\@glspl@#1#2{%
2377   \@glsxtr@record{#1}{#2}{glslink}%
2378   \@glsxtr@org@glspl@{#1}{#2}%
2379 }%
```

\@Gls@ Save the original definition and redefine.

```
2380 \let\@glsxtr@org@Gls@\@Gls@
2381 \def\@Gls@#1#2{%
2382   \@glsxtr@record{#1}{#2}{glslink}%
2383   \@glsxtr@org@Gls@{#1}{#2}%
2384 }%
```

\@Glspl@ Save the original definition and redefine.

```
2385 \let\@glsxtr@org@Glspl@\@Glspl@
2386 \def\@Glspl@#1#2{%
2387   \@glsxtr@record{#1}{#2}{glslink}%
2388   \@glsxtr@org@Glspl@{#1}{#2}%
2389 }%
```

\@GLS@ Save the original definition and redefine.

```
2390 \let\@glsxtr@org@GLS@\@GLS@
2391 \def\@GLS@#1#2{%
2392   \@glsxtr@record{#1}{#2}{glslink}%
2393   \@glsxtr@org@GLS@{#1}{#2}%
2394 }%
```

\@GLSpl@ Save the original definition and redefine.

```
2395 \let\@glsxtr@org@GLSpl@\@GLSpl@
2396 \def\@GLSpl@#1#2{%
2397   \@glsxtr@record{#1}{#2}{glslink}%
2398   \@glsxtr@org@GLSpl@{#1}{#2}%
2399 }%
```

\@glsdisp This is redefined to allow the recording on the first run. Can't save and restore \@glsdisp since it has an optional argument.

```
2400 \renewcommand*{\@glsdisp}[3][[]]{%
2401   \@glsxtr@record{#1}{#2}{glslink}%
2402   \glsdoifexists{#2}{%
2403     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
```

```

2404 \let\glsifplural\@secondoftwo
2405 \let\glsifscapscase\@firstofthree
2406 \def\glscustomtext{#3}%
2407 \def\glsinsert{}%
2408 \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
2409 \@gls@link[#1]{#2}{\@glo@text}%
2410 \ifKV@glslink@local
2411 \glslocalunset{#2}%
2412 \else
2413 \glsunset{#2}%
2414 \fi
2415 }%
2416 \glspostlinkhook
2417 }

```

`\@gls@@link@` Redefine to include `\@glsxtr@record`

```

2418 \renewcommand*{\@gls@@link}[3][{}]{%
2419 \@glsxtr@record{#1}{#2}{\glslink}%
2420 \glsdoifexistsordo{#2}%
2421 {%
2422 \let\do@gls@link@checkfirsthyper\relax
2423
2424 Post-link hook commands need initialising.
2425 \def\glscustomtext{#3}%
2426 \@glsxtr@field@linkdefs
2427 \@gls@link[#1]{#2}{#3}%
2428 }%
2429 {%
2430 \glsxtr@field@linkdefs
2431 \glsxtr@link@checkfirsthyper
2432 }%
2433 }%
2434 }%
2435 }%
2436 }%
2437 }%
2438 }%
2439 }%
2440 }

```

`\glsxtrinitwrgloss` Set the default if the `wrgloss` is omitted.

```

2432 \newcommand*{\glsxtrinitwrgloss}{%
2433 \glsifattribute{\glslabel}{wrgloss}{after}%
2434 {%
2435 \glsxtrinitwrglossbeforefalse
2436 }%
2437 {%
2438 \glsxtrinitwrglossbeforetrue
2439 }%
2440 }

```

`\glsxtrwrglossbefore` Conditional to determine if the indexing should be done before the link text.

```

2441 \newif\ifglsxtrinitwrglossbefore
2442 \glsxtrinitwrglossbeforetrue

```

Define a `wrgloss` key to determine whether to write the glossary information before or after the link text.

```

2443 \define@choicekey{glslink}{wrgloss}%
2444 [\@glsxtr@wrglossval\@glsxtr@wrglossnr]%
2445 {before,after}%
2446 {%
2447   \ifcase\@glsxtr@wrglossnr\relax
2448     \glsxtrinitwrglossbeforetrue
2449   \or
2450     \glsxtrinitwrglossbeforefalse
2451   \fi
2452 }

2453 \define@key{glslink}{thevalue}{\def\@glsxtr@thevalue{#1}}

2454 \define@key{glslink}{theHvalue}{\def\@glsxtr@theHvalue{#1}}

```

`\tr@hyperoutside` Define a hyperoutside key to determine whether `\hyperlink` should be outside `\glstextformat`.

```

2455 \define@boolkey{glslink}[glsxtr@]{hyperoutside}[true]{}
2456 \glsxtr@hyperoutsidettrue

```

`\ocal@textformat` Provide a key to locally change the text format.

```

2457 \define@key{glslink}{textformat}{%
2458   \ifcsdef{#1}
2459   {%
2460     \letcs{\@glsxtr@local@textformat}{#1}%
2461   }%
2462   {%
2463     \PackageError{glossaries-extra}{Unknown control sequence name ‘#1’}{}%
2464   }%
2465 }

2466 \define@key{glslink}{prefix}{\def\glolinkprefix{#1}}

```

`\nithyperoutside` Set the default if the hyperoutside is omitted.

```

2467 \newcommand*{\glsxtrinithyperoutside}{%
2468   \glsifattribute{\glslabel}{hyperoutside}{false}%
2469   {%
2470     \glsxtr@hyperoutsidedefalse
2471   }%
2472   {%
2473     \glsxtr@hyperoutsidettrue
2474   }%
2475 }

```

`\r@inc@linkcount` Does nothing by default.

```

2476 \newcommand*{\glsxtr@inc@linkcount}{}

```

`\slinkpresetkeys` User hook performed immediately before options are set. Does nothing by default.

```

2477 \newcommand*{\glslinkpresetkeys}{}

```

`\GlsXtrExpandedFmt` Helper command that (protected) fully expands second argument and then applies it to the first, which must be a command that takes a single argument.

```
2478 \newrobustcmd*{\GlsXtrExpandedFmt}[2]{%
2479   \protected@edef\@glstr@tmp{#2}%
2480   \expandafter#1\expandafter{\@glstr@tmp}%
2481 }
```

`\@glstr@use@equation@counter` If in a numbered equation, change the counter to equation. This can be overridden by explicitly setting the counter in the optional argument of commands like `\gls` and `\glslink`.

```
2482 \newcommand*{\@glstr@use@equation@counter}{%
2483   \@glstr@ifnum@mmode{\def\@glstr@counter{equation}}{}%
2484 }
```

`\@glstr@do@autoadd` If `\GlsXtrAutoAddOnFormat` is used, this will automatically use `\glsadd`. It's therefore only used with `\@glstr@link` not with `\glsadd` otherwise it could trigger an infinite loop. The argument indicates the key family (`glslink` or `glossadd`).

```
2485 \newcommand*{\@glstr@do@autoadd}[1]{}
```

`\GlsXtrAutoAddOnFormat`

$$\backslash\text{GlsXtrAutoAddOnFormat}[\langle label \rangle][\langle format\ list \rangle][\langle glsadd\ options \rangle]$$

If an entry is indexed with the format set to one identified in the comma-separated list, then automatically index it using `\glsadd` with the given options, which may override the current options. Scoping is needed to prevent leakage.

```
2486 \newcommand*{\GlsXtrAutoAddOnFormat}[3][\glslabel]{%
2487   \renewcommand*{\@glstr@do@autoadd}[1]{%
2488     \begingroup
2489       \protected@edef\@glstr@do@autoadd{%
2490         \noexpand\ifstrequal{##1}{glslink}{%
2491           {%
2492             \noexpand\DTLifinlist{\@glstr@numberformat}{#2}{\noexpand\glsadd[format={\@glstr@numberfor
2493           }}%
2494         }}%
2495       }%
2496       \@glstr@do@autoadd
2497     \endgroup
2498   }%
2499 }
```

`\@glstr@link` Redefine to allow the indexing to be placed after the link text. By default this is done before the link text to prevent problems that can occur from the `\whatsit`, but there may be times when the user would like the indexing done afterwards even though it causes a `\whatsit`.

```
2500 \def\@glstr@link[#1]#2#3{%
2501   \leavevmode
```



```

2502 \protected@edef\glslabel{\glsdetoklabel{#2}}%
2503 \def\@gls@link@opts{#1}%
2504 \let\@gls@link@label\glslabel
2505 \let\@gls@numberformat\@glsxtr@defaultnumberformat
2506 \protected@edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
2507 \protected@edef\gls@type{\csname glo@\glslabel @type\endcsname}%
2508 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

  Save current value of \glslinkprefix:
2509 \let\@glsxtr@org@glslinkprefix\glslinkprefix

  Initialise \@glsxtr@local@textformat
2510 \let\@glsxtr@local@textformat\relax

  Initialise thevalue and theHvalue (v1.19).
2511 \def\@glsxtr@thevalue{}%
2512 \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%

  Initialise when indexing should occur (new to v1.14).
2513 \glsxtrinitwrgloss

  Initialise whether \hyperlink should be outside \gls@textformat (new to v1.21).
2514 \glsxtrinithyperoutside

  Note that the default link options may override \glsxtrinitwrgloss.
2515 \@gls@setdefault@glslink@opts

  Increment link counter if enabled (new to v1.26).
2516 \glsxtr@inc@linkcount

  Check if the equations option has been set (new to v1.37).
2517 \if@glsxtr@equations
2518   \@glsxtr@use@equation@counter
2519 \fi

  As the original definition.
2520 \do@gls@disablehyperinlist
2521 \do@gls@link@checkfirsthyper

  User hook before options are set (new to v1.26):
2522 \glslinkpresetkeys

  Set options.
2523 \setkeys{glslink}{#1}%

  Perform auto add if set (new to v1.37)
2524 \glsxtr@do@autoadd{glslink}%

  User hook after options are set:
2525 \glslinkpostsetkeys

  Check thevalue and theHvalue before saving (v1.19).
2526 \ifdefempty{\@glsxtr@thevalue}%
2527 {%
2528   \@gls@saveentrycounter

```

```

2529 }%
2530 {%
2531     \let\theglsentrycounter\@glsxtr@thevalue
2532     \def\theHglentrycounter{\@glsxtr@theHvalue}%
2533 }%
2534 \@gls@setsort{\glslabel}%

    Check if the textformat key has been used.
2535 \ifx\@glsxtr@local@textformat\relax

    Check textformat attribute (new to v1.21).
2536     \gls@hasattribute{\glslabel}{textformat}%
2537     {%
2538         \protected@edef\@glsxtr@attrval{\gls@getattribute{\glslabel}{textformat}}%
2539         \ifcsdef{\@glsxtr@attrval}%
2540         {%
2541             \letcs{\@glsxtr@textformat}{\@glsxtr@attrval}%
2542         }%
2543         {%
2544             \GlossariesExtraWarning{Unknown control sequence name
2545             '@@glsxtr@attrval' supplied in textformat attribute
2546             for entry '\glslabel'. Reverting to default \string\glstextformat}%
2547             \let\@glsxtr@textformat\glstextformat
2548         }%
2549     }%
2550     {%
2551         \let\@glsxtr@textformat\glstextformat
2552     }%
2553 \else
2554     \let\@glsxtr@textformat\@glsxtr@local@textformat
2555 \fi

    Do write if it should occur before the link text:
2556 \ifglsxtr@initwrglossbefore
2557     \@do@wrglossary{#2}%
2558 \fi

    Do the link text:
2559 \ifKV@glslink@hyper
2560     \ifglsxtr@hyperoutside
2561         \@glslink{\glslinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
2562     \else
2563         \@glsxtr@textformat{\@glslink{\glslinkprefix\glslabel}{#3}}%
2564     \fi
2565 \else
2566     \ifglsxtr@hyperoutside
2567         \glsdonohyperlink{\glslinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
2568     \else
2569         \@glsxtr@textformat{\glsdonohyperlink{\glslinkprefix\glslabel}{#3}}%
2570     \fi
2571 \fi

```

Do write if it should occur after the link text:

```
2572 \ifglxtrinitwrglossbefore
2573 \else
2574   \@do@wrglossary{#2}%
2575 \fi
```

Restore original value of \glolinkprefix:

```
2576 \let\glolinkprefix\@glxtr@org@glolinkprefix
```

As the original definition:

```
2577 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
2578 }
```

```
2579 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}
```

```
2580 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}
```

lsaddpresetkeys

```
2581 \newcommand*{\gladdpresetkeys}{}
```

saddpostsetkeys

```
2582 \newcommand*{\gladdpostsetkeys}{}
```

\gladd Redefine to include \@glxtr@record and suppress in headings

```
2583 \renewrobustcmd*{\gladd}[2][\]{%
2584   \glxtrifinmark
2585   {}%
2586   {%
2587     \@gl@adjustmode
2588     \begingroup
2589     \@glxtr@record{#1}{#2}{glossadd}%
2590     \glsoifexists{#2}%
2591     {%
2592       \let\@gl@numberformat\@glxtr@defaultnumberformat

2593       \protected@edef\@gl@counter{\csname glo@\gl@detoklabel{#2}@counter\endcsname}%
2594       \def\@glxtr@thevalue{}%
2595       \def\@glxtr@theHvalue{\@glxtr@thevalue}%
```

Implement any default settings (before options are set)

```
2596   \gladdpresetkeys
2597   \setkeys{glossadd}{#1}%
```

Implement any default settings (after options are set)

```
2598   \gladdpostsetkeys
2599   \ifdefempty{\@glxtr@thevalue}%
2600   {%
2601     \@gl@saveentrycounter
2602   }%
2603   {%
```

```

2604      \let\theglsentrycounter\@glstr@thevalue
2605      \def\theHglentrycounter{\@glstr@theHvalue}%
2606  }%

```

Define sort key if necessary (in case of sort=use):

```

2607      \@gls@setsort{#2}%

```

Ensure that indexing occurs (since that's the point of \glsadd). If indexing has been switched off by default, don't want the setting to affect \glsadd. The ignored format \glsignore can be used for selection without location, but the indexing still needs to be performed.

```

2608      \KV@glslink@noindexfalse
2609      \@do@wrglossary{#2}%
2610  }%
2611  \endgroup
2612 }%
2613 }

```

**\glsaddeach** Performs \glsadd for each entry listed in the mandatory argument.

```

2614 \newrobustcmd{\glsaddeach}[2][]{%
2615   \@for\@gls@thislabel:=#2\do{\glsadd[#1]{\@gls@thislabel}}%
2616 }

```

**@field@linkdefs** Default settings for \@gls@field@link

```

2617 \newcommand*{\@glstr@field@linkdefs}{%
2618   \let\glstrifwasfirstuse\@secondoftwo
2619   \let\glssifplural\@secondoftwo
2620   \let\glscapscase\@firstofthree
2621   \let\glsinsert\@empty
2622 }

```

Redefine the field link commands that need to modify the above. Also add accessibility support and set the abbreviation styles if required.

**assignfieldfont**

```

2623 \newcommand*{\glstrassignfieldfont}[1]{%
2624   \ifglstryexists{#1}%
2625   {%
2626     \ifglshasshort{#1}%
2627     {%
2628       \glssetabbrvfmt{\glscategory{#1}}%
2629       \glsifregular{#1}%
2630       {\let\@gls@field@font\glstrregularfont}%
2631       {\let\@gls@field@font\@firstofone}%
2632     }%
2633     {%
2634       \glsifnotregular{#1}%
2635       {\let\@gls@field@font\@firstofone}%
2636       {\let\@gls@field@font\glstrregularfont}%
2637     }%
2638   }%

```

```

2639  {%
2640    \let\@gls@field@font\@gobble
2641  }%
2642 }

```

`\@gls@text@` The abbreviation format may also need setting.

```

2643 \def\@gls@text@#1#2[#3]{%
2644   \glsxtrassignfieldfont{#2}%
2645   \@gls@field@link{#1}{#2}{\@gls@field@font{\gls@text@{#2}{#3}}}%
2646 }

```

`\@GLStext@` All uppercase version of `\@gls@text@`. The abbreviation format may also need setting.

```

2647 \def\@GLStext@#1#2[#3]{%
2648   \glsxtrassignfieldfont{#2}%
2649   \@gls@field@link{\let\gls@caps@case\@thirdofthree}{#1}{#2}%
2650   {\@gls@field@font{\GLS@text@{#2}\mfirstucMakeUppercase{#3}}}%
2651 }

```

`\@Gls@text@` First letter uppercase version. The abbreviation format may also need setting.

```

2652 \def\@Gls@text@#1#2[#3]{%
2653   \glsxtrassignfieldfont{#2}%
2654   \@gls@field@link{\let\gls@caps@case\@secondofthree}{#1}{#2}%
2655   {\@gls@field@font{\Gls@text@{#2}{#3}}}%
2656 }

```

Version 1.07 ensures that `\glsfirst` etc honours the `nohyperfirst` attribute. Allow a convenient way for the user to revert to ignoring this attribute for these commands.

`\@gls@checknohyperfirst`

```

2657 \newcommand*\@gls@checknohyperfirst[1]{%
2658   \gls@ifattribute{#1}{nohyperfirst}{true}{\KV@gls@link@hyperfalse}{}}%
2659 }

```

`\@glsfirst@` No case changing version. The abbreviation format may also need setting.

```

2660 \def\@glsfirst@#1#2[#3]{%
2661   \glsxtrassignfieldfont{#2}%
2662   \@gls@field@link
2663   [\let\glsxtrifwasfirstuse\@firstoftwo
2664    \glsxtrchecknohyperfirst{#2}%
2665    ]{#1}{#2}%
2666    {\@gls@field@font{\gls@first@{#2}{#3}}}%
2667 }

```

Ensure that `\@glsfirst` honours the `nohyperfirst` attribute.

`\@Glsfirst@` First letter uppercase version. The abbreviation format may also need setting.

```

2668 \def\@Glsfirst@#1#2[#3]{%
2669   \glsxtrassignfieldfont{#2}%

```

Ensure that \Glsfirst honours the nohyperfirst attribute.

```
2670 \@gls@field@link
2671 [\let\glstrifwasfirstuse\@firstoftwo
2672 \let\glscapscase\@secondofthree
2673 \glstrchecknohyperfirst{#2}%
2674 ]%
2675 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}{#3}}}%
2676 }
```

\@GLSfirst@ All uppercase version. The abbreviation format may also need setting.

```
2677 \def\@GLSfirst@#1#2[#3]{%
2678 \glstrassignfieldfont{#2}%
2679 \@gls@field@link
2680 [\let\glstrifwasfirstuse\@firstoftwo
2681 \let\glscapscase\@thirdofthree
2682 \glstrchecknohyperfirst{#2}%
2683 ]%
2684 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
2685 }
```

\@glsplural@ No case changing version. The abbreviation format may also need setting.

```
2686 \def\@glsplural@#1#2[#3]{%
2687 \glstrassignfieldfont{#2}%
2688 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
2689 {\@gls@field@font{\glsaccessplural{#2}{#3}}}%
2690 }
```

\@Glsplural@ First letter uppercase version. The abbreviation format may also need setting.

```
2691 \def\@Glsplural@#1#2[#3]{%
2692 \glstrassignfieldfont{#2}%
2693 \@gls@field@link
2694 [\let\glsifplural\@firstoftwo
2695 \let\glscapscase\@secondofthree
2696 ]%
2697 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}{#3}}}%
2698 }
```

\@GLSplural@ All uppercase version. The abbreviation format may also need setting.

```
2699 \def\@GLSplural@#1#2[#3]{%
2700 \glstrassignfieldfont{#2}%
2701 \@gls@field@link
2702 [\let\glsifplural\@firstoftwo
2703 \let\glscapscase\@thirdofthree
2704 ]%
2705 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
2706 }
```

glsfirstplural@ No case changing version. The abbreviation format may also need setting.

```
2707 \def\@glsfirstplural@#1#2[#3]{%
2708   \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
2709   \@gls@field@link
2710   [\let\glstrifwasfirstuse\@firstoftwo
2711    \let\glsifplural\@firstoftwo
2712    \glstrchecknohyperfirst{#2}%
2713   ]%
2714   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
2715 }
```

Glsfirstplural@ First letter uppercase version. The abbreviation format may also need setting.

```
2716 \def\@Glsfirstplural@#1#2[#3]{%
2717   \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
2718   \@gls@field@link
2719   [\let\glstrifwasfirstuse\@firstoftwo
2720    \let\glsifplural\@firstoftwo
2721    \let\glscapscase\@secondofthree
2722    \glstrchecknohyperfirst{#2}%
2723   ]%
2724   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
2725 }
```

GLSfirstplural@ All uppercase version. The abbreviation format may also need setting.

```
2726 \def\@GLSfirstplural@#1#2[#3]{%
2727   \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
2728   \@gls@field@link
2729   [\let\glstrifwasfirstuse\@firstoftwo
2730    \let\glsifplural\@firstoftwo
2731    \let\glscapscase\@thirdofthree
2732    \glstrchecknohyperfirst{#2}%
2733   ]%
2734   {#1}{#2}%
2735   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2736 }
```

\@glsname@ Redefine to use accessibility support. The abbreviation format may also need setting.

```
2737 \def\@glsname@#1#2[#3]{%
2738   \glstrassignfieldfont{#2}%
2739   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2740 }
```

\@Glsname@ First letter uppercase version. The abbreviation format may also need setting.

```
2741 \def\@Glsname@#1#2[#3]{%
```

```

2742 \glstrassignfieldfont{#2}%
2743 \@gls@field@link
2744 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2745 {\@gls@field@font{\Glsaccessname{#2}#3}}%
2746 }

```

\@GLSname@ All uppercase version. The abbreviation format may also need setting.

```

2747 \def\@GLSname@#1#2[#3]{%
2748 \glstrassignfieldfont{#2}%
2749 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2750 {#1}{#2}%
2751 {\@gls@field@font{\Glsaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2752 }

```

\@glsdesc@

```

2753 \def\@glsdesc@#1#2[#3]{%
2754 \glstrassignfieldfont{#2}%
2755 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2756 }

```

\@Glsdesc@ First letter uppercase version.

```

2757 \def\@Glsdesc@#1#2[#3]{%
2758 \glstrassignfieldfont{#2}%
2759 \@gls@field@link
2760 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2761 {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2762 }

```

\@GLSdesc@ All uppercase version.

```

2763 \def\@GLSdesc@#1#2[#3]{%
2764 \glstrassignfieldfont{#2}%
2765 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2766 {#1}{#2}{\@gls@field@font{\Glsaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2767 }

```

@glsdescplural@ No case-changing version.

```

2768 \def\@glsdescplural@#1#2[#3]{%
2769 \glstrassignfieldfont{#2}%
2770 \@gls@field@link
2771 [\let\glscapscase\@secondoftwo
2772 \let\glsifplural\@firstoftwo
2773 ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2774 }

```

@Glsdescplural@ First letter uppercase version.

```

2775 \def\@Glsdescplural@#1#2[#3]{%
2776 \glstrassignfieldfont{#2}%
2777 \@gls@field@link

```



```

2778 [\let\glscapscase\@secondoftwo
2779 \let\glsifplural\@firstoftwo
2780 ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2781 }

```

@GLSdescplural@ All uppercase version.

```

2782 \def\@GLSdesc@#1#2[#3]{%
2783 \glstrassignfieldfont{#2}%
2784 \@gls@field@link
2785 [\let\glscapscase\@thirdoftwo
2786 \let\glsifplural\@firstoftwo
2787 ]%
2788 {#1}{#2}%
2789 {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2790 }

```

\@glssymbol@

```

2791 \def\@glssymbol@#1#2[#3]{%
2792 \glstrassignfieldfont{#2}%
2793 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2794 }

```

\@GLssymbol@ First letter uppercase version.

```

2795 \def\@GLssymbol@#1#2[#3]{%
2796 \glstrassignfieldfont{#2}%
2797 \@gls@field@link
2798 [\let\glscapscase\@secondoftwo]%
2799 {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2800 }

```

\@GLSsymbol@ All uppercase version.

```

2801 \def\@GLSsymbol@#1#2[#3]{%
2802 \glstrassignfieldfont{#2}%
2803 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2804 {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2805 }

```

lssymbolplural@ No case-changing version.

```

2806 \def\@glssymbolplural@#1#2[#3]{%
2807 \glstrassignfieldfont{#2}%
2808 \@gls@field@link
2809 [\let\glscapscase\@secondoftwo
2810 \let\glsifplural\@firstoftwo
2811 ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2812 }

```

lssymbolplural@ First letter uppercase version.

```

2813 \def\@Glssymbolplural@#1#2[#3]{%

```

```

2814 \glstrassignfieldfont{#2}%
2815 \@gls@field@link
2816 [\let\glscapscase\@secondoftwo
2817 \let\glsifplural\@firstoftwo
2818 ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2819 }

```

Lsymbolplural@ All uppercase version.

```

2820 \def\@GLSsymbol@#1#2[#3]{%
2821 \glstrassignfieldfont{#2}%
2822 \@gls@field@link
2823 [\let\glscapscase\@thirdoftwo
2824 \let\glsifplural\@firstoftwo
2825 ]%
2826 {#1}{#2}%
2827 {\@gls@field@font{\Glsaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2828 }

```

\@Glsuseri@ First letter uppercase version.

```

2829 \def\@Glsuseri@#1#2[#3]{%
2830 \glstrassignfieldfont{#2}%
2831 \@gls@field@link
2832 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2833 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2834 }

```

\@GLSuseri@ All uppercase version.

```

2835 \def\@GLSuseri@#1#2[#3]{%
2836 \glstrassignfieldfont{#2}%
2837 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2838 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuseri{#2}#3}}}%
2839 }

```

\@Glsuserii@ First letter uppercase version.

```

2840 \def\@Glsuserii@#1#2[#3]{%
2841 \glstrassignfieldfont{#2}%
2842 \@gls@field@link
2843 [\let\glscapscase\@secondoftwo]%
2844 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2845 }

```

\@GLSuserii@ All uppercase version.

```

2846 \def\@GLSuserii@#1#2[#3]{%
2847 \glstrassignfieldfont{#2}%
2848 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2849 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}}%
2850 }

```

\@Glsuseriii@ First letter uppercase version.

```
2851 \def\@Glsuseriii@#1#2[#3]{%
2852   \glstrassignfieldfont{#2}%
2853   \@gls@field@link
2854   [\let\glscapscase\@secondoftwo]%
2855   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2856 }
```

\@GLSuseriii@ All uppercase version.

```
2857 \def\@GLSuseriii@#1#2[#3]{%
2858   \glstrassignfieldfont{#2}%
2859   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2860   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\Glsentryuseriii{#2}#3}}}%
2861 }
```

\@Glsuseriv@ First letter uppercase version.

```
2862 \def\@Glsuseriv@#1#2[#3]{%
2863   \glstrassignfieldfont{#2}%
2864   \@gls@field@link
2865   [\let\glscapscase\@secondoftwo]%
2866   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2867 }
```

\@GLSuseriv@ All uppercase version.

```
2868 \def\@GLSuseriv@#1#2[#3]{%
2869   \glstrassignfieldfont{#2}%
2870   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2871   {#1}{#2}%
2872   {\@gls@field@font{\mfirstucMakeUppercase{\Glsentryuseriv{#2}#3}}}%
2873 }
```

\@Glsuserv@ First letter uppercase version.

```
2874 \def\@Glsuserv@#1#2[#3]{%
2875   \glstrassignfieldfont{#2}%
2876   \@gls@field@link
2877   [\let\glscapscase\@secondoftwo]%
2878   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
2879 }
```

\@GLSuserv@ All uppercase version.

```
2880 \def\@GLSuserv@#1#2[#3]{%
2881   \glstrassignfieldfont{#2}%
2882   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2883   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\Glsentryuserv{#2}#3}}}%
2884 }
```

\@Glsuservi@ First letter uppercase version.

```
2885 \def\@Glsuservi@#1#2[#3]{%
```

```

2886 \glxtrassignfieldfont{#2}%
2887 \@gls@field@link
2888 [\let\glscapscase\@secondoftwo]%
2889 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
2890 }

```

\@GLSuservi@ All uppercase version.

```

2891 \def\@GLSuservi@#1#2[#3]{%
2892 \glxtrassignfieldfont{#2}%
2893 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2894 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2895 }

```

Commands like \acrshort already set \glsifplural, but they don't set \glxtrifwasfirstuse so they need adjusting. These commands shouldn't be used with \newabbreviation, but the redefinitions below allow for users reverting \newacronym back to its base definition.

base@acrcmd@warn Warn user that they need to use to new abbreviation commands.

```

2896 \newcommand*{\@glsxtr@base@acrcmd@warn}[2]{%
2897 \GlossariesExtraWarning{Base acronym command \string#1\space
2898 should not be used with new abbreviation definitions. Use
2899 \string#2\space instead}%
2900 }

```

xtr@base@acrcmd Warn user that they need to use to new abbreviation commands.

```

2901 \let\@glsxtr@base@acrcmd\@glsxtr@base@acrcmd@warn

```

\@acrshort No case change.

```

2902 \def\@acrshort#1#2[#3]{%
2903 \@glsxtr@base@acrcmd\acrshort\glxtrshort
2904 \glsdoifexists{#2}%
2905 {%
2906 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2907 \let\glxtrifwasfirstuse\@secondoftwo
2908 \let\glsifplural\@secondoftwo
2909 \let\glscapscase\@firstofthree
2910 \let\glsinsert\@empty
2911 \def\glscustomtext{%
2912 \acronymfont{\glsaccessshort{#2}}#3%
2913 }%
2914 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2915 }%
2916 \glspostlinkhook
2917 }

```

\@Acrshort First letter uppercase.

```

2918 \def\@Acrshort#1#2[#3]{%
2919 \@glsxtr@base@acrcmd\Acrshort\Glsxtrshort

```

```

2920 \glsdoifexists{#2}%
2921 {%
2922   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2923   \let\glxtrifwasfirstuse\@secondoftwo
2924   \let\gl@ifplural\@secondoftwo
2925   \let\glscapscase\@secondofthree
2926   \let\glinsert\@empty
2927   \def\glscustomtext{%
2928     \acronymfont{\Glsaccesssshort{#2}}#3%
2929   }%
2930   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2931 }%
2932 \glspostlinkhook
2933 }

```

\@ACRshort All uppercase.

```

2934 \def\@ACRshort#1#2[#3]{%
2935   \@glxtr@base@acrcmd\ACRshort\GLSxtrshort
2936   \glsdoifexists{#2}%
2937   {%
2938     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2939     \let\glxtrifwasfirstuse\@secondoftwo
2940     \let\gl@ifplural\@secondoftwo
2941     \let\glscapscase\@thirdofthree
2942     \let\glinsert\@empty
2943     \def\glscustomtext{%
2944       \mfirstucMakeUppercase{\acronymfont{\Glsaccesssshort{#2}}#3}%
2945     }%
2946     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2947   }%
2948   \glspostlinkhook
2949 }

```

\@acrshortpl No case change.

```

2950 \def\@acrshortpl#1#2[#3]{%
2951   \@glxtr@base@acrcmd\acrshortpl\glxtrshortpl
2952   \glsdoifexists{#2}%
2953   {%
2954     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2955     \let\glxtrifwasfirstuse\@secondoftwo
2956     \let\gl@ifplural\@firstoftwo
2957     \let\glscapscase\@firstofthree
2958     \let\glinsert\@empty
2959     \def\glscustomtext{%
2960       \acronymfont{\Glsaccesssshortpl{#2}}#3%
2961     }%
2962     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2963   }%
2964   \glspostlinkhook

```

2965 }

\@Acrshortpl First letter uppercase.

```
2966 \def\@Acrshortpl#1#2[#3]{%
2967   \@glsxtr@base@acrcmd\Acrshortpl\Glsxtrshortpl
2968   \glsdoifexists{#2}%
2969   {%
2970     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2971     \let\glsxtrifwasfirstuse\@secondoftwo
2972     \let\glsifplural\@firstoftwo
2973     \let\glsapscase\@secondofthree
2974     \let\glsinsert\@empty
2975     \def\glscustomtext{%
2976       \acronymfont{\Glsaccessshortpl{#2}}#3%
2977     }%
2978     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2979   }%
2980   \glspostlinkhook
2981 }
```

\@ACRshortpl All uppercase.

```
2982 \def\@ACRshortpl#1#2[#3]{%
2983   \@glsxtr@base@acrcmd\ACRshortpl\GLSxtrshortpl
2984   \glsdoifexists{#2}%
2985   {%
2986     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2987     \let\glsxtrifwasfirstuse\@secondoftwo
2988     \let\glsifplural\@firstoftwo
2989     \let\glsapscase\@thirdofthree
2990     \let\glsinsert\@empty
2991     \def\glscustomtext{%
2992       \mfirstucMakeUppercase{\acronymfont{\Glsaccessshortpl{#2}}#3}%
2993     }%
2994     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2995   }%
2996   \glspostlinkhook
2997 }
```

\@acrlong No case change.

```
2998 \def\@acrlong#1#2[#3]{%
2999   \@glsxtr@base@acrcmd\acrlong\glsxtrlong
3000   \glsdoifexists{#2}%
3001   {%
3002     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3003     \let\glsxtrifwasfirstuse\@secondoftwo
3004     \let\glsifplural\@secondoftwo
3005     \let\glsapscase\@firstofthree
3006     \let\glsinsert\@empty
3007     \def\glscustomtext{%
```

```

3008      \acronymfont{\glsaccesslong{#2}}#3%
3009    }%
3010    \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3011  }%
3012  \glspostlinkhook
3013}

```

\@Acrlong First letter uppercase.

```

3014 \def\@Acrlong#1#2[#3]{%
3015   \@glsxtr@base@acrcmd\Acrlong\Glsxtrlong
3016   \glsdoifexists{#2}%
3017   {%
3018     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3019     \let\glsxtrifwasfirstuse\@secondoftwo
3020     \let\glsifplural\@secondoftwo
3021     \let\glscapscase\@secondofthree
3022     \let\glsinsert\@empty
3023     \def\glscustomtext{%
3024       \acronymfont{\Glsaccesslong{#2}}#3%
3025     }%
3026     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3027   }%
3028   \glspostlinkhook
3029}

```

\@ACRlong All uppercase.

```

3030 \def\@ACRlong#1#2[#3]{%
3031   \@glsxtr@base@acrcmd\ACRlong\GLSxtrlong
3032   \glsdoifexists{#2}%
3033   {%
3034     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3035     \let\glsxtrifwasfirstuse\@secondoftwo
3036     \let\glsifplural\@secondoftwo
3037     \let\glscapscase\@thirdofthree
3038     \let\glsinsert\@empty
3039     \def\glscustomtext{%
3040       \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
3041     }%
3042     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3043   }%
3044   \glspostlinkhook
3045}

```

\@acrlongpl No case change.

```

3046 \def\@acrlongpl#1#2[#3]{%
3047   \@glsxtr@base@acrcmd\acrlongpl\glsxtrlongpl
3048   \glsdoifexists{#2}%
3049   {%
3050     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

3051 \let\glxtrifwasfirstuse\@secondoftwo
3052 \let\glsifplural\@firstoftwo
3053 \let\glscapscase\@firstofthree
3054 \let\glsinsert\@empty
3055 \def\glscustomtext{%
3056   \acronymfont{\glsaccesslongpl{#2}}#3%
3057 }%
3058 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3059 }%
3060 \glspostlinkhook
3061 }

```

\@Acrlongpl First letter uppercase.

```

3062 \def\@Acrlongpl#1#2[#3]{%
3063   \@glxtr@base@acrcmd\Acrlongpl\Glsxtrlongpl
3064   \glsdoifexists{#2}%
3065   {%
3066     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3067     \let\glxtrifwasfirstuse\@secondoftwo
3068     \let\glsifplural\@firstoftwo
3069     \let\glscapscase\@secondofthree
3070     \let\glsinsert\@empty
3071     \def\glscustomtext{%
3072       \acronymfont{\Glsaccesslongpl{#2}}#3%
3073     }%
3074     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3075   }%
3076   \glspostlinkhook
3077 }

```

\@ACRlongpl All uppercase.

```

3078 \def\@ACRlongpl#1#2[#3]{%
3079   \@glxtr@base@acrcmd\ACRlongpl\GLSxtrlongpl
3080   \glsdoifexists{#2}%
3081   {%
3082     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
3083     \let\glxtrifwasfirstuse\@secondoftwo
3084     \let\glsifplural\@firstoftwo
3085     \let\glscapscase\@thirdofthree
3086     \let\glsinsert\@empty
3087     \def\glscustomtext{%
3088       \mfirstucMakeUppercase{\acronymfont{\glsaccesslongpl{#2}}#3}%
3089     }%
3090     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
3091   }%
3092   \glspostlinkhook
3093 }

```

The full formats use the internal long and short commands (such as \@acrshort and \@acrlong). Therefore they don't need adjustments, but they do need clearer warnings. This



means three warnings per use (once for the full command and once each for the short and long commands), but at least this way the most important warning (replace \acrfull with \glsxtrfull etc) is present.

\@acrfull

```
3094 \def\@acrfull#1#2[#3]{%
3095   \@glsxtr@base@acrcmd\acrfull\glsxtrfull
3096   \acrfullfmt{#1}{#2}{#3}%
3097 }
```

\@Acrfull

```
3098 \def\@Acrfull#1#2[#3]{%
3099   \@glsxtr@base@acrcmd\Acrfull\Glsxtrfull
3100   \Acrfullfmt{#1}{#2}{#3}%
3101 }
```

\@ACRfull

```
3102 \def\@ACRfull#1#2[#3]{%
3103   \@glsxtr@base@acrcmd\ACRfull\GLSxtrfull
3104   \ACRfullfmt{#1}{#2}{#3}%
3105 }
```

\@acrfullpl

```
3106 \def\@acrfullpl#1#2[#3]{%
3107   \@glsxtr@base@acrcmd\acrfullpl\glsxtrfullpl
3108   \acrfullplfmt{#1}{#2}{#3}%
3109 }
```

\@Acrfullpl

```
3110 \def\@Acrfullpl#1#2[#3]{%
3111   \@glsxtr@base@acrcmd\Acrfullpl\Glsxtrfullpl
3112   \Acrfullplfmt{#1}{#2}{#3}%
3113 }
```

\@ACRfullpl

```
3114 \def\@ACRfullpl#1#2[#3]{%
3115   \@glsxtr@base@acrcmd\ACRfullpl\GLSxtrfullpl
3116   \ACRfullplfmt{#1}{#2}{#3}%
3117 }
```

Modify \@glsaddkey so additional keys provided by the user can be treated in a similar way.

\@glsaddkey

```
3118 \renewcommand*{\@glsaddkey}[7]{%
3119   \key@ifundefined{glossentry}{#1}%
3120   {%
3121     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
```

```

3122 \appto\@gls@keymap{,{#1}{#1}}%
3123 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
3124 \appto\@newglossaryentryposthook{%
3125   \letcs{\@glo@tmp}{@glo@#1}%
3126   \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
3127 }%
3128 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
3129 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

Now for the commands with links. First the version with no case change (same as before):

```

3130 \ifcsdef{@gls@user@#1@}%
3131 {%
3132   \PackageError{glossaries}%
3133   {Can't define '\string#5' as helper command
3134     '\expandafter\string\csname @gls@user@#1@endcsname' already
3135     exists}%
3136 }%
3137 {%
3138   \expandafter\newcommand\expandafter*\expandafter
3139   {\csname @gls@user@#1@endcsname}[2][ ]{%
3140     \new@ifnextchar[%
3141       {\csuse{@gls@user@#1@}{##1}{##2}}%
3142       {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
3143   \csdef{@gls@user@#1@}##1##2[##3]{%
3144     \@gls@field@link{##1}{##2}{#3{##2}##3}%
3145   }%
3146   \newrobustcmd*{#5}{%
3147     \expandafter\@gls@hyp@opt\csname @gls@user@#1@endcsname}%
3148   }%
3149 }%

```

Next the version with the first letter converted to upper case (modified):

```

3150 \ifcsdef{@Gls@user@#1@}%
3151 {%
3152   \PackageError{glossaries}%
3153   {Can't define '\string#6' as helper command
3154     '\expandafter\string\csname @Gls@user@#1@endcsname' already
3155     exists}%
3156 }%
3157 {%
3158   \expandafter\newcommand\expandafter*\expandafter
3159   {\csname @Gls@user@#1@endcsname}[2][ ]{%
3160     \new@ifnextchar[%
3161       {\csuse{@Gls@user@#1@}{##1}{##2}}%
3162       {\csuse{@Gls@user@#1@}{##1}{##2}[ ]}}%
3163   \csdef{@Gls@user@#1@}##1##2[##3]{%
3164     \@gls@field@link[\let\glscaps@case\@secondofthree]%
3165     {##1}{##2}{#4{##2}##3}%
3166   }%
3167 }%

```

```

3168     \newrobustcmd*{#6}{%
3169         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
3170     }%
    Finally the all caps version (modified):
3171     \ifcsdef{@Gls@user@#1@}%
3172     {%
3173         \PackageError{glossaries}%
3174         {Can't define '\string#7' as helper command
3175         '\expandafter\string\csname @Gls@user@#1\endcsname' already
3176         exists}%
3177     }%
3178 }%
3179 {%
3180     \expandafter\newcommand\expandafter*\expandafter
3181     {\csname @Gls@user@#1\endcsname}[2][ ]{%
3182         \new@ifnextchar[%
3183             {\csuse{@Gls@user@#1@}{##1}{##2}}%
3184             {\csuse{@Gls@user@#1@}{##1}{##2}[ ]}}%
3185     \csdef{@Gls@user@#1@}##1##2[##3]{%
3186         \@gls@field@link[\let\glscaps@case\@thirdofthree]%
3187         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
3188     }%
3189     \newrobustcmd*{#7}{%
3190         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
3191     }%
3192 }%
3193 {%
3194     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
3195 }%
3196 }

```

checkfirsthyper Old versions of glossaries don't define this, so provide it just in case it hasn't been defined.

```

3197 \providecommand*\@gls@link@nocheckfirsthyper{}

```

checkfirsthyper Modify check to determine if the hyperlink should be automatically suppressed, but save the original in case the acronyms are restored.

```

3198 \let\@glxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
3199 \renewcommand*\@gls@link@checkfirsthyper{%

```

\ifglused isn't useful in the post link hook as it's already been unset by then, so define a command that can be used in the post link hook. Since \@gls@link@checkfirsthyper is only used by commands like \gls but not by other commands, this seems the best place to put it to automatically set the value for the commands that change the first use flag. The other commands should set \glxtrifwasfirstuse to \@secondoftwo (which is done in \@glxtr@field@linkdefs). Note that if the entry is undefined (as with bib2gls on the first  $\text{\LaTeX}$  run), \ifglused does neither true nor false parts. However, in that case, this macro won't be called anyway (since it's used in the argument of \glsdoifexistsordo).

```

3200 \ifglused{\glslabel}%

```

```

3201 {\let\glstrifwasfirstuse\@secondoftwo}
3202 {\let\glstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.
3203 \protected@edef\glscategorylabel{\glscategory{\glslabel}}%
3204 \ifglused{\glslabel}%
3205 {%
3206   \glscategoryattribute{\glscategorylabel}{nohypernext}{true}%
3207   {\KV@glslink@hyperfalse}{}}%
3208 }%
3209 {%
3210   \glscategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
3211   {\KV@glslink@hyperfalse}{}}%
3212 }%
3213 \glslinkcheckfirsthyperhook
3214 }

```

ablehyperinlist This command was introduced in glossaries v4.19. If it hasn't been defined, we're using an earlier version, in which case the nohyper attribute can't be implemented.

```

3215 \ifdefined\do@gl:disablehyperinlist
3216 {%
3217   \let\glstr@do@gl:disablehyperinlist\do@gl:disablehyperinlist
3218   \renewcommand*{\do@gl:disablehyperinlist}{%
3219     \@glstr@do@gl:disablehyperinlist
3220     \glscategoryattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}}%
3221   }
3222 }
3223 {}

```

Define a noindex key to prevent writing information to the external file.

```

3224 \define@boolkey{glslink}{noindex}[true]{}
3225 \KV@glslink@noindexfalse

```

If \@gls@setdefault@glslink@opts has been defined (glossaries v4.20) use it to set the default keys in \@glslink.

lt@glslink@opts

```

3226 \ifdefined\@gls@setdefault@glslink@opts
3227 {
3228   \renewcommand*{\@gls@setdefault@glslink@opts}{%
3229     \KV@glslink@noindexfalse
3230     \@glstrsetaliasnoindex
3231   }
3232 }
3233 {

```

Not defined so prepend it to \do@gl:disablehyperinlist to achieve the same effect.

```

3234 \newcommand*{\@gls@setdefault@glslink@opts}{%
3235   \KV@glslink@noindexfalse
3236   \@glstrsetaliasnoindex

```

```

3237 }
3238 \pretodo{@glsdisablehyperinlist{\@gls@setdefault@glslink@opts}
3239 }

```

`setaliasnoindex` Allow user to hook into the alias noindex setting. Default behaviour switches off indexing for aliases. If the record option is on, this will have been defined to do nothing. (bib2gls will deal with records for aliased entries.)

```

3240 \providecommand*\glxtrsetaliasnoindex{%
3241 \KV@glslink@noindextrue
3242 }

```

`setaliasnoindex` The change made in v1.46 to remove the grouping has had the knock-on effect of redefining `\glscurrentfieldvalue`, which may be a problem, so v1.47 has changed this to use `\ifcsvoid`.

```

3243 \newcommand*\@glxtrsetaliasnoindex{%
3244 \ifcsvoid{glo@\glsdetoklabel{\glslabel}@alias}%
3245 {}%
3246 {%
3247 \let\glxtrindexaliased\@glxtrindexaliased
3248 \glxtrsetaliasnoindex
3249 \let\glxtrindexaliased\@no@glxtrindexaliased
3250 }%
3251 }

```

`xtrindexaliased`

```

3252 \newcommand*\@glxtrindexaliased{%
3253 \ifKV@glslink@noindex
3254 \else
3255 \begingroup
3256 \let\@glsnumberformat\@glxtr@defaultnumberformat
3257 \protected@edef\@gls@counter{\csname glo@\glsdetoklabel{\glslabel}@counter\endcsname}%
3258 \glxtr@saveentrycounter
3259 \@@do@wrglossary{\glxtralias{\glslabel}}%
3260 \endgroup
3261 \fi
3262 }

```

`xtrindexaliased`

```

3263 \newcommand*\@no@glxtrindexaliased{%
3264 \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
3265 not permitted outside definition of \string\glxtrsetaliasnoindex}%
3266 {}%
3267 }

```

`xtrindexaliased` Provide a command to redirect alias indexing, but only allow it to be used within `\glxtrsetaliasnoindex`.

```

3268 \let\glxtrindexaliased\@no@glxtrindexaliased

```

tDefaultGlsOpts Set the default options for \glslink etc.

```
3269 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
3270   \renewcommand*{\@gls@setdefault@glslink@opts}{%
3271     \setkeys{glslink}{#1}%
3272     \@glsxtrsetaliasnoindex
3273   }%
3274 }
```

lsxtrifindexing Provide user level command to access it in \glswriteentry.

```
3275 \newcommand*{\glsxtrifindexing}[2]{%
3276   \ifKV@glslink@noindex #2\else #1\fi
3277 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute. This needs to use \GlsXtrIfUnusedOrUndefined instead of \ifglsused to allow it to work with bib2gls.

```
3278 \renewcommand*{\glswriteentry}[2]{%
3279   \glsxtrifindexing
3280   {%
3281     \ifglsindexonlyfirst
3282       \GlsXtrIfUnusedOrUndefined{#1}
3283       {#2}%
3284       {\glsxtrdoautoindexname{#1}{dualindex}}%
3285     \else
3286       \glsifattribute{#1}{indexonlyfirst}{true}%
3287       {%
3288         \GlsXtrIfUnusedOrUndefined{#1}%
3289         {#2}%
3290         {\glsxtrdoautoindexname{#1}{dualindex}}%
3291       }%
3292     {#2}%
3293   \fi
3294   }%
3295   {}%
3296 }
```

@do@wrglossary Hook into glossary indexing command so that it can also use \index at the same time if required and add user hook.

```
3297 \appto\@do@wrglossary{\@glsxtr@do@wrindex
3298   \glsxtrdowrglossaryhook{\@gls@label}%
3299 }
```

(The label can be obtained from \@gls@label at this point.)

Similarly for the “noidx” version:

s@noidxglossary

```
3300 \appto\gls@noidxglossary{\@glsxtr@do@wrindex
3301   \glsxtrdowrglossaryhook{\@gls@label}%
3302 }
```

xtr@do@@wrindex

```
3303 \newcommand*{\@glsxtr@do@@wrindex}{%
3304   \glsxtrdoautoindexname{\@gls@label}{dualindex}%
3305 }
```

owrglossaryhook Allow user to hook into indexing code. (Always used by \glsadd. Used by \gls when indexing, which may or may not occur depending on the indexing settings.)

```
3306 \newcommand*{\glsxtrdowrglossaryhook}[1]{}
```

gls@alt@hyp@opt Commands like \gls have a star or plus version. Provide a third symbol that the user can adapt for convenience.

```
3307 \newcommand*{\@gls@alt@hyp@opt}[1]{%
3308   \let\glslinkvar\@firstofthree
3309   \let\@gls@hyp@opt@cs#1\relax
3310   \@ifstar{\s@gls@hyp@opt}%
3311   {\@ifnextchar+%
3312     {\@firstoftwo{\p@gls@hyp@opt}}%
3313     {%
3314       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
3315       {\@firstoftwo{\@alt@gls@hyp@opt}}%
3316       {#1}%
3317     }%
3318   }%
3319 }
```

alt@gls@hyp@opt User version

```
3320 \newcommand*{\@alt@gls@hyp@opt}[1] []{%
3321   \let\glslinkvar\@firstofthree
3322   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

lt@hyp@opt@char Contains the character used as the command modifier.

```
3323 \newcommand*{\@gls@alt@hyp@opt@char}{{}
```

lt@hyp@opt@keys Contains the option list used as the command modifier.

```
3324 \newcommand*{\@gls@alt@hyp@opt@keys}{{}
```

rSetAltModifier

```
3325 \newcommand*{\GlsXtrSetAltModifier}[2]{%
3326   \let\@gls@hyp@opt\@gls@alt@hyp@opt

   Check that the supplied character isn't + or *
3327   \ifstrequal{#1}{+}%
3328   {\PackageError{glossaries-extra}%
3329     {Can't use '#1' as modifier (it's already in use)}}}%
3330   {%
3331     \ifstrequal{#1}{*}%
3332     {\PackageError{glossaries-extra}%
3333       {Can't use '#1' as modifier (it's already in use)}}}%

```

```

3334     {}%
3335 }%
3336 \def\@gls@alt@hyp@opt@char{#1}%
3337 \def\@gls@alt@hyp@opt@keys{#2}%
3338 \ifdefequal\@glsxtr@record@setting\@glsxtr@record@setting@off
3339 {}%
3340 {%

```

Let **bib2gls** know the modifier.

```

3341     \protected@write\@auxout{}\string\providecommand{\string\@glsxtr@altmodifier}[1]{}%
3342     \protected@write\@auxout{}\string\@glsxtr@altmodifier{#1}}%
3343 }%
3344 }

```

org@dohyperlink

```

3345 \let\glsxtr@org@dohyperlink\glsdohyperlink

```

glsnavhyperlink

Since `\glsnavhyperlink` uses `\glslink`, it's necessary to patch it uses `\glsdohyperlink` instead of `\glsxtrdohyperlink`. The simplest way to achieve this is to locally let `\glsxtrdohyperlink` to `\glsdohyperlink`.

This command is provided by `glossary-hypernav` so it may not exist.

```

3346 \ifdef\glsnavhyperlink
3347 {
3348   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
3349     \protected@edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%

```

Scope:

```

3350     {%
3351       \let\glsxtrdohyperlink\glsxtr@org@dohyperlink
3352       \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
3353     }%
3354   }%
3355 }
3356 {}

```

The redefinition of `\glsdohyperlink` has been causing problems so introduce a new command instead.

sxtrdohyperlink

Unpleasant complications can occur if the text or first key etc contains `\gls`, particularly if there are hyperlinks. To get around this problem, patch `\glsdohyperlink` so that it temporarily makes `\gls` behave like `\glstext[hyper=false,noindex]`. (This will be overridden if the user explicitly cancels either of those options in the optional argument of `\gls` or using the plus version.) This also patches the short form commands like `\acrshort` and `\glsxtrshort` to use `\glsentryshort` and, similarly, the long form commands like `\acrlong` and `\glsxtrlong` to use `\glsentrylong`. Added attribute check.

```

3357 \newcommand*{\glsxtrdohyperlink}[2]{%
3358   \glsattribute{\glslabel}{targeturl}%

```



```

3359 {%
3360   \glshasattribute{\glslabel}{targetname}%
3361   {%
3362     \glshasattribute{\glslabel}{targetcategory}%
3363     {%
3364       \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
3365         {\glsgetattribute{\glslabel}{targetcategory}}}%
3366         {\glsgetattribute{\glslabel}{targetname}}}%
3367         {{\glsxtrprotectlinks#2}}}%
3368     }%
3369   {%
3370     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
3371       {}%
3372       {\glsgetattribute{\glslabel}{targetname}}}%
3373       {{\glsxtrprotectlinks#2}}}%
3374   }%
3375 }%
3376 {%
3377   \href{\glsgetattribute{\glslabel}{targeturl}}{%
3378     {{\glsxtrprotectlinks#2}}}%
3379   }%
3380 }%
3381 {%

```

Check for alias.

```

3382   \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
3383   \ifdefvoid\gloaliaslabel
3384   {%
3385     \glsxtrhyperlink{#1}{{\glsxtrprotectlinks#2}}%
3386   }%
3387   {%

```

Redirect link to the alias target.

```

3388     \glsxtrhyperlink
3389     {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
3390     {{\glsxtrprotectlinks#2}}%
3391   }%
3392 }%
3393 }

```

`\glsxtrhyperlink` Allows integration with the base glossaries package's `debug=showtargets` option.

```

3394 \ifdef\@glsshowtarget
3395 {
3396   \newcommand{\glsxtrhyperlink}[2]{%
3397     \@glsshowtarget{#1}%
3398     \hyperlink{#1}{#2}%
3399   }%
3400 }
3401 {
3402   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%

```

3403 }

`glsdisablehyper` Redefine to set `\glslabel` (to allow it to be picked up by `\glsdohyperlink`). Also made it robust and added grouping to localise the definition of `\glslabel`. The original internal command `@glo@label` could probably be simply replaced with `\glslabel`, but it's retained in case its removal causes unexpected problems.

```
3404 \renewrobustcmd*{\glsdohyperlink}[2][\glsentrytext{\@glo@label}]{%
3405   \glsdoifexists{#2}%
3406   {%
3407     \def\@glo@label{#2}%
3408     {\protected@edef\glslabel{#2}%
3409      \@glslink{\glslinkprefix\glslabel}{#1}}%
3410   }%
3411 }
```

`glsdisablehyper` Redefine in case we have an old version of glossaries. This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdonohyperlink`.

```
3412 \renewcommand{\glsdisablehyper}{%
3413   \KV@glslink@hyperfalse
3414   \def\@glslink{\glsdonohyperlink}%
3415   \let\@glstarget\@secondoftwo
3416 }
```

`\glsenablehyper` This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdohypertarget` and `\glsdohyperlink`.

```
3417 \renewcommand{\glsenablehyper}{%
3418   \KV@glslink@hypertrue
3419   \def\@glslink{\glsxtrdohyperlink}%
3420   \def\@glstarget{\glsdohypertarget}%
3421 }
```

`glsdonohyperlink` This command was only introduced in glossaries v4.20, so it may not be defined (therefore use `\def`). For older glossaries versions, this won't be used if `hyperref` hasn't been loaded, which means the indexing will still take place. The generated text is scoped (the link text in `\hyperlink` is also scoped, so it's consistent).

```
3422 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}
```

`\@glslink` Reset `\@glslink` with patched versions:

```
3423 \ifcsundef{hyperlink}%
3424 {%
3425   \def\@glslink{\glsdonohyperlink}
3426 }%
3427 {%
3428   \def\@glslink{\glsxtrdohyperlink}
3429 }
```

`\glstrprotectlinks` Make `\gls` (and variants) behave like the corresponding `\gls{text}` (and variants) with hyperlinking and indexing off.

```

3430 \newcommand*{\glstrprotectlinks}{%
3431   \KV@glslink@hyperfalse
3432   \KV@glslink@noindextrue
3433   \let\@gls@\@glstr@p@text@
3434   \let\@Gls@\@Glsxtr@p@text@
3435   \let\@GLS@\@GLSxtr@p@text@
3436   \let\@glspl@\@glstr@p@plural@
3437   \let\@Glspl@\@Glsxtr@p@plural@
3438   \let\@GLSpl@\@GLSxtr@p@plural@
3439   \let\@glstrshort@\@glstr@p@short@
3440   \let\@Glsxtrshort@\@Glsxtr@p@short@
3441   \let\@GLSxtrshort@\@GLSxtr@p@short@
3442   \let\@glstrlong@\@glstr@p@long@
3443   \let\@Glsxtrlong@\@Glsxtr@p@long@
3444   \let\@GLSxtrlong@\@GLSxtr@p@long@
3445   \let\@glstrshortpl@\@glstr@p@shortpl@
3446   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
3447   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
3448   \let\@glstrlongpl@\@glstr@p@longpl@
3449   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
3450   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
3451   \let\@acrshort@\@glstr@p@acrshort@
3452   \let\@Acrshort@\@Glsxtr@p@acrshort@
3453   \let\@ACRshort@\@GLSxtr@p@acrshort@
3454   \let\@acrshortpl@\@glstr@p@acrshortpl@
3455   \let\@Acrshortpl@\@Glsxtr@p@acrshortpl@
3456   \let\@ACRshortpl@\@GLSxtr@p@acrshortpl@
3457   \let\@acrlong@\@glstr@p@acrlong@
3458   \let\@Acrlong@\@Glsxtr@p@acrlong@
3459   \let\@ACRlong@\@GLSxtr@p@acrlong@
3460   \let\@acrlongpl@\@glstr@p@acrlongpl@
3461   \let\@Acrlongpl@\@Glsxtr@p@acrlongpl@
3462   \let\@ACRlongpl@\@GLSxtr@p@acrlongpl@
3463 }

```

These protected versions need grouping to prevent the label from getting confused.

`@glstr@p@text@`

```

3464 \def\@glstr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}

```

`@Glsxtr@p@text@`

```

3465 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}

```

`@GLSxtr@p@text@`

```

3466 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}

```

`lsxtr@p@plural@`

```

3467 \def\@glstr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}

```

```

lsxtr@p@plural@
3468 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2} [#3]}}

LSxtr@p@plural@
3469 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2} [#3]}}

glxtr@p@short@
3470 \def\@glxtr@p@short@#1#2[#3]{%
3471 {%
3472   \glsssetabbrvfmt{\glscategory{#2}}%
3473   \glsabbrvfont{\glsentryshort{#2}}#3%
3474 }%
3475 }

Glsxtr@p@short@
3476 \def\@Glsxtr@p@short@#1#2[#3]{%
3477 {%
3478   \glsssetabbrvfmt{\glscategory{#2}}%
3479   \glsabbrvfont{\Glsentryshort{#2}}#3%
3480 }%
3481 }

GLSxtr@p@short@
3482 \def\@GLSxtr@p@short@#1#2[#3]{%
3483 {%
3484   \glsssetabbrvfmt{\glscategory{#2}}%
3485   \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
3486 }%
3487 }

sxtr@p@shortpl@
3488 \def\@glxtr@p@shortpl@#1#2[#3]{%
3489 {%
3490   \glsssetabbrvfmt{\glscategory{#2}}%
3491   \glsabbrvfont{\glsentryshortpl{#2}}#3%
3492 }%
3493 }

sxtr@p@shortpl@
3494 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
3495 {%
3496   \glsssetabbrvfmt{\glscategory{#2}}%
3497   \glsabbrvfont{\Glsentryshortpl{#2}}#3%
3498 }%
3499 }

Sxtr@p@shortpl@
3500 \def\@GLSxtr@p@shortpl@#1#2[#3]{%

```

```

3501  {%
3502    \glsssetabbrvfmt{\glscategory{#2}}%
3503    \mfirstucMakeUppercase{\glssabbrvfont{\glssentryshortpl{#2}}#3}%
3504  }%
3505 }

@glssxtr@p@long@
3506 \def\@glssxtr@p@long@#1#2[#3]{\glssentrylong{#2}#3}}

@Glsxtr@p@long@
3507 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}}

@GLSxtr@p@long@
3508 \def\@GLSxtr@p@long@#1#2[#3]{%
3509   {\mfirstucMakeUppercase{\glsslongfont{\glssentrylong{#2}}#3}}}

lssxtr@p@longpl@
3510 \def\@lssxtr@p@longpl@#1#2[#3]{\glssentrylongpl{#2}#3}}

Lssxtr@p@longpl@
3511 \def\@Lssxtr@p@longpl@#1#2[#3]{\glsslongfont{\Glsentrylongpl{#2}}#3}}

LSxtr@p@longpl@
3512 \def\@GLSxtr@p@longpl@#1#2[#3]{%
3513   {\mfirstucMakeUppercase{\glsslongfont{\glssentrylongpl{#2}}#3}}}

xtr@p@acrshort@
3514 \def\@glssxtr@p@acrshort@#1#2[#3]{\acronymfont{\glssentryshort{#2}}#3}}

Xtr@p@acrshort@
3515 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
3516 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
3517   {\mfirstucMakeUppercase{\acronymfont{\glssentryshort{#2}}#3}}}

r@p@acrshortpl@
3518 \def\@glssxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glssentryshortpl{#2}}#3}}

R@p@acrshortpl@
3519 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
3520 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
3521   {\mfirstucMakeUppercase{\acronymfont{\glssentryshortpl{#2}}#3}}}

ssxtr@p@acrlong@
3522 \def\@glssxtr@p@acrlong@#1#2[#3]{\glssentrylong{#2}#3}}

```

sxtr@p@acrlong@

```
3523 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}
```

Sxtr@p@acrlong@

```
3524 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
3525 {\mfirstucMakeUppercase{\Glsentrylong{#2}#3}}}
```

tr@p@acrlongpl@

```
3526 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
3527 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
3528 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
3529 {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glsxtrp@opt

```
3530 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}
```

\glsxtrsetpopts Used in glossary to switch hyperlinks on for the \@glsxtrp type of commands.

```
3531 \newcommand*{\glsxtrsetpopts}[1]{%
3532 \renewcommand*{\@glsxtrp@opt}{#1}%
3533 }
```

lossxtrsetpopts Used in glossary to switch hyperlinks on for the \glsxtrp type of commands.

```
3534 \newcommand*{\glossxtrsetpopts}{%
3535 \glsxtrsetpopts{noindex}%
3536 }
```

\@@glsxtrp

```
3537 \newrobustcmd*{\@@glsxtrp}[2]{%
```

Add scope.

```
3538 {%
3539 \let\glspostlinkhook\relax
3540 \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
3541 }%
3542 }
```

\@glsxtrp

```
3543 \newrobustcmd*{\@glsxtrp}[2]{%
3544 \ifcsdef{gls#1}%
3545 {%
3546 \@glsxtrp{gls#1}{#2}%
3547 }%
```

```

3548 {%
3549   \ifcsdef{glsxtr#1}%
3550   {%
3551     \@glsxtrp{glsxtr#1}{#2}%
3552   }%
3553   {%
3554     \PackageError{glossaries-extra}{‘#1’ not recognised by
3555       \string\glsxtrp}{}%
3556   }%
3557 }%
3558 }

```

\@Glsxtrp

```

3559 \newrobustcmd*{\@Glsxtrp}[2]{%
3560   \ifcsdef{Gls#1}%
3561   {%
3562     \@glsxtrp{Gls#1}{#2}%
3563   }%
3564   {%
3565     \ifcsdef{Glsxtr#1}%
3566     {%
3567       \@glsxtrp{Glsxtr#1}{#2}%
3568     }%
3569     {%
3570       \PackageError{glossaries-extra}{‘#1’ not recognised by
3571         \string\Glsxtrp}{}%
3572     }%
3573   }%
3574 }

```

\@GLSxtrp

```

3575 \newrobustcmd*{\@GLSxtrp}[2]{%
3576   \ifcsdef{GLS#1}%
3577   {%
3578     \@glsxtrp{GLS#1}{#2}%
3579   }%
3580   {%
3581     \ifcsdef{GLSxtr#1}%
3582     {%
3583       \@glsxtrp{GLSxtr#1}{#2}%
3584     }%
3585     {%
3586       \PackageError{glossaries-extra}{‘#1’ not recognised by
3587         \string\GLSxtrp}{}%
3588     }%
3589   }%
3590 }

```

\glsxtr@entry@p

```

3591 \newrobustcmd*{\glxtr@headentry@p}[2]{%
3592   \gl@ifattribute{#1}{headuc}{true}%
3593   {%
3594     \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
3595   }%
3596   {%
3597     \@gls@entry@field{#1}{#2}%
3598   }%
3599 }

```

`\glxtrp` Not robust as it needs to expand somewhat.

```

3600 \ifdef\teorpdfstring
3601 {
3602   \newcommand{\glxtrp}[2]{%
3603     \protect\NoCaseChange
3604     {%
3605       \protect\teorpdfstring
3606       {%
3607         \protect\glxtrifinmark
3608         {%
3609           \ifcsdef{glxtrhead#1}%
3610           {%
3611             {\protect\csuse{glxtrhead#1}{#2}}%
3612           }%
3613           {%
3614             \glxtr@headentry@p{#2}{#1}%
3615           }%
3616         }%
3617         {%
3618           \@glxtrp{#1}{#2}%
3619         }%
3620       }%
3621       {%
3622         \protect\@gls@entry@field{#2}{#1}%
3623       }%
3624     }%
3625   }
3626 }
3627 {
3628   \newcommand{\glxtrp}[2]{%
3629     \protect\NoCaseChange
3630     {%
3631       \protect\glxtrifinmark
3632       {%
3633         \ifcsdef{glxtrhead#1}%
3634         {%
3635           {\protect\csuse{glxtrhead#1}}%
3636         }%
3637         {%

```



```

3638         \glxtr@headentry@p{#2}{#1}%
3639     }%
3640 }%
3641 {%
3642     \@glxtrp{#1}{#2}%
3643 }%
3644 }%
3645 }
3646 }

```

Provide short synonyms for the most common option.

\glsp

```

3647 \newcommand*{\glsp}{\glxtrp{short}}

```

\glsp

```

3648 \newcommand*{\glsp}{\glxtrp{text}}

```

\Glsxtrp As above but use first letter upper case (but not for the bookmarks, which can't process \uppercase).

```

3649 \ifdef\texorpdfstring
3650 {
3651     \newcommand{\Glsxtrp}[2]{%
3652         \protect\NoCaseChange
3653         {%
3654             \protect\texorpdfstring
3655             {%
3656                 \protect\glxtrifinmark
3657                 {%
3658                     \ifcsdef{Glsxtrhead#1}%
3659                     {%
3660                         {\protect\csuse{Glsxtrhead#1}{#2}}%
3661                     }%
3662                     {%
3663                         \protect\@Gls@entry@field{#2}{#1}%
3664                     }%
3665                 }%
3666                 {%
3667                     \@Glsxtrp{#1}{#2}%
3668                 }%
3669             }%
3670             {%
3671                 \protect\@gl@entry@field{#2}{#1}%
3672             }%
3673         }%
3674     }
3675 }
3676 {
3677     \newcommand{\Glsxtrp}[2]{%

```

```

3678 \protect\NoCaseChange
3679 {%
3680 \protect\glxtrifinmark
3681 {%
3682 \ifcsdef{Glsxtrhead#1}%
3683 {%
3684 {\protect\csuse{Glsxtrhead#1}}%
3685 }%
3686 {%
3687 \protect\@Gls@entry@field{#2}{#1}%
3688 }%
3689 }%
3690 {%
3691 \@Glsxtrp{#1}{#2}%
3692 }%
3693 }%
3694 }
3695 }

```

\GLSxtrp As above but all upper case (but not for the bookmarks, which can't process \uppercase).

```

3696 \ifdef\texorpdfstring
3697 {
3698 \newcommand{\GLSxtrp}[2]{%
3699 \protect\NoCaseChange
3700 {%
3701 \protect\texorpdfstring
3702 {%
3703 \protect\glxtrifinmark
3704 {%
3705 \ifcsdef{GLSxtr#1}%
3706 {%
3707 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3708 }%
3709 {%
3710 \protect\mfirstucMakeUppercase
3711 {%
3712 \protect\@gls@entry@field{#2}{#1}%
3713 }%
3714 }%
3715 }%
3716 {%
3717 \@GLSxtrp{#1}{#2}%
3718 }%
3719 }%
3720 {%
3721 \protect\@gls@entry@field{#2}{#1}%
3722 }%
3723 }%
3724 }

```

```

3725 }
3726 {
3727   \newcommand{\GLSxtrp}[2]{%
3728     \protect\NoCaseChange
3729     {%
3730       \protect\glxtrifinmark
3731       {%
3732         \ifcsdef{GLSxtr#1}%
3733         {%
3734           {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
3735         }%
3736         {%
3737           \protect\mfirstucMakeUppercase
3738           {%
3739             \protect\@gls@entry@field{#2}{#1}%
3740           }%
3741         }%
3742       }%
3743     }%
3744     \@GLSxtrp{#1}{#2}%
3745   }%
3746 }%
3747 }
3748 }

```

### 1.3.5 Entry Counting

The (use) entry counting mechanism from glossaries is adjusted here to work with category attributes. Provide a convenient command to enable entry counting, set the entrycount attribute for given categories and redefine \gls etc to use \cgl instead. This form of entry counting is provided to adjust the formatting if the number of times an entry has been used (through commands that unset the first use flag) doesn't exceeding the specified threshold. For link counting, see Section 1.4.

First adjust definitions of the unset and reset commands to provide a hook, but changing the flag can cause problems in certain situations, so to allow the normal unsetting to be temporarily disabled, \@glsunset is let to \@glxtr@unset, which performs the actual unsetting through @@glsunset and then does the hook. This means that the unsetting (and the hook) can switched off by redefining \@glsunset and then switched back on again by changing the definition back to \@glxtr@unset.

```

\@glxtr@unset  Global unset.
3749 \newcommand*{\@glxtr@unset}[1]{%
3750   \@glsunset{#1}%
3751   \glxtrpostunset{#1}%
3752 }%

\@glsunset  Global unset.
3753 \let\@glsunset\@glxtr@unset

```

glsxtrpostunset

```
3754 \newcommand*{\glsxtrpostunset}[1]{}
```

Provide a command to store a list of labels that will need unsetting.

tUnsetBuffering

```
3755 \newcommand*{\GlsXtrStartUnsetBuffering}{%
3756   \ifstar\s@GlsXtrStartUnsetBuffering\@GlsXtrStartUnsetBuffering
3757 }
```

tUnsetBuffering Unstarred version doesn't check for duplicates.

```
3758 \newcommand*{\@GlsXtrStartUnsetBuffering}{%
3759   \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3760   \def\@glsxtr@unset@buffer{}%
3761   \let\@glsunset\@glsxtrbuffer@unset
3762 }
```

tUnsetBuffering Starred version checks for duplicates.

```
3763 \newcommand*{\s@GlsXtrStartUnsetBuffering}{%
3764   \let\@glsxtr@org@unset@buffer\@glsxtr@unset@buffer
3765   \def\@glsxtr@unset@buffer{}%
3766   \let\@glsunset\@glsxtrbuffer@nodup@unset
3767 }
```

xtrbuffer@unset This must use a global change since \gls may have to be placed inside \mbox (for example, with soul commands).

```
3768 \newcommand*{\@glsxtrbuffer@unset}[1]{%
3769   \listxadd\@glsxtr@unset@buffer{#1}%
3770 }
```

fer@nodup@unset Alternative version that avoids duplicates. One level of expansion is performed on the argument in case it's a control sequence containing the label. (Not using \xifinlist as the added complexity might cause problems that the buffering is trying to overcome.)

```
3771 \newcommand*{\@glsxtrbuffer@nodup@unset}[1]{%
3772   \expandafter\ifinlist\expandafter{#1}{\@glsxtr@unset@buffer}{}%
3773   {\listxadd\@glsxtr@unset@buffer{#1}}%
3774 }
```

pUnsetBuffering

```
3775 \newcommand*{\GlsXtrStopUnsetBuffering}{%
3776   \ifstar\s@GlsXtrStopUnsetBuffering\@GlsXtrStopUnsetBuffering
3777 }
```

pUnsetBuffering Unstarred form (global unset).

```
3778 \newcommand*{\@GlsXtrStopUnsetBuffering}{%
3779   \let\@glsunset\@glsxtr@unset
3780   \forlistloop\@glsunset\@glsxtr@unset@buffer
3781   \let\@glsxtr@unset@buffer\@glsxtr@org@unset@buffer
3782 }
```

pUnsetBuffering    Starred form (local unset).

```

3783 \newcommand*{\s@GlsXtrStopUnsetBuffering}{%
3784   \forlistloop\@glslocalunset\@glxtr@unset@buffer
3785   \let\@glsunset\@glxtr@unset
3786 }

```

dUnsetBuffering    Discards pending buffer and restores \glsunset.

```

3787 \newcommand*{\GlsXtrDiscardUnsetBuffering}{%
3788   \let\@glsunset\@glxtr@unset
3789   \let\@glxtr@unset@buffer\@glxtr@org@unset@buffer
3790 }

```

setBufferedList    Iterate over labels stored in the current buffer. The argument is the handler macro.

```

3791 \newcommand*{\GlsXtrForUnsetBufferedList}[1]{%
3792   \forlistloop#1\@glxtr@unset@buffer
3793 }

```

\@glslocalunset    Local unset.

```

3794 \renewcommand*{\@glslocalunset}[1]{%
3795   \@glslocalunset{#1}%
3796   \glxtrpostlocalunset{#1}%
3797 }%

```

rpostlocalunset

```

3798 \newcommand*{\glxtrpostlocalunset}[1]{%

```

\@glsreset    Global reset.

```

3799 \renewcommand*{\@glsreset}[1]{%
3800   \@glsreset{#1}%
3801   \glxtrpostreset{#1}%
3802 }%

```

glxtrpostreset

```

3803 \newcommand*{\glxtrpostreset}[1]{%

```

\@glslocalreset    Local reset.

```

3804 \renewcommand*{\@glslocalreset}[1]{%
3805   \@glslocalreset{#1}%
3806   \glxtrpostlocalreset{#1}%
3807 }%

```

rpostlocalreset

```

3808 \newcommand*{\glxtrpostlocalreset}[1]{%

```

slocalreseteach    Locally reset a list of entries.

```

3809 \newcommand*{\glslocalreseteach}[1]{%
3810   \gls@ifnotmeasuring
3811   {%

```

```

3812   \@for\@gls@thislabel:=#1\do{%
3813     \glsdoifexists{\@gls@thislabel}%
3814     {%
3815       \@glslocalreset{\@gls@thislabel}%
3816     }%
3817   }%
3818 }%
3819 }

```

`\glslocalunseteach` Locally unset a list of entries.

```

3820 \newcommand*{\glslocalunseteach}[1]{%
3821   \gls@ifnotmeasuring
3822   {%
3823     \@for\@gls@thislabel:=#1\do{%
3824       \glsdoifexists{\@gls@thislabel}%
3825       {%
3826         \@glslocalunset{\@gls@thislabel}%
3827       }%
3828     }%
3829   }%
3830 }

```

`\glsleEntryCounting` The first argument is the list of categories and the second argument is the value of the `entrycount` attribute.

```

3831 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%

```

Enable entry counting:

```

3832   \glsenableentrycount

```

Redefine `\gls` etc:

```

3833   \renewcommand*{\gls}{\cglsl}%
3834   \renewcommand*{\Gls}{\cGls}%
3835   \renewcommand*{\glspl}{\cglspl}%
3836   \renewcommand*{\Glspl}{\cGlspl}%
3837   \renewcommand*{\GLS}{\cGLS}%
3838   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the `entrycount` attribute:

```

3839   \@glsxtr@setentrycountunsetattr{#1}{#2}%

```

In case this command is used again:

```

3840   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
3841   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
3842     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
3843       can't be used with \string\GlsXtrEnableEntryCounting}%
3844     {Use one or other but not both commands}}%
3845 }

```

`\glsycountunsetattr`

```

3846 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%

```

```

3847 \@for\@glxtr@cat:=#1\do
3848 {%
3849   \ifdefempty{\@glxtr@cat}{}%
3850   {%
3851     \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
3852   }%
3853 }%
3854 }

```

Redefine the entry counting commands to take into account the entrycount attribute.

nableentrycount

```

3855 \renewcommand*\glsenableentrycount{%

```

Enable new fields:

```

3856   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%

```

Just in case the user has switched on the docdef option.

```

3857 \renewcommand*\gls@defdocnewglossaryentry{%
3858   \renewcommand*\newglossaryentry[2]{%
3859     \PackageError{glossaries}{\string\newglossaryentry\space
3860       may only be used in the preamble when entry counting has
3861       been activated}{If you use \string\glsenableentrycount\space
3862       you must place all entry definitions in the preamble not in
3863       the document environment}%
3864   }%
3865 }%

```

New commands to access new fields:

```

3866 \newcommand*\glsentrycurrcount[1]{%
3867   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
3868   {0}{\@gls@entry@field{##1}{currcount}}%
3869 }%
3870 \newcommand*\glsentryprevcount[1]{%
3871   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
3872   {0}{\@gls@entry@field{##1}{prevcount}}%
3873 }%

```

Adjust post unset and reset:

```

3874 \let\@glxtr@entrycount@org@unset\glxtrpostunset
3875 \renewcommand*\glxtrpostunset[1]{%
3876   \@glxtr@entrycount@org@unset{##1}%
3877   \@gls@increment@currcount{##1}%
3878 }%
3879 \let\@glxtr@entrycount@org@localunset\glxtrpostlocalunset
3880 \renewcommand*\glxtrpostlocalunset[1]{%
3881   \@glxtr@entrycount@org@localunset{##1}%
3882   \@gls@local@increment@currcount{##1}%
3883 }%
3884 \let\@glxtr@entrycount@org@reset\glxtrpostreset
3885 \renewcommand*\glxtrpostreset[1]{%

```

```

3886 \@glxtr@entrycount@org@reset{##1}%
3887 \csgdef{glo@glsetoklabel{##1}@currcount}{0}%
3888 }%
3889 \let\@glxtr@entrycount@org@localreset\glxtrpostlocalreset
3890 \renewcommand*{\glxtrpostlocalreset}[1]{%
3891 \@glxtr@entrycount@org@localreset{##1}%
3892 \csdef{glo@glsetoklabel{##1}@currcount}{0}%
3893 }%

```

Modifications to take into account the attributes that govern whether the entry should be unset.

```

3894 \let\@cgl@\@cgl@
3895 \let\@cgl@pl@\@cgl@pl@

3896 \let\@cGl@\@cGl@
3897 \let\@cGl@pl@\@cGl@pl@
3898 \let\@cGL@\@cGL@
3899 \let\@cGL@pl@\@cGL@pl@

```

The rest is as the original definition.

```

3900 \AtEndDocument{\@gl@write@entrycounts}%
3901 \renewcommand*{\@gl@entry@count}[2]{%
3902 \csgdef{glo@glsetoklabel{##1}@prevcount}{##2}%
3903 }%
3904 \let\gl@enableentrycount\relax
3905 \renewcommand*{\gl@enableentrycount}{%
3906 \PackageError{glossaries-extra}{\string\gl@enableentrycount\space
3907 can't be used with \string\gl@enableentrycount}%
3908 {Use one or other but not both commands}%
3909 }%
3910 }

```

`\@gl@write@entrycounts` Modify this command so that it only writes the information for entries with the entrycount attribute and issue warning if no entries have this attribute set.

```

3911 \renewcommand*{\@gl@write@entrycounts}{%
3912 \immediate\write\@auxout
3913 {\string\providecommand*{\string\@gl@entry@count}[2]{}}%
3914 \count@=0\relax
3915 \forallgl@sentries{\@gl@sentry}{%
3916 \gl@hasattribute{\@gl@sentry}{entrycount}%
3917 {%
3918 \ifgl@sused{\@gl@sentry}%
3919 {%
3920 \immediate\write\@auxout
3921 {\string\@gl@entry@count{\@gl@sentry}{\gl@sentrycurrcount{\@gl@sentry}}}%
3922 }%
3923 }%
3924 \advance\count@ by \@ne
3925 }%
3926 }%

```



```

3927 }%
3928 \ifnum\count@=0
3929   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3930   \MessageBreak with \string\glsenableentrycount\space but the
3931   \MessageBreak attribute ‘entrycount’ hasn’t
3932   \MessageBreak been assigned to any of the defined
3933   \MessageBreak entries}%
3934 \fi
3935 }

```

rifcounttrigger

`\glxtrifcounttrigger{<label>}{<trigger format>}{<normal>}`

```

3936 \newcommand*{\glxtrifcounttrigger}[3]{%
3937   \glshasattribute{#1}{entrycount}%
3938   {%
3939     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
3940     #3%
3941   \else
3942     #2%
3943   \fi
3944 }%
3945 {#3}%
3946 }

```

Actual internal definitions of `\cgl`s used when entry counting is enabled.

`\@@cgl@s@`

```

3947 \def\@@cgl@s@#1#2[#3]{%
3948   \glxtrifcounttrigger{#2}%
3949   {%
3950     \cgl sformat{#2}{#3}%
3951     \gl sunset{#2}%
3952   }%
3953   {%
3954     \@cgl@s@{#1}{#2}[#3]%
3955   }%
3956 }%

```

`\@@cgl spl@`

```

3957 \def\@@cgl spl@#1#2[#3]{%
3958   \glxtrifcounttrigger{#2}%
3959   {%
3960     \cgl splformat{#2}{#3}%
3961     \gl sunset{#2}%
3962   }%

```

```

3963  {%
3964    \@glsp1@{#1}-{#2}[#3]%
3965  }%
3966 }%

```

\@@cGls@

```

3967 \def\@@cGls@#1#2[#3]{%
3968   \glxtrifcounttrigger{#2}%
3969   {%
3970     \cGlsformat{#2}{#3}%
3971     \glunset{#2}%
3972   }%
3973   {%
3974     \@Gls@{#1}-{#2}[#3]%
3975   }%
3976 }%

```

\@@cGlsp1@

```

3977 \def\@@cGlsp1@#1#2[#3]{%
3978   \glxtrifcounttrigger{#2}%
3979   {%
3980     \cGlsp1format{#2}{#3}%
3981     \glunset{#2}%
3982   }%
3983   {%
3984     \@Glsp1@{#1}-{#2}[#3]%
3985   }%
3986 }%

```

\@@cGLS@

```

3987 \def\@@cGLS@#1#2[#3]{%
3988   \glxtrifcounttrigger{#2}%
3989   {%
3990     \cGLSformat{#2}{#3}%
3991     \glunset{#2}%
3992   }%
3993   {%
3994     \@GLS@{#1}-{#2}[#3]%
3995   }%
3996 }%

```

\@@cGLSp1@

```

3997 \def\@@cGLSp1@#1#2[#3]{%
3998   \glxtrifcounttrigger{#2}%
3999   {%
4000     \cGLSp1format{#2}{#3}%
4001     \glunset{#2}%
4002   }%
4003   {%

```

```

4004 \@GLSp1@{#1}-{#2}[#3]%
4005 }%
4006 }%

```

Remove default warnings from \cgl's etc so that it can be used interchangeable with \gls etc.

```

\@cgl's@
4007 \def\@cgl's@#1#2[#3]{\@gls@{#1}-{#2}[#3]}

```

```

\@cGls@
4008 \def\@cGls@#1#2[#3]{\@Gls@{#1}-{#2}[#3]}

```

```

\@cgl'spl@
4009 \def\@cgl'spl@#1#2[#3]{\@glspl@{#1}-{#2}[#3]}

```

```

\@cGlspl@
4010 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}-{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

```

\cGLS
4011 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}

```

\@cGLS Defined the un-starred form. Need to determine if there is a final optional argument

```

4012 \newcommand*{\@cGLS}[2][\@cGLS@{#1}-{#2}[#3]]{%
4013 \new@ifnextchar[\@cGLS@{#1}-{#2}]{\@cGLS@{#1}-{#2}[#3]}%
4014 }

```

```

\@cGLS@
4015 \def\@cGLS@#1#2[#3]{\@GLS@{#1}-{#2}[#3]}

```

\cGLSformat Format used by \cGLS if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```

4016 \newcommand*{\cGLSformat}[2]{%
4017 \expandafter\mfirstucMakeUppercase\expandafter{\cgl'sformat{#1}-{#2}}%
4018 }

```

```

\cGLSp1
4019 \newrobustcmd*{\cGLSp1}{\@gls@hyp@opt\@cGLSp1}

```

\@cGLSp1 Defined the un-starred form. Need to determine if there is a final optional argument

```

4020 \newcommand*{\@cGLSp1}[2][\@cGLSp1@{#1}-{#2}[#3]]{%
4021 \new@ifnextchar[\@cGLSp1@{#1}-{#2}]{\@cGLSp1@{#1}-{#2}[#3]}%
4022 }

```

```

\@cGLSp1@
4023 \def\@cGLSp1@#1#2[#3]{\@GLSp1@{#1}-{#2}[#3]}

```

`\cGLSplformat` Format used by `\cGLSpl` if entry only used once on previous run. The first argument is the label, the second argument is the insert text.

```
4024 \newcommand*{\cGLSplformat}[2]{%
4025   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
4026 }
```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
4027 \renewcommand*{\cglformat}[2]{%
4028   \glsifregular{#1}
4029   {\glsentryfirst{#1}}%
4030   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4031 }
```

`\cGlsformat`

```
4032 \renewcommand*{\cGlsformat}[2]{%
4033   \glsifregular{#1}
4034   {\Glsentryfirst{#1}}%
4035   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4036 }
```

`\cglsplformat`

```
4037 \renewcommand*{\cglsplformat}[2]{%
4038   \glsifregular{#1}
4039   {\glsentryfirstplural{#1}}%
4040   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
4041 }
```

`\cGlsplformat`

```
4042 \renewcommand*{\cGlsplformat}[2]{%
4043   \glsifregular{#1}
4044   {\Glsentryfirstplural{#1}}%
4045   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%#2%
4046 }
```

New code similar to above for unit counting.

`defunitcounters`

```
4047 \newcommand*{\@newglossaryentry@defunitcounters}{%
4048   \protected@edef\@glo@countunit{\csuse{@glxtr@categoryattr\@glo@category @unitcount}}%
4049   \ifdefvoid\@glo@countunit
4050   {}%
4051   {%
4052     \@glxtr@ifunitcounter{\@glo@countunit}%
4053     {}%
4054     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
4055   }%
4056 }
```

r@unitcountlist List to keep track of which counters are being used by the entry unit count facility.

```
4057 \newcommand*{\@glsxtr@unitcountlist}{}
```

@addunitcounter

```
4058 \newcommand*{\@glsxtr@addunitcounter}[1]{%
4059 \listadd{\@glsxtr@unitcountlist}{#1}%
4060 \ifcsundef{glsxtr@theunit@#1}
4061 {%
4062 \ifcsdef{theH#1}%
4063 {\csdef{glsxtr@theunit@#1}{\csuse{theH#1}}}%
4064 {\csdef{glsxtr@theunit@#1}{\csuse{the#1}}}%
4065 }%
4066 {}%
4067 }
```

r@ifunitcounter

```
4068 \newcommand*{\@glsxtr@ifunitcounter}[3]{%
4069 \xifinlist{#1}{\@glsxtr@unitcountlist}{#2}{#3}%
4070 }
```

urrentunitcount

```
4071 \newcommand*\@glsxtr@currentunitcount[1]{%
4072 glo@\glsdetoklabel{#1}@currunit@glsggetattribute{#1}{unitcount}.%
4073 \csuse{glsxtr@theunit@\glsggetattribute{#1}{unitcount}}%
4074 }
```

eviousunitcount

```
4075 \newcommand*\@glsxtr@previousunitcount[1]{%
4076 glo@\glsdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
4077 \csuse{glsxtr@theunit@\glsggetattribute{#1}{unitcount}}%
4078 }
```

t@currunitcount

```
4079 \newcommand*{\@gls@increment@currunitcount}[1]{%
4080 \glshasattribute{#1}{unitcount}%
4081 {%
4082 \protected@edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
4083 \ifcsundef{\@glsxtr@csname}%
4084 {%
4085 \csgdef{\@glsxtr@csname}{1}%
4086 \listcsxadd
4087 {glo@\glsdetoklabel{#1}@unitlist}%
4088 {\glsggetattribute{#1}{unitcount}.%
4089 \csuse{glsxtr@theunit@\glsggetattribute{#1}{unitcount}}%
4090 }%
4091 }%
4092 {%
4093 \csxdef{\@glsxtr@csname}%

```

```

4094      {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
4095    }%
4096  }%
4097  {}%
4098 }

```

t@currunitcount

```

4099 \newcommand*{\@glx@local@increment@currunitcount}[1]{%
4100   \glshasattribute{#1}{unitcount}%
4101   {%
4102     \protected@edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
4103     \ifcsundef{\@glxtr@csname}%
4104     {%
4105       \csdef{\@glxtr@csname}{1}%
4106       \listcseadd
4107         {glo@\glsdetoklabel{#1}@unitlist}%
4108         {\glsggetattribute{#1}{unitcount}.%
4109          \csuse{glxtr@theunit@\glsggetattribute{#1}{unitcount}}}%
4110       }%
4111     }%
4112     {%
4113       \csedef{\@glxtr@csname}%
4114         {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
4115       }%
4116     }%
4117   }%
4118 }

```

r@currunitcount

```

4119 \newcommand*{\@glxtr@currunitcount}[2]{%
4120   \ifcsundef
4121     {glo@\glsdetoklabel{#1}@currunit@#2}%
4122     {0}%
4123     {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
4124 }%

```

r@prevunitcount

```

4125 \newcommand*{\@glxtr@prevunitcount}[2]{%
4126   \ifcsundef
4127     {glo@\glsdetoklabel{#1}@prevunit@#2}%
4128     {0}%
4129     {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
4130 }%

```

eentryunitcount

```

4131 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
4132   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%

```

Just in case the user has switched on the docdef option.

```

4133 \renewcommand*{\gls@defdocnewglossaryentry}{%
4134 \renewcommand*\newglossaryentry[2]{%
4135 \PackageError{glossaries}{\string\newglossaryentry\space
4136 may only be used in the preamble when entry counting has
4137 been activated}{If you use \string\glsenableentryunitcount\space
4138 you must place all entry definitions in the preamble not in
4139 the document environment}%
4140 }%
4141 }%

```

New commands to access new fields:

```

4142 \newcommand*\glsentrycurrcount[1]{%
4143 \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
4144 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
4145 }%
4146 \newcommand*\glsentryprevcount[1]{%
4147 \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
4148 \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
4149 }%

```

Access total count:

```

4150 \newcommand*\glsentryprevtotalcount[1]{%
4151 \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
4152 {0}%
4153 {%
4154 \number\csuse{glo@\glsdetoklabel{##1}@prevunittotal}
4155 }%
4156 }%

```

Access max value:

```

4157 \newcommand*\glsentryprevmaxcount[1]{%
4158 \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
4159 {0}%
4160 {%
4161 \number\csuse{glo@\glsdetoklabel{##1}@prevunitmax}
4162 }%
4163 }%

```

Adjust post unset and reset:

```

4164 \let\@glsxtr@entryunitcount@org@unset\glsxtrpostunset
4165 \renewcommand*\glsxtrpostunset[1]{%
4166 \@glsxtr@entryunitcount@org@unset{##1}%
4167 \@gls@increment@currunitcount{##1}%
4168 }%
4169 \let\@glsxtr@entryunitcount@org@localunset\glsxtrpostlocalunset
4170 \renewcommand*\glsxtrpostlocalunset[1]{%
4171 \@glsxtr@entryunitcount@org@localunset{##1}%
4172 \@gls@local@increment@currunitcount{##1}%
4173 }%
4174 \let\@glsxtr@entryunitcount@org@reset\glsxtrpostreset

```

```

4175 \renewcommand*{\glstrpostreset}[1]{%
4176   \glshasattribute{##1}{unitcount}%
4177   {%
4178     \protected@edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
4179     \ifcsundef{\@glstr@csname}%
4180     {}%
4181     {\csgdef{\@glstr@csname}{0}}%
4182   }%
4183   {}%
4184 }%
4185 \let\@glstr@entryunitcount@org@localreset\glstrpostlocalreset
4186 \renewcommand*{\glstrpostlocalreset}[1]{%
4187   \@glstr@entryunitcount@org@localreset{##1}%
4188   \glshasattribute{##1}{unitcount}%
4189   {%
4190     \protected@edef\@glstr@csname{\@glstr@currentunitcount{##1}}%
4191     \ifcsundef{\@glstr@csname}%
4192     {}%
4193     {\csdef{\@glstr@csname}{0}}%
4194   }%
4195   {}%
4196 }%

```

Modifications to take into account the attributes that govern whether the entry should be unset.

```

4197 \let\@cgl@\@cgl@
4198 \let\@cgl@pl@\@cgl@pl@
4199 \let\@cgl@\@cgl@
4200 \let\@cgl@pl@\@cgl@pl@
4201 \let\@cgl@\@cgl@
4202 \let\@cgl@pl@\@cgl@pl@

```

Write information to the aux file.

```

4203 \AtEndDocument{\@gl@write@entryunitcounts}%
4204 \renewcommand*{\@gl@entry@unitcount}[3]{%
4205   \csgdef{glo@gl@detoklabel{##1}@prevunit@##3}{##2}%
4206   \ifcsundef{glo@gl@detoklabel{##1}@prevunittotal}%
4207   {\csgdef{glo@gl@detoklabel{##1}@prevunittotal}{##2}}%
4208   {%
4209     \csxdef{glo@gl@detoklabel{##1}@prevunittotal}{
4210       \number\numexpr\csuse{glo@gl@detoklabel{##1}@prevunittotal}+##2}%
4211     }%
4212     \ifcsundef{glo@gl@detoklabel{##1}@prevunitmax}%
4213     {\csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}}%
4214     {%
4215       \ifnum\csuse{glo@gl@detoklabel{##1}@prevunitmax}<##2
4216       \csgdef{glo@gl@detoklabel{##1}@prevunitmax}{##2}%

```



```

4217     \fi
4218   }%
4219 }%
4220 \let\glsenableentryunitcount\relax
4221 \renewcommand*{\glsenableentrycount}{%
4222   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
4223     can't be used with \string\glsenableentryunitcount}%
4224   {Use one or other but not both commands}%
4225 }%
4226 }
4227 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

4228 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

4229 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
4230   \immediate\write\@auxout
4231   {\string\@gls@entry@unitcount
4232     {\@glsentry}%
4233     {\@glsxtr@currunitcount{\@glsentry}{#1}%
4234     }%
4235     {#1}}%
4236 }

```

entryunitcounts

```

4237 \newcommand*{\@gls@write@entryunitcounts}{%
4238   \immediate\write\@auxout
4239   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
4240   \count@=0\relax
4241   \forallglsentries{\@glsentry}{%
4242     \gls@hasattribute{\@glsentry}{unitcount}%
4243     {%
4244       \ifglsused{\@glsentry}%
4245       {%
4246         \forlistcsloop
4247           {\@gls@write@entryunitcounts@do}%
4248           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
4249       }%
4250     }%
4251     \advance\count@ by \@ne
4252   }%
4253 }%
4254 }%
4255 \ifnum\count@=0
4256   \GlossariesExtraWarningNoLine{Entry counting has been enabled
4257     \MessageBreak with \string\glsenableentryunitcount\space but the
4258     \MessageBreak attribute 'unitcount' hasn't
4259     \MessageBreak been assigned to any of the defined

```

```

4260     \MessageBreak entries}%
4261 \fi
4262 }

```

`tryUnitCounting` The first argument is the list of categories, the second argument is the value of the entrycount attribute and the third is the counter name.

```

4263 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

4264 \glsenableentryunitcount

```

Redefine `\gls` etc:

```

4265 \renewcommand*{\gls}{\cglsl}%
4266 \renewcommand*{\Gls}{\cGls}%
4267 \renewcommand*{\glspl}{\cglspl}%
4268 \renewcommand*{\Glspl}{\cGlspl}%
4269 \renewcommand*{\GLS}{\cGLS}%
4270 \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

4271 \@glsextr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

4272 \let\GlsXtrEnableEntryUnitCounting\@glsextr@setentryunitcountunsetattr
4273 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
4274 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
4275 can't be used with \string\GlsXtrEnableEntryUnitCounting}%
4276 {Use one or other but not both commands}}%
4277 }

```

`countunsetattr`

```

4278 \newcommand*{\@glsextr@setentryunitcountunsetattr}[3]{%
4279 \@for\@glsextr@cat:=#1\do
4280 {%
4281 \ifdefempty{\@glsextr@cat}{}%
4282 {%
4283 \glsssetcategoryattribute{\@glsextr@cat}{entrycount}{#2}%
4284 \glsssetcategoryattribute{\@glsextr@cat}{unitcount}{#3}%
4285 }%
4286 }%
4287 }

```

### 1.3.6 Acronym Modifications

It's more consistent to use the abbreviation code for acronyms, but make some adjustments to allow for continued use of the glossaries package's custom acronym format. (For example, user may already have defined some acronym styles with `\newacronymstyle` which they would like to continue to use.) The original glossaries acronym code can be restored with `\RestoreAcronyms`, but adjust `\SetGenericNewAcronym` so that `\newacronym` adds the category.

nericNewAcronym

```
4288 \renewcommand*{\SetGenericNewAcronym}{%
    Make sure \RestoreAcronyms has been used.
4289 \ifdefequal\@addtoacronymlists\@glstr@org@addtoacronymlists
4290 {}%
4291 {%
4292     \GlossariesWarning{\string\SetGenericNewAcronym\space used
4293     without restoring base acronym functions with
4294     \string\RestoreAcronyms}%
4295 }%
4296 \let\@Gls@entryname\@Gls@acronymentryname

    Redefine \newacronym:
4297 \renewcommand{\newacronym}[4][{}]{%
4298     \ifdefempty{\@glssacronymlists}%
4299     {%
4300         \def\@glo@type{\acronymtype}%
4301         \setkeys{glossentry}{##1}%
4302         \DeclareAcronymList{\@glo@type}%
4303     }%
4304     {}%
4305     \glskeylisttok{##1}%
4306     \glslabeltok{##2}%
4307     \glsshorttok{##3}%
4308     \glslongtok{##4}%
4309     \newacronymhook
4310     \protected@edef\@do@newglossaryentry{%
4311         \noexpand\newglossaryentry{\the\glslabeltok}%
4312         {%
4313             type=\acronymtype,%
4314             name={\expandonce{\acronymentry{##2}}},%
4315             sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%
4316             text={\the\glsshorttok},%
4317             short={\the\glsshorttok},%
4318             shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
4319             long={\the\glslongtok},%
4320             longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
4321             category=acronym,%
4322             \GenericAcronymFields,%
4323             \the\glskeylisttok
4324         }%
4325     }%
4326     \@do@newglossaryentry
4327 }%
4328 \renewcommand*{\acrfullfmt}[3]{%
4329     \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}%
4330 \renewcommand*{\Acrfullfmt}[3]{%
4331     \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}%
4332 \renewcommand*{\ACRfullfmt}[3]{%
```

```

4333 \glslink{##1}{##2}{%
4334 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
4335 \renewcommand*{\acrfullplfmt}[3]{%
4336 \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
4337 \renewcommand*{\Acrfullplfmt}[3]{%
4338 \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
4339 \renewcommand*{\ACRfullplfmt}[3]{%
4340 \glslink{##1}{##2}{%
4341 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
4342 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
4343 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
4344 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
4345 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
4346 }

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbreviations, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```

4347 \let\@glxstr@org@setacronymstyle\setacronymstyle
4348 \let\@glxstr@org@newacronymstyle\newacronymstyle

```

Save the list of acronyms in case they are required.

tr@acronymlists

```

4349 \let\@glxstr@acronymlists\@glsacronymlists

```

doacronymlists

```

4350 \let\@glxstr@org@addtoacronymlists\@addtoacronymlists

```

setacronymlists

```

4351 \let\@glxstr@org@setacronymlists\SetAcronymLists

```

Need to provide a replacement for `\forallacronyms` since `\@glsacronymlists` isn't available.

lsxtr@abbrlists

```

4352 \newcommand{\@glxstr@abbrlists}{}

```

breivationlists

```

4353 \newcommand*{\forallabbreviationlists}[2]{%
4354 \@for#1:=\@glxstr@abbrlists\do{\ifdefempty{#1}{-}{#2}}%
4355 }

```

bbreviationlist

```

4356 \newcommand*{\@glxstr@addabbreviationlist}[1]{%
4357 \protected@edef\@glo@type{#1}%
4358 \ifdefempty\@glxstr@abbrlists
4359 {\let\@glxstr@abbrlists\@glo@type}%
4360 }

```

```

4361 \ifdefequal\@glstr@abbrlists\@glo@type
4362 {}%
4363 {%

4364 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glstr@abbrlists}{}%
4365 {\protected@eappto\@glstr@abbrlists{,\@glo@type}}%
4366 }%
4367 }%
4368 }

```

\forallacronyms Modify to add warning.

```

4369 \renewcommand*{\forallacronyms}[2]{%
4370 \@glstr@base@acrcmd\forallacronyms\forallabbreviationlists
4371 \@for#1:=\@glstr@acronymlists\do{\ifx#1\@empty\else#2\fi}%
4372 }

```

msAbbreviations Make acronyms use the same interface as abbreviations. Note that \newacronymstyle has a different implementation to \newabbreviationstyle so disable \newacronymstyle and \setacronymstyle.

```

4373 \newcommand*{\MakeAcronymsAbbreviations}{%

```

Undo acronym display style:

```

4374 \@for\@glstr@type:=\@glstr@acronymlists\do{%
4375 \csgdef{glstr\@glstr@type @entryfmt}{\glstr@entryfmt}%
4376 }%

```

Save and clear acronym list.

```

4377 \let\@glstr@acronymlists\@glstr@acronymlists
4378 \let\@glstr@acronymlists\@empty
4379 \let\@addtoacronymlists\@gobble
4380 \let\SetAcronymLists\@gobble

```

Warn if \acrshort etc are used.

```

4381 \let\@glstr@base@acrcmd\@glstr@base@acrcmd@warn

```

Redefine \newacronym to use same interface as \newabbreviation.

```

4382 \renewcommand*{\newacronym}[4][{}]{%
4383 \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
4384 }%
4385 \renewcommand*{\firstacronymfont}[1]{\glstr@firstabbrvfont{##1}}%
4386 \renewcommand*{\acronymfont}[1]{\glstr@abbrvfont{##1}}%
4387 \renewcommand*{\setacronymstyle}[1]{%
4388 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
4389 unavailable.
4390 Use \string\setabbreviationstyle[acronym]\space instead.
4391 The original acronym interface can be restored with
4392 \string\RestoreAcronyms}{}%
4393 }%
4394 \renewcommand*{\newacronymstyle}[1]{%
4395 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
4396 available unless you restore the original acronym interface with

```

```

4397 \string\RestoreAcronyms}%
4398 \@glxtr@org@newacronymstyle{##1}%
4399 }%
4400 }

```

Switch acronyms to abbreviations:

```
4401 \MakeAcronymsAbbreviations
```

**RestoreAcronyms** Restore acronyms to glossaries interface.

```

4402 \newcommand*\RestoreAcronyms{%
  Restore acronym list.
4403 \let\@glsacronymlists\@glxtr@acronymlists
4404 \let\@addtoacronymlists\@glxtr@org@addtoacronymlists
4405 \let\SetAcronymLists\@glxtr@org@setacronymlists
  Suppress warnings if \acrshort etc are used.
4406 \let\@glxtr@base@acrcmd\@gobbletwo
  Restore acronym display style:
4407 \@for\@gls@type:=\@glsacronymlists\do{%
4408   \SetDefaultAcronymDisplayStyle{\@gls@type}%
4409 }%

```

Switch to the generic acronym mechanism.

```

4410 \SetGenericNewAcronym
4411 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
4412 \renewcommand{\acronymfont}[1]{##1}%
4413 \let\setacronymstyle\@glxtr@org@setacronymstyle
4414 \let\newacronymstyle\@glxtr@org@newacronymstyle

```

Need to restore the original definition of \@gls@link@checkfirsthyper but \glxtrifwasfirstuse still needs setting for the benefit of the post-link hook.

```

4415 \renewcommand*\@gls@link@checkfirsthyper{%
4416   \ifglsused{\glslabel}%
4417   {\let\glxtrifwasfirstuse\@secondoftwo}
4418   {\let\glxtrifwasfirstuse\@firstoftwo}%
4419   \@glxtr@org@checkfirsthyper
4420 }
4421 \glssetcategoryattribute{acronym}{regular}{false}%
4422 \setacronymstyle{long-short}%
4423 }

```

**\glsacspace** Allow the user to customise the maximum value.

```

4424 \renewcommand*\glsacspace[1]{%
4425   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
4426   \ifdim\dimen@<\glsacspacemax~\else\space\fi
4427 }

```

**\glsacspacemax** Value used in the above.

```
4428 \newcommand*\glsacspacemax{3em}
```

### 1.3.7 Indexing and Displaying Glossaries

From time-to-time users ask if they can have one glossary sorted normally and another sorted by definition or usage. With the base glossaries package this can only be achieved with the “noidx” commands (Option 1). This is an attempt to mix and match.

First we need a list of the glossaries that require `makeindex/xindy`.

r@reg@glosslist

```
4429 \newcommand*{\@glxtr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```
4430 \let\@glxtr@org@makeglossaries\makeglossaries
```

noprintglossary This command was only introduced to glossaries v4.47 so it may not be defined.

```
4431 \providecommand\@makeglossaries@warn@noprintglossary{%
4432   \ifdefstring{\@glo@types}{,}%
4433   {%
4434     \GlossariesWarningNoLine{No glossaries have been defined}%
4435   }%
4436   {%
4437     \GlossariesWarningNoLine{No \string\printglossary\space
4438       or \string\printglossaries\space
4439       found. ^^J(Remove \string\makeglossaries\space if you
4440       don't want any glossaries.) ^^JThis document will not
4441       have a glossary}%
4442   }%
4443 }%
4444 %   \begin{macrocode}
4445 \end{macro}
4446 %
4447 %\begin{macro}{\@domakeglossaries}
4448 %\changes{1.42}{2020-02-03}{provided definition for \cs{@domakeglossaries}}
4449 % \sty{glossaries} v4.45 introduced \cs{@domakeglossaries} to
4450 % provide a way of disabling \cs{makeglossaries}. If it hasn't been
4451 % defined, define here to do its argument:
4452 %   \begin{macrocode}
4453 \providecommand{\@domakeglossaries}[1]{#1}
```

Redefine `\makeglossaries` to take an optional argument. This should be empty for the usual behaviour (all glossaries need processing with an indexing application) or a comma-separated list of glossary labels indicating those glossaries that should be processed with an indexing application. The optional argument version shouldn't be used with record.

\makeglossaries

```
4454 \renewcommand*{\makeglossaries}[1][]{%
4455   \@domakeglossaries
4456   {%
4457     \@glxtr@if@record@only
4458     {%
```

```

4459 \PackageError{glossaries-extra}{\string\makeglossaries\space
4460 not permitted\MessageBreak with record=\@glxtr@record@setting\space
4461 package option}%
4462 {You may only use \string\makeglossaries\space with
4463 record=off or record=hybrid options}%
4464 }%
4465 {%
4466 \ifblank{#1}%
4467 {%
4468 \@glxtr@org@makeglossaries

4469 \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4470 \let\warn@noprintglossary\@glxtr@warn@hybrid@noprintgloss
4471 \fi
4472 }%
4473 {%
4474 \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
4475 \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
4476 not permitted\MessageBreak with record=\@glxtr@record@setting\space package option}%
4477 {You may only use the hybrid \string\makeglossaries[...]\space with
4478 record=off option}%
4479 \else

```

\@gls@@automake@immediate was introduced to glossaries v4.42 so it may not be defined.

```

4480 \ifdef\@gls@@automake@immediate{\@gls@@automake@immediate}{}%
4481 \protected@edef\@glxtr@reg@glosslist{#1}%
4482 \ifundef{\glswrite}{\newwrite\glswrite}{}%
4483 \protected@write\@auxout{}{\string\providecommand
4484 \string\@glsorder[1]{}%
4485 \protected@write\@auxout{}{\string\providecommand
4486 \string\@istfilename[1]{}%
4487 \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
4488 \protected@write\@auxout{}{\string\@glsorder{\glsorder}}%
4489 \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}%
4490 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}%

```

Iterate through each supplied glossary type and activate it.

```

4491 \@for\@glo@type:=#1\do{%
4492 \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
4493 }%

```

New glossaries must be created before \makeglossaries:

```

4494 \renewcommand*\newglossary[4][{}]{%
4495 \PackageError{glossaries}{New glossaries
4496 must be created before \string\makeglossaries}{You need
4497 to move \string\makeglossaries\space after all your
4498 \string\newglossary\space commands}}%

```

Any subsequence instances of this command should have no effect.

```

4499 \let\@makeglossary\@gobble

```



Version 1.42 removed letting \makeglossary to \relax (no kernel redefs may be in effect).

```
4500 \renewcommand\makeglossaries[1] [] {}%
```

Disable all commands that have no effect after \makeglossaries

```
4501 \@disable@onlypremakeg
```

Allow see key:

```
4502 \let\gls@checkseeallowed\relax
```

Adjust \@do@seeglossary. This needs to check for the entry's existence but don't increment associated counter.

```
4503 \renewcommand*\@do@seeglossary}[2] {%
```

```
4504 \glsdoifexists{##1}%
```

```
4505 {%
```

```
4506 \protected@edef\@gls@label{\glsdetoklabel{##1}}%
```

```
4507 \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
```

```
4508 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
```

```
4509 {\@glsxtr@org@doseeglossary{##1}{##2}}%
```

```
4510 {%
```

```
4511 \@glsxtrwrglossmark
```

```
4512 \protected@write\@auxout{}{%
```

```
4513 \string\@gls@reference
```

```
4514 {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
```

```
4515 }%
```

```
4516 }%
```

```
4517 }%
```

```
4518 }%
```

Adjust \@do@@wrglossary

```
4519 \let\@glsxtr@@do@@wrglossary\@do@@wrglossary
```

```
4520 \def\@do@@wrglossary{%
```

```
4521 \protected@edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
```

```
4522 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
```

```
4523 {\@glsxtr@@do@@wrglossary}%
```

```
4524 {\@gls@noidxglossary}%
```

```
4525 }%
```

Suppress warning about no \makeglossaries

```
4526 \let\warn@nomakeglossaries\relax
```

```
4527 \let\warn@noprntglossary\@makeglossaries@warn@noprntglossary
```

Only warn for glossaries not listed.

```
4528 \renewcommand{\@gls@noref@warn}[1] {%
```

```
4529 \protected@edef\@gls@type{##1}%
```

```
4530 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
```

```
4531 {%
```

```
4532 \GlossariesExtraWarning{Can't use
```

```
4533 \string\printnoidxglossary[type={\@gls@type}]
```

```
4534 when '\@gls@type' is listed in the optional argument of
```

```

4535         \string\makeglossaries}%
4536     }%
4537     {%
4538         \GlossariesWarning{Empty glossary for
4539         \string\printnoidxglossary[type={##1}].
4540         Rerun may be required (or you may have forgotten to use
4541         commands like \string\gls)}%
4542     }%
4543 }%

```

Adjust display number list to check for type:

```

4544 \renewcommand*{\glsdisplaynumberlist}[1]{%
4545     \expandafter\DTLifinlist\expandafter{##1}{\@glstr@reg@glosslist}%
4546     {\@glstr@idx@displaynumberlist{##1}}%
4547     {\@glstr@noidx@displaynumberlist{##1}}%
4548 }%

```

Adjust entry list:

```

4549 \renewcommand*{\glsentrynumberlist}[1]{%
4550     \expandafter\DTLifinlist\expandafter{##1}{\@glstr@reg@glosslist}%
4551     {\@glstr@idx@entrynumberlist{##1}}%
4552     {\@glstr@noidx@entrynumberlist{##1}}%
4553 }%

```

Adjust number list loop

```

4554 \renewcommand*{\glsnumberlistloop}[2]{%
4555     \expandafter\DTLifinlist\expandafter{##1}{\@glstr@reg@glosslist}%
4556     {%
4557         \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
4558         not available for glossary ‘##1’}{}%
4559     }%
4560     {\@glstr@noidx@numberlistloop{##1}{##2}}%
4561 }%

```

Only sanitize sort for normal indexing glossaries.

```

4562 \renewcommand*{\glsprestandardsort}[3]{%
4563     \expandafter\DTLifinlist\expandafter{##2}{\@glstr@reg@glosslist}%
4564     {%
4565         \glsdosanitizesort
4566     }%
4567     {%
4568         \ifglssanitizesort
4569         \@gls@noidx@sanitizesort
4570     \else
4571         \@gls@noidx@nosanitizesort
4572     \fi
4573 }%
4574 }%

```

Unlike `\makenoidxglossaries` we can't automatically set `sanitizesort=false`. All entries must be defined in the preamble.

```

4575         \renewcommand*\new@glossaryentry[2]{%
4576             \PackageError{glossaries-extra}{Glossary entries must be defined
4577                 in the preamble\MessageBreak when you use the optional argument
4578                 of \string\makeglossaries}{Either move your definitions to the
4579                 preamble or don't use the optional argument of
4580                 \string\makeglossaries}}%
4581     }%

```

Only activate sort key for glossaries that aren't listed in #1 (glossary label is stored in `\@glo@type` but this defaults to `\glsdefaulttype` so some expansion is required).

```

4582         \let\@glo@assign@sortkey\@glstr@mixed@assign@sortkey
4583         \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

4584             \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
4585             type=\glsdefaulttype,\@end@glstr@gettype
4586             \def\@glo@sorttype{\@glo@default@sorttype}%
4587         }%

```

Check automake setting:

```

4588         \ifglautomake
4589             \renewcommand*\@glstr@doautomake{%
4590                 \@for\@glstr@type:=\@glstr@reg@glosslist\do{%
4591                     \ifdefempty{\@glstr@type}{\@glstr@automake{\@glstr@type}}%
4592                 }%
4593             }%
4594         \fi

```

Check the sort setting (glossaries v4.30 onwards):

```

4595         \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
4596     \fi
4597 }%
4598 }%
4599 }%
4600 }

```

The optional argument version of `\makeglossaries` needs an adjustment to `\@printglossary` to allow `\@glo@assign@sortkey` to pick up the glossary type.

Earlier versions of `glossaries-extra` simply saved the original version of `\@printglossary` with `\let \@glstr@orgprintglossary`. This was later changed to actually defining `\@glstr@orgprintglossary` to something similar with some alterations to allow for ignored glossaries, which don't have an associated title and to by-pass the existence check with `\ifglossaryexists` which doesn't recognise ignored glossaries. (`bib2gls` writes `\provideignoredglossary` to the `glstex` file for some settings, so the glossary might not been defined on the first  $\LaTeX$  run and it needs to be allowed with `\printunsortedglossary` on subsequent runs.)

Unfortunately, removing the existence check will cause an error if `\printglossary` is used with an ignored glossary.

As from `glossaries v4.46`, some new commands have been included to allow the existence check to be varied depending on whether or not ignored glossaries should be allowed, so check for them:

oss@checkexists

```
4601 \ifdef\@printgloss@checkexists
4602 {\newcommand{\glxtr@printgloss@checkexists}{\@printgloss@checkexists}}
4603 {\newcommand{\glxtr@printgloss@checkexists}[2]{#2}}
```

rgprintglossary (This command is also used for on-the-fly setting.)

```
4604 \newcommand{\@glxtr@orgprintglossary}[2]{%
4605   \def\@glo@type{\glsdefaulttype}%

  Add check here.

4606   \def\glossarytitle{%
4607     \ifcsdef{\@glo@type\@glo@type @title}%
4608       {\csuse{\@glo@type\@glo@type @title}}%
4609       {\glossaryname}}%
4610   \def\glossarytoctitle{\glossarytitle}%
4611   \let\org@glossarytitle\glossarytitle
4612   \def\@glossarystyle{%
4613     \ifx\@glossary@default@style\relax
4614       \GlossariesWarning{No default glossary style provided \MessageBreak
4615         for the glossary '\@glo@type'. \MessageBreak
4616         Using deprecated fallback. \MessageBreak
4617         To fix this set the style with \MessageBreak
4618         \string\setglossarystyle\space or use the \MessageBreak
4619         style key=value option}%
4620     \fi
4621   }%
4622   \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
4623   \let\@org@glossaryentrynumbers\glossaryentrynumbers
4624   \bgroup
4625     \@printgloss@setsort
4626     \setkeys{printgloss}{#1}%
4627     \ifx\glossarytitle\org@glossarytitle
4628     \else
4629       \cslet{\@glo@type\@glo@type @title}{\glossarytitle}%
4630     \fi
4631     \let\currentglossary\@glo@type
4632     \let\org@glossaryentrynumbers\glossaryentrynumbers
4633     \let\glsnonextpages\@glsnonextpages
4634     \let\glsnextpages\@glsnextpages

4635     \glxtractivatenopost
4636     \gls@dotoc@title
4637     \@glossarystyle
4638     \let\gls@org@glossaryentryfield\glossentry
4639     \let\gls@org@glossarysubentryfield\subglossentry
4640     \renewcommand{\glossentry}[1]{%
4641       \protected@xdef\gls@currententrylabel{\gls@detoklabel{##1}}%
4642       \gls@org@glossaryentryfield{##1}%
4643     }%
4644     \renewcommand{\subglossentry}[2]{%
```

```

4645     \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
4646     \gls@org@glossarysubentryfield{##1}{##2}%
4647   }%
4648   \@gls@preglossaryhook
4649   \glxtr@printgloss@checkexists{\@glo@type}{#2}%
4650 \egroup
4651 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
4652 \global\let\warn@noprntglossary\relax
4653 }

```

**ractivatenopost** Change \nopostdesc and \glxtrnopostpunc to behave as they do in the glossary.

```

4654 \newcommand*{\glxtractivatenopost}{%
4655   \let\nopostdesc\@nopostdesc
4656   \let\glxtrnopostpunc\@glxtr@nopostpunc
4657 }

```

**lsxtrnopostpunc**

```

4658 \newrobustcmd*{\glxtrnopostpunc}{%

```

**glxtr@nopostpunc** Provide a command that works like \nopostdesc but only switches of the punctuation without suppressing the post-description hook.

```

4659 \newcommand{\@glxtr@nopostpunc}{%
4660   \let\@glxtr@org@postdescription\glspostdescription
4661   \ifglsnopostdot
4662     \renewcommand{\glspostdescription}{%
4663       \glsnopostdottrue
4664       \let\glspostdescription\@glxtr@org@postdescription
4665       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
4666       \glxtrpostdescription
4667       \@glxtr@nopostpunc@postdesc}%
4668   \else
4669     \renewcommand{\glspostdescription}{%
4670       \let\glspostdescription\@glxtr@org@postdescription
4671       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
4672       \glxtrpostdescription
4673       \@glxtr@nopostpunc@postdesc}%
4674   \fi
4675   \glsnopostdotfalse
4676 }

```

**stpunc@postdesc**

```

4677 \newcommand*{\@glxtr@nopostpunc@postdesc}{%

```

**estore@postpunc**

```

4678 \newcommand*{\@glxtr@restore@postpunc}{%
4679   \def\@glxtr@nopostpunc@postdesc{%
4680     \@glxtr@org@postdescription
4681     \let\@glxtr@nopostpunc@postdesc\@empty

```

```

4682 \let\glxtrrestorepostpunc\@empty
4683 }%
4684 }

```

restorepostpunc Does nothing outside of glossary.

```

4685 \newcommand*\glxtrrestorepostpunc{}

```

\@printglossary Redefine.

```

4686 \renewcommand{\@printglossary}[2]{%
4687 \def\@glxtr@printglossopts{#1}%
4688 \@glxtr@orgprintglossary{#1}{#2}%
4689 }

```

Add a key that switches off the entry targets:

```

4690 \define@choicekey{printgloss}{target}
4691 [\@glxtr@printglossval\@glxtr@printglossnr]%
4692 {true,false}[true]%
4693 {%
4694 \ifcase\@glxtr@printglossnr
4695 \def\@glstarget{\glsdohypertarget}%
4696 \else
4697 \let\@glstarget\@secondoftwo
4698 \fi
4699 }

```

hypernameprefix

```

4700 \newcommand*\@glxtrhypernameprefix{}

```

New to v1.20:

```

4701 \define@key{printgloss}{targetnameprefix}{%
4702 \renewcommand*\@glxtrhypernameprefix{#1}%
4703 }

4704 \define@key{printgloss}{prefix}{%
4705 \renewcommand*\glolinkprefix{#1}%
4706 }

4707 \define@key{printgloss}{label}{%
4708 \glxtrsetglossarylabel{#1}%
4709 }

```

setglossarylabel Set the label for subsequent glossaries. If the label is fixed (that is, doesn't change with each glossary) this will need to be scoped or changed again to prevent duplicate labels.

```

4710 \newcommand*\glxtrsetglossarylabel[1]{%
4711 \renewcommand*\@@glossaryseclabel{%
4712 \protected@edef\@currentlabelname{\glossarytoctitle}%
4713 \label{#1}%
4714 }%
4715 }

```

xtr@leveloffset

```
4716 \newcount\@glxtr@leveloffset
```

New to vl.44:

```
4717 \define@key{printgloss}{leveloffset}{%
```

```
4718   \@glxtr@assign@leveloffset#1\relax
```

```
4719 }
```

ign@leveloffset

```
4720 \newcommand*{\@glxtr@assign@leveloffset}{%
```

```
4721   \@ifnextchar+{\p@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
```

```
4722 }
```

ign@leveloffset Discard initial + character.

```
4723 \newcommand*{\p@glxtr@assign@leveloffset}[1]{%
```

```
4724   \@ifnextchar+{\pp@glxtr@assign@leveloffset}{\np@glxtr@assign@leveloffset}%
```

```
4725 }
```

ign@leveloffset

```
4726 \def\np@glxtr@assign@leveloffset#1\relax{\@glxtr@leveloffset=#1\relax}
```

ign@leveloffset

```
4727 \def\pp@glxtr@assign@leveloffset#1\relax{\advance\@glxtr@leveloffset by #1\relax}
```

```
4728 \define@boolkey{printgloss}[glxtr@printgloss@]{groups}[true]{}
```

```
4729 \glxtr@printgloss@groupstrue
```

lsdohypertarget Redefine to insert \@glxtrhypertextprefix before the target name.

```
4730 \let\@glxtr@org@glldohypertarget\glldohypertarget
```

```
4731 \renewcommand{\glldohypertarget}[2]{%
```

```
4732   \@glxtr@org@glldohypertarget{\@glxtrhypertextprefix#1}{#2}%
```

```
4733 }
```

Update \@glstarget to use \def instead being assigned with \let so that it can pick up the new definition and allow any further redefinitions:

```
4734 \ifx\@glstarget\@glxtr@org@glldohypertarget
```

```
4735   \def\@glstarget{\glldohypertarget}%
```

```
4736 \fi
```

@makeglossaries For the benefit of makeglossaries

```
4737 \newcommand*{\glxtr@makeglossaries}[1]{}
```

@glxtr@gettype Get just the type.

```
4738 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
```

```
4739   \def\@glo@type{#2}%
```

```
4740 }
```

@assign@sortkey Assign the sort key.

```
4741 \newcommand\@glstr@mixed@assign@sortkey[1]{%
4742   \protected@edef\@glo@type{\@glo@type}%
4743   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glstr@reg@glosslist}%
4744   {%
4745     \@glo@no@assign@sortkey{#1}%
4746   }%
4747   {%
4748     \@glo@assign@sortkey{#1}%
4749   }%
4750 }%
```

Display number list for the regular version:

splaynumberlist

```
4751 \let\@glstr@idx@displaynumberlist\glsdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
4752 \newcommand*{\@glstr@noidx@displaynumberlist}[1]{%
4753   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4754   \ifdef\@gls@loclist
4755   {%
4756     \def\@gls@noidxloclist@sep{%
4757       \def\@gls@noidxloclist@sep{%
4758         \def\@gls@noidxloclist@sep{%
4759           \glsnumlistsep
4760         }%
4761         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
4762       }%
4763     }%
4764     \def\@gls@noidxloclist@finalsep{}%
4765     \def\@gls@noidxloclist@prev{}%
4766     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
4767     \@gls@noidxloclist@finalsep
4768     \@gls@noidxloclist@prev
4769   }%
4770   {%
4771     \glstrundeftag
4772     \glsdoifexists{#1}%
4773     {%
4774       \GlossariesWarning{Missing location list for ‘#1’. Either
4775         a rerun is required or you haven’t referenced the entry.}%
4776     }%
4777   }%
4778 }%
4779
```



And for the number list loop:

@numberlistloop

```

4780 \newcommand*{\@glxtr@noidx@numberlistloop}[3]{%
4781   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4782   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
4783   \let\@gls@org@glssееformat\glssееformat
4784   \let\glsnoidxdisplayloc#2\relax
4785   \let\glssееformat#3\relax
4786   \ifdef\@gls@loclist
4787   {%
4788     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
4789   }%
4790   {%

4791     \glxtrundeftag
4792     \glsdоifexists{#1}%
4793     {%
4794       \GlossariesWarning{Missing location list for ‘##1’. Either
4795         a rerun is required or you haven’t referenced the entry.}%
4796     }%
4797   }%
4798   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
4799   \let\glssееformat\@gls@org@glssееformat
4800 }%

```

Same for entry number list.

entrynumberlist

```

4801 \newcommand*{\@glxtr@noidx@entrynumberlist}[1]{%
4802   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4803   \ifdef\@gls@loclist
4804   {%
4805     \glsnoidxloclist{\@gls@loclist}%
4806   }%
4807   {%

4808     \glxtrundeftag
4809     \glsdоifexists{#1}%
4810     {%
4811       \GlossariesWarning{Missing location list for ‘#1’. Either
4812         a rerun is required or you haven’t referenced the entry.}%
4813     }%
4814   }%
4815 }%

```

entrynumberlist

```

4816 \newcommand*{\@glxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

4817 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
4818   \protected@edef\@glsxtr@titlelabel{#1}%
4819   \ifdefvoid\@glsxtr@titlelabel
4820   {%
4821   {%
4822     \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
4823   }%
4824   \ifdefvoid{\@glsxtr@titlelabel}%
4825   {%
4826     \DTLifint{#1}%
4827     {%
4828       \ifnum#1<256\relax
4829       \edef#2{\char#1\relax}%
4830     \else
4831     \edef#2{#1}%
4832     \fi
4833   }%
4834   {%
4835     \ifcsundef{#1groupname}%
4836     {\def#2{#1}}%
4837     {\letcs#2{#1groupname}}%
4838   }%
4839 }%
4840 {%
4841   \let#2\@glsxtr@titlelabel
4842 }%
4843 }

```

**g@getgrouptitle** Save original definition of \@gls@getgrouptitle

```

4844 \let\glsxtr@org@getgrouptitle\@gls@getgrouptitle

```

**trgetgrouptitle** Provide a user-level command to fetch the group title. The first argument is the group label. The second argument is a control sequence in which to store the title.

```

4845 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
4846   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
4847   \@onelevel@sanitize\@glsxtr@titlelabel
4848   \ifcsdef{\@glsxtr@titlelabel}
4849   {\letcs{#2}{\@glsxtr@titlelabel}}%
4850   {\glsxtr@org@getgrouptitle{#1}{#2}}%
4851 }
4852 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

**trsetgrouptitle** Sets the title for the given group label.

```

4853 \newcommand{\glsxtrsetgrouptitle}[2]{%
4854   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
4855   \@onelevel@sanitize\@glsxtr@titlelabel
4856   \protected@csxdef{\@glsxtr@titlelabel}{#2}%
4857 }

```

`\alsetgrouptitle` As above put only locally defines the title.

```

4858 \newcommand{\glxtrlocalsetgrouptitle}[2]{%
4859   \protected@edef\glxtr@titlelabel{\glxtr@grouptitle@#1}%
4860   \@onelevel@sanitize\glxtr@titlelabel
4861   \protected@csdef{\glxtr@titlelabel}{#2}%
4862 }
```

`\glsnavigation` Redefine to use new user-level command.

```

4863 \renewcommand*{\glsnavigation}{%
4864   \def\@gls@between{}%
4865   \ifcsundef\@gls@hypergroup\list\@glo@type}%
4866   {%
4867     \def\@gls@list{}%
4868   }%
4869   {%
4870     \expandafter\let\expandafter\@gls@list
4871       \csname @gls@hypergroup\list\@glo@type\endcsname
4872   }%
4873   \@for\@gls@tmp:=\@gls@list\do{%
4874     \@gls@between
4875     \glxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
4876     \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
4877     \let\@gls@between\glshypernavsep
4878   }%
4879 }
```

`@noidx@glossary`

```

4880 \renewcommand*{\@print@noidx@glossary}{%
4881   \ifcsdef\@glsref\@glo@type}%
4882   {%
4883     \ifcsdef\@glo@sortmacro\@glo@sorttype}%
4884     {%
4885       \csuse\@glo@sortmacro\@glo@sorttype}{\@glo@type}%
4886     }%
4887     {%
4888       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
4889     }%
4890     \glossarysection[\glossarytoctitle]{\glossarytitle}%
4891     \glossarypreamble
```

Moved this command definition outside of environment in case of scoping issues (e.g. in tabular-like styles).

```

4892   \def\@gls@currentlettergroup{}%
4893   \begin{theglossary}%
4894     \glossaryheader
4895     \glsresetentrylist
4896     \forlistcsloop{\@gls@noidx@do}{\@glsref\@glo@type}%
4897     \end{theglossary}%
4898     \glossarypostamble
```

```

4899 }%
4900 {%

```

Add section header if there are actually entries defined in this glossary as the document is likely pending a re-run.

```

4901 \glstrifemptyglossary{\@glo@type}%
4902 {}%
4903 {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
4904 \@gls@noref@warn{\@glo@type}%
4905 }%
4906 }

```

`noidxdisplayloc` Patch to check for range formations.

```

4907 \renewcommand*{\glsnoidxdisplayloc}[4]{%
4908 \setentrycounter[#1]{#2}%
4909 \@glstr@display@loc#3\empty\end@glstr@display@loc{#4}%
4910 }

```

`xtr@display@loc` Patch to check for range formations.

```

4911 \def\@glstr@display@loc#1#2\end@glstr@display@loc#3{%
4912 \ifx#1(\relax
4913 \glstrdisplaystartloc{#2}{#3}%
4914 \else
4915 \ifx#1)\relax
4916 \glstrdisplayendloc{#2}{#3}%
4917 \else
4918 \glstrdisplaysingleloc{#1#2}{#3}%
4919 \fi
4920 \fi
4921 }

```

`isplaysingleloc` Single location.

```

4922 \newcommand*{\glstrdisplaysingleloc}[2]{%
4923 \csuse{#1}{#2}%
4924 }

```

By default the range identifiers are simply ignored. A custom list loop handler can be defined by the user to test for ranges by checking the definition of `\glstrlocrangefmt`.

`displaystartloc` Start of a location range.

```

4925 \newcommand*{\glstrdisplaystartloc}[2]{%
4926 \protected@edef\glstrlocrangefmt{#1}%
4927 \ifx\glstrlocrangefmt\empty
4928 \def\glstrlocrangefmt{glsnumberformat}%
4929 \fi
4930 \expandafter\glstrdisplaysingleloc
4931 \expandafter{\glstrlocrangefmt}{#2}%
4932 }

```

trdisplayendloc End of a location range.

```
4933 \newcommand*{\glxtrdisplayendloc}[2]{%
4934   \protected@edef\@glxtr@tmp{#1}%
4935   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glsnumberformat}}{}%
4936   \ifx\glxtrlocrangefmt\@glxtr@tmp
4937   \else
4938     \GlossariesExtraWarning{Mismatched end location range
4939       (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
4940   \fi
4941   \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
4942   \expandafter\glxtrdisplaysingleloc
4943   \expandafter{\glxtrlocrangefmt}{#2}%
4944   \def\glxtrlocrangefmt{}%
4945 }
```

splayendloohook Allow the user to hook into the end of range command.

```
4946 \newcommand*{\glxtrdisplayendloohook}[2]{}
```

sxtrlocrangefmt Current range format. Empty if not in a range.

```
4947 \newcommand*{\glxtrlocrangefmt}{}
```

setentrycounter Adjust \setentrycounter to save the original prefix.

```
4948 \renewcommand*{\setentrycounter}[2][{}]{%
4949   \def\glxtrcounterprefix{#1}%
4950   \ifx\glxtrcounterprefix\@empty
4951     \def\@glo@counterprefix{.}%
4952   \else
4953     \def\@glo@counterprefix{.#1.}%
4954   \fi
4955   \def\glsetentrycounter{#2}%
4956 }
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
4957 \def\@gls@removespaces#1 #2\@nil{%
4958   \toks@=\expandafter{\the\toks@#1}%
4959   \ifx\@#2\%

4960     \edef\@glo@tmp{\the\toks@}%
4961     \ifx\@glo@tmp\empty
4962     \else

      Expand location (just in case \toks@ is needed for something else).

4963       \expandafter\glxtrlocationhyperlink\expandafter
4964       \glsetentrycounter\expandafter\@glo@counterprefix\expandafter{\the\toks@}%
4965       \fi
4966     \else
4967       \@gls@ReturnAfterFi{%
4968         \@gls@removespaces#2\@nil
4969       }%

```

```

4970 \fi
4971 }

```

cationhyperlink

<code>\glxtrlocationhyperlink{&lt;counter&gt;}{&lt;prefix&gt;}{&lt;location&gt;}</code>
---

```

4972 \newcommand*\glxtrlocationhyperlink}[3]{%
4973   \ifdefvoid\glxtrsupplocationurl
4974   {%
4975     \GlsXtrInternalLocationHyperlink{#1}{#2}{#3}%
4976   }%
4977   {%
4978     \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
4979   }%
4980 }

```

supphypernumber

```

4981 \newcommand*\glxtrsupphypernumber}[1]{%
4982   {%
4983     \glshasattribute{\glscurrententrylabel}{externallocation}%
4984     {%
4985       \def\glxtrsupplocationurl{%
4986         \glsggetattribute{\glscurrententrylabel}{externallocation}}%
4987     }%
4988     {%
4989       \def\glxtrsupplocationurl{}%
4990     }%
4991     \glshypernumber{#1}%
4992   }%
4993 }

```

Give a bit of assistance to new users who are confused and don't know how to read transcript messages.

@print@glossary

```

4994 \renewcommand{\@print@glossary}{%
4995   \makeatletter
4996   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4997   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4998   {}%
4999   {\glxtrNoGlossaryWarning{\@glo@type}}%
5000   \ifglxindy
5001     \ifcsundef{@xdy@\@glo@type @language}%
5002     {%
5003       \edef\@do@auxoutstuff{%
5004         \noexpand\AtEndDocument{%
5005           \noexpand\immediate\noexpand\write\@auxout{%

```

```

5006         \string\providecommand\string\@xdylanguage[2]{}%
5007         \noexpand\immediate\noexpand\write\@auxout{%
5008         \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
5009     }%
5010 }%
5011 }%
5012 {%
5013     \edef\@do@auxoutstuff{%
5014     \noexpand\AtEndDocument{%
5015     \noexpand\immediate\noexpand\write\@auxout{%
5016     \string\providecommand\string\@xdylanguage[2]{}%
5017     \noexpand\immediate\noexpand\write\@auxout{%
5018     \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
5019     @language\endcsname}}%
5020     }%
5021     }%
5022     }%
5023     \@do@auxoutstuff
5024     \edef\@do@auxoutstuff{%
5025     \noexpand\AtEndDocument{%
5026     \noexpand\immediate\noexpand\write\@auxout{%
5027     \string\providecommand\string\@gls@codepage[2]{}%
5028     \noexpand\immediate\noexpand\write\@auxout{%
5029     \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
5030     }%
5031     }%
5032     \@do@auxoutstuff
5033     \fi
5034     \renewcommand*{\@warn@nomakeglossaries}{%
5035     \GlossariesWarningNoLine{\string\makeglossaries\space
5036     hasn't been used,^^Jthe glossaries will not be updated}%
5037     }%
5038 }

```

Setup the warning text to display if the external file for the given glossary is missing.

`\GlsWarningHead` Header message.

```

5039 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
5040 This document is incomplete. The external file associated with
5041 the glossary '#1' (which should be called \texttt{#2})
5042 hasn't been created.%
5043 }

```

`\GlsWarningEmptyStart` No entries have been added to the glossary.

```

5044 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
5045 This has probably happened because there are no entries defined
5046 in this glossary.%
5047 }

```

`\GlsWarningEmptyMain` The default “main” glossary is empty.

```

5048 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
5049 If you don't want this glossary,
5050 add \texttt{nomain} to your package option list when you load
5051 \texttt{glossaries-extra.sty}. For example:%
5052 }

```

ingEmptyNotMain A glossary that isn't the default "main" glossary is empty.

```

5053 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
5054 Did you forget to use \texttt{type=#1} when you defined your
5055 entries? If you tried to load entries into this glossary with
5056 \texttt{\string\loadglsentries} did you remember to use
5057 \texttt{[#1]} as the optional argument? If you did, check that
5058 the definitions in the file you loaded all had the type set
5059 to \texttt{\string\glsdefaulttype}.%
5060 }

```

arningCheckFile Advisory message to check the file contents.

```

5061 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
5062 Check the contents of the file \texttt{#1}. If
5063 it's empty, that means you haven't indexed any of your entries in this
5064 glossary (using commands like \texttt{\string\gls} or
5065 \texttt{\string\glsadd}) so this list can't be generated.
5066 If the file isn't empty, the document build process hasn't been
5067 completed.%
5068 }

```

WarningAutoMake Message when automake option has been used.

```

5069 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
5070 You may need to rerun \LaTeX. If you already have, it may be that
5071 \TeX's shell escape doesn't allow you to run
5072 \ifglxindy xindy\else makeindex\fi. Check the
5073 transcript file \texttt{\jobname.log}. If the shell escape is
5074 disabled, try one of the following:
5075
5076 \begin{itemize}
5077 \item Run the external (Lua) application:
5078
5079 \texttt{makeglossaries-lite \string\jobname\string}
5080
5081 \item Run the external (Perl) application:
5082
5083 \texttt{makeglossaries \string\jobname\string}
5084 \end{itemize}
5085
5086 Then rerun \LaTeX\ on this document.
5087 \GlossariesExtraWarning{Rerun required to build the
5088 glossary '#1' or check TeX's shell escape allows
5089 you to run \ifglxindy xindy\else makeindex\fi}%
5090 }

```



WarningMismatch Mismatching \makenoidxglossaries.

```
5091 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
5092   You need to either replace \texttt{\string\makenoidxglossaries}
5093   with \texttt{\string\makeglossaries} or replace
5094   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
5095   \texttt{\string\printnoidxglossary}
5096   (or \texttt{\string\printnoidxglossaries}) and then rebuild
5097   this document.%
5098 }
```

WarningBuildInfo Build advice.

```
5099 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
5100   Try one of the following:
5101   \begin{itemize}
5102     \item Add \texttt{automake} to your package option list when you load
5103           \texttt{glossaries-extra.sty}. For example:
5104
5105           \texttt{\string\usepackage[automake]%
5106                 \glssopenbrace glossaries-extra\glsclosebrace}
5107
5108     \item Run the external (Lua) application:
5109
5110           \texttt{makeglossaries-lite.lua \string"\jobname\string"}
5111
5112     \item Run the external (Perl) application:
5113
5114           \texttt{makeglossaries \string"\jobname\string"}
5115   \end{itemize}
5116
5117   Then rerun \LaTeX\ on this document.%
5118 }
```

trRecordWarning Paragraph for record=only.

```
5119 \newcommand{\GlsXtrRecordWarning}[1]{%
5120   \texttt{\string\printglossary} doesn't work
5121   with the \texttt{record=@glsxtr@record@setting} package option
5122   use\par\texttt{\string\printunsrtglossary[type=#1]}\par
5123   instead (or change the package option).%
5124 }
```

oGlsWarningTail Final paragraph.

```
5125 \newcommand{\GlsXtrNoGlsWarningTail}{%
5126   This message will be removed once the problem has been fixed.%
5127 }
```

GlsWarningNoOut No out file created. Build advice.

```
5128 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
5129   The file \texttt{#1} doesn't exist. This most likely means you haven't used
5130   \texttt{\string\makeglossaries} or you have used
```

```

5131 \texttt{\string\nofiles}. If this is just a draft version of the
5132 document, you can suppress this message using the
5133 \texttt{nomissingglstext} package option.%
5134 }

```

glossarywarning

```

5135 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
5136 \glossarysection[\glossarytoctitle]{\glossarytitle}
5137 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
5138 \par
5139 \glstrifemptyglossary{#1}%
5140 {%
5141 \GlsXtrNoGlsWarningEmptyStart\space
5142 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
5143 \medskip
5144 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
5145 \glsoopenbrace glossaries-extra\glsclosebrace}
5146 \medskip
5147 }%
5148 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
5149 }%
5150 {%
5151 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
5152 {%
5153 \GlsXtrNoGlsWarningCheckFile
5154 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
5155
5156 \ifglssautomake
5157
5158 \GlsXtrNoGlsWarningAutoMake{#1}
5159
5160 \else
5161
5162 \ifthenelse{\equal{#1}{main}}{%
5163 {%
5164 \GlsXtrNoGlsWarningEmptyMain\par
5165 \medskip
5166 \noindent\texttt{\string\usepackage[nomain]%
5167 \glsoopenbrace glossaries-extra\glsclosebrace}
5168 \medskip
5169 }%
5170 {}%
5171
5172 \ifdefequal\makeglossaries\@no@makeglossaries
5173 {%
5174 \GlsXtrNoGlsWarningMisMatch
5175 }%
5176 {}%
5177 \GlsXtrNoGlsWarningBuildInfo

```

```

5178     }%
5179     \fi
5180 }%
5181 {%
5182     \GlsXtrNoGlsWarningNoOut
5183     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
5184 }%
5185 }%
5186 \par
5187 \GlsXtrNoGlsWarningTail
5188 }

```

**glossarywarning** Warn about using `\printglossary` with `record`

```

5189 \newcommand*{\@glxtr@record@noglossarywarning}[1]{%
5190     \GlossariesExtraWarning{\string\printglossary\space doesn't work\MessageBreak
5191     with record=\@glxtr@record@setting\space package option\MessageBreak(use
5192     \string\printunsrtglossary[type=#1])\MessageBreak
5193     instead (or change the package option)}%
5194     \glossarysection[\glossarytoctitle]{\glossarytitle}
5195     \GlsXtrRecordWarning{#1}
5196     \GlsXtrNoGlsWarningTail
5197 }

```

Provide some commands to accompany the `record` option for use with **bib2gls**.

**ResourceOptions** Default resource options.

```

5198 \newcommand*{\GlsXtrDefaultResourceOptions}{}

```

**xtrresourcefile** Since it's dangerous for an external application to create a file with a `.tex` extension, as from v1.11 this enforces a `.glstex` extension to avoid conflict.

```

5199 \newcommand*{\glxtrresourcefile}[2] []{%

```

The `record` option can't be set after this command.

```

5200     \disable@keys{glossaries-extra.sty}{record}%
5201     \glxtr@writefields
5202     \ifdefempty\GlsXtrDefaultResourceOptions
5203     {%
5204         \protected@write\@auxout{\glxtrresourceinit}%
5205         {\string\glxtr@resource{#1}{#2}}%
5206     }%
5207     {%
5208         \protected@write\@auxout{\glxtrresourceinit}%
5209         {\string\glxtr@resource{\GlsXtrDefaultResourceOptions,#1}{#2}}%
5210     }%
5211     \let\@glxtr@org@see@noindex\@glxtr@see@noindex
5212     \let\@glxtr@see@noindex\relax
5213     \IfFileExists{#2.glstex}%
5214     {%

```

Can't scope \@input so save and restore the category code of @ to allow for internal commands in the location list.

```
5215 \edef\@bibgls@restoreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
5216 \makeatletter
5217 \@input{#2.glstex}%
5218 \@bibgls@restoreat
```

If the record=nameref option has been set, check if this is supported by the installed version of bib2gls.

```
5219 \@glxstr@check@bibgls@nameref
5220 }%
5221 {%
5222 \GlossariesExtraWarning{No file '#2.glstex'}%
5223 }%
5224 \let\@gls@see@noindex\@glxstr@org@see@noindex
5225 }
5226 \@onlypreamble\glxstrresourcefile
```

@bibgls@nameref This will only warn after bib2gls has created the .glstex file, but there's way to check before.

```
5227 \newcommand{\@glxstr@check@bibgls@nameref}{%
5228 \ifx\@glxstr@record@setting\@glxstr@record@setting@nameref
5229 \ifdef\bibglshrefchar
5230 {}%
5231 {%
5232 \GlossariesExtraWarning{record=nameref requires at least
5233 version 1.8 of bib2gls}%
5234 }%
5235 \fi
5236 \let\@glxstr@check@bibgls@nameref\relax
5237 }
```

glxstrresourceinit Code used during the protected write operation.

```
5238 \newcommand*\glxstrresourceinit{}
```

glxstrresourcecount

```
5239 \newcount\glxstrresourcecount
```

glxstrLoadResources Short cut that uses \glxstrresourcefile with \jobname as the mandatory argument.

```
5240 \newcommand*\glxstrLoadResources[1][\jobname]{%
5241 \ifnum\glxstrresourcecount=0\relax
5242 \glxstrresourcefile[#1]{\jobname}%
5243 \else
5244 \glxstrresourcefile[#1]{\jobname-\the\glxstrresourcecount}%
5245 \fi
5246 \advance\glxstrresourcecount by 1\relax
5247 }
```

glxstr@resource

```
5248 \newcommand*\glxstr@resource[2]{}
```

```

\glxtr@fields
5249 \newcommand*{\glxtr@fields}[1]{}

xtr@texencoding
5250 \newcommand*{\glxtr@texencoding}[1]{}

\glxtr@langtag
5251 \newcommand*{\glxtr@langtag}[1]{}

@pluralsuffixes
5252 \newcommand*{\glxtr@pluralsuffixes}[4]{}

tr@shortcutsval
5253 \newcommand*{\glxtr@shortcutsval}[1]{}

sxtr@linkprefix
5254 \newcommand*{\glxtr@linkprefix}[1]{}

xtr@writefields This information only needs to be written once, so disable it after it's been used.
5255 \newcommand*{\glxtr@writefields}{%

5256 \protected@write\@auxout{}%
5257 {\string\providecommand*{\string\glxtr@fields}[1]{}}%
5258 \protected@write\@auxout{}%
5259 {\string\providecommand*{\string\glxtr@resource}[2]{}}%
5260 \protected@write\@auxout{}%
5261 {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
5262 \protected@write\@auxout{}%
5263 {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
5264 \protected@write\@auxout{}%
5265 {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
5266 \protected@write\@auxout{}{\string\glxtr@fields{\@gls@keymap}}%

5267 \protected@write\@auxout{}%
5268 {\string\providecommand*{\string\glxtr@record}[5]{}}%

5269 \ifx\@glxtr@record@setting\@glxtr@record@setting@nameref
5270 \protected@write\@auxout{}%
5271 {\string\providecommand*{\string\glxtr@record@nameref}[8]{}}%
5272 \fi

If any languages have been loaded, the language tag will be available in \CurrentTrackedLanguageTag
(provided by tracklang). For multilingual documents, the required locale will have to be indi-
cated in the sort key when using \glxtrresourcefile.

5273 \ifdef\CurrentTrackedLanguageTag
5274 {%
5275 \protected@write\@auxout{}%
5276 {\string\glxtr@langtag{\CurrentTrackedLanguageTag}}%

```

```

5277 }%
5278 {}%
5279 \protected@write\@auxout{}\string\glxtr@pluralsuffixes
5280   {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
5281   {\glxtrabbrvpluralsuffix}}%
5282 \ifdef\inputencodingname
5283 {%
5284   \protected@write\@auxout{}\string\glxtr@texencoding{\inputencodingname}}%
5285 }%
5286 {%

```

If fontspec has been loaded, assume UTF-8. (The encoding can be changed with `\XeTeXinputencoding`, but I can't work out how to determine the current encoding.)

```

5287   \@ifpackageloaded{fontspec}%
5288   {\protected@write\@auxout{}\string\glxtr@texencoding{utf8}}}%
5289   {}%
5290 }%
5291 \protected@write\@auxout{}\string\glxtr@shortcutsval{\glxtr@shortcutsval}}%

```

Prefix deferred until the beginning of the document in case it's redefined later in the preamble. This is picked up by bib2gls when the external option is used.

```

5292 \AtBeginDocument
5293   {\protected@write\@auxout{}\string\glxtr@linkprefix{\glolinkprefix}}}%
5294 \let\glxtr@writefields\relax

```

If the automake option is on, try running bib2gls if the aux file exists. This has to be done before the aux file is opened (so package options `automake=immediate` and `automake=true` are identical if just bib2gls is used). The double-quotes around `\jobname` have been removed (v1.19) since `\jobname` will include double-quotes if the file name has spaces.

```

5295 \ifglautomake
5296   \IfFileExists{\jobname.aux}%
5297   {\immediate\write18{bib2gls \jobname}}}%

```

If `\makeglossaries` is also used, allow `makeindex/xindy` to also be run, otherwise disable the error message about requiring `\makeglossaries` with `automake=true`.

```

5298   \ifx\@gls@doautomake\@gls@doautomake@err
5299   \let\@gls@doautomake\relax
5300   \fi
5301 \fi

```

Check if `order=letter` has been used by mistake (but not if `record=alsoindex` has been used).

```

5302 \@glxtr@if@record@only
5303 {\ifdefstring{\glsorder}{letter}%
5304   {\GlossariesExtraWarningNoLine{Package option 'order=letter' isn't
5305     supported with 'record=\@glxtr@record@setting'. Use 'break-at=none'
5306     resource option instead}}%
5307   {}%
5308 }%
5309 {}%
5310 }

```

do@automake@err

```
5311 \newcommand*{\@gls@doautomake@err}{%
5312   \PackageError{glossaries}{You must use
5313   \string\makeglossaries\space with automake=true}
5314   {%
5315     Either remove the automake=true setting or
5316     add \string\makeglossaries\space to your document preamble.%
5317   }%
5318 }
```

Allow locations specific to a particular counter to be recorded.

\glsxtr@record

```
5319 \newcommand*{\glsxtr@record}[5]{}
```

@record@nameref Used with record=nameref to include current label information.

```
5320 \newcommand*{\glsxtr@record@nameref}[8]{}
```

r@counterrecord Aux file command.

```
5321 \newcommand*{\glsxtr@counterrecord}[3]{%
5322   \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
5323 }
```

unterrecordhook Hook used by \@glsxtr@dorecord.

```
5324 \newcommand*{\@glsxtr@counterrecordhook}{}%
```

trRecordCounter Activate recording for a particular counter (identified in the argument).

```
5325 \newcommand*{\GlsXtrRecordCounter}[1]{%
5326   \@glsxtr@recordcounter{#1}%
5327 }
5328 \@onlypreamble\GlsXtrRecordCounter
```

docounterrecord

```
5329 \newcommand*{\@glsxtr@docounterrecord}[1]{%
5330   \protected@write\@auxout{}{\string\glsxtr@counterrecord
5331     {\@gls@label}{#1}{\csuse{the#1}}}%
5332 }
```

lsxtrglossentry Users may prefer to have entries displayed throughout the document rather than gathered together in a list. This command emulates the way \glossentry behaves (without the style formatting commands like \item). This needs to define \currentglossary to the current glossary type (normally set at the start of \@printglossary) and needs to define \glscurrententrylabel to the entry's label (normally set before \glossentry and \subglossentry). This needs some protection in case it's used in a section heading.

```
5333 \newcommand*{\glsxtrglossentry}[1]{%
5334   \glsxtrtitleorpdforheading
5335   {\@glsxtrglossentry{#1}}%
5336   {\glsentryname{#1}}%
```

```

5337 {\glxtrheadname{#1}}%
5338 }

```

`\glxtrglossentry` Another test is needed in case `\@glxtrglossentry` has been written to the table of contents.

```

5339 \newrobustcmd*{\@glxtrglossentry}[1]{%
5340   \glxtrtitleorpdforheading
5341   {%
5342     \glstoifexists{#1}%
5343     {%
5344       \begingroup
5345         \protected@edef\glscurrententrylabel{\glsetoklabel{#1}}%
5346         \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5347         \ifglshasparent{#1}%
5348         {\GlsXtrStandaloneSubEntryItem{#1}}%
5349         {\glentryitem{#1}}%
5350         \GlsXtrStandaloneEntryName{#1}%
5351       \endgroup
5352     }%
5353   }%
5354   {\glsentryname{#1}}%
5355   {\glxtrheadname{#1}}%
5356 }

```

`\GlsXtrStandaloneEntryName`

```

5357 \newcommand*{\GlsXtrStandaloneEntryName}[1]{%
5358   \glstarget{#1}{\glossentryname{#1}}%
5359 }

```

`\GlsXtrStandaloneGlossaryType` To make it easier to adjust the definition of `\currentglossary` within `\glxtrglossentry`, this expands to the default definition. (If redefined, it must fully expand to the appropriate label.)

```

5360 \newcommand{\GlsXtrStandaloneGlossaryType}{\glsentrytype{\glscurrententrylabel}}

```

`\GlsXtrStandaloneSubEntryItem` Used for sub-entries in standalone format. The argument is the entry's label.

```

5361 \newcommand*{\GlsXtrStandaloneSubEntryItem}[1]{%
5362   \GlsXtrIfFieldEqNum{level}{#1}{1}{\glssubentryitem{#1}}{}%
5363 }

```

`\glxtrglossentryother` As `\glxtrglossentry` but uses a different field. First argument is code to use in the header. The second argument is the entry's label. The third argument is the internal field label. This needs to be expandable in case it occurs in a sectioning command so it can't have an optional argument.

```

5364 \newcommand*{\glxtrglossentryother}[3]{%
5365   \ifstrempy{#1}%
5366   {%
5367     \ifcsdef{glxtrhead#3}%

```



```

5368   {%
5369     \glxtrtitleorpdforheading
5370     {\@glxtrglossentryother{#2}{#3}{#1}}%
5371     {\@gls@entry@field{#2}{#3}}%
5372     {\csuse{glxtrhead#3}{#2}}%
5373   }%
5374   {%
5375     \glxtrtitleorpdforheading
5376     {\@glxtrglossentryother{#2}{#3}{#1}}%
5377     {\@gls@entry@field{#2}{#3}}%
5378     {\@gls@entry@field{\NoCaseChange{#2}}{#3}}%
5379   }%
5380 }%
5381 {%
5382   \glxtrtitleorpdforheading
5383   {\@glxtrglossentryother{#2}{#3}{#1}}%
5384   {\@gls@entry@field{#2}{#3}}%
5385   {#1}%
5386 }%
5387 }

```

`glossentryother` As `\@glxtrglossentry` but uses a different field.

```

5388 \newrobustcmd*{\@glxtrglossentryother}[3]{%
5389   \glxtrtitleorpdforheading
5390   {%
5391     \glsdoifexists{#1}%
5392     {%
5393       \begingroup
5394         \protected@edef\glscurrententrylabel{\glsetoklabel{#1}}%
5395         \protected@edef\currentglossary{\GlsXtrStandaloneGlossaryType}%
5396         \ifglshasparent{#1}%
5397         {\GlsXtrStandaloneSubEntryItem{#1}}%
5398         {\glsentryitem{#1}}%
5399         \GlsXtrStandaloneEntryOther{#1}%
5400       \endgroup
5401     }%
5402   }%
5403   {\@gls@entry@field{#1}{#2}}%
5404   {#3}%
5405 }

```

`aloneEntryOther`

```

5406 \newcommand*{\GlsXtrStandaloneEntryOther}[2]{%
5407   \glstarget{#1}{\glossentrynameother{#1}{#2}}%
5408 }

```

`ntunsrtglossary` Similar to `\printnoidxglossary` but it displays all entries defined for the given glossary without sorting. Check for `\@printgloss@checkexists` which was introduced to glossaries v4.46.

```

5409 \ifdef \@printgloss@checkexists
5410 {
5411   \newcommand*{\printunsrtglossary}{%
5412     \let\@printgloss@checkexists\@printgloss@checkexists@allowignored
5413     \ifstar\s@printunsrtglossary\@printunsrtglossary
5414   }
5415 }
5416 {
5417   \newcommand*{\printunsrtglossary}{%
5418     \@ifstar\s@printunsrtglossary\@printunsrtglossary
5419   }
5420 }

```

`\printunsrtglossary` Unstarred version.

```

5421 \newcommand*{\@printunsrtglossary}[1][\@print@unsrt@glossary]{%
5422   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
5423 }

```

`\printunsrtglossary` Starred version.

```

5424 \newcommand*{\s@printunsrtglossary}[2][\@print@unsrt@glossary]{%
5425   \begin{group}
5426     #2%
5427   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
5428   \end{group}
5429 }

```

`\printunsrtglossaries` Similar to `\printnoidxglossaries` but it displays all entries defined for the given glossary without sorting.

```

5430 \newcommand*{\printunsrtglossaries}{%
5431   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
5432 }

```

`\@print@unsrt@glossary`

```

5433 \newcommand*{\@print@unsrt@glossary}{%
5434   \glossarysection[\glossarytoctitle]{\glossarytitle}%
5435   \glossarypreamble
5436   check for empty list
5437   \glsxtrifemptyglossary{\@glo@type}%
5438   {%
5439     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
5440   }%
5441   {%
5442     \key@ifundefined{glossentry}{group}%
5443     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
5444     {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
5445     \def\@gls@currentlettergroup{}}%

```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```

5445 \def\@glxtr@doglossary{%
5446   \begin{theglossary}%
5447   \glossaryheader
5448   \glsresetentrylist
5449 }%
5450 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
5451   :\expandafter=\csname glo@list@\@glo@type\endcsname\do{%
5452   \ifdefempty{\glscurrententrylabel}
5453   {}%
5454   {%

```

Provide a hook (for example to measure width).

```

5455   \let\glxtr@process\@firstofone
5456   \let\printunsrtglossaryskipentry
5457   \@glxtr@printunsrtglossaryskipentry
5458   \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

5459   \glxtr@process
5460   {%
5461   \ifglxtr@printgloss@groups

```

This still uses `\ifglshasparent` to determine whether or not to check for a change in the letter group. (It doesn't take the level offset into account because `bib2gls` only saves the group information for parentless entries.)

```

5462   \ifglshasparent{\glscurrententrylabel}{}%
5463   {%
5464   \@glxtr@checkgroup\glscurrententrylabel
5465   \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
5466     {\@glxtr@groupheading}%
5467   }%
5468   \fi

5469   \protected@appto\@glxtr@doglossary{%
5470     \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5471   }%
5472 }%
5473 }%
5474 \appto\@glxtr@doglossary{\end{theglossary}}%
5475 \printunsrtglossarypredoglossary
5476 \@glxtr@doglossary
5477 }%
5478 \glossarypostamble
5479 }

```

`rtinnerglossary` Similar to `\printunsrtglossary` but doesn't add the section heading, preamble, postamble or start and end of theglossary. Grouping is automatically applied so it may cause a problem within tabular-like environments. The beginning and ending of theglossary should be added

around this command (but ensure the style has been set first). The simplest way of doing this is to place `\printunsrtinnerglossary` inside the `printunsrtglossarywrap` environment.

```

5480 \newcommand*{\printunsrtinnerglossary}[3] [] {%
5481   \begingroup
5482   \def\@glstr@printglossopts{#1}%
5483   \def\@glo@type{\glsdefaulttype}%
5484   \setkeys{printgloss}[title,toctitle,style,numberedsection,sort,label]{#1}%
5485   \let\currentglossary\@glo@type
5486   #2%
5487   \@print@unsrt@innerglossary
5488   #3%
5489   \endgroup
5490 }

```

`srtglossarywrap`

```

5491 \newenvironment{printunsrtglossarywrap}[1] [] {%
5492 {%
5493   \def\@glstr@printglossopts{#1}%
5494   \def\@glo@type{\glsdefaulttype}%
5495   \def\glossarytitle{\csname \@glotype@\@glo@type @title\endcsname}%
5496   \def\glossarytoctitle{\glossarytitle}%
5497   \let\org@glossarytitle\glossarytitle
5498   \def\@glossarystyle{%
5499     \ifx\@glossary@default@style\relax
5500       \GlossariesWarning{No default glossary style provided \MessageBreak
5501         for the glossary '@@glo@type'. \MessageBreak
5502       Using deprecated fallback. \MessageBreak
5503       To fix this set the style with \MessageBreak
5504       \string\setglossarystyle\space or use the \MessageBreak
5505       style key=value option}%
5506     \fi
5507   }%
5508   \def\gls@dotoc@title{\glssettoc@title{\@glo@type}}%
5509   \let\@org@glossaryentrynumbers\glossaryentrynumbers
5510   \@printgloss@setsort
5511   \setkeys{printgloss}{#1}%

```

The `type` key simply allows the title to be set if the title key isn't supplied.

```

5512 \ifglossaryexists*{\@glo@type}%
5513 {%
5514   \ifx\glossarytitle\org@glossarytitle
5515   \else
5516     \expandafter\let\csname \@glotype@\@glo@type @title\endcsname
5517       \glossarytitle
5518   \fi
5519   \let\currentglossary\@glo@type
5520 }%
5521 {}%
5522 \let\org@glossaryentrynumbers\glossaryentrynumbers

```

```

5523 \let\glsnonextpages\@glsnonextpages
5524 \let\glsnextpages\@glsnextpages
5525 \let\nopostdesc\@nopostdesc
5526 \gls@dotoctitle
5527 \@glossarystyle
5528 \let\gls@org@glossaryentryfield\glossentry
5529 \let\gls@org@glossarysubentryfield\subglossentry

5530 \renewcommand{\glossentry}[1]{%
5531   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
5532   \gls@org@glossaryentryfield{##1}%
5533 }%
5534 \renewcommand{\subglossentry}[2]{%
5535   \protected@xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
5536   \gls@org@glossarysubentryfield{##1}{##2}%
5537 }%
5538 \@gls@preglossaryhook
5539 \glossarysection[\glossarytoctitle]{\glossarytitle}%
5540 \glossarypreamble
5541 \begin{theglossary}%
5542 \glossaryheader
5543 \glsresetentrylist
5544 }%
5545 {%
5546 \end{theglossary}%
5547 \glossarypostamble
5548 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
5549 \global\let\warn@noprintglossary\relax
5550 }

```

t@innerglossary This is much like \@print@unsrt@innerglossary but only contains what would normally be the content of the theglossary.

```

5551 \newcommand*{\@print@unsrt@innerglossary}{%
  No section header or preamble.

5552   \glstrifemptyglossary{\@glo@type}%
5553   {%
5554     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
5555   }%
5556   {%
5557     \key@ifundefined{glossentry}{group}%
5558     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
5559     {\let\@gls@getgrouptitle\@glstr@unsrt@getgrouptitle}%
5560     \def\@gls@currentlettergroup{}%

  No header or reset.

5561     \def\@glstr@doglossary{}%
5562     \expandafter\for\expandafter\glscurrententrylabel\expandafter
5563       :\expandafter=\csname glist@\@glo@type\endcsname\do{%
5564       \ifdefempty{\glscurrententrylabel}

```

```

5565     {}%
5566     {%

```

Provide a hook (for example to measure width).

```

5567     \let\glxtr@process\@firstofone
5568     \let\printunsrtglossaryskipentry
5569         \@glxtr@printunsrtglossaryskipentry
5570     \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

5571     \glxtr@process
5572     {%
5573     \ifglxtr@printgloss@groups

```

This still uses `\ifglshasparent` to determine whether or not to check for a change in the letter group. (It doesn't take the level offset into account because `bib2gls` only saves the group information for parentless entries.)

```

5574         \ifglshasparent{\glscurrententrylabel}{}%
5575         {%
5576             \@glxtr@checkgroup\glscurrententrylabel
5577             \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
5578                 {\@glxtr@groupheading}%
5579             }%
5580         \fi

5581         \protected@eappto\@glxtr@doglossary{%
5582             \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
5583         }%
5584     }%
5585 }%
5586 \printunsrtglossarypredoglossary
5587 \@glxtr@doglossary
5588 }%

```

No postamble.

```

5589 }

```

entryprocesshook

```

5590 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}

```

glossaryskipentry

```

5591 \newcommand*{\printunsrtglossaryskipentry}{%
5592     \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
5593 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
5594 }

```

entryprocesshook

```

5595 \newcommand*{\@glxtr@printunsrtglossaryskipentry}{%
5596     \let\glxtr@process\@gobble
5597 }

```

rypredoglossary

```
5598 \newcommand*{\printunsrtglossarypredoglossary}{}%
```

lossary@handler

```
5599 \newcommand{\@printunsrtglossary@handler}[1]{%
5600   \protected@xdef\glscurrententrylabel{#1}%
5601   \printunsrtglossaryhandler\glscurrententrylabel
5602 }
```

glossaryhandler

```
5603 \newcommand{\printunsrtglossaryhandler}[1]{%
5604   \glsxtrunsrtdo{#1}%
5605 }
```

triflabelinlist

`\glsxtriflabelinlist{<label>}{<list>}{<true>}{<false>}`

Might be useful for the handler to check if an entry label or category label is contained in a list, so provide a user-level version of `\@gls@ifinlist` which ensures the label and list are fully expanded.

```
5606 \newrobustcmd*{\glsxtriflabelinlist}[4]{%
5607   \protected@edef\@glxtr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
5608   \@glxtr@doiflabelinlist{#3}{#4}%
5609 }
```

srtglossaryunit

```
5610 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
5611   \s@printunsrtglossary[type=\glstypedefaulttype,#1]{%
5612     \printunsrtglossaryunitsetup{#2}%
5613   }%
5614 }
```

ossaryunitsetup

```
5615 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
5616   \renewcommand{\printunsrtglossaryhandler}[1]{%
5617     \glsxtrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
5618     {\glsxtrunsrtdo{##1}}%
5619   }%
5620 }
```

Only the target names should have the prefixes adjusted as `\gls` etc need the original `\glolinkprefix`. The `\@gobble` part discards `\glolinkprefix`.

```
5621 \ifcsundef{theH#1}%
5622 {%
5623   \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
5624 }
```

```

5625 {%
5626   \renewcommand*{\@glxtrhypernameprefix}{record.#1.\cuse{theH#1}.\@gobble}%
5627 }%
5628 \renewcommand*\glossarysection[2][{}]{%
5629   \appto\glossarypostamble{\glspare\medskip\glspare}%
5630 }

```

unsrtglossaryunit

```

5631 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
5632   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
5633     requires the record=only or record=alsoindex package option}{}%
5634 }

```

glxtr@getgrouptitle

```

5635 \newrobustcmd*{\@glxtr@unsrt@getgrouptitle}[2]{%
5636   \protected@edef\@glxtr@titlelabel{\glxtr@grouptitle@#1}%
5637   \@onelevel@sanitize\@glxtr@titlelabel
5638   \ifcsdef{\@glxtr@titlelabel}
5639     {\letcs{#2}{\@glxtr@titlelabel}}%
5640     {\def#2{#1}}%
5641 }

```

\glxtr@unsrtdo Provide a user-level call to \@glxtr@noidx@do to make it easier to define a new handler.

```

5642 \newcommand{\glxtr@unsrtdo}{\@glxtr@noidx@do}

```

glxtrgroupfield bib2gls provides a supplementary field labelled secondarygroup for secondary glossaries, so provide a way of switching to that field. (The group key still needs checking. There's no associated key with the internal field).

```

5643 \newcommand*\glxtrgroupfield{group}

```

The tabular-like glossary styles cause quite a problem with the iterative approach. In particular for the group skip. To compensate for this, the groups are now determined while \@glxtr@doglossary is being constructed rather than in the handler.

glxtr@checkgroup The argument is the entry's label. (This block of code was formerly in \@glxtr@noidx@do.) Now that this is no longer within a tabular environment, the global definitions aren't needed. The result is now stored in \@glxtr@groupheading, which will be empty if no heading is required.

```

5644 \newcommand*\@glxtr@checkgroup[1]{%
5645   \def\@glxtr@groupheading{}%
5646   \key@ifundefined{glossentry}{group}%
5647   {%
5648     \letcs{\@gls@sort}{glo@glstdetoklabel{#1}@sort}%
5649     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
5650   }%
5651   {%

```



```

5652 \protected@edef\@glo@thislettergrp{%
5653 \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
5654 }%
5655 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
5656 {}%
5657 {%
5658 \ifdefempty{\@gls@currentlettergroup}{}%
5659 {\def\@glsxtr@groupheading{\gls@groupskip}}%
5660 \protected@eappto\@glsxtr@groupheading{%
5661 \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
5662 }%
5663 }%
5664 \let\@gls@currentlettergroup\@glo@thislettergrp
5665 }

```

`trLocationField` Stores the internal name of the location field.

```
5666 \newcommand*\GlsXtrLocationField{location}
```

`glsxtr@noidx@do` Minor modification of `\@gls@noidx@do` to check for location field if present, but also need to check for the group field.

```

5667 \newcommand{\@glsxtr@noidx@do}[1]{%
5668 \ifglsentryexists{#1}%
5669 {%
5670 \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
5671 \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@\GlsXtrLocationField}%

```

Use level number to determine whether or not this entry has a parent.

```

5672 \gls@level=\numexpr\csuse{glo@\glsdetoklabel{#1}@level}+\@glsxtr@leveloffset\relax
5673 \ifnum\gls@level>0
5674 \let\@glsxtr@ifischild\@firstoftwo
5675 \else
5676 \let\@glsxtr@ifischild\@secondoftwo
5677 \fi

```

Some glossary styles (such as `topicmcols`) save the level using `\def` so make sure `\gls@level` is expanded before being passed to `\subglossentry`.

```

5678 \@glsxtr@ifischild
5679 {%
5680 \ifdefvoid{\@gls@location}%
5681 {%
5682 \ifdefvoid{\@gls@loclist}%
5683 {%
5684 \expandafter\subglossentry\expandafter{\number\gls@level}{#1}{}%
5685 }%
5686 {%
5687 \expandafter\subglossentry\expandafter{\number\gls@level}{#1}%
5688 {%
5689 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
5690 }%
5691 }%

```

```

5692     }%
5693     {%
5694         \expandafter\subglossentry\expandafter
5695         {\number\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}}%
5696     }%
5697 }%
5698 {%

5699     \ifdefvoid{\@gls@location}%
5700     {%
5701         \ifdefvoid{\@gls@loclist}
5702         {%
5703             \glossentry{#1}{}%
5704         }%
5705         {%
5706             \glossentry{#1}%
5707             {%
5708                 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}}%
5709             }%
5710         }%
5711     }%
5712     {%
5713         \glossentry{#1}%
5714         {%
5715             \glossaryentrynumbers{\@gls@location}}%
5716         }%
5717     }%
5718 }%
5719 }%
5720 {}%
5721 }

```

Provide a way to conveniently define commands that behaves like `\gls` with a label prefix.

It's possible that the user might want minor variations with the same prefix but different default options, so use a counter to provide unique inner commands.

`\glsxtrnewgls`

```

5722 \newcount\@glsxtrnewgls@inner

```

(The default options supplied in *<options>* below could possibly be used to form the inner control sequence name to help make it unique, but it might feasibly contain the value where the value might contain commands.)

`r@providenewgls`

```

5723 \newcommand*{\@glsxtr@providenewgls}{%
5724     \protected@write\@auxout{}{\string\providecommand{\string\@glsxtr@newglslike}[2]{}}%
5725     \let\@glsxtr@providenewgls\relax
5726 }

```

identifyglslike Identify the command given in the second argument for the benefit of bib2gls.

```
5727 \newcommand{\glxtridentifyglslike}[2]{%
5728 \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off
5729 }%
5730 {%
5731 \@glxtr@providenewgls
5732 \protected@write\@auxout{}\string\@glxtr@newglslike{#1}\string#2}%
5733 }%
5734 }
```

\@glxtrnewgls

<code>\glxtrnewgls[&lt;options&gt;]{&lt;prefix&gt;}{&lt;cs&gt;}{&lt;inner cs name&gt;}</code>
---

```
5735 \newcommand*\@glxtrnewgls[4]{%
5736 \ifdef{#3}%
5737 {%
5738 \PackageError{glossaries-extra}{Command \string#3\space already
5739 defined}{}%
5740 }%
5741 {%
```

Write information to the aux file for bib2gls.

```
5742 \glxtridentifyglslike{#2}{#3}%
5743 \ifcsdef{@#4like@#2}%
5744 {%
5745 \advance\@glxtrnewgls@inner by \@ne
5746 \def\@glxtrnewgls@innercsname{@#4like\number\@glxtrnewgls@inner @#2}%
5747 }%
5748 {\def\@glxtrnewgls@innercsname{@#4like@#2}}%
5749 \expandafter\newrobustcmd\expandafter*\expandafter
5750 #3\expandafter{\expandafter\@glshyp@opt\csname\@glxtrnewgls@innercsname\endcsname}%
5751 \ifstrempy{#1}%
5752 {%
5753 \expandafter\newcommand\expandafter*\csname\@glxtrnewgls@innercsname\endcsname[2][]{%
5754 \new@ifnextchar[%
5755 {\csname @#4@\endcsname{##1}{#2##2}}%
5756 {\csname @#4@\endcsname{##1}{#2##2} []}%
5757 }%
5758 }%
5759 {%
5760 \expandafter\newcommand\expandafter*\csname\@glxtrnewgls@innercsname\endcsname[2][]{%
5761 \new@ifnextchar[%
5762 {\csname @#4@\endcsname{#1,##1}{#2##2}}%
5763 {\csname @#4@\endcsname{#1,##1}{#2##2} []}%
5764 }%
5765 }%
5766 }%
```

5767 }

\glxstrnewgls

`\glxstrnewgls[\langle options \rangle]{\langle prefix \rangle}{\langle cs \rangle}`

The first argument prepends to the options and the second argument is the prefix.

```
5768 \newrobustcmd*{\glxstrnewgls}[3] [] {%
5769   \@glxstrnewgls{#1}{#2}{#3}{gls}%
5770 }
```

**lsxtrnewglslike** Provide a way to conveniently define commands that behave like \gls, \glsp1, \Gls and \Glsp1 with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
5771 \newrobustcmd*{\glxtrnewglslike}[6] [] {%
5772   \@glxtrnewgls{#1}{#2}{#3}{gls}%
5773   \@glxtrnewgls{#1}{#2}{#4}{glsp1}%
5774   \@glxtrnewgls{#1}{#2}{#5}{Gls}%
5775   \@glxtrnewgls{#1}{#2}{#6}{Glsp1}%
5776 }
```

**lsxtrnewGLSlike** Provide a way to conveniently define commands that behave like \GLS, \GLSp1 with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
5777 \newrobustcmd*{\glxtrnewGLSlike}[4] [] {%
5778   \@glxtrnewgls{#1}{#2}{#3}{GLS}%
5779   \@glxtrnewgls{#1}{#2}{#4}{GLSp1}%
5780 }
```

**\glxtrnewrgls** As \glxtrnewgls but for \rgls.

```
5781 \newrobustcmd*{\glxtrnewrgls}[3] [] {%
5782   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
5783 }
```

**sxtrnewrglslike** As \glxtrnewglslike but for \rgls etc.

```
5784 \newrobustcmd*{\glxtrnewrglslike}[6] [] {%
5785   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
5786   \@glxtrnewgls{#1}{#2}{#4}{rglsp1}%
5787   \@glxtrnewgls{#1}{#2}{#5}{rGls}%
5788   \@glxtrnewgls{#1}{#2}{#6}{rGlsp1}%
5789 }
```

**sxtrnewrGLSlike** As \glxtrnewGLSlike but for \rGLS etc.

```
5790 \newrobustcmd*{\glxtrnewrGLSlike}[4] [] {%
5791   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
5792   \@glxtrnewgls{#1}{#2}{#4}{rGLSp1}%
5793 }
```

Provide easy access to record count fields.

**totalRecordCount** Access total record count. This is designed to be expandable. The argument is the label.

```
5794 \newcommand*{\GlsXtrTotalRecordCount}[1]{%
5795 \ifcsdef{glo@glstdetoklabel{#1}@recordcount}%
5796 {\csname glo@glstdetoklabel{#1}@recordcount\endcsname}%
5797 {0}%
5798 }
```

**sXtrRecordCount** Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
5799 \newcommand*{\GlsXtrRecordCount}[2]{%
5800 \ifcsdef{glo@glstdetoklabel{#1}@recordcount.#2}%
5801 {\csname glo@glstdetoklabel{#1}@recordcount.#2\endcsname}%
5802 {0}%
5803 }
```

**tionRecordCount** Access record count for a particular counter and location. The first argument is the label. The second argument is the counter name. The third argument is the location. This command shouldn't be used if the location doesn't fully expand unless `\glstrdetoklocation` can be set to something sensible.

```
5804 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
5805 \ifcsdef{glo@glstdetoklabel{#1}@recordcount.#2.\glstrdetoklocation{#3}}%
5806 {\csname glo@glstdetoklabel{#1}@recordcount.#2.\glstrdetoklocation{#3}\endcsname}%
5807 {0}%
5808 }
```

**trdetoklocation**

```
5809 \newcommand*{\glstrdetoklocation}[1]{#1}
```

**ablerecordcount**

```
5810 \newcommand*{\glxtrenablerecordcount}{%
5811 \renewcommand*{\gls}{\rgls}%
5812 \renewcommand*{\Gls}{\rGls}%
5813 \renewcommand*{\glspl}{\rglspl}%
5814 \renewcommand*{\Glspl}{\rGlspl}%
5815 \renewcommand*{\GLS}{\rGLS}%
5816 \renewcommand*{\GLSpl}{\rGLSpl}%
5817 }
```

**ordtriggervalue** The value used by the record trigger test. The argument is the entry's label.

```
5818 \newcommand*{\glstrrecordtriggervalue}[1]{%
5819 \GlsXtrTotalRecordCount{#1}%
5820 }
```

**dCountAttribute**

```
5821 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
5822 \@for\@glstrcat:=#1\do
```

```

5823 {%
5824   \ifdefempty{\@glstr@cat}{}%
5825   {%
5826     \glsssetcategoryattribute{\@glstr@cat}{recordcount}{#2}%
5827   }%
5828 }%
5829 }

```

ifrecordtrigger

`\glstrifrecordtrigger{<label>}{<trigger format>}{<normal>}`

```

5830 \newcommand*{\glstrifrecordtrigger}[3]{%
5831   \glshasattribute{#1}{recordcount}%
5832   {%
5833     \ifnum\glstrrecordtriggervalue{#1}>\glsggetattribute{#1}{recordcount}\relax
5834     #3%
5835   \else
5836     #2%
5837   \fi
5838 }%
5839 {#3}%
5840 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

5841 \newcommand*{\@glstr@rglstrigger@record}[3]{%
5842   \protected@edef\glslabel{\glsdetoklabel{#2}}%
5843   \let\@gls@link@label\glslabel
5844   \def\@glstr@thevalue{%
5845     \def\@glstr@theHvalue{\@glstr@thevalue}%
5846     \def\@glsnumberformat{glstriggerrecordformat}%
5847     \protected@edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
5848     \protected@edef\glstype{\csname glo@\glslabel @type\endcsname}%
5849     \def\@glstr@thevalue{%
5850       \def\@glstr@theHvalue{\@glstr@thevalue}%
5851       \glstrinitwrgloss
5852       \glslinkpresetkeys
5853       \setkeys{glslink}{#1}%
5854       \glslinkpostsetkeys
5855       \ifdefempty{\@glstr@thevalue}%
5856       {%
5857         \@gls@saveentrycounter
5858       }%
5859     }%
5860     \let\theglssentrycounter\@glstr@thevalue
5861     \def\theHglssentrycounter{\@glstr@theHvalue}%
5862   }%

```

```

5863 \ifglxtrinitwrglossbefore
5864 \do@wrglossary{#2}%
5865 \fi
5866 #3%
5867 \ifglxtrinitwrglossbefore
5868 \else
5869 \do@wrglossary{#2}%
5870 \fi
5871 \ifKV@glslink@local
5872 \glsllocalunset{#2}%
5873 \else
5874 \glsunset{#2}%
5875 \fi
5876 }

```

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```

5877 \newcommand*{\glstriggerrecordformat}[1]{

```

\rgls

```

5878 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

```

\@rgls

```

5879 \newcommand*{\@rgls}[2][{}]{%
5880 \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
5881 }

```

\@rgls@

```

5882 \def\@rgls@#1#2[#3]{%
5883 \glxtrifrecordtrigger{#2}%
5884 {%
5885 \@glxtr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
5886 }%
5887 {%
5888 \@gls@{#1}{#2}[#3]%
5889 }%
5890 }%

```

\rglsp1

```

5891 \newrobustcmd*{\rglsp1}{\@gls@hyp@opt\@rglsp1}

```

\@rglsp1

```

5892 \newcommand*{\@rglsp1}[2][{}]{%
5893 \new@ifnextchar[{\@rglsp1@{#1}{#2}}{\@rglsp1@{#1}{#2}[]}%
5894 }

```

\@rglsp1@

```

5895 \def\@rglsp1@#1#2[#3]{%

```

```

5896 \glxtrifrecordtrigger{#2}%
5897 {%
5898 \glxtr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
5899 }%
5900 {%
5901 \@Glspl@{#1}{#2}[#3]%
5902 }%
5903 }%

```

\rGls

```
5904 \newrobustcmd*{\rGls}{\@Gls@hyp@opt\rGls}
```

\@rGls

```

5905 \newcommand*{\@rGls}[2][{}]{%
5906 \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
5907 }

```

\@rGls@

```

5908 \def\@rGls@#1#2[#3]{%
5909 \glxtrifrecordtrigger{#2}%
5910 {%
5911 \glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
5912 }%
5913 {%
5914 \@Gls@{#1}{#2}[#3]%
5915 }%
5916 }%

```

\rGlspl

```
5917 \newrobustcmd*{\rGlspl}{\@Gls@hyp@opt\rGlspl}
```

\@rGlspl

```

5918 \newcommand*{\@rGlspl}[2][{}]{%
5919 \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
5920 }

```

\@rGlspl@

```

5921 \def\@rGlspl@#1#2[#3]{%
5922 \glxtrifrecordtrigger{#2}%
5923 {%
5924 \glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
5925 }%
5926 {%
5927 \@Glspl@{#1}{#2}[#3]%
5928 }%
5929 }%

```

\rGLS

```
5930 \newrobustcmd*{\rGLS}{\@Gls@hyp@opt\rGLS}
```



\@rGLS

```
5931 \newcommand*{\@rGLS}[2] [] {%
5932   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2} []}%
5933 }
```

\@rGLS@

```
5934 \def\@rGLS@#1#2[#3] {%
5935   \glstrifrecordtrigger{#2}%
5936   {%
5937     \@glstr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
5938   }%
5939   {%
5940     \@GLS@{#1}{#2}[#3]%
5941   }%
5942 }%
```

\rGLSpl

```
5943 \newrobustcmd*{\rGLSpl}{\@glshyp@opt\rGLSpl}
```

\@rGLSpl

```
5944 \newcommand*{\@rGLSpl}[2] [] {%
5945   \new@ifnextchar[{\@rGLSpl@{#1}{#2}}{\@rGLSpl@{#1}{#2} []}%
5946 }
```

\@rGLSpl@

```
5947 \def\@rGLSpl@#1#2[#3] {%
5948   \glstrifrecordtrigger{#2}%
5949   {%
5950     \@glstr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
5951   }%
5952   {%
5953     \@GLSpl@{#1}{#2}[#3]%
5954   }%
5955 }%
```

\rglsformat

```
5956 \newcommand*{\rglsformat}[2] {%
5957   \glrifregular{#1}
5958   {\glentryfirst{#1}}%
5959   {\ifglshaslong{#1}{\glentrylong{#1}}{\glentryfirst{#1}}}%#2%
5960 }
```

\rglsplformat

```
5961 \newcommand*{\rglsplformat}[2] {%
5962   \glrifregular{#1}
5963   {\glentryfirstplural{#1}}%
5964   {\ifglshaslong{#1}{\glentrylongplural{#1}}{\glentryfirstplural{#1}}}%#2%
5965 }
```

`\rGlsformat`

```
5966 \newcommand*{\rGlsformat}[2]{%
5967   \glsifregular{#1}
5968   {\Glsentryfirst{#1}}%
5969   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2}%
5970 }
```

`\rGlsplformat`

```
5971 \newcommand*{\rGlsplformat}[2]{%
5972   \glsifregular{#1}
5973   {\Glsentryfirstplural{#1}}%
5974   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}#2}%
5975 }
```

`\rGLSformat`

```
5976 \newcommand*{\rGLSformat}[2]{%
5977   \expandafter\mfirstucMakeUppercase\expandafter{\rGlsformat{#1}{#2}}%
5978 }
```

`\rGLSplformat`

```
5979 \newcommand*{\rGLSplformat}[2]{%
5980   \expandafter\mfirstucMakeUppercase\expandafter{\rGlsplformat{#1}{#2}}%
5981 }
```

## 1.4 Link Counting

This is different to the entry counting provided by the base package (which counts the number of times the first use flag is unset). Instead, this method hooks into `\@gls@link` (through `\glsxtr@inc@linkcount`) to increment an associated counter. To preserve resources, the counter is only defined if it needs to be incremented. This method is independent of the presence of hyperlinks. (The “link” part of the name refers to `\@gls@link` not `\hyperlink`.)

`\@inc@linkcount` This performs the actual incrementing and counter definition. The counter is given by `\c@glsxtr@linkcount@<label>` where *<label>* is the entry’s label. Since this is performed within `\@gls@link` the label can be accessed with `\glslabel`.

```
5982 \newcommand{\@glsxtr@do@inc@linkcount}{%
  Does this entry have the linkcount attribute set?
5983   \glsifattribute{\glslabel}{linkcount}{true}%
5984   {%
    Does the counter exist?
5985     \ifcsdef{c@glsxtr@linkcount@\glslabel}{}%
5986     {%
      Counter doesn't exist, so define it.
5987       \newcounter{glsxtr@linkcount@\glslabel}%
```

If linkcountmaster is set, add to counter reset.

```
5988 \glshasattribute{\glslabel}{linkcountmaster}%  
5989 {%
```

Need to ensure values are fully expanded.

```
5990 \begingroup  
5991 \edef\@glo@tmp{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%  
5992 {\glsggetattribute{\glslabel}{linkcountmaster}}}%  
5993 \@glo@tmp  
5994 }%  
5995 {}%  
5996 }%
```

Increment counter:

```
5997 \glxtrinlinkcounter{glxtr@linkcount@\glslabel}%  
5998 }%  
5999 {}%  
6000 }
```

`\refstepcounter` May be redefined to use `\refstepcounter` if required.

```
6001 \newcommand*{\glxtrinlinkcounter}[1]{\stepcounter{#1}}
```

`\linkCounterValue` Expands to the associated link counter register or 0 if not defined.

```
6002 \newcommand*{\GlsXtrLinkCounterValue}[1]{%  
6003 \ifcsundef{c@glxtr@linkcount@#1}{0}{\csname c@glxtr@linkcount@#1\endcsname}%  
6004 }
```

`\rTheLinkCounter` Expands to the display value of the associated link counter or 0 if not defined.

```
6005 \newcommand*{\GlsXtrTheLinkCounter}[1]{%  
6006 \ifcsundef{theglxtr@linkcount@#1}{0}%  
6007 {\csname theglxtr@linkcount@#1\endcsname}%  
6008 }
```

`\ifLinkCounterDef` Tests if the counter has been defined

```
6009 \newcommand*{\GlsXtrIfLinkCounterDef}[3]{%  
6010 \ifcsundef{theglxtr@linkcount@#1}{#3}{#2}%  
6011 }
```

`\LinkCounterName` Expands to the associated link counter name. (No check for existence.)

```
6012 \newcommand*{\GlsXtrLinkCounterName}[1]{glxtr@linkcount@#1}
```

`\bleLinkCounting`

```
\GlsXtrEnableLinkCounting[master counter]{categories}
```

Enable link counting for the given categories.

```
6013 \newcommand*{\GlsXtrEnableLinkCounting}[2][{}]{%
```

```

6014 \let\glxstr@inc@linkcount\@glxstr@do@inc@linkcount
6015 \@for\@glxstr@label:=#2\do
6016 {%
6017 \glsetcategoryattribute{\@glxstr@label}{linkcount}{true}%
6018 \ifstrepty{#1}{}%
6019 {%
6020 \ifcsundef{c@#1}%
6021 {\@nocounterr{#1}}%
6022 {\glsetcategoryattribute{\@glxstr@label}{linkcountmaster}{#1}}%
6023 }%
6024 }%
6025 }
6026 \@onlypreamble\GlsXtrEnableLinkCounting

```

## 1.5 Integration with glossaries-accsupp

Provide better integration with the glossaries-accsupp package. (Must be loaded before the main code of glossaries-extra either explicitly or through the accsupp package option.)

These commands have their definitions set according to whether or not glossaries-extra has been loaded.

```

6027 \@ifpackageloaded{glossaries-accsupp}
6028 {

```

Define (or redefine) commands to use the accessibility information.

**\glsaccessname** Display the name value (no link and no check for existence).

```

6029 \newcommand*\glsaccessname[1]{%
6030 \glsnameaccessdisplay
6031 {%
6032 \glstentryname{#1}%
6033 }%
6034 {#1}%
6035 }

```

**\Glsaccessname** Display the name value (no link and no check for existence) with the first letter converted to upper case.

```

6036 \newcommand*\Glsaccessname[1]{%
6037 \glsnameaccessdisplay
6038 {%
6039 \Glstentryname{#1}%
6040 }%
6041 {#1}%
6042 }

```

**\GLSaccessname** Display the name value (no link and no check for existence) converted to upper case.

```

6043 \newcommand*\GLSaccessname[1]{%
6044 \glsnameaccessdisplay
6045 {%

```

```

6046      \mfirstucMakeUppercase{\glentryname{#1}}%
6047    }%
6048    {#1}%
6049  }

```

`\glsaccesstext` Display the text value (no link and no check for existence).

```

6050  \newcommand*{\glsaccesstext}[1]{%
6051    \glstextaccessdisplay
6052    {%
6053      \glentrytext{#1}%
6054    }%
6055    {#1}%
6056  }

```

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.

```

6057  \newcommand*{\Glsaccesstext}[1]{%
6058    \glstextaccessdisplay
6059    {%
6060      \Glsentrytext{#1}%
6061    }%
6062    {#1}%
6063  }

```

`\GLSaccesstext` Display the text value (no link and no check for existence) converted to upper case.

```

6064  \newcommand*{\GLSaccesstext}[1]{%
6065    \glstextaccessdisplay
6066    {%
6067      \mfirstucMakeUppercase{\glentrytext{#1}}%
6068    }%
6069    {#1}%
6070  }

```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

6071  \newcommand*{\glsaccessplural}[1]{%
6072    \glspluralaccessdisplay
6073    {%
6074      \glentryplural{#1}%
6075    }%
6076    {#1}%
6077  }

```

`Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

6078  \newcommand*{\Glsaccessplural}[1]{%
6079    \glspluralaccessdisplay
6080    {%
6081      \Glsentryplural{#1}%

```

```

6082     }%
6083     {#1}%
6084 }

```

**GLSaccessplural** Display the plural value (no link and no check for existence) converted to upper case.

```

6085 \newcommand*{\GLSaccessplural}[1]{%
6086 \glspluralaccessdisplay
6087 {%
6088 \mfirstucMakeUppercase{\glsentryplural{#1}}%
6089 }%
6090 {#1}%
6091 }

```

**\glsaccessfirst** Display the first value (no link and no check for existence).

```

6092 \newcommand*{\glsaccessfirst}[1]{%
6093 \glsfirstaccessdisplay
6094 {%
6095 \glsentryfirst{#1}%
6096 }%
6097 {#1}%
6098 }

```

**\Glsaccessfirst** Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

6099 \newcommand*{\Glsaccessfirst}[1]{%
6100 \glsfirstaccessdisplay
6101 {%
6102 \Glsentryfirst{#1}%
6103 }%
6104 {#1}%
6105 }

```

**\GLSaccessfirst** Display the first value (no link and no check for existence) converted to upper case.

```

6106 \newcommand*{\GLSaccessfirst}[1]{%
6107 \glsfirstaccessdisplay
6108 {%
6109 \mfirstucMakeUppercase{\glsentryfirst{#1}}%
6110 }%
6111 {#1}%
6112 }

```

**cessfirstplural** Display the firstplural value (no link and no check for existence).

```

6113 \newcommand*{\glsaccessfirstplural}[1]{%
6114 \glsfirstpluralaccessdisplay
6115 {%
6116 \glsentryfirstplural{#1}%
6117 }%
6118 {#1}%
6119 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```
6120 \newcommand*{\Glsaccessfirstplural}[1]{%
6121 \glfirstpluralaccessdisplay
6122 {%
6123 \Glsentryfirstplural{#1}%
6124 }%
6125 {#1}%
6126 }
```

`cessfirstplural` Display the firstplural value (no link and no check for existence) converted to upper case.

```
6127 \newcommand*{\GLSaccessfirstplural}[1]{%
6128 \glfirstpluralaccessdisplay
6129 {%
6130 \mfirstucMakeUppercase{\Glsentryfirstplural{#1}}%
6131 }%
6132 {#1}%
6133 }
```

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

```
6134 \newcommand*{\glsaccesssymbol}[1]{%
6135 \glssymbolaccessdisplay
6136 {%
6137 \Glsentrysymbol{#1}%
6138 }%
6139 {#1}%
6140 }
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```
6141 \newcommand*{\GLSaccesssymbol}[1]{%
6142 \glssymbolaccessdisplay
6143 {%
6144 \Glsentrysymbol{#1}%
6145 }%
6146 {#1}%
6147 }
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence) converted to upper case.

```
6148 \newcommand*{\GLSaccesssymbol}[1]{%
6149 \glssymbolaccessdisplay
6150 {%
6151 \mfirstucMakeUppercase{\Glsentrysymbol{#1}}%
6152 }%
6153 {#1}%
6154 }
```

`esssymbolplural` Display the symbolplural value (no link and no check for existence).

```

6155 \newcommand*{\glsaccesssymbolplural}[1]{%
6156   \glssymbolpluralaccessdisplay
6157   {%
6158     \glsentrysymbolplural{#1}%
6159   }%
6160   {#1}%
6161 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

6162 \newcommand*{\Glsaccesssymbolplural}[1]{%
6163   \glssymbolpluralaccessdisplay
6164   {%
6165     \Glsentrysymbolplural{#1}%
6166   }%
6167   {#1}%
6168 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence) converted to upper case.

```

6169 \newcommand*{\GLSaccesssymbolplural}[1]{%
6170   \glssymbolpluralaccessdisplay
6171   {%
6172     \mfirstucMakeUppercase{\glsentrysymbolplural{#1}}%
6173   }%
6174   {#1}%
6175 }

```

`\glsaccessdesc` Display the desc value (no link and no check for existence).

```

6176 \newcommand*{\glsaccessdesc}[1]{%
6177   \glsdescriptionaccessdisplay
6178   {%
6179     \glsentrydesc{#1}%
6180   }%
6181   {#1}%
6182 }

```

`\Glsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

6183 \newcommand*{\Glsaccessdesc}[1]{%
6184   \glsdescriptionaccessdisplay
6185   {%
6186     \Glsentrydesc{#1}%
6187   }%
6188   {#1}%
6189 }

```

`\GLSaccessdesc` Display the desc value (no link and no check for existence) converted to upper case.

```

6190 \newcommand*{\GLSaccessdesc}[1]{%

```



```

6191 \glsdescriptionaccessdisplay
6192 {%
6193     \mfirstucMakeUppercase{\glsentrydesc{#1}}%
6194 }%
6195 {#1}%
6196 }

```

`\glsaccessdescplural` Display the descplural value (no link and no check for existence).

```

6197 \newcommand*{\glsaccessdescplural}[1]{%
6198     \glsdescriptionpluralaccessdisplay
6199     {%
6200         \glsentrydescplural{#1}%
6201     }%
6202     {#1}%
6203 }

```

`\Glsaccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

```

6204 \newcommand*{\Glsaccessdescplural}[1]{%
6205     \glsdescriptionpluralaccessdisplay
6206     {%
6207         \Glsentrydescplural{#1}%
6208     }%
6209     {#1}%
6210 }

```

`\glsaccessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

6211 \newcommand*{\GLSaccessdescplural}[1]{%
6212     \glsdescriptionpluralaccessdisplay
6213     {%
6214         \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
6215     }%
6216     {#1}%
6217 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

6218 \newcommand*{\glsaccessshort}[1]{%
6219     \glsshortaccessdisplay
6220     {%
6221         \glsentryshort{#1}%
6222     }%
6223     {#1}%
6224 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

6225 \newcommand*{\Glsaccessshort}[1]{%
6226     \glsshortaccessdisplay

```

```

6227     {%
6228         \Glsentryshort{#1}%
6229     }%
6230     {#1}%
6231 }

```

`\GLSaccessshort` Display the short value (no link and no check for existence) converted to upper case.

```

6232 \newcommand*{\GLSaccessshort}[1]{%
6233     \glsshortaccessdisplay
6234     {%
6235         \mfirstucMakeUppercase{\Glsentryshort{#1}}%
6236     }%
6237     {#1}%
6238 }

```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

6239 \newcommand*{\lsaccessshortpl}[1]{%
6240     \glsshortpluralaccessdisplay
6241     {%
6242         \Glsentryshortpl{#1}%
6243     }%
6244     {#1}%
6245 }

```

`lSaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

6246 \newcommand*{\lSaccessshortpl}[1]{%
6247     \glsshortpluralaccessdisplay
6248     {%
6249         \Glsentryshortpl{#1}%
6250     }%
6251     {#1}%
6252 }

```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence) converted to upper case.

```

6253 \newcommand*{\LSaccessshortpl}[1]{%
6254     \glsshortpluralaccessdisplay
6255     {%
6256         \mfirstucMakeUppercase{\Glsentryshortpl{#1}}%
6257     }%
6258     {#1}%
6259 }

```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

6260 \newcommand*{\glsaccesslong}[1]{%
6261     \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
6262 }

```

`\GLsaccesslong` Display the long form (no link and no check for existence).

```
6263
6264 \newcommand*\GLsaccesslong}[1]{%
6265   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
6266 }
```

`\GLSaccesslong` Display the long value (no link and no check for existence) converted to upper case.

```
6267 \newcommand*\GLSaccesslong}[1]{%
6268   \glslongaccessdisplay
6269   {%
6270     \mfirstucMakeUppercase{\Glsentrylong{#1}}%
6271   }%
6272   {#1}%
6273 }
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
6274 \newcommand*\glsaccesslongpl}[1]{%
6275   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
6276 }
```

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
6277
6278 \newcommand*\Glsaccesslongpl}[1]{%
6279   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
6280 }
```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```
6281 \newcommand*\GLSaccesslongpl}[1]{%
6282   \glslongpluralaccessdisplay
6283   {%
6284     \mfirstucMakeUppercase{\Glsentrylongpl{#1}}%
6285   }%
6286   {#1}%
6287 }
```

Keys for accessibility support.

```
6288 \define@key{glsxtrabrv}{access}{%
6289   \def\@gls@nameaccess{#1}%
6290 }
6291 \define@key{glsxtrabrv}{textaccess}{%
6292   \def\@gls@textaccess{#1}%
6293 }
6294 \define@key{glsxtrabrv}{pluralaccess}{%
6295   \def\@gls@pluralaccess{#1}%
6296 }
6297 \define@key{glsxtrabrv}{firstaccess}{%
6298   \def\@gls@firstaccess{#1}%
6299 }
```

```

6300 \define@key{glsxtrabrv}{firstpluralaccess}{%
6301   \def\@gls@firstpluralaccess{#1}%
6302 }
6303 \define@key{glsxtrabrv}{shortaccess}{%
6304   \def\@gls@shortaccess{#1}%
6305 }
6306 \define@key{glsxtrabrv}{shortpluralaccess}{%
6307   \def\@gls@shortaccesspl{#1}%
6308 }
6309 \define@key{glsxtrabrv}{longaccess}{%
6310   \def\@gls@longaccess{#1}%
6311 }
6312 \define@key{glsxtrabrv}{shortlonglaccess}{%
6313   \def\@gls@longaccesspl{#1}%
6314 }

```

@initaccesskeys

```

6315 \newcommand*{\@gls@initaccesskeys}{%
6316   \def\@gls@nameaccess{%
6317     \def\@gls@textaccess{%
6318       \def\@gls@pluralaccess{%
6319         \def\@gls@firstaccess{%
6320           \def\@gls@firstpluralaccess{%
6321             \def\@gls@shortaccess{%
6322               \def\@gls@shortaccesspl{%
6323                 \def\@gls@longaccess{%
6324                   \def\@gls@longaccesspl{%
6325                     }

```

ssattribute@set

$\backslash\mathrm{gls@ifaccessattribute@set}\{\langle attribute\rangle\}\{\langle true\rangle\}\{\langle false\rangle\}$
---

```

6326 \newcommand*{\@gls@ifaccessattribute@set}[3]{%
6327   \glsifcategoryattribute{\glscategorylabel}{access#1}{true}%
6328   {#2}%
6329   {%
6330     \glsifcategoryattribute{\glscategorylabel}{access#1}{false}%
6331     {#3}%
6332     {%
6333       \glsifcategoryattribute{\glscategorylabel}{#1}{true}%
6334       {#2}%
6335       {#3}%
6336     }%
6337   }%
6338 }

```

As from glossaries v4.45, the replacement text support has been corrected so that the accessibility support for abbreviations use the “E” (expanded value) element. This should actually contain the long form since it’s supposed to explain the abbreviation. This is a bit redundant on first use for styles like long-short.

aultshortaccess

```
\glsdefaultshortaccess{<long>}{<short>}
```

This command was only introduced to glossaries-accsupp 1.42 so it may not be defined.

```
6339 \def\glsdefaultshortaccess#1#2{#1 (#2)}
```

signactualsetup

```
6340 \newcommand{\glstrassignactualsetup}{%
6341 \let\@empty
6342 \let\emph\@firstofone
6343 \let\textbf\@firstofone
6344 \let\textmd\@firstofone
6345 \let\textit\@firstofone
6346 \let\textsl\@firstofone
6347 \let\textsc\@firstofone
6348 \let\textrm\@firstofone
6349 \let\textsf\@firstofone
6350 \let\texttt\@firstofone
6351 }
```

s@assign@actual

```
6352 \ifdef\pdfstringdef
6353 {
6354 \newcommand{\@gls@assign@actual}{%
6355 \begingroup
6356 \glstrassignactualsetup
6357 \pdfstringdef\@gls@actualshort{\glstrorgshort}%
6358 \pdfstringdef\@gls@actuaallong{\glstrorglong}%
6359 \pdfstringdef\@gls@actualshortpl{\@gls@shortpl}%
6360 \pdfstringdef\@gls@actuaallongpl{\@gls@longpl}%
6361 \protected@edef\@gls@tmp{\endgroup
6362 \def\noexpand\@gls@actualshort{\expandonce\@gls@actualshort}%
6363 \def\noexpand\@gls@actuaallong{\expandonce\@gls@actuaallong}%
6364 \def\noexpand\@gls@actualshortpl{\expandonce\@gls@actualshortpl}%
6365 \def\noexpand\@gls@actuaallongpl{\expandonce\@gls@actuaallongpl}%
6366 }%
6367 \@gls@tmp
6368 }
6369 }
6370 {
6371 \newcommand{\@gls@assign@actual}{%
```

```

6372 \beginngroup
6373 \glstrassignactualsetup
6374 \protected@edef\@gls@tmp{\endgroup
6375 \def\noexpand\@gls@actualshort{\glstrorgshort}%
6376 \def\noexpand\@gls@actallong{\glstrorglong}%
6377 \def\noexpand\@gls@actualshortpl{\@gls@shortpl}%
6378 \def\noexpand\@gls@actallongpl{\@gls@longpl}%
6379 }%
6380 \@gls@tmp
6381 }
6382 }

```

lt@short@access Renamed \@gls@setup@default@access and removed argument since it can be obtained from \glstrorgshort.

@default@access Assign the default value of the shortaccess key. The argument is the short value passed to \newabbreviation. The shortaccess value should explain the abbreviation.

```

6383 \newcommand{\@gls@setup@default@access}{%
6384 \@gls@assign@actual
6385 \ifdefempty\@gls@shortaccess
6386 {%

```

Check if the accessinsertdots attribute has been set but only if shortaccess hasn't been set.

```

6387 \@gls@ifaccessattribute@set{insertdots}%
6388 {%
6389 \expandafter\@glstr@insertdots\expandafter\@gls@actualshort\expandafter
6390 {\@gls@actualshort}%
6391 }%
6392 {}%
6393 \ifdefempty\@gls@longaccess
6394 {%
6395 \protected@edef\@gls@shortaccess{\glsdefaultshortaccess
6396 {\expandonce\@gls@actallong}{\expandonce\@gls@actualshort}}%
6397 }%
6398 {%
6399 \protected@edef\@gls@shortaccess{\glsdefaultshortaccess
6400 {\expandonce\@gls@longaccess}{\expandonce\@gls@actualshort}}%
6401 }%
6402 \eappto\ExtraCustomAbbreviationFields{shortaccess={\@gls@shortaccess},}%

```

If shortaccessplural hasn't been set, assign plural form.

```

6403 \ifdefempty\@gls@shortaccesspl
6404 {%
6405 \@gls@ifaccessattribute@set{aposplural}%
6406 {%
6407 \expandafter\def\expandafter\@gls@shortaccesspl\expandafter{%
6408 \@gls@actualshort'\glstrabbrvpluralsuffix}%
6409 }%
6410 {%

```

```

6411         \@gls@ifaccessattribute@set{noshortplural}%
6412     {%
6413         \let\@gls@shortaccesspl\@gls@shortaccess
6414     }%
6415     {%
6416         \let\@gls@shortaccesspl\@gls@actualshortpl
6417     }%
6418 }%
6419 \ifdefempty\@gls@longaccesspl
6420 {%
6421     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6422         {\expandonce\@gls@actuallongpl}{\expandonce\@gls@actualshortpl}}%
6423 }%
6424 {%
6425     \protected@edef\@gls@shortaccesspl{\glsdefaultshortaccess
6426         {\expandonce\@gls@longaccesspl}{\expandonce\@gls@actualshort}}%
6427 }%
6428 \eappto\ExtraCustomAbbreviationFields{shortpluralaccess={\@gls@shortaccesspl},}%
6429 }%
6430 {}%
6431 }%
6432 {%
6433     \ifdefempty\@gls@shortaccesspl
6434     {\let\@gls@shortaccesspl\@gls@shortaccess}%
6435     {}%
6436 }%

```

If access key hasn't been set, check if the nameshortaccess attribute has been set.

```

6437 \ifdefempty\@gls@nameaccess
6438 {%
6439     \glsifcategoryattribute{\glscategorylabel}{nameshortaccess}{true}%
6440     {%
6441         \eappto\ExtraCustomAbbreviationFields{access={\@gls@shortaccess},}%
6442     }%
6443     {}%
6444 }%
6445 {}%

```

If textaccess key hasn't been set, check if the textshortaccess attribute has been set.

```

6446 \ifdefempty\@gls@textaccess
6447 {%
6448     \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6449     {%
6450         \eappto\ExtraCustomAbbreviationFields{textaccess={\@gls@shortaccess},}%
6451     }%
6452     {}%
6453 }%
6454 {}%
6455 \ifdefempty\@gls@pluralaccess

```

```

6456   {%
6457     \glsifcategoryattribute{\glscategorylabel}{textshortaccess}{true}%
6458     {%
6459       \eappto\ExtraCustomAbbreviationFields{%
6460         pluralaccess={\@gls@shortaccesspl},%
6461       }%
6462     }%
6463   }%
6464 }%
6465 {}%

  If firstaccess key hasn't been set, check if the firstshortaccess attribute has been set.
6466   \ifdefempty\@gls@firstaccess
6467   {%
6468     \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6469     {%
6470       \eappto\ExtraCustomAbbreviationFields{firstaccess={\@gls@shortaccess},}%
6471     }%
6472   }%
6473 }%
6474 {}%
6475   \ifdefempty\@gls@firstpluralaccess
6476   {%
6477     \glsifcategoryattribute{\glscategorylabel}{firstshortaccess}{true}%
6478     {%
6479       \eappto\ExtraCustomAbbreviationFields{%
6480         firstpluralaccess={\@gls@shortaccesspl},%
6481       }%
6482     }%
6483   }%
6484 }%
6485 {}%
6486 }

```

Provide hooks for `\setabbreviationstyle` that automatically set the attributes appropriate for the style. If the name is just the short form and the description contains the long form, then it may not be necessary to set `nameshortaccess` but it would depend on the glossary style. Need to provide `\glstr<category><field>accsupp` if not already defined.

provideaccsuppcmd

```

6487   \newcommand*{\glstrprovideaccsuppcmd}[2] {%
6488     \ifcsundef{glstr#1#2accsupp}%
6489     {\csdef{glstr#1#2accsupp}{\glsshortaccsupp}}%
6490     {}%
6491   }

```

rrSetNoLongAttrs For styles where the name, first and text are just the abbreviation.

```

6492   \newcommand*{\glstrAccSuppAbbrSetNoLongAttrs}[1] {%
6493     \glssetcategoryattribute{#1}{nameshortaccess}{true}%

```



```

6494 \glssetcategoryattribute{#1}{firstshortaccess}{true}%
6495 \glssetcategoryattribute{#1}{textshortaccess}{true}%
6496 \glsxtrprovideaccsuppcmd{#1}{name}%
6497 \glsxtrprovideaccsuppcmd{#1}{first}%
6498 \glsxtrprovideaccsuppcmd{#1}{firstpl}%
6499 \glsxtrprovideaccsuppcmd{#1}{text}%
6500 \glsxtrprovideaccsuppcmd{#1}{plural}%
6501 }

```

**tFirstLongAttrs** For styles where the name and text are just the abbreviation. The first form may just be long or may be short and long.

```

6502 \newcommand*{\glsxtrAccSuppAbbrSetFirstLongAttrs}[1]{%
6503 \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6504 \glssetcategoryattribute{#1}{textshortaccess}{true}%
6505 \glsxtrprovideaccsuppcmd{#1}{name}%
6506 \glsxtrprovideaccsuppcmd{#1}{text}%
6507 \glsxtrprovideaccsuppcmd{#1}{plural}%
6508 }

```

**tTextShortAttrs** For styles where only the text is just the abbreviation. The name and first form may just be long or may be short and long. The name may also be short but followed by the long form in the description.

```

6509 \newcommand*{\glsxtrAccSuppAbbrSetTextShortAttrs}[1]{%
6510 \glssetcategoryattribute{#1}{textshortaccess}{true}%
6511 \glsxtrprovideaccsuppcmd{#1}{text}%
6512 \glsxtrprovideaccsuppcmd{#1}{plural}%
6513 }

```

**tNameShortAttrs** For styles where only the name is just the abbreviation. The first and subsequent form may just be long or may be short and long.

```

6514 \newcommand*{\glsxtrAccSuppAbbrSetNameShortAttrs}[1]{%
6515 \glssetcategoryattribute{#1}{nameshortaccess}{true}%
6516 \glsxtrprovideaccsuppcmd{#1}{name}%
6517 }

```

**etNameLongAttrs** For styles where the first and text are just the abbreviation. The name may just be long or may be short and long or the name may be short.

```

6518 \newcommand*{\glsxtrAccSuppAbbrSetNameLongAttrs}[1]{%
6519 \glssetcategoryattribute{#1}{firstshortaccess}{true}%
6520 \glssetcategoryattribute{#1}{textshortaccess}{true}%
6521 \glsxtrprovideaccsuppcmd{#1}{first}%
6522 \glsxtrprovideaccsuppcmd{#1}{firstpl}%
6523 \glsxtrprovideaccsuppcmd{#1}{text}%
6524 \glsxtrprovideaccsuppcmd{#1}{plural}%
6525 }

```

End of if accsupp part

```

6526 }
6527 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

`\glsaccessname` Display the name value (no link and no check for existence).  
6528 `\newcommand*{\glsaccessname}[1]{\glsentryname{#1}}`

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.  
6529 `\newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}`

`\GLSaccessname` Display the name value (no link and no check for existence). converted to upper case.  
6530 `\newcommand*{\GLSaccessname}[1]{%`  
6531 `\protect\mfirstucMakeUppercase{\glsentryname{#1}}}`

`\glsaccesstext` Display the text value (no link and no check for existence).  
6532 `\newcommand*{\glsaccesstext}[1]{\glsentrytext{#1}}`

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.  
6533 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.  
6534 `\newcommand*{\GLSaccesstext}[1]{%`  
6535 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).  
6536 `\newcommand*{\glsaccessplural}[1]{\glsentryplural{#1}}`

`Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.  
6537 `\newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.  
6538 `\newcommand*{\GLSaccessplural}[1]{%`  
6539 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).  
6540 `\newcommand*{\glsaccessfirst}[1]{\glsentryfirst{#1}}`

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.  
6541 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.  
6542 `\newcommand*{\GLSaccessfirst}[1]{%`  
6543 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).  
6544 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`cessfirstplural`    Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.  
6545    `\newcommand*{\GLsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

`cessfirstplural`    Display the firstplural value (no link and no check for existence). converted to upper case.  
6546    `\newcommand*{\GLSaccessfirstplural}[1]{%`  
6547    `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`glsaccesssymbol`    Display the symbol value (no link and no check for existence).  
6548    `\newcommand*{\glsaccesssymbol}[1]{\glsentrysymbol{#1}}`

`GLsaccesssymbol`    Display the symbol value (no link and no check for existence) with the first letter converted to upper case.  
6549    `\newcommand*{\GLsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

`GLSaccesssymbol`    Display the symbol value (no link and no check for existence). converted to upper case.  
6550    `\newcommand*{\GLSaccesssymbol}[1]{%`  
6551    `\protect\mfirstucMakeUppercase{\glsentrysymbol{#1}}}`

`esssymbolplural`    Display the symbolplural value (no link and no check for existence).  
6552    `\newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}`

`esssymbolplural`    Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.  
6553    `\newcommand*{\GLsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`esssymbolplural`    Display the symbolplural value (no link and no check for existence). converted to upper case.  
6554    `\newcommand*{\GLSaccesssymbolplural}[1]{%`  
6555    `\protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}`

`\glsaccessdesc`    Display the desc value (no link and no check for existence).  
6556    `\newcommand*{\glsaccessdesc}[1]{\glsentrydesc{#1}}`

`\GLsaccessdesc`    Display the desc value (no link and no check for existence) with the first letter converted to upper case.  
6557    `\newcommand*{\GLsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc`    Display the desc value (no link and no check for existence). converted to upper case.  
6558    `\newcommand*{\GLSaccessdesc}[1]{%`  
6559    `\protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}`

`ccessdescplural`    Display the descplural value (no link and no check for existence).  
6560    `\newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}`

ccessdesclplural    Display the desclplural value (no link and no check for existence) with the first letter converted to upper case.

6561    \newcommand\*{\GLsaccessdesclplural}[1]{\Glsentrydesclplural{#1}}

ccessdesclplural    Display the desclplural value (no link and no check for existence). converted to upper case.

6562    \newcommand\*{\GLSaccessdesclplural}[1]{%

6563    \protect\mfirstucMakeUppercase{\glsentrydesclplural{#1}}}

\glsaccessshort    Display the short form (no link and no check for existence).

6564    \newcommand\*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort    Display the short form with first letter converted to uppercase (no link and no check for existence).

6565    \newcommand\*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort    Display the short value (no link and no check for existence). converted to upper case.

6566    \newcommand\*{\GLSaccessshort}[1]{%

6567    \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl    Display the short plural form (no link and no check for existence).

6568    \newcommand\*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

lsaccessshortpl    Display the short plural form with first letter converted to uppercase (no link and no check for existence).

6569    \newcommand\*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}

LSaccessshortpl    Display the shortplural value (no link and no check for existence). converted to upper case.

6570    \newcommand\*{\GLSaccessshortpl}[1]{%

6571    \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}

\glsaccesslong    Display the long form (no link and no check for existence).

6572    \newcommand\*{\glsaccesslong}[1]{\glsentrylong{#1}}

\Glsaccesslong    Display the long form (no link and no check for existence).

6573    \newcommand\*{\Glsaccesslong}[1]{\Glsentrylong{#1}}

\GLSaccesslong    Display the long value (no link and no check for existence). converted to upper case.

6574    \newcommand\*{\GLSaccesslong}[1]{%

6575    \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}

glsaccesslongpl    Display the long plural form (no link and no check for existence).

6576    \newcommand\*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}

Glsaccesslongpl    Display the long plural form (no link and no check for existence).

6577    \newcommand\*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}

GLSaccesslongpl Display the longplural value (no link and no check for existence). converted to upper case.

```
6578 \newcommand*{\GLSaccesslongpl}[1]{%
6579 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

@initaccesskeys This does nothing if there's no accessibility support.

```
6580 \newcommand*{\@gls@initaccesskeys}{}
```

@default@access This does nothing if there's no accessibility support.

```
6581 \newcommand{\@gls@setup@default@access}{}
```

rSetNoLongAttrs This does nothing if there's no accessibility support.

```
6582 \newcommand*{\glsextrAccSuppAbbrSetNoLongAttrs}[1]{}
```

tFirstLongAttrs This does nothing if there's no accessibility support.

```
6583 \newcommand*{\glsextrAccSuppAbbrSetFirstLongAttrs}[1]{}
```

tTextShortAttrs This does nothing if there's no accessibility support.

```
6584 \newcommand*{\glsextrAccSuppAbbrSetTextShortAttrs}[1]{}
```

tNameShortAttrs This does nothing if there's no accessibility support.

```
6585 \newcommand*{\glsextrAccSuppAbbrSetNameShortAttrs}[1]{}
```

etNameLongAttrs This does nothing if there's no accessibility support.

```
6586 \newcommand*{\glsextrAccSuppAbbrSetNameLongAttrs}[1]{}
```

End of else part

```
6587 }
```

## 1.6 Categories

\glscategory Add a new storage key that can be used to indicate a category. The default category is general.

```
6588 \glsaddstoragekey{category}{general}{\glscategory}
```

\glsifcategory Convenient shortcut to determine if an entry has the given category.

```
6589 \newcommand{\glsifcategory}[4]{%
6590 \ifglstfield{#1}{category}{#2}{#3}{#4}%
6591 }
```

Categories can have attributes.

tegorattribute

```
\glissetcategoryattribute{<category>}{<attribute-label>}{<value>}
```

Set (or override if already set) an attribute for the given category.

```
6592 \newcommand*{\glissetcategoryattribute}[3]{%
6593 \csdef{\glsextr@categoryattr@@#1@#2}{#3}%
6594 }
```

tegrityattribute

```
\glsggetcategoryattribute{<category>}{<attribute-label>}
```

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
6595 \newcommand*{\glsggetcategoryattribute}[2]{%  
6596   \csuse{@glsxtr@categoryattr@@#1@#2}%  
6597 }
```

tegrityattribute

```
\glgsunsetcategoryattribute{<category>}{<attribute-label>}
```

Unsets the given attribute for the given category.

```
6598 \newcommand*{\glgsunsetcategoryattribute}[2]{%  
6599   \csundef{@glsxtr@categoryattr@@#1@#2}%  
6600 }
```

tegrityattribute

```
\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}
```

Tests if the category has the given attribute set.

```
6601 \newcommand*{\glshascategoryattribute}[4]{%  
6602   \ifcvoid{@glsxtr@categoryattr@@#1@#2}{#4}{#3}%  
6603 }
```

glissetattribute

```
\glissetattribute{<entry label>}{<attribute-label>}{<value>}
```

Short cut where the category label is obtained from the entry information.

```
6604 \newcommand*{\glissetattribute}[3]{%  
6605   \glsgsetcategoryattribute{\glscategory{#1}}{#2}{#3}%  
6606 }
```

glsggetattribute

```
\glsggetattribute{<entry label>}{<attribute-label>}
```

Short cut where the category label is obtained from the entry information.

```

6607 \newcommand*\glsggetattribute}[2]{%
6608   \glsggetcategoryattribute{\glscategory{#1}}{#2}%
6609 }

```

glshasattribute

```
\glshasattribute{<entry label>}{<attribute-label>}{<true>}{<false>}
```

Short cut to test if the given attribute has been set where the category label is obtained from the entry information.

```

6610 \newcommand*\glshasattribute}[4]{%
6611   \ifglsgentryexists{#1}%
6612   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
6613   {#4}%
6614 }

```

categoryattribute

```
\glsgcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}
```

True if category has the attribute with the given value.

```

6615 \newcommand{\glsgcategoryattribute}[5]{%
6616   \ifcsundef{@glsgxtr@categoryattr@#1@#2}%
6617   {#5}%
6618   {\ifcsstring{@glsgxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
6619 }

```

\glsgifattribute

```
\glsgifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}
```

Short cut to determine if the given entry has a category with the given attribute set.

```

6620 \newcommand{\glsgifattribute}[5]{%
6621   \ifglsgentryexists{#1}%
6622   {\glsgcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
6623   {#5}%
6624 }

```

Set attributes for the default general category:

```
6625 \glsgsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
6626 \glsgsetcategoryattribute{acronym}{regular}{true}
```

regularcategory Convenient shortcut to add the regular attribute.

```
6627 \newcommand*\glsetregularcategory[1]{%
6628   \glsetcategoryattribute{#1}{regular}{true}%
6629 }
```

regularcategory

`\gl@ifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
6630 \newcommand{\gl@ifregularcategory}[3]{%
6631   \gl@ifcategoryattribute{#1}{regular}{true}{#2}{#3}%
6632 }
```

regularcategory

`\gl@ifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
6633 \newcommand{\gl@ifnotregularcategory}[3]{%
6634   \gl@ifcategoryattribute{#1}{regular}{false}{#2}{#3}%
6635 }
```

\gl@ifregular

`\gl@ifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
6636 \newcommand{\gl@ifregular}[3]{%
6637   \gl@ifregularcategory{\glscategory{#1}}{#2}{#3}%
6638 }
```

gl@ifnotregular

`\gl@ifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
6639 \newcommand{\gl@ifnotregular}[3]{%
6640   \gl@ifnotregularcategory{\glscategory{#1}}{#2}{#3}%
6641 }
```



reachincategory

```
\glsforeachincategory[<glossary labels>]{<category-label>}  
{<glossary-cs>}{<label-cs>}{<body>}
```

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category matches *<category-label>*. The control sequences *<glossary-cs>* and *<label-cs>* may be used in *<body>* to access the glossary label and entry label for the current iteration.

```
6642 \newcommand{\glsforeachincategory}[5][\@glo@types]{%  
6643   \forallglossaries[#1]{#3}%  
6644   {%  
6645     \forlgsentries[#3]{#4}%  
6646     {%  
6647       \glsifcategory{#4}{#2}{#5}{}%  
6648     }%  
6649   }%  
6650 }
```

chwithattribute

```
\glsforeachwithattribute[<glossary labels>]{<attribute-label>}  
{<attribute-value>}{<glossary-cs>}{<label-cs>}{<body>}
```

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category attribute *<attribute-label>* matches *<attribute-value>*. The control sequences *<glossary-cs>* and *<label-cs>* may be used in *<body>* to access the glossary label and entry label for the current iteration.

```
6651 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%  
6652   \forallglossaries[#1]{#4}%  
6653   {%  
6654     \forlgsentries[#4]{#5}%  
6655     {%  
6656       \glsifattribute{#5}{#2}{#3}{#6}{}%  
6657     }%  
6658   }%  
6659 }
```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```
6660 \ifdef\newterm  
6661 {%
```

`\newterm`

```
6662   \renewcommand*{\newterm}[2][ ]{%
```

```

6663 \newglossaryentry{#2}%
6664 {type={index},category=index,name={#2},%
6665   description={\glstrpostdescription\nopostdesc},#1}%
6666 }

```

Indexed terms are regular by default.

```

6667 \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

6668 \newcommand*{\glstrpostdescindex}{}

6669 }
6670 {}

```

If the symbols package option was used, define a similar command for symbols, but set the default sort to the label rather than the name as the symbols will typically contain commands that will confuse makeindex and xindy.

```

6671 \ifdef\printsymbols
6672 {%

```

glstrnewsymbol Unlike \newterm, this has a separate argument for the label (since the symbol will likely contain commands).

```

6673 \newcommand*{\glstrnewsymbol}[3] [] {%
6674   \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
6675 }

```

Symbols are regular by default.

```

6676 \glsssetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

6677 \newcommand*{\glstrpostdescsymbol}{}

6678 }
6679 {}

```

Similar for the numbers option.

```

6680 \ifdef\printnumbers
6681 {%

```

glstrnewnumber

```

6682 \ifdef\printnumbers
6683 \newcommand*{\glstrnewnumber}[3] [] {%
6684   \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
6685 }

```

Numbers are regular by default.

```

6686 \glsssetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```
6687 \newcommand*{\glxtrpostdescnumber}{}
6688 }
6689 }
```

setcategory Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
6690 \newcommand*{\glxtrsetcategory}[2]{%
6691   \@for\@glxtr@label:=#1\do
6692   {%
6693     \glsfieldxddef{\@glxtr@label}{category}{#2}%
6694   }%
6695 }
```

categoryforall Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```
6696 \newcommand*{\glxtrsetcategoryforall}[2]{%
6697   \forallglossaries[#1]{\@glxtr@type}{%
6698     \forallsentries[\@glxtr@type]{\@glxtr@label}%
6699     {%
6700       \glsfieldxddef{\@glxtr@label}{category}{#2}%
6701     }%
6702   }%
6703 }
```

rfieldtitlecase

`\glxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```
6704 \newcommand*{\glxtrfieldtitlecase}[2]{%
6705   \expandafter\glxtrfieldtitlecasecs\expandafter
6706   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
6707 }
```

fieldtitlecasecs The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`. Check for `\glscapitalisewords`, which was added to glossaries v4.48.

```
6708 \ifdef\glscapitalisewords
6709 {
6710   \newcommand*{\glxtrfieldtitlecasecs}[1]{%
6711     \expandafter\glscapitalisewords\expandafter{#1}}
6712 }
6713 {
6714   \newcommand*{\glxtrfieldtitlecasecs}[1]{\xcapitalisewords{#1}}
6715 }
```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

`\glossentrydesc` If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

6716 \ifpackageloaded{glossaries-accsupp}
6717 {
6718   \renewcommand*{\glossentrydesc}[1]{%
6719     \glsdoifexistsorwarn{#1}%
6720     {%
6721       \glssetabbrvfmt{\glscategory{#1}}%
        As from version 1.04, allow the glossdescfont attribute to determine the font applied.
6722       \glshasattribute{#1}{glossdescfont}%
6723       {%
6724         \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
6725         \ifcsdef{\@glxtr@attrval}%
6726         {%
6727           \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
6728         }%
6729         {%
6730           \GlossariesExtraWarning{Unknown control sequence name
6731             ‘\@glxtr@attrval’ supplied in glossdescfont attribute
6732             for entry ‘#1’. Ignoring}%
6733           \let\@glxtr@glossdescfont\@firstofone
6734         }%
6735       }%
6736       {\let\@glxtr@glossdescfont\@firstofone}%
6737       \glsifattribute{#1}{glossdesc}{firstuc}%
6738       {%
6739         \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
6740       }%
6741       {%
6742         \glsifattribute{#1}{glossdesc}{title}%
6743         {%
6744           \@glxtr@do@titlecaps@warn
6745           \glsdescriptionaccessdisplay
6746           {%
6747             \@glxtr@glossdescfont{\@glxtr@fieldtitlecase{#1}{desc}}%
6748           }%
6749           {#1}%
6750         }%
6751         {%
6752           \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
6753         }%
6754       }%
6755     }%
6756   }
6757 }
6758 {

```

```

6759 \renewcommand*{\glossentrydesc}[1]{%
6760   \glsdoifexistsorwarn{#1}%
6761   {%
6762     \glssetabbrvfmt{\glscategory{#1}}%
6763     \glsattribute{#1}{glossdescfont}%
6764     {%
6765       \protected@edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
6766       \ifcsdef{\@glxtr@attrval}%
6767       {%
6768         \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
6769       }%
6770       {%
6771         \GlossariesExtraWarning{Unknown control sequence name
6772           '\@glxtr@attrval' supplied in glossdescfont attribute
6773           for entry '#1'. Ignoring}%
6774         \let\@glxtr@glossdescfont\@firstofone
6775       }%
6776     }%
6777     {\let\@glxtr@glossdescfont\@firstofone}%
6778     \glsifattribute{#1}{glossdesc}{firstuc}%
6779     {%
6780       \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
6781     }%
6782     {%
6783       \glsifattribute{#1}{glossdesc}{title}%
6784     }%
6785     \@glxtr@do@titlecaps@warn
6786     \@glxtr@glossdescfont{\@glxtr@field@titlecase{#1}{desc}}%
6787     }%
6788     {%
6789       \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
6790     }%
6791     }%
6792   }%
6793 }
6794 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

6795 \@ifpackageloaded{glossaries-accsupp}
6796 {
6797   \renewcommand*{\glossentryname}[1]{%
6798     \glsdoifexistsorwarn{#1}%
6799     {%
6800       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

6801     \glsattribute{#1}{glossnamefont}%
6802     {%

```

```

6803 \protected@edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
6804 \ifcsdef{\@glxtr@attrval}%
6805 {%
6806 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6807 }%
6808 {%
6809 \GlossariesExtraWarning{Unknown control sequence name
6810 '\@glxtr@attrval' supplied in glossnamefont attribute
6811 for entry '#1'. Reverting to default \string\glsnamefont}%
6812 \let\@glxtr@glossnamefont\glsnamefont
6813 }%
6814 }%
6815 {\let\@glxtr@glossnamefont\glsnamefont}%
6816 \glsifattribute{#1}{glossname}{firstuc}%
6817 {%
6818 \glsnameaccessdisplay
6819 {%
6820 \@glxtr@glossnamefont{\Glsentryname{#1}}%
6821 }%
6822 {#1}%
6823 }%
6824 {%
6825 \glsifattribute{#1}{glossname}{title}%
6826 {%
6827 \@glxtr@do@titlecaps@warn
6828 \glsnameaccessdisplay
6829 {%
6830 \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
6831 }%
6832 {#1}%
6833 }%
6834 {%
6835 \glsifattribute{#1}{glossname}{uc}%
6836 {%
6837 \glsnameaccessdisplay
6838 {%

```

Hide the label from the upper-casing command.

```

6839 \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
6840 \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
6841 }%
6842 {#1}%
6843 }%
6844 {%
6845 \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
6846 \glsnameaccessdisplay
6847 {%
6848 \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
6849 }%
6850 {#1}%

```

```

6851         }%
6852     }%
6853 }%

    Do post-name hook:
6854     \glstrpostnamehook{#1}%
6855 }%
6856 }
6857 }
6858 {
6859     \renewcommand*{\glossentryname}[1]{%
6860         \@glsdoifexistsorwarn{#1}%
6861         {%
6862             \glssetabbrvfmt{\glscategory{#1}}%
6863             \glsattribute{#1}{glossnamefont}%
6864             {%
6865                 \protected@edef\@glstr@attrval{\glsattribute{#1}{glossnamefont}}%
6866                 \ifcsdef{\@glstr@attrval}%
6867                 {%
6868                     \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
6869                 }%
6870                 {%
6871                     \GlossariesExtraWarning{Unknown control sequence name
6872                     '\@glstr@attrval' supplied in glossnamefont attribute
6873                     for entry '#1'. Reverting to default \string\glsnamefont}%
6874                     \let\@glstr@glossnamefont\glsnamefont
6875                 }%
6876             }%
6877             {\let\@glstr@glossnamefont\glsnamefont}%
6878             \glsifattribute{#1}{glossname}{firstuc}%
6879             {%
6880                 \@glstr@glossnamefont{\Glsentryname{#1}}%
6881             }%
6882             {%
6883                 \glsifattribute{#1}{glossname}{title}%
6884                 {%
6885                     \@glstr@do@titlecaps@warn
6886                     \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
6887                 }%
6888                 {%
6889                     \glsifattribute{#1}{glossname}{uc}%
6890                     {%

```

Hide the label from the upper-casing command.

```

6891         \letcs{\glo@name}{\glo\glsdetoklabel{#1}@name}%
6892         \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
6893     }%
6894 }%

```

This little trick is used by glossaries to allow the user to redefine `\glsnamefont` to use `\makefirstuc`. Support it even though they can now use the `firstuc` attribute.

```
6895         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
6896         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
6897     }%
6898 }%
6899 }%
```

Do post-name hook.

```
6900     \glxtrpostnamehook{#1}%
6901 }%
6902 }
6903 }
```

`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
6904 \ifpackageloaded{glossaries-accsupp}
6905 {
6906     \renewcommand*{\Glossentryname}[1]{%
6907         \@glstdoifexistsorwarn{#1}%
6908         {%
6909             \glsssetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
6910     \glshasattribute{#1}{glossnamefont}%
6911     {%
6912         \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
6913         \ifcsdef{\@glxtr@attrval}%
6914         {%
6915             \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6916         }%
6917         {%
6918             \GlossariesExtraWarning{Unknown control sequence name
6919                 '\@glxtr@attrval' supplied in glossnamefont attribute
6920                 for entry '#1'. Reverting to default \string\glsnamefont}%
6921             \let\@glxtr@glossnamefont\glsnamefont
6922         }%
6923     }%
6924     {\let\@glxtr@glossnamefont\glsnamefont}%
6925     \glsnameaccessdisplay
6926     {%
6927         \@glxtr@glossnamefont{\Glsentryname{#1}}%
6928     }%
6929     {#1}%
```

Do post-name hook:

```
6930     \glxtrpostnamehook{#1}%
6931 }%
6932 }
6933 }
```



```

6934 {
6935   \renewcommand*{\Glossentryname}[1]{%
6936     \@glsdoifexistsorwarn{#1}%
6937     {%
6938       \glssetabbrvfmt{\glscategory{#1}}%
6939       \glsattribute{#1}{glossnamefont}%
6940       {%
6941         \protected@edef\@glxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
6942         \ifcsdef{\@glxtr@attrval}%
6943         {%
6944           \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
6945           }%
6946         {%
6947           \GlossariesExtraWarning{Unknown control sequence name
6948             '\@glxtr@attrval' supplied in glossnamefont attribute
6949             for entry '#1'. Reverting to default \string\glsnamefont}%
6950           \let\@glxtr@glossnamefont\glsnamefont
6951           }%
6952         }%
6953       {\let\@glxtr@glossnamefont\glsnamefont}%
6954       \@glxtr@glossnamefont{\Glsentryname{#1}}%
6955     }%
6956   }%
6957 }
6958 }

```

Do post-name hook:

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`\glstrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```

6959 \newcommand*{\glstrpostnamehook}[1]{%
6960   \let\@glsnumberformat\@glxtr@defaultnumberformat
6961   \glstrdoautoindexname{#1}{indexname}%

```

Allow additional code regardless of category:

```

6962   \glsextrapostnamehook{#1}%

```

Allow categories to hook in here.

```

6963   \csuse{glstrpostname\glscategory{#1}}%
6964 }

```

`\glstrpostnamehook`

```

6965 \newcommand*{\glsextrapostnamehook}[1]{%

```

`\glsdefpostname` Provide a convenient command for defining the post-name hook for the given category.

```

6966 \newcommand*{\glsdefpostname}[2]{%

```

```

6967 \csdef{glxstrpostname#1}{#2}%
6968 }

```

etaccessdisplay

```

6969 \@ifpackageloaded{glossaries-accsupp}
6970 {
6971   \newcommand*{\glxtr@setaccessdisplay}[1]{%
6972     \ifcsdef{glx#1accessdisplay}%
6973       {\letcs\glxtr@accessdisplay{glx#1accessdisplay}}%
6974       {%

```

This is essentially the reverse of `\@glx@fetchfield`, since the field supplied to `\glossentryname` has to be the internal label, but the `\glx{field}accessdisplay` commands use the key name.

```

6975     \protected@edef\@glx@thisval{#1}%
6976     \@for\@glx@map:=\@glx@keymap\do{%
6977       \protected@edef\@this@key{\expandafter\@secondoftwo\@glx@map}%
6978       \ifdefequal{\@this@key}{\@glx@thisval}%
6979       {%
6980         \protected@edef\@glx@thisval{\expandafter\@firstoftwo\@glx@map}%
6981         \@endfortrue
6982       }%
6983     }%
6984   }%
6985   \ifcsdef{glx\@glx@thisval accessdisplay}%
6986     {\letcs\glxtr@accessdisplay{glx\@glx@thisval accessdisplay}}%
6987     {\let\glxtr@accessdisplay\@firstoftwo}%
6988   }%
6989 }
6990 }
6991 {%
6992   \newcommand*{\glxtr@setaccessdisplay}[1]{%
6993     \let\glxtr@accessdisplay\@firstoftwo
6994   }

```

sentrynameother Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

6995 \newrobustcmd*{\glossentrynameother}[2]{%
6996   \@glxdoifexistsorwarn{#1}%
6997   {%

```

Accessibility support:

```

6998   \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

6999   \glxsetabbrvfmt{\glscategory{#1}}%
7000   \glxhasattribute{#1}{glossnamefont}%
7001   {%
7002     \protected@edef\@glxtr@attrval{\glxgetattribute{#1}{glossnamefont}}%
7003     \ifcsdef{\@glxtr@attrval}%

```

```

7004      {%
7005      \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
7006      }%
7007      {%
7008      \GlossariesExtraWarning{Unknown control sequence name
7009      '\@glsxtr@attrval' supplied in glossnamefont attribute
7010      for entry '#1'. Reverting to default \string\glsnamefont}%
7011      \let\@glsxtr@glossnamefont\glsnamefont
7012      }%
7013      }%
7014      {\let\@glsxtr@glossnamefont\glsnamefont}%
7015      \glsifattribute{#1}{glossname}{firstuc}%
7016      {%
7017      \@glsxtr@accessdisplay
7018      {\@glsxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
7019      {#1}%
7020      }%
7021      {%
7022      \glsifattribute{#1}{glossname}{title}%
7023      {%
7024      \@glsxtr@do@titlecaps@warn
7025      \@glsxtr@accessdisplay
7026      {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
7027      {#1}%
7028      }%
7029      {%
7030      \glsifattribute{#1}{glossname}{uc}%
7031      {%
7032      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
7033      \@glsxtr@accessdisplay
7034      {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
7035      {#1}%
7036      }%
7037      {%
7038      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
7039      \@glsxtr@accessdisplay
7040      {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
7041      {#1}%
7042      }%
7043      }%
7044      }%

```

Do post-name hook.

```

7045      \glsxtrpostnamehook{#1}%
7046      }%
7047 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

7048 \newif\if@glsxtr@format@override
7049 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

7050 \@ifpackageloaded{hyperref}
7051 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
7052   \ifHy@hyperindex
7053     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7054       \@glstr@format@override true
7055       \appto\theindex{\let\glshypernumber\@firstofone}%
7056     }
7057   \else
7058     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7059       \@glstr@format@override true
7060       \appto\theindex{\let\glshypernumber\hyperpage}%
7061     }
7062   \fi
7063 }
7064 {
7065   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
7066     \@glstr@format@override true
7067   }
7068 }
7069 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

7070 \newcommand*{\glstrdoautoindexname}[2]{%
7071   \glshasattribute{#1}{#2}%
7072   {%
    Escape any makeindex/xindy characters in the value of the name field. Take care with babel
    as this won't work if the category code has changed for those characters.
7073     \@glstr@autoindex@setname{#1}%
    If the attribute value is simply "true" don't add an encap, otherwise use the value as the encap.
7074     \protected@edef\@glstr@attrval{\glsggetattribute{#1}{#2}}%
7075     \if@glstr@format@override
7076       \ifx\@glstrnumberformat\@glstr@defaultnumberformat
7077       \else
7078         \let\@glstr@attrval\@glstrnumberformat
7079       \fi
7080     \fi
7081     \ifdefstring{\@glstr@attrval}{true}%
7082     {}%
7083     {\protected@eappto\@glo@name{\@glstr@autoindex@encap\@glstr@attrval}}%

```

```

7084 \expandafter\glxstrautoindex\expandafter{\@glo@name}%
7085 }%
7086 {}%
7087 }

glxstrautoindex
7088 \newcommand*{\glxstrautoindex}{\index}

xtrautoindexesc
7089 \newcommand{\glxstrautoindexesc}{%
7090 \@gls@checkmkidxchars\@glo@sort
7091 \@glxstr@autoindex@doextra@esc\@glo@sort
7092 }

toindex@setname Assign \@glo@name for use with indexname attribute.
7093 \newcommand*{\@glxstr@autoindex@setname}[1]{%
7094 \protected@edef\@glo@name{\glxstrautoindexentry{#1}}%
7095 \glxstrautoindexassignsort{\@glo@sort}{#1}%
7096 \glxstrautoindexesc
7097 \epreto\@glo@name{\@glo@sort\@glxstr@autoindex@at}%
7098 }

rautoindexentry Command used for the actual part when auto-indexing.
7099 \newcommand*{\glxstrautoindexentry}[1]{\string\glstryname{#1}}

indexassignsort Used to assign the sort value when auto-indexing.
7100 \newcommand*{\glxstrautoindexassignsort}[2]{%
7101 \glsletentryfield{#1}{#2}{sort}%
7102 }

dex@doextra@esc
7103 \newcommand*{\@glxstr@autoindex@doextra@esc}[1]{%
Escape the escape character unless it has already been escaped.
7104 \ifx\@glxstr@autoindex@esc\@gls@quotechar
7105 \else
7106 \def\@gls@checkedmkidx{}%
7107 \edef\@glxstr@checkspch{%
7108 \noexpand\@glxstr@autoindex@escquote\expandonce{#1}%
7109 \noexpand\@empty\@glxstr@autoindex@esc\noexpand\@nnil
7110 \@glxstr@autoindex@esc\noexpand\@empty\noexpand\@glxstr@endescspch}%
7111 \@glxstr@checkspch
7112 \let#1\@gls@checkedmkidx\relax
7113 \fi
Escape actual character unless it has already been escaped.
7114 \ifx\@glxstr@autoindex@at\@gls@actualchar
7115 \else
7116 \def\@gls@checkedmkidx{}%

```

```

7117 \edef\@glsxtr@checkspch{%
7118     \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
7119     \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
7120     \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
7121 \@@glsxtr@checkspch
7122 \let#1\@gls@checkedmkidx\relax
7123 \fi

Escape level character unless it has already been escaped.
7124 \ifx\@glsxtr@autoindex@level\@gls@levelchar
7125 \else
7126     \def\@gls@checkedmkidx{%
7127         \edef\@glsxtr@checkspch{%
7128             \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
7129             \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
7130             \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
7131         \@@glsxtr@checkspch
7132         \let#1\@gls@checkedmkidx\relax
7133     \fi

Escape encap character unless it has already been escaped.
7134 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
7135 \else
7136     \def\@gls@checkedmkidx{%
7137         \edef\@glsxtr@checkspch{%
7138             \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
7139             \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
7140             \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
7141         \@@glsxtr@checkspch
7142         \let#1\@gls@checkedmkidx\relax
7143     \fi
7144 }

```

The user commands here have a preamble-only restriction to ensure they are set before required and also to reduce the chances of complications caused by babel's shorthands.

tr@autoindex@at Actual character for use with \index.

```
7145 \newcommand*{\@glsxtr@autoindex@at}{}
```

trSetActualChar Set the actual character.

```

7146 \newcommand*{\GlsXtrSetActualChar}[1]{%
7147     \gdef\@glsxtr@autoindex@at{#1}%
7148     \def\@glsxtr@autoindex@escat##1##2##3\@glsxtr@endescspch{%
7149         \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
7150     }%
7151 }
7152 \@onlypreamble\GlsXtrSetActualChar
7153 \makeatother
7154 \GlsXtrSetActualChar{@}
7155 \makeatletter

```

autoindex@encap Encap character for use with \index.

```
7156 \newcommand*{\@glxtr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
7157 \newcommand*{\GlsXtrSetEncapChar}[1]{%
7158   \gdef\@glxtr@autoindex@encap{#1}%
7159   \def\@glxtr@autoindex@escencap##1#1##2#1##3\@glxtr@endescspch{%
7160     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
7161   }%
7162 }
7163 \GlsXtrSetEncapChar{ }
7164 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
7165 \newcommand*{\@glxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
7166 \newcommand*{\GlsXtrSetLevelChar}[1]{%
7167   \gdef\@glxtr@autoindex@level{#1}%
7168   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
7169     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
7170   }%
7171 }
7172 \GlsXtrSetLevelChar{!}
7173 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
7174 \newcommand*{\@glxtr@autoindex@esc}{"}
```

lsXtrSetEscChar Set the escape character.

```
7175 \newcommand*{\GlsXtrSetEscChar}[1]{%
7176   \gdef\@glxtr@autoindex@esc{#1}%
7177   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
7178     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
7179   }%
7180 }
7181 \GlsXtrSetEscChar{"}
7182 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
7183 \ifdef\actualchar
7184   {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
7185 }
```

Quote character \quotechar:

```
7186 \ifdef\quotechar
7187   {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
7188 }
```

Level character \levelchar:

```
7189 \ifdef\levelchar
7190 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
7191 {}}
```

Encap character \encapchar:

```
7192 \ifdef\encapchar
7193 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
7194 {}}
```

bleto@endescspch

```
7195 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

oindex@esc@spch

<code>\@glxtr@autoindex@escspch{&lt;char&gt;}{&lt;cs&gt;}{&lt;pre&gt;}{&lt;mid&gt;}{&lt;post&gt;}</code>
--

```
7196 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
7197   \@glstmpb=\expandafter{\@gl@checkedmkidx}%
7198   \toks@={#3}%
7199   \ifx\@nnil#3\relax
7200     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
7201   \else
7202     \ifx\@nnil#4\relax
7203       \edef\@gl@checkedmkidx{\the\@glstmpb\the\toks@}%
7204       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
7205         #4#5\@glxtr@endescspch}%
7206     \else
7207       \edef\@gl@checkedmkidx{\the\@glstmpb\the\toks@
7208         \@glxtr@autoindex@esc#1}%
7209       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
7210     \fi
7211   \fi
7212   \@glxtr@checkspch
7213 }
```

\Glossentrydesc   Redefine to set the abbreviation format and accessibility support.

```
7214 \renewcommand*{\Glossentrydesc}[1]{%
7215   \glsoifexistsorwarn{#1}%
7216   {%
7217     \glsetabbrvfmt{\glscategory{#1}}%
7218     \Glsaccessdesc{#1}%
7219   }%
7220 }
```

lossentrysymbol   Redefine to set the format and accessibility support. Allow for the possibility of being used in a section heading for standalone entry definitions.



```

7221 \ifdef\teorpdfstring
7222 {
7223   \renewcommand*{\glossentrysymbol}[1]{%
7224     \teorpdfstring{\@glossentrysymbol{#1}}{\glentrypdfsymbol{#1}}%
7225   }
7226 }
7227 {
7228   \renewcommand*{\glossentrysymbol}[1]{\@glossentrysymbol{#1}}
7229 }

```

`\glentrypdfsymbol` May be redefined to a field that expands to a value that's more suitable for PDF bookmarks.

```

7230 \newcommand{\glentrypdfsymbol}[1]{\glentrysymbol{#1}}

```

`\glossentrysymbol` There are no case-changing attributes as it's less usual for symbols.

```

7231 \newrobustcmd*{\@glossentrysymbol}[1]{%
7232   \glsoifexistsorwarn{#1}%
7233   {%
7234     \begingroup
7235     \glsetabbrvfmt{\glscategory{#1}}%
7236     \glshasattribute{#1}{glosssymbolfont}%
7237     {%
7238       \protected@edef\@glxtr@attrval{\glsetattribute{#1}{glosssymbolfont}}%
7239       \ifcsdef{\@glxtr@attrval}%
7240       {%
7241         \letcs{\@glxtr@glosssymbolfont}{\@glxtr@attrval}%
7242       }%
7243       {%
7244         \GlossariesExtraWarning{Unknown control sequence name
7245           '\@glxtr@attrval' supplied in glosssymbolfont attribute
7246           for entry '#1'. Ignoring}%
7247         \let\@glxtr@glosssymbolfont\@firstofone
7248       }%
7249     }%
7250     {\let\@glxtr@glosssymbolfont\@firstofone}%
7251     \@glxtr@glosssymbolfont{\glaccesssymbol{#1}}%
7252   }%
7253 }%
7254 }

```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

7255 \renewcommand*{\Glossentrysymbol}[1]{%
7256   \glsoifexistsorwarn{#1}%
7257   {%
7258     \glsetabbrvfmt{\glscategory{#1}}%
7259     \Glsaccesssymbol{#1}%
7260   }%
7261 }

```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\enableInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
7262 \newcommand*{\GlsXtrEnableInitialTagging}{%
7263   \@ifstar\s@glstr@enabletagging\@glstr@enabletagging
7264 }
7265 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
7266 \newcommand*{\s@glstr@enabletagging}[2]{%
7267   \undef#2%
7268   \@glstr@enabletagging{#1}{#2}%
7269 }
```

`\r@enabletagging` Internal command.

```
7270 \newcommand*{\@glstr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
7271   \@for\@glstr@cat:=#1\do
7272   {%
7273     \ifdefempty\@glstr@cat
7274     {}%
7275     {\glsssetcategoryattribute{\@glstr@cat}{tagging}{true}}%
7276   }%
7277   \newrobustcmd*#2[1]{##1}%
7278   \def\@glstr@taggingcs{#2}%
7279   \renewcommand*\@glstr@activate@initialtagging{%
7280     \let#2\@glstr@tag
7281   }%
7282   \ifundef\@glstr@preglossaryhook
7283   {\GlossariesExtraWarning{Initial tagging requires at least
7284     glossaries.sty v4.19 to work correctly}}%
7285   {}%
7286 }
```

Are we using an old version of `mfirstuc` that has a bug in `\capitalisewords`? If so, patch it so we don't have a problem with a combination of tagging and title case.

`\mfu@checkword@do` If this command hasn't been defined, then we have pre v2.02 of `mfirstuc`

```
7287 \ifundef\mfu@checkword@do
7288 {
7289   \newcommand*{\mfu@checkword@do}[1]{%
7290     \ifdefstring{\mfu@checkword@arg}{#1}%
7291     {%
7292       \let\@mfu@do@makefirstuc\@firstofone
7293       \listbreak
7294     }%
7295     {}%
7296   }
```

`\mfu@checkword` `\capitalisewords` was introduced in `mfirstuc` v1.06. If `\mfu@checkword` hasn't been defined `mfirstuc` is too old to support the title case attribute.

```
7297 \ifundef\mfu@checkword
7298 {
7299   \newcommand{\@glxtr@do@titlecaps@warn}{%
7300     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
7301       support not available}%
7302     \let\@glxtr@do@titlecaps@warn\relax
7303   }
7304 }
7305 {
7306   \renewcommand*\mfu@checkword[1]{%
7307     \def\mfu@checkword@arg{#1}%
7308     \let\@mfu@do@makefirstuc\makefirstuc
7309     \forlistloop\mfu@checkword@do\@mfu@nocaplist
7310   }
7311 }
7312 }
7313 {}% no patch required
```

`@titlecaps@warn` Do warning if title case not supported.

```
7314 \newcommand*\@glxtr@do@titlecaps@warn{}
```

`@initialtagging` Used in `\printglossary` but at least v4.19 of `glossaries` required.

```
7315 \newcommand*\@glxtr@activate@initialtagging{}
```

`\@glxtr@tag` Definition of tagging command when used in glossary.

```
7316 \newrobustcmd*\@glxtr@tag[1]{%
7317   \gl@ifattribute{\gl@currententrylabel}{tagging}{true}%
7318   {\glxtr@tagfont{#1}}{#1}%
7319 }
```

`\glxtr@tagfont` Used in the glossary.

```
7320 \newcommand*\glxtr@tagfont[1]{\underline{#1}}
```

`preglossaryhook` This macro was introduced in `glossaries` version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```
7321 \ifdef\@gl@preglossaryhook
7322 {
7323   \renewcommand*\@gl@preglossaryhook{%
7324     \@glxtr@activate@initialtagging
```

Since the `glossaries` are automatically scoped, `\@glxtr@org@postdescription` shouldn't already be defined, but check anyway just as a precautionary measure.

```

7325 \ifundef\@glxtr@org@postdescription
7326 {%
7327 \let\@glxtr@org@postdescription\glspostdescription
7328 \renewcommand*{\glspostdescription}{%
7329 \ifglstryexists{\glscurrententrylabel}%
7330 {%
7331 \glxtrpostdescription
7332 \@glxtr@org@postdescription
7333 }%
7334 {}%
7335 }%
7336 }%
7337 {}%

```

Enable the options used by \@glxtrp:

```

7338 \glossxtrsetopts
7339 }%
7340 }
7341 {}

```

`postdescription` This command will only be used if \@gls@preglossaryhook is available *and* the glossary style uses \glspostdescription without modifying it. (\nopostdesc will suppress this.) The glossaries-extra-stylemods package will add the post description hook to all the predefined styles that don't include it.

```

7342 \newcommand*{\glxtrpostdescription}{%
7343 \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
7344 }

```

`postdescgeneral`

```

7345 \newcommand*{\glxtrpostdescgeneral}{}

```

`xtrpostdescterm`

```

7346 \newcommand*{\glxtrpostdescterm}{}

```

`postdescacronym`

```

7347 \newcommand*{\glxtrpostdescacronym}{}

```

`descabbreviation`

```

7348 \newcommand*{\glxtrpostdescabbreviation}{}

```

`\glsdefpostdesc` Provide a convenient command for defining the post-description hook for the given category.

```

7349 \newcommand*{\glsdefpostdesc}[2]{%
7350 \csdef{glxtrpostdesc#1}{#2}%
7351 }

```

`glspostlinkhook` Redefine the post link hook used by commands like \gls to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of

commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```
7352 \renewcommand*{\glspostlinkhook}{%
7353   \ifglstryexists{\glslabel}{\glstrpostlinkhook}{}%
7354 }
```

`\glstrpostlinkhook` The entry label should already be stored in `\glslabel` by `\@gls@link`.

```
7355 \newcommand*{\glstrpostlinkhook}{%
7356   \glstrdiscardperiod{\glslabel}%
7357   {\glstrpostlinkendsentence}%
7358   {\glstrifcustomdiscardperiod
7359     {\glstrifperiod{\glstrpostlinkendsentence}{\glstrpostlink}}}%
7360   {\glstrpostlink}%
7361   }%
7362 }
```

`\glstrdiscardperiod` Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
7363 \newcommand*{\glstrifcustomdiscardperiod}[2]{#2}
```

`\glstrpostlink`

```
7364 \newcommand*{\glstrpostlink}{%
7365   \csuse{glstrpostlink\glscategory{\glslabel}}%
7366 }
```

`\glstrdefpostlink` Provide a convenient command for defining the post-link hook for the given category. Doesn't allow an empty argument (which) would overwrite `\glstrpostlink`.

```
7367 \newcommand*{\glstrdefpostlink}[2]{%
  \ifthenelse is used to ensure that the expanded value is tested. (The category label must
  be fully expandable.)
7368   \ifthenelse{\equal{#1}{}}{%
7369     {\PackageError{glossaries-extra}
7370       {Invalid empty category label in \string\glstrdefpostlink}{}}%
7371     {\csdef{glstrpostlink#1}{#2}}%
7372 }
```

`\glstrpostlinkendsentence` Done by `\glstrpostlinkhook` if a full stop is discarded.

```
7373 \newcommand*{\glstrpostlinkendsentence}{%
7374   \ifcsdef{glstrpostlink\glscategory{\glslabel}}
7375   {%
7376     \csuse{glstrpostlink\glscategory{\glslabel}}%
  Put the full stop back.
7377   .\spacefactor\sfcode'\. \relax
7378   }%
7379   {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```
7380 \spacefactor\sfcode'\. \relax
7381 }%
7382 }
```

**DescOnFirstUse** Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
7383 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
7384 \glxtrifwasfirstuse{\space\glxtrparen{\glsaccessdesc{\glslabel}}}{}%
7385 }
```

**SymbolOnFirstUse** Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
7386 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
7387 \glxtrifwasfirstuse
7388 {%
7389 \ifglshassymbol{\glslabel}%
7390 {\space\glxtrparen{\glsaccesssymbol{\glslabel}}}%
7391 {}}%
7392 }%
7393 {}%
7394 }
```

**1DescOnFirstUse** Provide a command for appending the symbol (if defined) and description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
7395 \newcommand*{\glxtrpostlinkAddSymbolDescOnFirstUse}{%
7396 \glxtrifwasfirstuse
7397 {%
7398 \space\glxtrparen
7399 {%
7400 \ifglshassymbol{\glslabel}%
7401 {\glsaccesssymbol{\glslabel}, }%
7402 {}}%
7403 \glsaccessdesc{\glslabel}%
7404 }%
7405 }%
7406 {}%
7407 }
```

**trdiscardperiod** Discard following period (if present) if the discardperiod attribute is true. If a period is discarded, do the second argument otherwise do the third argument. The entry label is in the first argument. Since this is designed for abbreviations that end with a period, check if the plural form was used (which typically won't end with a period).

```
7408 \newcommand*{\glxtrdiscardperiod}[3]{%
7409 \glxtrifwasfirstuse
7410 {%
7411 \glsifattribute{#1}{retainfirstuseperiod}{true}%
```

```

7412   {#3}%
7413   {%
7414     \glsifattribute{#1}{discardperiod}{true}%
7415     {%
7416       \glsifplural
7417       {%
7418         \glsifattribute{#1}{pluraldiscardperiod}{true}%
7419         {\glxtrifperiod{#2}{#3}}%
7420         {#3}%
7421       }%
7422     {%
7423       \glxtrifperiod{#2}{#3}%
7424     }%
7425   }%
7426   {#3}%
7427 }%
7428 }%
7429 {%
7430   \glsifattribute{#1}{discardperiod}{true}%
7431   {%
7432     \glsifplural
7433     {%
7434       \glsifattribute{#1}{pluraldiscardperiod}{true}%
7435       {\glxtrifperiod{#2}{#3}}%
7436       {#3}%
7437     }%
7438   {%
7439     \glxtrifperiod{#2}{#3}%
7440   }%
7441 }%
7442 {#3}%
7443 }%
7444 }

```

`\glxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

7445 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.{\@firstoftwo{#1}}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```

7446 \newcommand*{\glxtr@punclist}{.,;?!}

```

`\punctuationmark` Add character to punctuation list.

```

7447 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}

```

`\punctuationmarks` Reset the punctuation list.

```

7448 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}

```

\glxstrifpunc

`\glxstrifnextpunc{<true part>}{<false part>}`

Test if this is followed by a punctuation mark. (Adapted from \new@ifnextchar.)

```
7449 \newcommand*{\glxstrifnextpunc}[2]{%
7450   \def\reserved@a{#1}%
7451   \def\reserved@b{#2}%
7452   \futurelet\@glspunc@token\glxstr@ifnextpunc
7453 }
```

glxstr@ifnextpunc

```
7454 \newcommand*{\glxstr@ifnextpunc}{%
7455   \glxstr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
7456   \reserved@b
7457 }
```

glxstr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
7458 \newcommand*{\glxstr@ifpunctoken}[1]{%
7459   \expandafter\@glxstr@ifpunctoken\expandafter#1\glxstr@punclist\@nnil
7460 }
```

glxstr@ifpunctoken

```
7461 \def\@glxstr@ifpunctoken#1#2{%
7462   \let\reserved@d=#2%
7463   \ifx\reserved@d\@nnil
7464     \let\glxstr@next\@glxstr@notfoundinlist
7465   \else
7466     \ifx#1\reserved@d
7467       \let\glxstr@next\@glxstr@foundinlist
7468     \else
7469       \let\glxstr@next\@glxstr@ifpunctoken
7470     \fi
7471   \fi
7472   \glxstr@next#1%
7473 }
```

glxstr@foundinlist

```
7474 \def\@glxstr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
7475 \def\@glxstr@notfoundinlist#1{\@secondoftwo}
```

glxstr@dopostpunc



```
\glxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do <code> after the character otherwise do <code> before whatever comes next.

```
7476 \newcommand{\glxtrdopostpunc}[1]{%
7477   \glxtrifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
7478 }
```

@glxtr@swaptwo

```
7479 \newcommand{\@glxtr@swaptwo}[2]{#2#1}
```

## 1.7 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, it needs to be applied by `\newabbreviation`.

```
7480 \define@key{glxtrabbrv}{category}{%
7481   \protected@edef\glscategorylabel{#1}%
7482 }
```

Save the short plural form. This may be needed before the entry is defined.

```
7483 \define@key{glxtrabbrv}{shortplural}{%
7484   \def\@gl@shortpl{#1}%
7485 }
```

Similarly for the long plural form.

```
7486 \define@key{glxtrabbrv}{longplural}{%
7487   \def\@gl@longpl{#1}%
7488 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
7489 \newtoks\glsshortpltok
```

\glslongpltok

```
7490 \newtoks\glslongpltok
```

`\glxtr@insertdots` Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to `\newabbreviation`. Note that explicitly using the short or shortplural keys will override this.

```

7491 \newcommand*{\@glsxtr@insertdots}[2]{%
7492   \def#1{}%
7493   \@glsxtr@insert@dots#1#2\@nnil
7494 }

```

xtr@insert@dots

```

7495 \newcommand*{\@glsxtr@insert@dots}[2]{%
7496   \ifx\@nnil#2\relax
7497   \let\@glsxtr@insert@dots@next\@gobble
7498   \else
7499   \ifx\relax#2\relax
7500   \else
7501     \appto#1{#2.}%
7502   \fi
7503   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
7504   \fi
7505   \@glsxtr@insert@dots@next#1%
7506 }

```

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep

```

7507 \newcommand*{\glsxtrwordsep}{\space}

```

Each word is marked with

\glsxtrword

```

7508 \newcommand*{\glsxtrword}[1]{#1}

```

tr@markwordseps

```

7509 \newcommand*{\@glsxtr@markwordseps}[2]{%
7510   \def#1{}%
7511   \@glsxtr@mark@wordseps#1#2 \@nnil
7512 }

```

r@mark@wordseps

```

7513 \def\@glsxtr@mark@wordseps#1#2 #3{%
7514   \ifdefempty{#1}%
7515   {\def#1{\protect\glsxtrword{#2}}}%
7516   {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
7517   \ifx\@nnil#3\relax
7518   \let\@glsxtr@mark@wordseps@next\relax
7519   \else
7520   \def\@glsxtr@mark@wordseps@next{%
7521     \@glsxtr@mark@wordseps#1#3}%
7522   \fi
7523   \@glsxtr@mark@wordseps@next
7524 }

```

`\newabbreviation` Define a new generic abbreviation.

```
7525 \newcommand*{\newabbreviation}[4] [] {%
7526   \glstr@newabbreviation{#1}{#2}{#3}{#4}%
7527 }
```

`\newabbreviation` Internal macro. (bib2gls has an option that needs to temporarily redefine `\newabbreviation`. This is just makes it easier to save and restore the original definition.)

```
7528 \newcommand*{\glstr@newabbreviation}[4] {%
7529   \glstok{#1}%
7530   \glstok{#2}%
7531   \glstok{#3}%
7532   \glstok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```
7533 \def\glstr@orgshort{#3}%
7534 \def\glstr@orglong{#4}%

```

Provide extra settings for hooks (if modified, this command must end with a comma).

```
7535 \def\ExtraCustomAbbreviationFields{%

```

Initialise accessibility settings if required.

```
7536 \@glstr@initaccesskeys

```

Get the category.

```
7537 \def\glscategorylabel{abbreviation}%

```

Ignore the shortplural and longplural keys.

```
7538 \setkeys*{\glstr@abbrv}{shortplural,longplural}{#1}%

```

Set the abbreviation style.

```
7539 \ifcsdef{\glstr@current@\glscategorylabel}%
7540 {%

```

Warning should already have been issued.

```
7541 \let\glstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
7542 \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
7543 \glstr@applyabbrvstyle{\csname \glstr@current@\glscategorylabel\endcsname}%
7544 \let\GlsXtrWarnDeprecatedAbbrStyle\glstr@orgwarndep
7545 }%
7546 {%

```

If no style has been associated with this category, fallback on the style for the abbreviation category.

```
7547 \glstr@applyabbrvstyle{\glstr@current@abbreviation}%
7548 }%

```

Set the default long plural

```
7549 \def\glstr@longpl{#4\glsppluralsuffix}%
7550 \let\glstr@default@longpl\glstr@longpl

```

Has the markwords attribute been set?

```
7551 \glstr@ifcategoryattribute{\glscategorylabel}{markwords}{true}%
7552 {%

```

```

7553 \glsxtr@markwordseps\@gls@long{#4}%
7554 \expandafter\def\expandafter\@gls@longpl\expandafter
7555 {\@gls@long\glspluralsuffix}%
7556 \let\@gls@default@longpl\@gls@longpl

```

Update \glslongtok.

```

7557 \expandafter\glslongtok\expandafter{\@gls@long}%
7558 }%
7559 {}%

```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```

7560 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
7561 {%
7562 \glsxtr@markwordseps\@gls@short{#3}%
7563 }%
7564 {%

```

Has the insertdots attribute been set?

```

7565 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
7566 {%
7567 \glsxtr@insertdots\@gls@short{#3}%
7568 \appto\@gls@short{\@}%
7569 }%
7570 {\def\@gls@short{#3}}%
7571 }%

```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```

7572 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
7573 {%
7574 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
7575 '\abbrvpluralsuffix}%
7576 }%
7577 {%

```

Has the noshortplural attribute been set?

```

7578 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
7579 {%
7580 \let\@gls@shortpl\@gls@short
7581 }%
7582 {%
7583 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
7584 \abbrvpluralsuffix}%
7585 }%
7586 }%

```

Update \glsshorttok:

```

7587 \expandafter\glsshorttok\expandafter{\@gls@short}%

```

Hook for further customisation if required:

```

7588 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary. Ignore the category key (already obtained).

```
7589 \setkeys*{glsxtrabbrv}[category]{#1}%
```

Save in case required.

```
7590 \let@gls@org@longpl@gls@longpl
```

```
7591 \let@gls@org@shortpl@gls@shortpl
```

Has the plural been explicitly set?

```
7592 \ifx@gls@default@longpl@gls@longpl
```

```
7593 \else
```

Has the markwords attribute been set?

```
7594 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
```

```
7595 {%
```

```
7596 \expandafter@glsxtr@keywordseps\expandafter@gls@longpl\expandafter
```

```
7597 {\@gls@longpl}%
```

```
7598 }%
```

```
7599 {}%
```

```
7600 \fi
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
7601 \expandafter@gls@shortpltok\expandafter{\@gls@shortpl}%
```

```
7602 \expandafter@gls@longpltok\expandafter{\@gls@longpl}%
```

Hook for accessibility support (does nothing if glossaries-accsupp hasn't been loaded).

```
7603 \@gls@setup@default@access
```

Do any extra setup provided by hook:

```
7604 \newabbreviationhook
```

Define this entry:

```
7605 \protected@edef\@do@newglossaryentry{%
```

```
7606 \noexpand\newglossaryentry{\the\glslabeltok}%
```

```
7607 {%
```

```
7608 type=glsxtrabbrvtype,%
```

```
7609 category=abbreviation,%
```

```
7610 short={\the\glsshorttok},%
```

```
7611 shortplural={\the\glsshortpltok},%
```

```
7612 long={\the\glslongtok},%
```

```
7613 longplural={\the\glslongpltok},%
```

```
7614 name={\the\glsshorttok},%
```

```
7615 \CustomAbbreviationFields,%
```

Hook may override abbreviation style default settings (this hook must end with a comma if set).

```
7616 \ExtraCustomAbbreviationFields
```

Any explicit fields set in the optional argument override all other settings.

```
7617 \the\glskeylisttok
```

```
7618 }%
```

```
7619 }%
```

```
7620 \@do@newglossaryentry
```

Obtain the type and add it to the list of abbreviations.

```
7621 \glsxtr@addabbreviationlist{\glsentrytype{\the\glslabeltok}}%
7622 \GlsXtrPostNewAbbreviation
7623 }
```

evpresetkeyhook Hook for extra stuff in \newabbreviation

```
7624 \newcommand*{\glsxtrnewabbrevpresetkeyhook}[3]{}%
```

NewAbbreviation Hook used by abbreviation styles.

```
7625 \newcommand*{\GlsXtrPostNewAbbreviation}{}%
```

bbreviationhook Hook for use with \newabbreviation.

```
7626 \newcommand*{\newabbreviationhook}{}%
```

reviationFields

```
7627 \newcommand*{\CustomAbbreviationFields}{}%
```

\glsxtrparen For the parenthetical styles.

```
7628 \newcommand*{\glsxtrparen}[1]{(#1)}
```

lsxtrfullformat Full format without case change.

```
7629 \newcommand*{\glsxtrfullformat}[2]{%
7630 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
7631 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
7632 }
```

lsxtrfullformat Full format with case change.

```
7633 \newcommand*{\Glsxtrfullformat}[2]{%
7634 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
7635 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
7636 }
```

xtrfullplformat Plural full format without case change.

```
7637 \newcommand*{\glsxtrfullplformat}[2]{%
7638 \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
7639 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
7640 }
```

xtrfullplformat Plural full format with case change.

```
7641 \newcommand*{\Glsxtrfullplformat}[2]{%
7642 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
7643 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
7644 }
```

\glsxtrfullsep Separator used by full format is a space by default. The argument is the entry's label.

```
7645 \newcommand*{\glsxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glsentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glsentryfullformat` Full format without case change.  
 7646 `\newcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}`

`\glsentryfullformat` Full format with case change.  
 7647 `\newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}`

`\glsxtrfullplformat` Plural full format without case change.  
 7648 `\newcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}`

`\glsentryfullplformat` Plural full format with case change.  
 7649 `\newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}`

Redefine `\glsentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glsxtrfull` set of commands instead.

`\glsentryfull`  
 7650 `\renewcommand*{\glsentryfull}[1]{\glsxtrinlinefullformat{#1}{}}`

`\Glsentryfull`  
 7651 `\renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}`

`\glsentryfullpl`  
 7652 `\renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}`

`\Glsentryfullpl`  
 7653 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.  
 7654 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.  
 7655 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use. This is redefined by the abbreviation styles, as appropriate.  
 7656 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont`  
 7657 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.  
 7658 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

longdefaultfont Default font changing command used for the long form in commands like \glxtrlong.

```
7659 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
7660 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
7661 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
7662 \newcommand*{\glxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
7663 \newcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}
```

\glxtrfull Full form (no case-change).

```
7664 \newrobustcmd*{\glxtrfull}{\@gls@hyp@opt\@ns@glxtrfull}
```

```
7665 \newcommand*{\ns@glxtrfull}[2][\%
```

```
7666 \new@ifnextchar[{\@glxtr@full{#1}{#2}}]{\%
```

```
7667 \glxtr@full{#1}{#2}[]]{\%
```

```
7668 }
```

\@glxtr@full Low-level macro:

```
7669 \def\@glxtr@full#1#2[#3]{\%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
7670 \@glxtr@record{#1}{#2}{\glslink}%
```

```
7671 \glstdoifexists{#2}%
```

```
7672 {\%
```

```
7673 \glsssetabbrvfmt{\glscategory{#2}}}%
```

```
7674 \let\do@glslink@checkfirsthyper\@glslink@nocheckfirsthyper
```

```
7675 \let\glslifplural\@secondoftwo
```

```
7676 \let\glscapscase\@firstofthree
```

```
7677 \let\glsininsert\@empty
```

```
7678 \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%
```

What should \glxtrifwasfirstuse be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
7679 \glxtrsetupfulldefs
```

```
7680 \@glslink[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
```

```
7681 }%
```

```
7682 \glspostlinkhook
```

```
7683 }
```



trsetupfulldefs

```
7684 \newcommand*{\glxtrsetupfulldefs}{%
7685   \let\glxtrifwasfirstuse\@firstoftwo
7686 }
```

\Glsxtrfull Full form (first letter uppercase).

```
7687 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
7688 \newcommand*\ns@Glsxtrfull[2][{}]{%
7689   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}]{%
7690     {\@Glsxtr@full{#1}{#2}[]}%
7691 }
```

\@Glsxtr@full Low-level macro:

```
7692 \def\@Glsxtr@full#1#2[#3]{%
7693   \glsdoifexists{#2}%
7694   {%
7695     \glssetabbrvfmt{\glscategory{#2}}%
7696     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7697     \let\glsifplural\@secondoftwo
7698     \let\glscapscase\@secondofthree
7699     \let\glsinsert\@empty
7700     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
7701     \glxtrsetupfulldefs
7702     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7703   }%
7704   \glspostlinkhook
7705 }
```

\GLSxtrfull Full form (all uppercase).

```
7706 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
7707 \newcommand*\ns@GLSxtrfull[2][{}]{%
7708   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}]{%
7709     {\@GLSxtr@full{#1}{#2}[]}%
7710 }
```

\@GLSxtr@full Low-level macro:

```
7711 \def\@GLSxtr@full#1#2[#3]{%
7712   \glsdoifexists{#2}%
7713   {%
7714     \glssetabbrvfmt{\glscategory{#2}}%
7715     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7716     \let\glsifplural\@secondoftwo
7717     \let\glscapscase\@thirdofthree
7718     \let\glsinsert\@empty
7719     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
7720     \glxtrsetupfulldefs
7721     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7722   }%
7723   \glspostlinkhook
```

7724 }

`\glstrfullpl` Plural full form (no case-change).

```
7725 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\@ns@glstrfullpl}
7726 \newcommand*\ns@glstrfullpl[2][\%
7727   \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}]{%
7728     {\@glstr@fullpl{#1}{#2}[]}%
7729 }
```

`\@glstr@fullpl` Low-level macro:

```
7730 \def\@glstr@fullpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7731   \@glstr@record{#1}{#2}{glslink}%
7732   \glsdoifexists{#2}%
7733   {%
7734     \glssetabbrvfmt{\glscategory{#2}}%
7735     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7736     \let\glsifplural\@firstoftwo
7737     \let\glscapscase\@firstofthree
7738     \let\glsinsert\@empty
7739     \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
7740     \glstrsetupfulldefs
7741     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7742   }%
7743   \glspostlinkhook
7744 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
7745 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
7746 \newcommand*\ns@Glsxtrfullpl[2][\%
7747   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}]{%
7748     {\@Glsxtr@fullpl{#1}{#2}[]}%
7749 }
```

`\@Glsxtr@fullpl` Low-level macro:

```
7750 \def\@Glsxtr@fullpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7751   \@glstr@record{#1}{#2}{glslink}%
7752   \glsdoifexists{#2}%
7753   {%
7754     \glssetabbrvfmt{\glscategory{#2}}%
7755     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7756     \let\glsifplural\@firstoftwo
7757     \let\glscapscase\@secondofthree
7758     \let\glsinsert\@empty
```

```

7759 \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
7760 \glstrsetupfulldefs
7761 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7762 }%
7763 \glspostlinkhook
7764 }

```

`\Glsxtrfullpl` Plural full form (all upper case).

```

7765 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
7766 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
7767 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
7768 {\@Glsxtr@fullpl{#1}{#2}[]}%
7769 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

7770 \def\@Glsxtr@fullpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7771 \@glsxtr@record{#1}{#2}{glslink}%
7772 \glsdoifexists{#2}%
7773 {%
7774 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7775 \let\glsifplural\@firstoftwo
7776 \let\glscapscase\@thirdofthree
7777 \let\glsinsert\@empty
7778 \def\glscustomtext{%
7779 \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
7780 \glstrsetupfulldefs
7781 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7782 }%
7783 \glspostlinkhook
7784 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

7785 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
7786 \newcommand*\ns@glsxtrshort[2][{}]{%
7787 \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2}[]}%
7788 }

```

Read in the final optional argument:

```

7789 \def\@glsxtrshort#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7790 \@glsxtr@record{#1}{#2}{glslink}%
7791 \glsdoifexists{#2}%
7792 {%

```

Need to make sure \glsabbrvfont is set correctly.

```

7793 \glssetabbrvfmt{\glscategory{#2}}%
7794 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
7795 \let\glxtrifwasfirstuse\@secondoftwo
7796 \let\gl@sifplural\@secondoftwo
7797 \let\glscapscase\@firstofthree
7798 \let\glsininsert\@empty
7799 \def\glscustomtext{%
7800 \glsabbrvfont{\glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%
7801 \ifglxtrininsertinside\else#3\fi
7802 }%
7803 \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7804 }%
7805 \glspostlinkhook
7806 }

```

\Glsxtrshort

```

7807 \newrobustcmd*{\Glsxtrshort}{\@gl@s@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

7808 \newcommand*{\ns@Glsxtrshort}[2] [] {%
7809 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
7810 }

```

Read in the final optional argument:

```

7811 \def\@Glsxtrshort#1#2[#3] {%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7812 \@glxtr@record{#1}{#2}{glslink}%
7813 \glsoifexists{#2}%
7814 {%
7815 \glssetabbrvfmt{\glscategory{#2}}%
7816 \let\do@gl@s@link@checkfirsthyper\@gl@s@link@nocheckfirsthyper
7817 \let\glxtrifwasfirstuse\@secondoftwo
7818 \let\gl@sifplural\@secondoftwo
7819 \let\glscapscase\@secondofthree
7820 \let\glsininsert\@empty
7821 \def\glscustomtext{%
7822 \glsabbrvfont{\Glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%
7823 \ifglxtrininsertinside\else#3\fi
7824 }%
7825 \@gl@s@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7826 }%
7827 \glspostlinkhook
7828 }

```

\GLSxtrshort

```

7829 \newrobustcmd*{\GLSxtrshort}{\@gl@s@hyp@opt\ns@GLSxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
7830 \newcommand*{\ns@GLSxtrshort}[2][\%
7831 \new@ifnextchar[\@GLSxtrshort{#1}{#2}]{\@GLSxtrshort{#1}{#2}[]}%
7832 }
```

Read in the final optional argument:

```
7833 \def\@GLSxtrshort#1#2[#3]{%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
7834 \@GLSxtr@record{#1}{#2}{glslink}%
7835 \glsdoifexists{#2}%
7836 {%
7837 \glsssetabbrvfmt{\glscategory{#2}}%
7838 \let\do@GLS@link@checkfirsthyper\@GLS@link@nocheckfirsthyper
7839 \let\GLSxtrifwasfirstuse\@secondoftwo
7840 \let\GLSifplural\@secondoftwo
7841 \let\GLScapscase\@thirdofthree
7842 \let\GLSinsert\@empty
7843 \def\GLScustomtext{%
7844 \mfirstucMakeUppercase
7845 {\GLSabbrvfont{\GLSaccessshort{#2}\ifGLSxtrininsertinside#3\fi}%
7846 \ifGLSxtrininsertinside\else#3\fi
7847 }%
7848 }%
7849 \@GLS@link[#1]{#2}{\csname gls@\GLStype @entryfmt\endcsname}%
7850 }%
7851 \GLSpostlinkhook
7852 }
```

\GLSxtrlong

```
7853 \newrobustcmd*{\GLSxtrlong}{\@GLS@hyp@opt\ns@GLSxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
7854 \newcommand*{\ns@GLSxtrlong}[2][\%
7855 \new@ifnextchar[\@GLSxtrlong{#1}{#2}]{\@GLSxtrlong{#1}{#2}[]}%
7856 }
```

Read in the final optional argument:

```
7857 \def\@GLSxtrlong#1#2[#3]{%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
7858 \@GLSxtr@record{#1}{#2}{glslink}%
7859 \glsdoifexists{#2}%
7860 {%
7861 \let\do@GLS@link@checkfirsthyper\@GLS@link@nocheckfirsthyper
7862 \let\GLSxtrifwasfirstuse\@secondoftwo
7863 \let\GLSifplural\@secondoftwo
7864 \let\GLScapscase\@firstofthree
7865 \let\GLSinsert\@empty
```

```

7866 \def\glscustomtext{%
7867 \glslongfont{\glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
7868 \ifglxtrinsertinside\else#3\fi
7869 }%
7870 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7871 }%
7872 \glspostlinkhook
7873 }

```

\Glsxtrlong

```

7874 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\ns@Glsxtrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

7875 \newcommand*{\ns@Glsxtrlong}[2][]{%
7876 \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
7877 }

```

Read in the final optional argument:

```

7878 \def\@Glsxtrlong#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7879 \@glsxtr@record{#1}{#2}{glslink}%
7880 \glsdoifexists{#2}%
7881 {%
7882 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7883 \let\glsxtrifwasfirstuse\@secondoftwo
7884 \let\glsifplural\@secondoftwo
7885 \let\glscapscase\@secondofthree
7886 \let\glsinsert\@empty
7887 \def\glscustomtext{%
7888 \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
7889 \ifglxtrinsertinside\else#3\fi
7890 }%
7891 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7892 }%
7893 \glspostlinkhook
7894 }

```

\GLSxtrlong

```

7895 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

7896 \newcommand*{\ns@GLSxtrlong}[2][]{%
7897 \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2}[]}%
7898 }

```

Read in the final optional argument:

```

7899 \def\@GLSxtrlong#1#2[#3]{%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7900 \@glsxtr@record{#1}{#2}{glslink}%
7901 \glsdoifexists{#2}%
7902 {%
7903   \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
7904   \let\glsxtrifwasfirstuse\@secondoftwo
7905   \let\glsifplural\@secondoftwo
7906   \let\glscapscase\@thirdofthree
7907   \let\glsinsert\@empty
7908   \def\glscustomtext{%
7909     \mfirstucMakeUppercase
7910     {\glslongfont{\glsaccessslong{#2}\ifglsxtrininsertinside#3\fi}%
7911     \ifglsxtrininsertinside\else#3\fi
7912   }%
7913 }%
7914 \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7915 }%
7916 \glspostlinkhook
7917 }

```

Plural short forms:

\glsxtrshortpl

```

7918 \newrobustcmd*{\glsxtrshortpl}{\@glsl@hyp@opt\@ns@glsl@xtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

7919 \newcommand*{\ns@glsl@xtrshortpl}[2] [] {%
7920   \new@ifnextchar[{\@glsl@xtrshortpl{#1}{#2}}{\@glsl@xtrshortpl{#1}{#2} [] }%
7921 }

```

Read in the final optional argument:

```

7922 \def\@glsl@xtrshortpl#1#2[#3] {%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7923 \@glsl@xtr@record{#1}{#2}{glslink}%
7924 \glsdoifexists{#2}%
7925 {%
7926   \glsssetabbrvfmt{\glscategory{#2}}%
7927   \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
7928   \let\glsxtrifwasfirstuse\@secondoftwo
7929   \let\glsifplural\@firstoftwo
7930   \let\glscapscase\@firstofthree
7931   \let\glsinsert\@empty
7932   \def\glscustomtext{%
7933     \glsl@abbrvfont{\glsaccesssshortpl{#2}\ifglsxtrininsertinside#3\fi}%
7934     \ifglsxtrininsertinside\else#3\fi
7935   }%
7936   \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

```

```

7937 }%
7938 \glspostlinkhook
7939 }

```

\Glsxtrshortpl

```

7940 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\@ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
7941 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
7942   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[ ]}%
7943 }

```

Read in the final optional argument:

```

7944 \def\@Glsxtrshortpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7945   \@glsxtr@record{#1}{#2}{glslink}%
7946   \glsdoifexists{#2}%
7947   {%
7948     \glssetabbrvfmt{\glscategory{#2}}%
7949     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
7950     \let\glsxtrifwasfirstuse\@secondoftwo
7951     \let\glsifplural\@firstoftwo
7952     \let\glscapscase\@secondofthree
7953     \let\glsinsert\@empty
7954     \def\glscustomtext{%
7955       \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
7956       \ifglsxtrininsertinside\else#3\fi
7957     }%
7958     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7959   }%
7960   \glspostlinkhook
7961 }

```

\GLSxtrshortpl

```

7962 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\@ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
7963 \newcommand*{\ns@GLSxtrshortpl}[2][ ]{%
7964   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2}[ ]}%
7965 }

```

Read in the final optional argument:

```

7966 \def\@GLSxtrshortpl#1#2[#3]{%
    If the record option has been used, the information needs to be written to the aux file regard-
    less of whether the entry exists (unless indexing has been switched off).
7967   \@glsxtr@record{#1}{#2}{glslink}%
7968   \glsdoifexists{#2}%
7969   {%

```



```

7970 \glsssetabbrvfmt{\glscategory{#2}}%
7971 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7972 \let\glxtrifwasfirstuse\@secondoftwo
7973 \let\gl@sifplural\@firstoftwo
7974 \let\glscapscase\@thirdofthree
7975 \let\gl$insert\@empty
7976 \def\glscustomtext{%
7977     \mfirstucMakeUppercase
7978     {\glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
7979     \ifglxtrininsertinside\else#3\fi
7980 }%
7981 }%
7982 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
7983 }%
7984 \glspostlinkhook
7985 }

```

Plural long forms:

\glxtrlongpl

```

7986 \newrobustcmd*{\glxtrlongpl}{\@gl@hyp@opt\@ns@glxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
7987 \newcommand*{\ns@glxtrlongpl}[2] [] {%
7988     \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2} []}%
7989 }

```

Read in the final optional argument:

```

7990 \def\@glxtrlongpl#1#2[#3] {%

```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```

7991 \@glxtr@record{#1}{#2}{glslink}%
7992 \glsoifexists{#2}%
7993 {%
7994     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
7995     \let\glxtrifwasfirstuse\@secondoftwo
7996     \let\gl@sifplural\@firstoftwo
7997     \let\glscapscase\@firstofthree
7998     \let\gl$insert\@empty
7999     \def\glscustomtext{%
8000         \glslongfont{\glaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
8001         \ifglxtrininsertinside\else#3\fi
8002     }%
8003     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8004 }%
8005 \glspostlinkhook
8006 }

```

\Glsxtrlongpl

```

8007 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\@ns@Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
8008 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
8009   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
8010 }
```

Read in the final optional argument:

```
8011 \def\@Glsxtrlongpl#1#2[#3]{%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
8012   \@glsxtr@record{#1}{#2}{glslink}%
8013   \glsdoifexists{#2}%
8014   {%
8015     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
8016     \let\glsxtrifwasfirstuse\@secondoftwo
8017     \let\glsifplural\@firstoftwo
8018     \let\glscapscase\@secondofthree
8019     \let\glsinsert\@empty
8020     \def\glscustomtext{%
8021       \glsfont{\Glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
8022       \ifglsxtrininsertinside\else#3\fi
8023     }%
8024     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8025   }%
8026   \glspostlinkhook
8027 }
```

\Glsxtrlongpl

```
8028 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
8029 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
8030   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
8031 }
```

Read in the final optional argument:

```
8032 \def\@Glsxtrlongpl#1#2[#3]{%
```

If the record option has been used, the information needs to be written to the aux file regardless of whether the entry exists (unless indexing has been switched off).

```
8033   \@glsxtr@record{#1}{#2}{glslink}%
8034   \glsdoifexists{#2}%
8035   {%
8036     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
8037     \let\glsxtrifwasfirstuse\@secondoftwo
8038     \let\glsifplural\@firstoftwo
8039     \let\glsapsase\@thirdofthree
8040     \let\glsinsert\@empty
8041     \def\glscustomtext{%
8042       \mfirstucMakeUppercase
8043       {\glsfont{\Glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
8044     }%
8045   }%
```

```

8044      \ifglxtrinsertinside\else#3\fi
8045      }%
8046      }%
8047      \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
8048      }%
8049      \glspostlinkhook
8050      }

```

`\glsetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

8051 \newcommand*{\glsetabbrvfmt}[1]{%
8052   \ifcsdef{@glsabbrv@current@#1}%
8053   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
8054   {\glxtr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
8055 }

```

`\gluseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

8056 \newrobustcmd*{\gluseabbrvfont}[2]{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\gluselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

8057 \newrobustcmd*{\gluselongfont}[2]{\glsetabbrvfmt{#2}\glsfont{#1}}

```

`\glxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

8058 \newcommand*{\glxtrgenabbrvfmt}{%
8059   \ifdefempty\glscustomtext
8060   {%
8061     \ifglsused\glslabel
8062     {%

```

Subsequent use:

```

8063     \glsifplural
8064     {%

```

Subsequent plural form:

```

8065     \glscapscase
8066     {%

```

Subsequent plural form, don't adjust case:

```

8067     \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%
8068     }%
8069     {%

```

Subsequent plural form, make first letter upper case:

```

8070     \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%
8071     }%
8072     {%

```

Subsequent plural form, all caps:

```

8073     \mfirstucMakeUppercase
8074     {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%
8075     }%

```

8076       }%  
8077       {%

Subsequent singular form

8078       \glscapscase  
8079       {%

Subsequent singular form, don't adjust case:

8080       \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
8081       }%  
8082       {%

Subsequent singular form, make first letter upper case:

8083       \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
8084       }%  
8085       {%

Subsequent singular form, all caps:

8086       \mfirstucMakeUppercase  
8087       {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
8088       }%  
8089       }%  
8090       }%  
8091       {%

First use:

8092       \glsifplural  
8093       {%

First use plural form:

8094       \glscapscase  
8095       {%

First use plural form, don't adjust case:

8096       \glxtrfullplformat{\glslabel}{\glsinsert}%  
8097       }%  
8098       {%

First use plural form, make first letter upper case:

8099       \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
8100       }%  
8101       {%

First use plural form, all caps:

8102       \mfirstucMakeUppercase  
8103       {\glxtrfullplformat{\glslabel}{\glsinsert}}%  
8104       }%  
8105       }%  
8106       {%

First use singular form

8107       \glscapscase  
8108       {%

First use singular form, don't adjust case:

```
8109      \glxtrfullformat{\glslabel}{\glsinsert}%
8110      }%
8111      {%
```

First use singular form, make first letter upper case:

```
8112      \Glsxtrfullformat{\glslabel}{\glsinsert}%
8113      }%
8114      {%
```

First use singular form, all caps:

```
8115      \mfirstucMakeUppercase
8116      {\glxtrfullformat{\glslabel}{\glsinsert}}%
8117      }%
8118      }%
8119      }%
8120      }%
8121      {%
```

User supplied text.

```
8122      \glscustomtext
8123      }%
8124 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
8125 \newcommand*{\glxtrsubsequentfmt}[2]{%
8126   \glsabbrvfont{\glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8127   \ifglxtrininsertinside \else#2\fi
8128 }
8129 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
8130 \newcommand*{\glxtrsubsequentplfmt}[2]{%
8131   \glsabbrvfont{\glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8132   \ifglxtrininsertinside \else#2\fi
8133 }
8134 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
8135 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
8136   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
8137   \ifglxtrininsertinside \else#2\fi
8138 }
8139 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```
8140 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
8141   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
8142   \ifglxtrininsertinside \else#2\fi
8143 }
8144 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt
```

## 1.7.1 Abbreviation Styles Setup

abbreviationstyle

```

8145 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
8146   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
8147   {%
8148     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
8149   }%
8150   {%

```

Have abbreviations already been defined for this category?

```

8151     \ifcsstring{@glsabbrv@current@#1}{#2}%
8152     {%

```

Style already set.

```

8153     }%
8154     {%
8155       \def\@glsxtr@dostylewarn{%
8156         \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
8157         {%
8158           \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
8159             style has been switched \MessageBreak
8160             for category ‘#1’, \MessageBreak
8161             but there have already been entries \MessageBreak
8162             defined for this category. Unwanted \MessageBreak
8163             side-effects may result}}%
8164           \@endfortrue
8165         }%
8166         \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

8167       \csdef{@glsabbrv@current@#1}{#2}%
8168       \protected@edef\glscategorylabel{#1}%
8169       \glsxtr@applyabbrvstyle{#2}%
8170     }%
8171   }%
8172 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

8173 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
8174   \csuse{@glsabbrv@dispstyle@setup@#1}%
8175   \csuse{@glsabbrv@dispstyle@fmts@#1}%
8176 }

```

r@applyabbrvfmt Only apply the style formats.

```

8177 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
8178   \csuse{@glsabbrv@dispstyle@fmts@#1}%
8179 }

```

breivationstyle This is different from `\newacronymstyle`. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
8180 \newcommand*{\newabbreviationstyle}[3]{%
8181   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
8182   {%
8183     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
8184       defined}{}}%
8185   }%
8186   {%
8187     \csdef{@glsabbrv@dispstyle@setup@#1}{%
```

Initialise hook to do nothing. The style may change this.

```
8188     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8189     #2}%
8190     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
```

Assume in-line form is the same as first use. The style may change this.

```
8191     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
8192     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8193     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
8194     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glsxtrsubsequentfmt` etc in case a style changes this.

```
8195     \let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt
8196     \let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt
8197     \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
8198     \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
8199     #3}%
8200   }%
8201 }
```

breivationstyle

```
8202 \newcommand*{\renewabbreviationstyle}[3]{%
8203   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
8204   {%
8205     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
8206   }%
8207   {%
8208     \csdef{@glsabbrv@dispstyle@setup@#1}{%
```

Initialise hook to do nothing. The style may change this.

```
8209     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
8210     #2}%
8211     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
```

Assume in-line form is the same as first use. The style may change this.

```
8212     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
8213     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
8214     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
8215     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

```

8216     #3}%
8217 }%
8218 }

```

**abbreviationstyle** Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

8219 \newcommand*{\letabbreviationstyle}[2]{%
8220   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
8221   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8222 }

```

**deprecated@abbrstyle**

```
\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}
```

Define a synonym for a deprecated abbreviation style.

```

8223 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
8224   \csdef{@glsabbrv@dispstyle@setup@#1}{%
8225     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
8226     \csuse{@glsabbrv@dispstyle@setup@#2}%
8227   }%
8228   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
8229 }

```

**deprecatedAbbrStyle** Generate warning for deprecated style use.

```

8230 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
8231   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
8232   use ‘#2’ instead}%
8233 }

```

**useAbbrStyleSetup**

```

8234 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
8235   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
8236   {%
8237     \PackageError{glossaries-extra}%
8238     {Unknown abbreviation style definitions ‘#1’}{}%
8239   }%
8240   {%
8241     \csname @glsabbrv@dispstyle@setup@#1\endcsname
8242   }%
8243 }

```

**useAbbrStyleFmt**

```

8244 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
8245   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
8246   {%
8247     \PackageError{glossaries-extra}%

```



```

8248     {Unknown abbreviation style formats ‘#1’}{}%
8249 }%
8250 {%
8251     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
8252 }%
8253 }

```

## 1.7.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn’t set to “true”. If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```

8254 \newif\ifglsxtrinsertinside
8255 \glsxtrinsertinsidefalse

```

`trlongshortname`

```

8256 \newcommand*{\glsxtrlongshortname}{%
8257   \protect\glsabbrvfont{\the\glsshorttok}%
8258 }

```

`long-short`

```

8259 \newabbreviationstyle{long-short}%
8260 {%

```

Set accessibility attributes if enabled.

```
8261   \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```

8262   \renewcommand*{\CustomAbbreviationFields}{%
8263     name={\glsxtrlongshortname},
8264     sort={\the\glsshorttok},
8265     first={\protect\glsfirstlongfont{\the\glslongtok}%
8266       \protect\glsxtrfullsep{\the\glslabeltok}%
8267       \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
8268     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
8269       \protect\glsxtrfullsep{\the\glslabeltok}%
8270       \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
8271     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
8272     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8273     description={\the\glslongtok}%

```

Unset the regular attribute if it has been set.

```

8274 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8275   \glshasattribute{\the\glslabeltok}{regular}%
8276   {%
8277     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8278   }%
8279   {}%
8280 }%
8281 }%
8282 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8283 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
8284 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
8285 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
8286 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
8287 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8288 \renewcommand*{\glsxtrfullformat}[2]{%
8289   \glssfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8290   \ifglsxtrininsertinside\else##2\fi
8291   \glsxtrfullsep{##1}%
8292   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}%
8293 }%
8294 \renewcommand*{\glsxtrfullplformat}[2]{%
8295   \glssfirstlongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8296   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8297   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}%
8298 }%
8299 \renewcommand*{\Glsxtrfullformat}[2]{%
8300   \glssfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8301   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8302   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}%
8303 }%
8304 \renewcommand*{\Glsxtrfullplformat}[2]{%
8305   \glssfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8306   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8307   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}%
8308 }%
8309 }

```

Set this as the default style for general abbreviations:

```

8310 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

8311 \newcommand*{\glsxtrlongshortdescsort}{%
8312   \expandonce\glsxtrorglong\space (\expandonce\glsxtrorgshort)%
8313 }

```

ngshortdescname

```
8314 \newcommand*{\glxtrlongshortdescname}{%
8315   \protect\glslongfont{\the\glslongtok}
8316   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
8317 }
```

long-short-desc User supplies description. The long form is included in the name.

```
8318 \newabbreviationstyle{long-short-desc}%
8319 {%
  Set accessibility attributes if enabled.
8320   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
  Setup the default fields.
8321   \renewcommand*{\CustomAbbreviationFields}{%
8322     name={\glxtrlongshortdescname},
8323     sort={\glxtrlongshortdescsort},%
8324     first={\protect\glsfirstlongfont{\the\glslongtok}%
8325       \protect\glxtrfullsep{\the\glslabeltok}%
8326       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
8327     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
8328       \protect\glxtrfullsep{\the\glslabeltok}%
8329       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
  The text key should only have the short form.
8330     text={\protect\glsabbrvfont{\the\glsshorttok}},%
8331     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
8332   }%
```

Unset the regular attribute if it has been set.

```
8333 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8334   \glshasattribute{\the\glslabeltok}{regular}%
8335   {%
8336     \glissetattribute{\the\glslabeltok}{regular}{false}%
8337   }%
8338   {}%
8339 }%
8340 }%
8341 {%
8342   \GlsXtrUseAbbrStyleFmts{long-short}%
8343 }
```

trshortlongname

```
8344 \newcommand*{\glxtrshortlongname}{%
8345   \protect\glsabbrvfont{\the\glsshorttok}%
8346 }
```

short-long Short form followed by long form in parenthesis on first use.

```
8347 \newabbreviationstyle{short-long}%
8348 {%
```

Set accessibility attributes if enabled.

```
8349 \glxstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
8350 \renewcommand*{\CustomAbbreviationFields}{%
8351   name={\glxstrshortlongname},
8352   sort={\the\glsshorttok},
8353   description={\the\glslongtok},%
8354   first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8355     \protect\glxstrfullsep{\the\glslabeltok}%
8356     \glxstrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
8357   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8358     \protect\glxstrfullsep{\the\glslabeltok}%
8359     \glxstrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%

8360   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8361   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
```

Unset the regular attribute if it has been set.

```
8362 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8363   \glshasattribute{\the\glslabeltok}{regular}%
8364   {%
8365     \glssetattribute{\the\glslabeltok}{regular}{false}%
8366   }%
8367   {}%
8368 }%
8369 }%
8370 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
8371 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8372 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8373 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8374 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8375 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8376 \renewcommand*{\glxstrfullformat}[2]{%
8377   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrinsertinside##2\fi}%
8378   \ifglxtrinsertinside\else##2\fi
8379   \glxstrfullsep{##1}%
8380   \glxstrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8381 }%
8382 \renewcommand*{\glxstrfullplformat}[2]{%
8383   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrinsertinside##2\fi}%
8384   \ifglxtrinsertinside\else##2\fi
8385   \glxstrfullsep{##1}%
8386   \glxstrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8387 }%
8388 \renewcommand*{\Glsxtrfullformat}[2]{%
8389   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglxtrinsertinside##2\fi}%

```

```

8390 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8391 \glxtrparen{\glsfirstlongfont{\glssaccesslong{##1}}}%
8392 }%
8393 \renewcommand*{\Glsxtrfullplformat}[2]{%
8394 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8395 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8396 \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
8397 }%
8398 }

```

ortlongdescsort

```
8399 \newcommand*{\glxtrshortlongdescsort}{\the\glssshorttok}
```

ortlongdescname

```

8400 \newcommand*{\glxtrshortlongdescname}{%
8401 \protect\glssabbrvfont{\the\glssshorttok}
8402 \glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8403 }

```

short-long-desc User supplies description. The long form is included in the name.

```

8404 \newabbreviationstyle{short-long-desc}%
8405 {%

```

Set accessibility attributes if enabled.

```
8406 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```

8407 \renewcommand*{\CustomAbbreviationFields}{%
8408 name={\glxtrshortlongdescname},
8409 sort={\glxtrshortlongdescsort},
8410 first={\protect\glsfirstabbrvfont{\the\glssshorttok}%
8411 \protect\glxtrfullsep{\the\glslabeltok}%
8412 \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
8413 firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}%
8414 \protect\glxtrfullsep{\the\glslabeltok}%
8415 \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
8416 text={\protect\glssabbrvfont{\the\glssshorttok}},%
8417 plural={\protect\glssabbrvfont{\the\glssshortpltok}}}%
8418 }%

```

Unset the regular attribute if it has been set.

```

8419 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8420 \glshasattribute{\the\glslabeltok}{regular}%
8421 {%
8422 \glissetattribute{\the\glslabeltok}{regular}{false}%
8423 }%
8424 {}%
8425 }%

```

```

8426 }%
8427 {%
8428   \GlsXtrUseAbbrStyleFmts{short-long}%
8429 }

```

`\longfootnotefont` Only used by the “footnote” styles.

```

8430 \newcommand*{\glstfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%

```

`\longfootnotefont` Only used by the “footnote” styles.

```

8431 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%

```

`\trabbrvfootnote`

`\glstextrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument `<long>` includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gl`s or `\gl`spl).

```

8432 \newcommand*{\glstextrabbrvfootnote}[2]{\footnote{#2}}

```

`\xtrfootnotename`

```

8433 \newcommand*{\glstxtrfootnotename}{%
8434   \protect\glstabbrvfont{\the\glsshorttok}%
8435 }

```

`\footnote` Short form followed by long form in footnote on first use.

```

8436 \newabbreviationstyle{footnote}%
8437 {%

```

Set accessibility attributes if enabled. (Add `\firstshortaccess` since long form is hidden in a footnote on first use.)

```

8438   \glstxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

8439   \renewcommand*{\CustomAbbreviationFields}{%
8440     name={\glstxtrfootnotename},
8441     sort={\the\glsshorttok},
8442     description={\the\glslongtok},%

8443     first={\protect\glstfirstabbrvfont{\the\glsshorttok}%
8444       \protect\glstextrabbrvfootnote{\the\glslabeltok}%
8445       {\protect\glstfirstlongfootnotefont{\the\glslongtok}}},%
8446     firstplural={\protect\glstfirstabbrvfont{\the\glsshortpltok}%
8447       \protect\glstextrabbrvfootnote{\the\glslabeltok}%
8448       {\protect\glstfirstlongfootnotefont{\the\glslongpltok}}},%

```

```

8449   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8450   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8451   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8452     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8453     \glshasattribute{\the\glslabeltok}{regular}%
8454     {%
8455       \glssetattribute{\the\glslabeltok}{regular}{false}%
8456     }%
8457   }%
8458 }%
8459 }%
8460 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8461   \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
8462   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8463   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8464   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
8465   \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8466   \renewcommand*{\glsxtrfullformat}[2]{%
8467     \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8468     \ifglsxtrininsertinside\else##2\fi
8469     \protect\glsxtrabbrvfootnote{##1}%
8470     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8471   }%
8472   \renewcommand*{\glsxtrfullplformat}[2]{%
8473     \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8474     \ifglsxtrininsertinside\else##2\fi
8475     \protect\glsxtrabbrvfootnote{##1}%
8476     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8477   }%
8478   \renewcommand*{\Glsxtrfullformat}[2]{%
8479     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8480     \ifglsxtrininsertinside\else##2\fi
8481     \protect\glsxtrabbrvfootnote{##1}%
8482     {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8483   }%
8484   \renewcommand*{\Glsxtrfullplformat}[2]{%
8485     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8486     \ifglsxtrininsertinside\else##2\fi
8487     \protect\glsxtrabbrvfootnote{##1}%
8488     {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8489   }%

```

The first use full form and the inline full form use the short (long) style.

```

8490   \renewcommand*{\glsxtrinlinefullformat}[2]{%

```

```

8491 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8492 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8493 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8494 }%
8495 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8496 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8497 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8498 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8499 }%
8500 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8501 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8502 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8503 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8504 }%
8505 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8506 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8507 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8508 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8509 }%
8510 }

```

short-footnote

```
8511 \letabbreviationstyle{short-footnote}{footnote}
```

ootnotedesname

```

8512 \newcommand*{\glxtrfootnotedesname}{%
8513 \protect\glsabbrvfont{\the\glsshorttok}%
8514 \protect\glxtrfullsep{\the\glslabeltok}%
8515 \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
8516 }

```

ootnotedesort

```
8517 \newcommand*{\glxtrfootnotedesort}{\the\glsshorttok}
```

t-footnote-desc Like short-footnote but with user supplied description.

```

8518 \newabbreviationstyle{short-footnote-desc}%
8519 {%

```

Set accessibility attributes if enabled

```
8520 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

8521 \renewcommand*{\CustomAbbreviationFields}{%
8522 name={\glxtrfootnotedesname},
8523 sort={\glxtrfootnotedesort},
8524 first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
8525 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8526 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8527 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
8528 \protect\glxtrabbrvfootnote{\the\glslabeltok}%

```



```

8529      {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8530      text={\protect\glsabbrvfont{\the\glsshorttok}},%
8531      plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8532 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8533   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8534   \glsattribute{\the\glslabeltok}{regular}%
8535   {%
8536     \glssetattribute{\the\glslabeltok}{regular}{false}%
8537   }%
8538   {}}%
8539 }%
8540}%
8541{%
8542 \GlsXtrUseAbbrStyleFmts{footnote}%
8543 }

```

**footnote-desc**    Synonym.

```
8544 \letabbreviationstyle{footnote-desc}{short-footnote-desc}
```

**postfootnote**    Similar to footnote but the footnote is placed afterwards, outside the link. This avoids nested links and can also move the footnote marker after any following punctuation mark. Pre v1.07 included `\footnote` in the first keys, which was incorrect as it caused duplicate footnotes.

```

8545 \newabbreviationstyle{postfootnote}%
8546 {%

```

Set accessibility attributes if enabled. (Add `firstshortaccess` since long form is hidden in a footnote on first use.)

```
8547 \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```

8548 \renewcommand*{\CustomAbbreviationFields}{%
8549   name={\glsxtrfootnotename},
8550   sort={\the\glsshorttok},
8551   description={\the\glslongtok},%
8552   first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8553   firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%

8554   text={\protect\glsabbrvfont{\the\glsshorttok}},%
8555   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

8556 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8557   \csdef{glsxtrpostlink\glscategorylabel}{%
8558     \glsxtrifwasfirstuse
8559     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8560      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}}%
8561      {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
8562      }%
8563      {}%
8564      }%
8565      \glshasattribute{\the\glslabeltok}{regular}%
8566      {%
8567      \glissetattribute{\the\glslabeltok}{regular}{false}%
8568      }%
8569      {}%
8570      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8571  \renewcommand*{\glxtrsetupfulldefs}{%
8572  \let\glxtrifwasfirstuse\@secondoftwo
8573  }%
8574 }%
8575 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8576  \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
8577  \renewcommand*{\glxtrabbrvfont}[1]{\glxtrabbrvdefaultfont{##1}}%
8578  \renewcommand*{\glxtrfirstabbrvfont}[1]{\glxtrfirstabbrvdefaultfont{##1}}%
8579  \renewcommand*{\glxtrfirstlongfont}[1]{\glxtrfirstlongfootnotefont{##1}}%
8580  \renewcommand*{\glxtrlongfont}[1]{\glxtrlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8581  \renewcommand*{\glxtrfullformat}[2]{%
8582  \glxtrfirstabbrvfont{\glxtraccessshort{##1}\ifglxtrinsertinside##2\fi}%
8583  \ifglxtrinsertinside\else##2\fi
8584  }%
8585  \renewcommand*{\glxtrfullplformat}[2]{%
8586  \glxtrfirstabbrvfont{\glxtraccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8587  \ifglxtrinsertinside\else##2\fi
8588  }%
8589  \renewcommand*{\Glsxtrfullformat}[2]{%
8590  \glxtrfirstabbrvfont{\Glsxtraccessshort{##1}\ifglxtrinsertinside##2\fi}%
8591  \ifglxtrinsertinside\else##2\fi
8592  }%
8593  \renewcommand*{\Glsxtrfullplformat}[2]{%
8594  \glxtrfirstabbrvfont{\Glsxtraccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8595  \ifglxtrinsertinside\else##2\fi
8596  }%

```

The first use full form and the inline full form use the short (long) style.

```

8597  \renewcommand*{\glxtrininlinefullformat}[2]{%
8598  \glxtrfirstabbrvfont{\glxtraccessshort{##1}\ifglxtrinsertinside##2\fi}%
8599  \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

8600 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8601 }%
8602 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8603 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
8604 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8605 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8606 }%
8607 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8608 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
8609 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8610 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8611 }%
8612 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8613 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
8614 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8615 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8616 }%
8617 }

```

rt-postfootnote

```
8618 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

stfootnote-desc Like short-postfootnote but with user supplied description.

```

8619 \newabbreviationstyle{short-postfootnote-desc}%
8620 {%

```

Set accessibility attributes if enabled.

```
8621 \glsxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

8622 \renewcommand*{\CustomAbbreviationFields}{%
8623 name={\glsxtrfootnotedesname},
8624 sort={\glsxtrfootnotedesort},
8625 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
8626 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
8627 text={\protect\glsabbrvfont{\the\glsshorttok}},%
8628 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

8629 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8630 \csdef{glsxtrpostlink\glscategorylabel}{%
8631 \glsxtrifwasfirstuse
8632 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8633 \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
8634 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%

```

```

8635     }%
8636     {}%
8637     }%
8638     \glshasattribute{\the\glslabeltok}{regular}%
8639     {%
8640         \glissetattribute{\the\glslabeltok}{regular}{false}%
8641     }%
8642     {}%
8643     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8644 \renewcommand*{\glxtrsetupfulldefs}{%
8645     \let\glxtrifwasfirstuse\@secondoftwo
8646 }%
8647 }%
8648 {%
8649     \GlsXtrUseAbbrStyleFmts{postfootnote}%
8650 }

```

stfootnote-desc

```

8651 \letabbreviationstyle{postfootnote-desc}{short-postfootnote-desc}

```

shortnolongname

```

8652 \newcommand*{\glxtrshortnolongname}{%
8653     \protect\glsabbrvfont{\the\glsshorttok}%
8654 }

```

**short** Provide a style that only displays the short form on first use, but the short and long form can be displayed with the “full” commands that use the inline format. If the user supplies a description, the long form won’t be displayed in the predefined glossary styles, but the post description hook can be employed to automatically insert it.

```

8655 \newabbreviationstyle{short}%
8656 {%

```

Set accessibility attributes if enabled.

```

8657 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

8658 \renewcommand*{\CustomAbbreviationFields}{%
8659     name={\glxtrshortnolongname},
8660     sort={\the\glsshorttok},
8661     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8662     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
8663     text={\protect\glsabbrvfont{\the\glsshorttok}},
8664     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
8665     description={\the\glslongtok}}%
8666 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8667     \glissetattribute{\the\glslabeltok}{regular}{true}}%
8668 }%

```

8669 {%

In case the user wants to mix and match font styles, these are redefined here.

```
8670 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
8671 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8672 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
8673 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8674 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
8675 \renewcommand*\glxtrinlinefullformat[2]{%
8676   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8677   \ifglxtrininsertinside##2\fi}%
8678   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8679   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8680 }%
8681 \renewcommand*\glxtrinlinefullplformat[2]{%
8682   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8683   \ifglxtrininsertinside##2\fi}%
8684   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8685   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8686 }%
8687 \renewcommand*\Glsxtrinlinefullformat[2]{%
8688   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
8689   \ifglxtrininsertinside##2\fi}%
8690   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8691   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
8692 }%
8693 \renewcommand*\Glsxtrinlinefullplformat[2]{%
8694   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
8695   \ifglxtrininsertinside##2\fi}%
8696   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8697   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
8698 }%
```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```
8699 \renewcommand*\glxtrfullformat[2]{%
8700   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8701   \ifglxtrininsertinside\else##2\fi
8702 }%
8703 \renewcommand*\glxtrfullplformat[2]{%
8704   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8705   \ifglxtrininsertinside\else##2\fi
8706 }%
8707 \renewcommand*\Glsxtrfullformat[2]{%
8708   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
8709   \ifglxtrininsertinside\else##2\fi
8710 }%
8711 \renewcommand*\Glsxtrfullplformat[2]{%
8712   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
8713 }
```

```

8713 \ifglxtrinsertinside\else##2\fi
8714 }%
8715 }

```

Set this as the default style for acronyms:

```
8716 \setabbreviationstyle[acronym]{short}
```

**short-nolong**

```
8717 \letabbreviationstyle{short-nolong}{short}
```

**rt-nolong-noreg** Like short-nolong but doesn't set the regular attribute.

```

8718 \newabbreviationstyle{short-nolong-noreg}%
8719 {%
8720 \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

8721 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8722 \glshasattribute{\the\glslabeltok}{regular}%
8723 {%
8724 \glissetattribute{\the\glslabeltok}{regular}{false}%
8725 }%
8726 {}%
8727 }%
8728 }%
8729 {%
8730 \GlsXtrUseAbbrStyleFmts{short-nolong}%
8731 }

```

**trshortdescname**

```

8732 \newcommand*{\glxtrshortdescname}{%
8733 \protect\glsabbrvfont{\the\glsshorttok}%
8734 \protect\glxtrfullsep{\the\glslabeltok}%
8735 \protect\glxtrparen{\protect\glslongfont{\the\glslongtok}}%
8736 }

```

**short-desc** The user must supply the description in this style. The long form is added to the name. The short style (possibly with the post-description hooks set) might be a better option.

```

8737 \newabbreviationstyle{short-desc}%
8738 {%

```

Set accessibility attributes if enabled.

```
8739 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```

8740 \renewcommand*{\CustomAbbreviationFields}{%
8741 name={\glxtrshortdescname},
8742 sort={\the\glsshorttok},
8743 first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
8744 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},

```

```

8745     text={\protect\glsabbrvfont{\the\glsshorttok}},
8746     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
8747 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8748   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8749 }%
8750 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8751 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
8752 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
8753 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8754 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8755 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8756 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8757   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8758   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8759   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8760 }%
8761 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8762   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8763   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8764   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8765 }%
8766 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8767   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8768   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8769   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
8770 }%
8771 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8772   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8773   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8774   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
8775 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

8776 \renewcommand*{\glsxtrfullformat}[2]{%
8777   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8778   \ifglsxtrininsertinside\else##2\fi
8779 }%
8780 \renewcommand*{\glsxtrfullplformat}[2]{%
8781   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8782   \ifglsxtrininsertinside\else##2\fi
8783 }%
8784 \renewcommand*{\Glsxtrfullformat}[2]{%
8785   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8786   \ifglsxtrininsertinside\else##2\fi
8787 }%
8788 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

8789 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8790 \ifglxtrinsertinside\else##2\fi
8791 }%
8792 }

```

ort-nolong-desc

```

8793 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

8794 \newabbreviationstyle{short-nolong-desc-noreg}%
8795 {%
8796 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
      Unset the regular attribute if it has been set.
8797 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8798 \glshasattribute{\the\glslabeltok}{regular}%
8799 {%
8800 \glissetattribute{\the\glslabeltok}{regular}{false}%
8801 }%
8802 {}%
8803 }%
8804 }%
8805 {%
8806 \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
8807 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

8808 \newabbreviationstyle{nolong-short}%
8809 {%
8810 \GlsXtrUseAbbrStyleSetup{short-nolong}%
8811 }%
8812 {%
8813 \GlsXtrUseAbbrStyleFmts{short-nolong}%
      The inline full form displays the long form followed by the short form in parentheses.
8814 \renewcommand*{\glxtrinlinefullformat}[2]{%
8815 \protect\glsfirstlongfont{\glsaccesslong{##1}%
8816 \ifglxtrinsertinside##2\fi}%
8817 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8818 \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8819 }%
8820 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8821 \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
8822 \ifglxtrinsertinside##2\fi}%
8823 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8824 \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8825 }%
8826 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8827 \protect\glsfirstlongfont{\glsaccesslong{##1}%

```



```

8828     \ifglxtrinsertinside##2\fi}%
8829     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8830     \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
8831 }%
8832 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8833     \protect\glsfirstlongfont{\Glsaccesslongpl{##1}%
8834         \ifglxtrinsertinside##2\fi}%
8835     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8836     \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
8837 }%
8838 }

```

**ong-short-noreg** Like `nolong-short` but doesn't set the regular attribute.

```

8839 \newabbreviationstyle{nolong-short-noreg}%
8840 {%
8841     \GlsXtrUseAbbrStyleSetup{nolong-short}%
8842     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8843         \glshasattribute{\the\glslabeltok}{regular}%
8844         {%
8845             \glissetattribute{\the\glslabeltok}{regular}{false}%
8846         }%
8847     }%
8848 }%
8849 }%
8850 {%
8851     \GlsXtrUseAbbrStyleFmts{nolong-short}%
8852 }

```

**noshortdesname**

```

8853 \newcommand*{\glxtrlongnoshortdesname}{%
8854     \protect\glslongfont{\the\glslongtok}%
8855 }

```

**long-desc** Provide a style that only displays the long form, but the long and short form can be displayed with the “full” commands that use the inline format. The predefined glossary styles won't show the short form. The user must supply a description for this style. The accessibility attributes don't need setting here.

```

8856 \newabbreviationstyle{long-desc}%
8857 {%
8858     \renewcommand*{\CustomAbbreviationFields}{%
8859         name={\glxtrlongnoshortdesname},
8860         sort={\the\glslongtok},
8861         first={\protect\glsfirstlongfont{\the\glslongtok}},
8862         firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
8863         text={\glslongfont{\the\glslongtok}},
8864         plural={\glslongfont{\the\glslongpltok}}%
8865     }%

```

```

8866 \renewcommand*\GlsXtrPostNewAbbreviation}{%
8867 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8868 }%
8869 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

8870 \renewcommand*\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
8871 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
8872 \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
8873 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8874 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8875 \renewcommand*\glsxtrsubsequentfmt}[2]{%
8876 \glslongfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8877 \ifglsxtrininsertinside \else##2\fi
8878 }%
8879 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
8880 \glslongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8881 \ifglsxtrininsertinside \else##2\fi
8882 }%
8883 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8884 \glslongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8885 \ifglsxtrininsertinside \else##2\fi
8886 }%
8887 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8888 \glslongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8889 \ifglsxtrininsertinside \else##2\fi
8890 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8891 \renewcommand*\glsxtrinlinefullformat}[2]{%
8892 \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8893 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8894 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8895 }%
8896 \renewcommand*\glsxtrinlinefullplformat}[2]{%
8897 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8898 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8899 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8900 }%
8901 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8902 \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8903 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8904 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
8905 }%
8906 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8907 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8908 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8909 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
8910 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

8911 \renewcommand*{\glstrfullformat}[2]{%
8912   \glstrfirstlongfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
8913   \ifglstrinsertinside\else##2\fi
8914 }%
8915 \renewcommand*{\glstrfullplformat}[2]{%
8916   \glstrfirstlongfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
8917   \ifglstrinsertinside\else##2\fi
8918 }%
8919 \renewcommand*{\Glsxtrfullformat}[2]{%
8920   \glstrfirstlongfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
8921   \ifglstrinsertinside\else##2\fi
8922 }%
8923 \renewcommand*{\Glsxtrfullplformat}[2]{%
8924   \glstrfirstlongfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
8925   \ifglstrinsertinside\else##2\fi
8926 }%
8927 }

```

**long-noshort-desc** Provide a synonym that matches similar styles.

```

8928 \letabbreviationstyle{long-noshort-desc}{long-desc}

```

**short-desc-noreg** Like long-noshort-desc but doesn't set the regular attribute.

```

8929 \newabbreviationstyle{long-noshort-desc-noreg}%
8930 {%
8931   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8932   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8933     \glshasattribute{\the\glslabeltok}{regular}%
8934     {%
8935       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
8936     }%
8937   }%
8938 }%
8939 }%
8940 {%
8941   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
8942 }

```

**longnoshortname**

```

8943 \newcommand*{\glstrlongnoshortname}{%
8944   \protect\glabbrvfont{\the\glsshorttok}%
8945 }

```

**long** It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

8946 \newabbreviationstyle{long}%
8947 {%
    Set accessibility attributes if enabled.
8948   \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel
    Setup the default fields.
8949   \renewcommand*{\CustomAbbreviationFields}{%
8950     name={\glxtrlongnoshortname},
8951     sort={\the\glsshorttok},
8952     first={\protect\glsfirstlongfont{\the\glslongtok}},
8953     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
8954     text={\glslongfont{\the\glslongtok}},
8955     plural={\glslongfont{\the\glslongpltok}},%
8956     description={\the\glslongtok}%
8957   }%
8958   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8959     \glsetattribute{\the\glslabeltok}{regular}{true}}%
8960 }%
8961 {%
8962   \GlsXtrUseAbbrStyleFmts{long-desc}%
8963 }

```

**long-noshort** Provide a synonym that matches similar styles.

```

8964 \letabbreviationstyle{long-noshort}{long}

```

**g-noshort-noreg** Like long-noshort but doesn't set the regular attribute.

```

8965 \newabbreviationstyle{long-noshort-noreg}%
8966 {%
8967   \GlsXtrUseAbbrStyleSetup{long-noshort}%
    Unset the regular attribute if it has been set.
8968   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8969     \glshasattribute{\the\glslabeltok}{regular}%
8970     {%
8971       \glsetattribute{\the\glslabeltok}{regular}{false}%
8972     }%
8973   }%
8974 }%
8975 }%
8976 {%
8977   \GlsXtrUseAbbrStyleFmts{long-noshort}%
8978 }

```

### 1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

**\glxtrscfont** Maintained for backward-compatibility.

```

8979 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}

```

`\glsabbrvscfont` Added for consistent naming.

```
8980 \newcommand*{\glsabbrvscfont}{\glsxtrscfont}
```

`\glsxtrfirstscfont` Maintained for backward-compatibility.

```
8981 \newcommand*{\glsxtrfirstscfont}[1]{\glsabbrvscfont{#1}}
```

`\glsfirstabbrvscfont` Added for consistent naming.

```
8982 \newcommand*{\glsfirstabbrvscfont}{\glsxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix` `\protect` needs to come inside `\s` to avoid interfering with all caps.

```
8983 \newcommand*{\glsxtrscsuffix}{\protect\glstextup{\glsxtrabbrvpluralsuffix}}
```

`long-short-sc`

```
8984 \newabbreviationstyle{long-short-sc}%
8985 {%
    Set accessibility attributes if enabled.
8986 \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
    Setup the default fields.
8987 \renewcommand*{\CustomAbbreviationFields}{%
8988     name={\glsxtrlongshortname},
8989     sort={\the\glsshorttok},
8990     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8991         \protect\glsxtrfullsep{\the\glslabeltok}%
8992         \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
8993     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8994         \protect\glsxtrfullsep{\the\glslabeltok}%
8995         \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
8996     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
8997     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},%
8998     description={\the\glslongtok}}%
8999 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9000 \glshasattribute{\the\glslabeltok}{regular}%
9001 {%
9002 \glssetattribute{\the\glslabeltok}{regular}{false}%
9003 }%
9004 {}%
9005 }%
9006 }%
9007 {%
    Use smallcaps and adjust the plural suffix to revert to upright.
9008 \renewcommand*{\abbrvpluralsuffix}{\glsxtrscsuffix}%
9009 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{#1}}%
9010 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{#1}}%
```

Use the default long fonts.

```
9011 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9012 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9013 \renewcommand*{\glsxtrfullformat}[2]{%
9014   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9015   \ifglsxtrinsertinside\else##2\fi
9016   \glsxtrfullsep{##1}%
9017   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9018 }%
9019 \renewcommand*{\glsxtrfullplformat}[2]{%
9020   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9021   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9022   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9023 }%
9024 \renewcommand*{\Glsxtrfullformat}[2]{%
9025   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9026   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9027   \glsxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
9028 }%
9029 \renewcommand*{\Glsxtrfullplformat}[2]{%
9030   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9031   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9032   \glsxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
9033 }%
9034 }
```

g-short-sc-desc

```
9035 \newabbreviationstyle{long-short-sc-desc}%
9036 {%
```

Set accessibility attributes if enabled.

```
9037 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
9038 \renewcommand*{\CustomAbbreviationFields}{%
9039   name={\glsxtrlongshortdescname},
9040   sort={\glsxtrlongshortdescsort},%
9041   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
9042     \protect\glsxtrfullsep{\the\glslabeltok}%
9043     \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
9044   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
9045     \protect\glsxtrfullsep{\the\glslabeltok}%
9046     \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
9047   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9048   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
9049 }%
```

Unset the regular attribute if it has been set.

```
9050 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```

9051 \glshasattribute{\the\glslabeltok}{regular}%
9052 {%
9053 \glissetattribute{\the\glslabeltok}{regular}{false}%
9054 }%
9055 {}%
9056 }%
9057 }%
9058 {%

```

As long-short-sc style:

```

9059 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
9060 }

```

short-sc-long Now the short (long) version

```

9061 \newabbreviationstyle{short-sc-long}%
9062 {%

```

Set accessibility attributes if enabled.

```

9063 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

9064 \renewcommand*{\CustomAbbreviationFields}{%
9065 name={\glxtrshortlongname},
9066 sort={\the\glsshorttok},
9067 description={\the\glslongtok},%
9068 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9069 \protect\glxtrfullsep{\the\glslabeltok}%
9070 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9071 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9072 \protect\glxtrfullsep{\the\glslabeltok}%
9073 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9074 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9075 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9076 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9077 \glshasattribute{\the\glslabeltok}{regular}%
9078 {%
9079 \glissetattribute{\the\glslabeltok}{regular}{false}%
9080 }%
9081 {}%
9082 }%
9083 }%
9084 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9085 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9086 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9087 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9088 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9089 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline form are the same for this style.

```

9090 \renewcommand*{\glxtrfullformat}[2]{%
9091   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9092   \ifglxtrinsertinside\else##2\fi
9093   \glxtrfullsep{##1}%
9094   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9095 }%
9096 \renewcommand*{\glxtrfullplformat}[2]{%
9097   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9098   \ifglxtrinsertinside\else##2\fi
9099   \glxtrfullsep{##1}%
9100   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9101 }%
9102 \renewcommand*{\Glsxtrfullformat}[2]{%
9103   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9104   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9105   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9106 }%
9107 \renewcommand*{\Glsxtrfullplformat}[2]{%
9108   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9109   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9110   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9111 }%
9112 }

```

rt-sc-long-desc As before but user provides description

```

9113 \newabbreviationstyle{short-sc-long-desc}%
9114 {%

```

Set accessibility attributes if enabled.

```

9115 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9116 \renewcommand*{\CustomAbbreviationFields}{%
9117   name={\glxtrshortlongdescname},
9118   sort={\glxtrshortlongdescsort},
9119   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9120     \protect\glxtrfullsep{\the\glslabeltok}%
9121     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9122   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9123     \protect\glxtrfullsep{\the\glslabeltok}%
9124     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9125   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9126   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
9127 }%

```

Unset the regular attribute if it has been set.

```

9128 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9129   \glshasattribute{\the\glslabeltok}{regular}%
9130   {%

```



```

9131 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9132 }%
9133 {}%
9134 }%
9135 }%
9136 {%

```

As short-sc-long style:

```

9137 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
9138 }

```

short-sc

```

9139 \newabbreviationstyle{short-sc}%
9140 {%

```

Set accessibility attributes if enabled.

```

9141 \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9142 \renewcommand*{\CustomAbbreviationFields}{%
9143   name={\glsxtrshortnolongname},
9144   sort={\the\glssshorttok},
9145   first={\protect\glsfirstabbrvscfont{\the\glssshorttok}},
9146   firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}},
9147   text={\protect\glsabbrvscfont{\the\glssshorttok}},
9148   plural={\protect\glsabbrvscfont{\the\glssshortpltok}},
9149   description={\the\glslongtok}}%
9150 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9151   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9152 }%
9153 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9154 \renewcommand*{\abbrvpluralsuffix}{\glsxtrscsuffix}%
9155 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9156 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9157 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9158 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9159 \renewcommand*{\glxtrinlinefullformat}[2]{%
9160   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
9161   \ifglxtrininsertinside##2\fi}%
9162   \ifglxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9163   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9164 }%
9165 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9166   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
9167   \ifglxtrininsertinside##2\fi}%
9168   \ifglxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9169   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9170 }%

```

```

9171 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9172   \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}%
9173     \ifglxtrinsertinside##2\fi}%
9174   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9175   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
9176 }%
9177 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9178   \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}%
9179     \ifglxtrinsertinside##2\fi}%
9180   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9181   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
9182 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

9183 \renewcommand*{\glxtrfullformat}[2]{%
9184   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglxtrinsertinside##2\fi}%
9185   \ifglxtrinsertinside\else##2\fi
9186 }%
9187 \renewcommand*{\glxtrfullplformat}[2]{%
9188   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglxtrinsertinside##2\fi}%
9189   \ifglxtrinsertinside\else##2\fi
9190 }%
9191 \renewcommand*{\Glsxtrfullformat}[2]{%
9192   \glsfirstabbrvscfont{\Glsaccesssshort{##1}\ifglxtrinsertinside##2\fi}%
9193   \ifglxtrinsertinside\else##2\fi
9194 }%
9195 \renewcommand*{\Glsxtrfullplformat}[2]{%
9196   \glsfirstabbrvscfont{\Glsaccesssshortpl{##1}\ifglxtrinsertinside##2\fi}%
9197   \ifglxtrinsertinside\else##2\fi
9198 }%
9199 }

```

short-sc-nolong

```
9200 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```

9201 \newabbreviationstyle{short-sc-desc}{%
9202 {%

```

Set accessibility attributes if enabled.

```
9203 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

9204 \renewcommand*{\CustomAbbreviationFields}{%
9205   name={\glxtrshortdescname},
9206   sort={\the\glsshorttok},
9207   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
9208   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
9209   text={\protect\glsabbrvscfont{\the\glsshorttok}},

```

```

9210 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
9211 \renewcommand*\GlsXtrPostNewAbbreviation}{%
9212   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9213 }%
9214 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9215 \renewcommand*\abbrvpluralsuffix{\glsxtrscsuffix}%
9216 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
9217 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
9218 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9219 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9220 \renewcommand*\glsxtrinlinefullformat[2]{%
9221   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9222   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9223   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9224 }%
9225 \renewcommand*\glsxtrinlinefullplformat[2]{%
9226   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9227   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9228   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9229 }%
9230 \renewcommand*\Glsxtrinlinefullformat[2]{%
9231   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9232   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9233   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
9234 }%
9235 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9236   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9237   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9238   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
9239 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

9240 \renewcommand*\glsxtrfullformat[2]{%
9241   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9242   \ifglsxtrininsertinside\else##2\fi
9243 }%
9244 \renewcommand*\glsxtrfullplformat[2]{%
9245   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9246   \ifglsxtrininsertinside\else##2\fi
9247 }%
9248 \renewcommand*\Glsxtrfullformat[2]{%
9249   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9250   \ifglsxtrininsertinside\else##2\fi
9251 }%
9252 \renewcommand*\Glsxtrfullplformat[2]{%
9253   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

9254 \ifglxtrinsertinside\else##2\fi
9255 }%
9256 }

```

-sc-nolong-desc

```

9257 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

9258 \newabbreviationstyle{nolong-short-sc}%
9259 {%
9260 \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
9261 }%
9262 {%
9263 \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

9264 \renewcommand*{\glxtrinlinefullformat}[2]{%
9265 \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
9266 \ifglxtrinsertinside##2\fi}%
9267 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9268 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9269 }%
9270 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9271 \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9272 \ifglxtrinsertinside##2\fi}%
9273 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9274 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9275 }%
9276 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9277 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9278 \ifglxtrinsertinside##2\fi}%
9279 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9280 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9281 }%
9282 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9283 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9284 \ifglxtrinsertinside##2\fi}%
9285 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9286 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9287 }%
9288 }

```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`. No accessibility attributes needed here.

```

9289 \newabbreviationstyle{long-noshort-sc}%
9290 {%
9291 \renewcommand*{\CustomAbbreviationFields}{%
9292 name={\glxtrlongnoshortname},
9293 sort={\the\glsshorttok},

```

```

9294     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9295     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9296     text={\protect\glslongdefaultfont{\the\glslongtok}},
9297     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9298     description={\the\glslongtok}%
9299 }%
9300 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9301   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9302 }%
9303 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9304 \renewcommand*{\abbrvpluralsuffix}{\glsxtrscsuffix}%
9305 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9306 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9307 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9308 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9309 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9310   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9311   \ifglsxtrininsertinside \else##2\fi
9312 }%
9313 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9314   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9315   \ifglsxtrininsertinside \else##2\fi
9316 }%
9317 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9318   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9319   \ifglsxtrininsertinside \else##2\fi
9320 }%
9321 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9322   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9323   \ifglsxtrininsertinside \else##2\fi
9324 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9325 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9326   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9327   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9328   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9329 }%
9330 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9331   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9332   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9333   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9334 }%
9335 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9336   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9337   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9338   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%

```

```

9339 }%
9340 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9341   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9342   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9343   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
9344 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

9345 \renewcommand*{\glxtrfullformat}[2]{%
9346   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9347   \ifglxtrinsertinside\else##2\fi
9348 }%
9349 \renewcommand*{\glxtrfullplformat}[2]{%
9350   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9351   \ifglxtrinsertinside\else##2\fi
9352 }%
9353 \renewcommand*{\Glsxtrfullformat}[2]{%
9354   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9355   \ifglxtrinsertinside\else##2\fi
9356 }%
9357 \renewcommand*{\Glsxtrfullplformat}[2]{%
9358   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9359   \ifglxtrinsertinside\else##2\fi
9360 }%
9361 }

```

long-sc Backward compatibility:

```

9362 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}

```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9363 \newabbreviationstyle{long-noshort-sc-desc}%
9364 {%
9365   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9366 }%
9367 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9368 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9369 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9370 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9371 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9372 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9373 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9374   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9375   \ifglxtrinsertinside \else##2\fi
9376 }%

```

```

9377 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9378   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9379   \ifglxtrinsertinside \else##2\fi
9380 }%
9381 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9382   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9383   \ifglxtrinsertinside \else##2\fi
9384 }%
9385 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9386   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9387   \ifglxtrinsertinside \else##2\fi
9388 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9389 \renewcommand*{\glxtrinlinefullformat}[2]{%
9390   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9391   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9392   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9393 }%
9394 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9395   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9396   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9397   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9398 }%
9399 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9400   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9401   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9402   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
9403 }%
9404 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9405   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9406   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9407   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
9408 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

9409 \renewcommand*{\glxtrfullformat}[2]{%
9410   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9411   \ifglxtrinsertinside\else##2\fi
9412 }%
9413 \renewcommand*{\glxtrfullplformat}[2]{%
9414   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9415   \ifglxtrinsertinside\else##2\fi
9416 }%
9417 \renewcommand*{\Glsxtrfullformat}[2]{%
9418   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9419   \ifglxtrinsertinside\else##2\fi
9420 }%
9421 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

9422 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9423 \ifglxtrinsertinside\else##2\fi
9424 }%
9425 }

```

long-desc-sc Backward compatibility:

```

9426 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

9427 \newabbreviationstyle{short-sc-footnote}%
9428 {%

```

Set accessibility attributes if enabled.

```

9429 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9430 \renewcommand*{\CustomAbbreviationFields}{%
9431   name={\glxtrfootnotename},
9432   sort={\the\glsshorttok},
9433   description={\the\glslongtok},%
9434   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
9435     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9436     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9437   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
9438     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9439     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9440   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9441   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

9442 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9443   \glsetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9444   \glshasattribute{\the\glslabeltok}{regular}%
9445   {%
9446     \glsetattribute{\the\glslabeltok}{regular}{false}%
9447   }%
9448   {}%
9449 }%
9450 }%
9451 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

9452 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9453 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
9454 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
9455 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9456 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9457 \renewcommand*{\glxtrfullformat}[2]{%

```



```

9458 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9459 \ifglxtrinsertinside\else##2\fi
9460 \protect\glxtrabbrvfootnote{##1}%
9461 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9462 }%
9463 \renewcommand*{\glxtrfullplformat}[2]{%
9464 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9465 \ifglxtrinsertinside\else##2\fi
9466 \protect\glxtrabbrvfootnote{##1}%
9467 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9468 }%
9469 \renewcommand*{\Glsxtrfullformat}[2]{%
9470 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9471 \ifglxtrinsertinside\else##2\fi
9472 \protect\glxtrabbrvfootnote{##1}%
9473 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9474 }%
9475 \renewcommand*{\Glsxtrfullplformat}[2]{%
9476 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9477 \ifglxtrinsertinside\else##2\fi
9478 \protect\glxtrabbrvfootnote{##1}%
9479 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9480 }%

```

The first use full form and the inline full form use the short (long) style.

```

9481 \renewcommand*{\glxtrinlinefullformat}[2]{%
9482 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9483 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9484 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9485 }%
9486 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9487 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9488 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9489 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9490 }%
9491 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9492 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9493 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9494 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9495 }%
9496 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9497 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9498 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9499 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9500 }%
9501 }

```

footnote-sc Backward compatibility:

```

9502 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

c-footnote-desc Like short-sc-footnote but with user supplied description.

```
9503 \newabbreviationstyle{short-sc-footnote-desc}%  
9504 {%
```

Set accessibility attributes if enabled.

```
9505 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
9506 \renewcommand*{\CustomAbbreviationFields}{%  
9507   name={\glxtrfootnotedesname},  
9508   sort={\glxtrfootnotedesort},  
9509   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%  
9510     \protect\glxtrabbrvfootnote{\the\glslabeltok}%  
9511     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%  
9512   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%  
9513     \protect\glxtrabbrvfootnote{\the\glslabeltok}%  
9514     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%  
9515   text={\protect\glsabbrvscfont{\the\glsshorttok}},%  
9516   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
9517 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
9518   \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%  
9519   \glshasattribute{\the\glslabeltok}{regular}%  
9520   {%  
9521     \glsssetattribute{\the\glslabeltok}{regular}{false}%  
9522   }%  
9523   {}%  
9524 }%  
9525 }%  
9526 {%  
9527   \GlsXtrUseAbbrStyleFmts{short-sc-footnote}%  
9528 }
```

sc-postfootnote

```
9529 \newabbreviationstyle{short-sc-postfootnote}%  
9530 {%
```

Set accessibility attributes if enabled.

```
9531 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
9532 \renewcommand*{\CustomAbbreviationFields}{%  
9533   name={\glxtrfootnotename},  
9534   sort={\the\glsshorttok},  
9535   description={\the\glslongtok},%  
9536   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%  
9537   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%  
9538   text={\protect\glsabbrvscfont{\the\glsshorttok}},%  
9539   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```
9540 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9541   \csdef{glxtrpostlink\glscategorylabel}{%
9542     \glxtrifwasfirstuse
9543     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
9544       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9545       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9546     }%
9547   }%
9548 }%
9549 \glshasattribute{\the\glslabeltok}{regular}%
9550 {%
9551   \glissetattribute{\the\glslabeltok}{regular}{false}%
9552   }%
9553   }%
9554 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
9555 \renewcommand*{\glxtrsetupfulldefs}{%
9556   \let\glxtrifwasfirstuse\@secondoftwo
9557 }%
9558 }%
9559 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
9560 \renewcommand*{\abbrvpluralsuffix}{\glxtrscsuffix}%
9561 \renewcommand*{\glssuffixfont}[1]{\glssuffixfont{##1}}%
9562 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvfont{##1}}%
9563 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9564 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
9565 \renewcommand*{\glxtrfullformat}[2]{%
9566   \glsfirstabbrvfont{\glssuccessshort{##1}\ifglxtrininsertinside##2\fi}%
9567   \ifglxtrininsertinside\else##2\fi
9568 }%
9569 \renewcommand*{\glxtrfullplformat}[2]{%
9570   \glsfirstabbrvfont{\glssuccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9571   \ifglxtrininsertinside\else##2\fi
9572 }%
9573 \renewcommand*{\Glsxtrfullformat}[2]{%
9574   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9575   \ifglxtrininsertinside\else##2\fi
9576 }%
9577 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

9578 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9579 \ifglxtrinsertinside\else##2\fi
9580 }%

```

The first use full form and the inline full form use the short (long) style.

```

9581 \renewcommand*{\glxtrinlinefullformat}[2]{%
9582 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9583 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9584 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9585 }%
9586 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9587 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9588 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9589 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9590 }%
9591 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9592 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9593 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9594 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9595 }%
9596 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9597 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9598 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9599 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9600 }%
9601 }

```

postfootnote-sc Backward compatibility:

```

9602 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

stfootnote-desc Like short-sc-footnote but with user supplied description.

```

9603 \newabbreviationstyle{short-sc-postfootnote-desc}%
9604 {%

```

Set accessibility attributes if enabled.

```

9605 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9606 \renewcommand*{\CustomAbbreviationFields}{%
9607 name={\glxtrfootnotedesname},
9608 sort={\glxtrfootnotedesort},
9609 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
9610 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
9611 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
9612 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

9613 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9614 \csdef{glxtrpostlink\glscategorylabel}{%
9615 \glxtrifwasfirstuse

```

```

9616      {%
Needs the specific font command here as the style may have been lost by the time the foot-
note occurs.
9617      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9618      {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
9619      }%
9620      {%
9621      }%
9622      \glshasattribute{\the\glslabeltok}{regular}%
9623      {%
9624      \glissetattribute{\the\glslabeltok}{regular}{false}%
9625      }%
9626      {%
9627      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9628 \renewcommand*{\glxtrsetupfulldefs}{%
9629 \let\glxtrifwasfirstuse\@secondoftwo
9630 }%
9631 }%
9632 {%
9633 \GlsXtrUseAbbrStyleFmts{short-sc-postfootnote}%
9634 }

```

#### 1.7.4 Predefined Styles (Fake Small Capitals)

These styles require the `relsize` package, which must be loaded by the user. These styles all use:

```

\glxtrsmfont    Maintained for backward compatibility.
9635 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

\glsabbrvsmfont Added for consistent naming.
9636 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
9637 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
9638 \newcommand*{\glsfirstabbrvsmfont}{\glxtrfirstsmfont}

and for the default short form suffix:

\glxtrsmsuffix
9639 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}

```

long-short-sm

```
9640 \newabbreviationstyle{long-short-sm}%  
9641 {%
```

Set accessibility attributes if enabled.

```
9642 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9643 \renewcommand*{\CustomAbbreviationFields}{%  
9644   name={\glxtrlongshortname},  
9645   sort={\the\glsshorttok},  
9646   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%  
9647     \protect\glxtrfullsep{\the\glslabeltok}%  
9648     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%  
9649   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%  
9650     \protect\glxtrfullsep{\the\glslabeltok}%  
9651     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%  
9652   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%  
9653   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%  
9654   description={\the\glslongtok}}%  
9655 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
9656   \glshasattribute{\the\glslabeltok}{regular}%  
9657   {%  
9658     \glissetattribute{\the\glslabeltok}{regular}{false}%  
9659   }%  
9660   {}%  
9661 }%  
9662 }%  
9663 {%
```

```
9664 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%  
9665 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%  
9666 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
```

Use the default long fonts.

```
9667 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%  
9668 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9669 \renewcommand*\glxtrfullformat[2]{%  
9670   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%  
9671   \ifglxtrininsertinside\else##2\fi  
9672   \glxtrfullsep{##1}%  
9673   \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshort{##1}}}%  
9674 }%  
9675 \renewcommand*\glxtrfullplformat[2]{%  
9676   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%  
9677   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
9678   \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%  
9679 }%  
9680 \renewcommand*\Glsxtrfullformat[2]{%
```

```

9681 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9682 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9683 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
9684 }%
9685 \renewcommand*{\Glsxtrfullplformat}[2]{%
9686 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9687 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9688 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
9689 }%
9690 }

```

g-short-sm-desc

```

9691 \newabbreviationstyle{long-short-sm-desc}%
9692 {%
    Set accessibility attributes if enabled.
9693 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9694 \renewcommand*{\CustomAbbreviationFields}{%
9695   name={\glxtrlongshortdescname},
9696   sort={\glxtrlongshortdescsort},%
9697   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
9698     \protect\glxtrfullsep{\the\glslabeltok}%
9699     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
9700   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
9701     \protect\glxtrfullsep{\the\glslabeltok}%
9702     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
9703   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9704   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9705 }%

```

Unset the regular attribute if it has been set.

```

9706 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9707   \glshasattribute{\the\glslabeltok}{regular}%
9708   {%
9709     \glissetattribute{\the\glslabeltok}{regular}{false}%
9710   }%
9711   {}%
9712 }%
9713 }%
9714 {%

```

As long-short-sm style:

```

9715 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
9716 }

```

short-sm-long Now the short (long) version

```

9717 \newabbreviationstyle{short-sm-long}%
9718 {%

```

Set accessibility attributes if enabled.

```
9719 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
9720 \renewcommand*{\CustomAbbreviationFields}{%
9721   name={\glxtrshortlongname},
9722   sort={\the\glsshorttok},
9723   description={\the\glslongtok},%
9724   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
9725     \protect\glxtrfullsep{\the\glslabeltok}%
9726     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9727   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
9728     \protect\glxtrfullsep{\the\glslabeltok}%
9729     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9730   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9731   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
```

Unset the regular attribute if it has been set.

```
9732 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9733   \glshasattribute{\the\glslabeltok}{regular}%
9734   {%
9735     \glssetattribute{\the\glslabeltok}{regular}{false}%
9736   }%
9737   {}%
9738 }%
9739 }%
9740 {%
```

```
9741 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
9742 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
9743 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
9744 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9745 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9746 \renewcommand*\glxtrfullformat[2]{%
9747   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9748   \ifglxtrininsertinside\else##2\fi
9749   \glxtrfullsep{##1}%
9750   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9751 }%
9752 \renewcommand*\glxtrfullplformat[2]{%
9753   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9754   \ifglxtrininsertinside\else##2\fi
9755   \glxtrfullsep{##1}%
9756   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9757 }%
9758 \renewcommand*\Glsxtrfullformat[2]{%
9759   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9760   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
```



```

9761 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9762 }%
9763 \renewcommand*{\Glsxtrfullplformat}[2]{%
9764 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9765 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9766 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9767 }%
9768 }

```

rt-sm-long-desc As before but user provides description

```

9769 \newabbreviationstyle{short-sm-long-desc}%
9770 {%

```

Set accessibility attributes if enabled.

```

9771 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

9772 \renewcommand*{\CustomAbbreviationFields}{%
9773 name={\glxtrshortlongdescname},
9774 sort={\glxtrshortlongdescsort},
9775 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
9776 \protect\glxtrfullsep{\the\glslabeltok}%
9777 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
9778 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
9779 \protect\glxtrfullsep{\the\glslabeltok}%
9780 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
9781 text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
9782 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9783 }%

```

Unset the regular attribute if it has been set.

```

9784 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9785 \glshasattribute{\the\glslabeltok}{regular}%
9786 {%
9787 \glssetattribute{\the\glslabeltok}{regular}{false}%
9788 }%
9789 {}%
9790 }%
9791 }%
9792 {%

```

As short-sm-long style:

```

9793 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
9794 }

```

short-sm

```

9795 \newabbreviationstyle{short-sm}%
9796 {%

```

Set accessibility attributes if enabled.

```

9797 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

9798 \renewcommand*{\CustomAbbreviationFields}{%
9799     name={\glxtrshortnolongname},
9800     sort={\the\glsshorttok},
9801     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
9802     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
9803     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
9804     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
9805     description={\the\glslongtok}}%
9806 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9807     \glssetAttribute{\the\glslabeltok}{regular}{true}}%
9808 }%
9809 {%
9810 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
9811 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
9812 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
9813 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9814 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

9815 \renewcommand*{\glxtrinlinefullformat}[2]{%
9816     \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
9817     \ifglxtrininsertinside##2\fi}%
9818     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9819     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
9820 }%
9821 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9822     \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
9823     \ifglxtrininsertinside##2\fi}%
9824     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9825     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
9826 }%
9827 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9828     \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}%
9829     \ifglxtrininsertinside##2\fi}%
9830     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9831     \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
9832 }%
9833 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9834     \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}%
9835     \ifglxtrininsertinside##2\fi}%
9836     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9837     \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
9838 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

9839 \renewcommand*{\glxtrfullformat}[2]{%

```

```

9840 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9841 \ifglxtrinsertinside\else##2\fi
9842 }%
9843 \renewcommand*{\glxtrfullplformat}[2]{%
9844 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9845 \ifglxtrinsertinside\else##2\fi
9846 }%
9847 \renewcommand*{\Glsxtrfullformat}[2]{%
9848 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9849 \ifglxtrinsertinside\else##2\fi
9850 }%
9851 \renewcommand*{\Glsxtrfullplformat}[2]{%
9852 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9853 \ifglxtrinsertinside\else##2\fi
9854 }%
9855 }

```

short-sm-nolong

```

9856 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

9857 \newabbreviationstyle{short-sm-desc}%
9858 {%

```

Set accessibility attributes if enabled.

```

9859 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

9860 \renewcommand*{\CustomAbbreviationFields}{%
9861 name={\glxtrshortdescname},
9862 sort={\the\glsshorttok},
9863 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
9864 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
9865 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
9866 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
9867 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9868 \glssetattribute{\the\glslabeltok}{regular}{true}}%
9869 }%
9870 {%

```

```

9871 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
9872 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
9873 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
9874 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9875 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

9876 \renewcommand*{\glxtrinlinefullformat}[2]{%
9877 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9878 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

9879 \glstrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
9880 }%
9881 \renewcommand*{\glstrinlinefullplformat}[2]{%
9882 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
9883 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
9884 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
9885 }%
9886 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9887 \glfirstabbrvsmfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
9888 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
9889 \glstrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
9890 }%
9891 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9892 \glfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
9893 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
9894 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
9895 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

9896 \renewcommand*{\glstrfullformat}[2]{%
9897 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglstrinsertinside##2\fi}%
9898 \ifglstrinsertinside\else##2\fi
9899 }%
9900 \renewcommand*{\glstrfullplformat}[2]{%
9901 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
9902 \ifglstrinsertinside\else##2\fi
9903 }%
9904 \renewcommand*{\Glsxtrfullformat}[2]{%
9905 \glfirstabbrvsmfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
9906 \ifglstrinsertinside\else##2\fi
9907 }%
9908 \renewcommand*{\Glsxtrfullplformat}[2]{%
9909 \glfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
9910 \ifglstrinsertinside\else##2\fi
9911 }%
9912 }

```

-sm-nolong-desc

```

9913 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

9914 \newabbreviationstyle{nolong-short-sm}%
9915 {%
9916 \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
9917 }%
9918 {%
9919 \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

9920 \renewcommand*{\glxtrinlinefullformat}[2]{%
9921   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
9922     \ifglxtrinsertinside##2\fi}%
9923   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9924   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9925 }%
9926 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9927   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9928     \ifglxtrinsertinside##2\fi}%
9929   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9930   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9931 }%
9932 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9933   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9934     \ifglxtrinsertinside##2\fi}%
9935   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9936   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9937 }%
9938 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9939   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9940     \ifglxtrinsertinside##2\fi}%
9941   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9942   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9943 }%
9944 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9945 \newabbreviationstyle{long-noshort-sm}%
9946 {}%

```

Set accessibility attributes if enabled.

```

9947 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

9948 \renewcommand*{\CustomAbbreviationFields}{%
9949   name={\glxtrlongnoshortname},
9950   sort={\the\glsshorttok},
9951   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9952   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9953   text={\protect\glslongdefaultfont{\the\glslongtok}},
9954   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9955   description={\the\glslongtok}%
9956 }%
9957 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9958   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9959 }%
9960 {}%
9961 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%

```

```

9962 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
9963 \renewcommand*\glsabbrvpluralsuffix{\glsxtrsmsuffix}%
9964 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
9965 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9966 \renewcommand*\glsxtrsubsequentfmt}[2]{%
9967   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9968   \ifglsxtrininsertinside \else##2\fi
9969 }%
9970 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
9971   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9972   \ifglsxtrininsertinside \else##2\fi
9973 }%
9974 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
9975   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9976   \ifglsxtrininsertinside \else##2\fi
9977 }%
9978 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
9979   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9980   \ifglsxtrininsertinside \else##2\fi
9981 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9982 \renewcommand*\glsxtrinlinefullformat}[2]{%
9983   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9984   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9985   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9986 }%
9987 \renewcommand*\glsxtrinlinefullplformat}[2]{%
9988   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9989   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9990   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
9991 }%
9992 \renewcommand*\Glsxtrinlinefullformat}[2]{%
9993   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9994   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9995   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
9996 }%
9997 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
9998   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9999   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10000   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10001 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

10002 \renewcommand*\glsxtrfullformat}[2]{%
10003   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10004   \ifglsxtrininsertinside\else##2\fi

```

```

10005 }%
10006 \renewcommand*{\glxtrfullplformat}[2]{%
10007   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10008   \ifglxtrinsertinside\else##2\fi
10009 }%
10010 \renewcommand*{\Glsxtrfullformat}[2]{%
10011   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10012   \ifglxtrinsertinside\else##2\fi
10013 }%
10014 \renewcommand*{\Glsxtrfullplformat}[2]{%
10015   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10016   \ifglxtrinsertinside\else##2\fi
10017 }%
10018 }

```

long-sm Backward compatibility:

```

10019 \@glxtr@deprecated@abbrstyle{long-sm}{long-noshort-sm}

```

noshort-sm-desc The smaller font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

10020 \newabbreviationstyle{long-noshort-sm-desc}%
10021 {%
10022   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
10023 }%
10024 {%
10025   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
10026   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10027   \renewcommand*\abbrvpluralsuffix{\glsxtrsmsuffix}%
10028   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
10029   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10030 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10031   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10032   \ifglxtrinsertinside \else##2\fi
10033 }%
10034 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10035   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10036   \ifglxtrinsertinside \else##2\fi
10037 }%
10038 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10039   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
10040   \ifglxtrinsertinside \else##2\fi
10041 }%
10042 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10043   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
10044   \ifglxtrinsertinside \else##2\fi
10045 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10046 \renewcommand*{\glxtrinlinefullformat}[2]{%
10047   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10048   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10049   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
10050 }%
10051 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10052   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10053   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10054   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
10055 }%
10056 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10057   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10058   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10059   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
10060 }%
10061 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10062   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10063   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10064   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
10065 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

10066 \renewcommand*{\glxtrfullformat}[2]{%
10067   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10068   \ifglxtrininsertinside\else##2\fi
10069 }%
10070 \renewcommand*{\glxtrfullplformat}[2]{%
10071   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10072   \ifglxtrininsertinside\else##2\fi
10073 }%
10074 \renewcommand*{\Glsxtrfullformat}[2]{%
10075   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10076   \ifglxtrininsertinside\else##2\fi
10077 }%
10078 \renewcommand*{\Glsxtrfullplformat}[2]{%
10079   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10080   \ifglxtrininsertinside\else##2\fi
10081 }%
10082 }

```

long-desc-sm Backward compatibility:

```

10083 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

10084 \newabbreviationstyle{short-sm-footnote}%
10085 {}

```



Set accessibility attributes if enabled.

```
10086 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10087 \renewcommand*{\CustomAbbreviationFields}{%
10088   name={\glxtrfootnotename},
10089   sort={\the\glsshorttok},
10090   description={\the\glslongtok},%
10091   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
10092     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10093     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10094   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10095     \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10096     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10097   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
10098   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
10099 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10100   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
10101   \glsattribute{\the\glslabeltok}{regular}%
10102   {%
10103     \glssetattribute{\the\glslabeltok}{regular}{false}%
10104   }%
10105   {}%
10106 }%
10107 }%
10108 {%

10109 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
10110 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
10111 \renewcommand*{\abbrvpluralsuffix}{\glxtrsmsuffix}%
10112 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
10113 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
10114 \renewcommand*{\glxtrfullformat}[2]{%
10115   \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10116   \ifglxtrininsertinside\else##2\fi
10117   \protect\glxtrabbrvfootnote{##1}%
10118   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10119 }%
10120 \renewcommand*{\glxtrfullplformat}[2]{%
10121   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
10122   \ifglxtrininsertinside\else##2\fi
10123   \protect\glxtrabbrvfootnote{##1}%
10124   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10125 }%
10126 \renewcommand*{\Glsxtrfullformat}[2]{%
```

```

10127 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10128 \ifglxtrinsertinside\else##2\fi
10129 \protect\glxtrabbrvfootnote{##1}%
10130 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10131 }%
10132 \renewcommand*{\Glsxtrfullplformat}[2]{%
10133 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10134 \ifglxtrinsertinside\else##2\fi
10135 \protect\glxtrabbrvfootnote{##1}%
10136 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10137 }%

```

The first use full form and the inline full form use the short (long) style.

```

10138 \renewcommand*{\glxtrinlinefullformat}[2]{%
10139 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10140 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10141 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
10142 }%
10143 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10144 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10145 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10146 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
10147 }%
10148 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10149 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10150 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10151 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
10152 }%
10153 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10154 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10155 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10156 \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
10157 }%
10158 }

```

**footnote-sm** Backward compatibility:

```
10159 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

**m-footnote-desc** Like short-footnote but with user supplied description.

```

10160 \newabbreviationstyle{short-sm-footnote-desc}%
10161 {%

```

Set accessibility attributes if enabled.

```
10162 \glxtrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```

10163 \renewcommand*{\CustomAbbreviationFields}{%
10164 name={\glxtrfootnotedesname},
10165 sort={\glxtrfootnotedesort},
10166 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%

```

```

10167 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10168 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
10169 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
10170 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
10171 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
10172 text={\protect\glsabbrvsmfont{\the\glsshorttok}}},%
10173 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

10174 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10175 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
10176 \glshasattribute{\the\glslabeltok}{regular}%
10177 {%
10178 \glssetattribute{\the\glslabeltok}{regular}{false}%
10179 }%
10180 {}%
10181 }%
10182}%
10183{%
10184 \GlsXtrUseAbbrStyleFmts{short-sm-footnote}%
10185}

```

sm-postfootnote

```

10186 \newabbreviationstyle{short-sm-postfootnote}%
10187 {%

```

Set accessibility attributes if enabled.

```

10188 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10189 \renewcommand*{\CustomAbbreviationFields}{%
10190 name={\glxtrfootnotename},
10191 sort={\the\glsshorttok},
10192 description={\the\glslongtok},%
10193 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
10194 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
10195 text={\protect\glsabbrvsmfont{\the\glsshorttok}}},%
10196 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

10197 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10198 \csdef{glxtrpostlink\glscategorylabel}{%
10199 \glxtrifwasfirstuse
10200 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

10201 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
10202 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%

```

```

10203     }%
10204     {}%
10205     }%
10206     \glshasattribute{\the\glslabeltok}{regular}%
10207     {%
10208         \glissetattribute{\the\glslabeltok}{regular}{false}%
10209     }%
10210     {}%
10211 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

10212 \renewcommand*\glxtrsetupfulldefs{%
10213     \let\glxtrifwasfirstuse\@secondoftwo
10214 }%
10215 }%
10216 {%
10217 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
10218 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
10219 \renewcommand*\abbrvpluralsuffix{\glxtrsmsuffix}%
10220 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
10221 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

10222 \renewcommand*\glxtrfullformat[2]{%
10223     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10224     \ifglxtrininsertinside\else##2\fi
10225 }%
10226 \renewcommand*\glxtrfullplformat[2]{%
10227     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10228     \ifglxtrininsertinside\else##2\fi
10229 }%
10230 \renewcommand*\Glsxtrfullformat[2]{%
10231     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10232     \ifglxtrininsertinside\else##2\fi
10233 }%
10234 \renewcommand*\Glsxtrfullplformat[2]{%
10235     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10236     \ifglxtrininsertinside\else##2\fi
10237 }%

```

The first use full form and the inline full form use the short (long) style.

```

10238 \renewcommand*\glxtrininlinefullformat[2]{%
10239     \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10240     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10241     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
10242 }%
10243 \renewcommand*\glxtrininlinefullplformat[2]{%
10244     \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10245     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

10246 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
10247 }%
10248 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10249 \glfirstabbrvsmfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
10250 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
10251 \glstrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
10252 }%
10253 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10254 \glfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
10255 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
10256 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
10257 }%
10258 }

```

postfootnote-sm Backward compatibility:

```

10259 \@glstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

stfootnote-desc Like short-sm-postfootnote but with user supplied description.

```

10260 \newabbreviationstyle{short-sm-postfootnote-desc}%
10261 {%

```

Set accessibility attributes if enabled.

```

10262 \glstrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

10263 \renewcommand*{\CustomAbbreviationFields}{%
10264 name={\glstrfootnotedesname},
10265 sort={\glstrfootnotedesort},
10266 first={\protect\glfirstabbrvsmfont{\the\glsshorttok}},%
10267 firstplural={\protect\glfirstabbrvsmfont{\the\glsshortpltok}},%
10268 text={\protect\glabbrvsmfont{\the\glsshorttok}},%
10269 plural={\protect\glabbrvsmfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

10270 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10271 \csdef{glstrpostlink\glscategorylabel}{%
10272 \glstrifwasfirstuse
10273 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

10274 \glstrdopostpunc{\protect\glstrabbrvfootnote{\glslabel}%
10275 {\glfirstlongfootnotefont{\glentrylong{\glslabel}}}}%
10276 }%
10277 }%
10278 }%
10279 \glshasattribute{\the\glslabeltok}{regular}%
10280 {%
10281 \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

10282 }%
10283 {}%
10284 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

10285 \renewcommand*{\glxtrsetupfulldefs}{%
10286 \let\glxtrifwasfirstuse\@secondoftwo
10287 }%
10288 }%
10289 {%
10290 \GlsXtrUseAbbrStyleFmts{short-sm-postfootnote}%
10291 }

```

### 1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

10292 \newcommand*{\glsabbrvemfont}[1]{\emph{#1}}%

```

`\glsfirstabbrvemfont`

```

10293 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{#1}}%

```

The default short form suffix:

`\glxtremsuffix`

```

10294 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```

10295 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{#1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```

10296 \newcommand*{\glslongemfont}[1]{\emph{#1}}%

```

`\long-short-em` The long form is just set in the default long font.

```

10297 \newabbreviationstyle{long-short-em}%
10298 {%

```

Set accessibility attributes if enabled.

```

10299 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

10300 \renewcommand*{\CustomAbbreviationFields}{%
10301 name={\glxtrlongshortname},
10302 sort={\the\glsshorttok},
10303 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}}%
10304 \protect\glxtrfullsep{\the\glslabeltok}}%
10305 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10306 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}}%

```

```

10307 \protect\glxtrfullsep{\the\glslabeltok}%
10308 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10309 text={\protect\glssabbbrvemfont{\the\glsshortttok}}},%
10310 plural={\protect\glssabbbrvemfont{\the\glsshortpltok}}},%
10311 description={\the\glslongtok}}%
10312 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10313 \glshasattribute{\the\glslabeltok}{regular}}%
10314 {%
10315 \glissetattribute{\the\glslabeltok}{regular}{false}}%
10316 }%
10317 {}%
10318 }%
10319 }%
10320 {%

10321 \renewcommand*\glssabbbrvfont[1]{\glssabbbrvemfont{##1}}%
10322 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
10323 \renewcommand*{\abbrvpluralsuffix}{\glstremssuffix}%

```

Use the default long fonts.

```

10324 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
10325 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10326 \renewcommand*\glxtrfullformat[2]{%
10327 \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10328 \ifglxtrininsertinside\else##2\fi
10329 \glxtrfullsep{##1}}%
10330 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10331 }%
10332 \renewcommand*\glxtrfullplformat[2]{%
10333 \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10334 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10335 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
10336 }%
10337 \renewcommand*\Glsxtrfullformat[2]{%
10338 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10339 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10340 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10341 }%
10342 \renewcommand*\Glsxtrfullplformat[2]{%
10343 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10344 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}}%
10345 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
10346 }%
10347 }

```

g-short-em-desc

```

10348 \newabbreviationstyle{long-short-em-desc}%
10349 {%

```

Set accessibility attributes if enabled.

```
10350 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
```

Setup the default fields.

```
10351 \renewcommand*{\CustomAbbreviationFields}{%
10352   name={\glxtrlongshortdescname},
10353   sort={\glxtrlongshortdescsort},%
10354   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
10355     \protect\glxtrfullsep{\the\glslabeltok}%
10356     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10357   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
10358     \protect\glxtrfullsep{\the\glslabeltok}%
10359     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10360   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10361   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
10362 }%
```

Unset the regular attribute if it has been set.

```
10363 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10364   \glshasattribute{\the\glslabeltok}{regular}%
10365   {%
10366     \glissetattribute{\the\glslabeltok}{regular}{false}%
10367   }%
10368   {}%
10369 }%
10370 }%
10371 {%
```

As long-short-em style:

```
10372 \GlsXtrUseAbbrStyleFmts{long-short-em}%
10373 }
```

ong-em-short-em

```
10374 \newabbreviationstyle{long-em-short-em}%
10375 {%
```

Set accessibility attributes if enabled.

```
10376 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields. \glslongemfont is used in the description since \glstdesc doesn't set the style.

```
10377 \renewcommand*{\CustomAbbreviationFields}{%
10378   name={\glxtrlongshortname},
10379   sort={\the\glsshorttok},
10380   first={\protect\glsfirstlongemfont{\the\glslongtok}%
10381     \protect\glxtrfullsep{\the\glslabeltok}%
10382     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10383   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10384     \protect\glxtrfullsep{\the\glslabeltok}%
10385     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
```



```

10386     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10387     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
10388     description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10389 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10390   \glsattribute{\the\glslabeltok}{regular}%
10391   {%
10392     \glsattribute{\the\glslabeltok}{regular}{false}%
10393   }%
10394   {}%
10395 }%
10396}%
10397{%

10398 \renewcommand*{\abbrvpluralsuffix}{\glstremsuffix}%
10399 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10400 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10401 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10402 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10403 \renewcommand*{\glsxtrfullformat}[2]{%
10404   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10405   \ifglsxtrininsertinside\else##2\fi
10406   \glsxtrfullsep{##1}%
10407   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10408 }%
10409 \renewcommand*{\glsxtrfullplformat}[2]{%
10410   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10411   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10412   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10413 }%
10414 \renewcommand*{\Glsxtrfullformat}[2]{%
10415   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10416   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10417   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10418 }%
10419 \renewcommand*{\Glsxtrfullplformat}[2]{%
10420   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10421   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10422   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10423 }%
10424 }

```

m-short-em-desc

```

10425 \newabbreviationstyle{long-em-short-em-desc}%
10426 {%

```

Set accessibility attributes if enabled.

```

10427   \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```
10428 \renewcommand*{\CustomAbbreviationFields}{%
10429     name={\glxtrlongshortdescname},
10430     sort={\glxtrlongshortdescsort},%
10431     first={\protect\glsfirstlongemfont{\the\glslongtok}%
10432         \protect\glxtrfullsep{\the\glslabeltok}%
10433         \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
10434     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
10435         \protect\glxtrfullsep{\the\glslabeltok}%
10436         \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
10437     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10438     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
10439 }
```

Unset the regular attribute if it has been set.

```
10440 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10441     \glshasattribute{\the\glslabeltok}{regular}%
10442     {%
10443         \glissetattribute{\the\glslabeltok}{regular}{false}%
10444     }%
10445     {}%
10446 }%
10447 }%
10448 {%
10449     \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
10450 }
```

short-em-long Now the short (long) version

```
10451 \newabbreviationstyle{short-em-long}%
10452 {%
```

Set accessibility attributes if enabled.

```
10453 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
10454 \renewcommand*{\CustomAbbreviationFields}{%
10455     name={\glxtrshortlongname},
10456     sort={\the\glsshorttok},
10457     description={\the\glslongtok},%
10458     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10459         \protect\glxtrfullsep{\the\glslabeltok}%
10460         \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
10461     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10462         \protect\glxtrfullsep{\the\glslabeltok}%
10463         \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
10464     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10465     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
10466 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10467     \glshasattribute{\the\glslabeltok}{regular}%

```

```

10468   {%
10469     \glsetattribute{\the\glslabelltok}{regular}{false}%
10470   }%
10471   {}%
10472 }%
10473 }%
10474 {%

```

Mostly as short-long style:

```

10475 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%
10476 \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
10477 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
10478 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
10479 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10480 \renewcommand*{\glxtrfullformat}[2]{%
10481   \glfirstabbrvemfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10482   \ifglxtrininsertinside\else##2\fi
10483   \glxtrfullsep{##1}%
10484   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
10485 }%
10486 \renewcommand*{\glxtrfullplformat}[2]{%
10487   \glfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10488   \ifglxtrininsertinside\else##2\fi
10489   \glxtrfullsep{##1}%
10490   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
10491 }%
10492 \renewcommand*{\Glsxtrfullformat}[2]{%
10493   \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
10494   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10495   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
10496 }%
10497 \renewcommand*{\Glsxtrfullplformat}[2]{%
10498   \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
10499   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10500   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
10501 }%
10502 }

```

rt-em-long-desc As before but user provides description

```

10503 \newabbreviationstyle{short-em-long-desc}%
10504 {%

```

Set accessibility attributes if enabled.

```

10505 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10506 \renewcommand*{\CustomAbbreviationFields}{%
10507   name={\glxtrshortlongdescname},
10508   sort={\glxtrshortlongdescsort},

```

```

10509   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10510     \protect\glsextrfullsep{\the\glslabeltok}%
10511     \glsextrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
10512   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10513     \protect\glsextrfullsep{\the\glslabeltok}%
10514     \glsextrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
10515   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10516   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
10517 }%

```

Unset the regular attribute if it has been set.

```

10518 \renewcommand*\GlsXtrPostNewAbbreviation{%
10519   \glshasattribute{\the\glslabeltok}{regular}%
10520   {%
10521     \glissetattribute{\the\glslabeltok}{regular}{false}%
10522   }%
10523   {}%
10524 }%
10525}%
10526{%
10527 \GlsXtrUseAbbrStyleFmts{short-em-long}%
10528 }

```

hort-em-long-em

```

10529 \newabbreviationstyle{short-em-long-em}%
10530 {%

```

Set accessibility attributes if enabled.

```

10531 \glsextrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields. \glslongemfont is used in the description since \glsdesc doesn't set the style.

```

10532 \renewcommand*\CustomAbbreviationFields{%
10533   name={\glsextrshortlongname},
10534   sort={\the\glsshorttok},
10535   description={\protect\glslongemfont{\the\glslongtok}},%
10536   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
10537     \protect\glsextrfullsep{\the\glslabeltok}%
10538     \glsextrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
10539   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10540     \protect\glsextrfullsep{\the\glslabeltok}%
10541     \glsextrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
10542   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10543   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%

```

Unset the regular attribute if it has been set.

```

10544 \renewcommand*\GlsXtrPostNewAbbreviation{%
10545   \glshasattribute{\the\glslabeltok}{regular}%
10546   {%
10547     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

10548 }%
10549 {}%
10550 }%
10551 }%
10552 {%

10553 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10554 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10555 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10556 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
10557 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10558 \renewcommand*{\glsxtrfullformat}[2]{%
10559   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10560   \ifglsxtrininsertinside\else##2\fi
10561   \glsxtrfullsep{##1}%
10562   \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
10563 }%
10564 \renewcommand*{\glsxtrfullplformat}[2]{%
10565   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10566   \ifglsxtrininsertinside\else##2\fi
10567   \glsxtrfullsep{##1}%
10568   \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
10569 }%
10570 \renewcommand*{\Glsxtrfullformat}[2]{%
10571   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
10572   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10573   \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
10574 }%
10575 \renewcommand*{\Glsxtrfullplformat}[2]{%
10576   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
10577   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10578   \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
10579 }%
10580 }

```

em-long-em-desc

```

10581 \newabbreviationstyle{short-em-long-em-desc}%
10582 {%

```

Set accessibility attributes if enabled.

```

10583 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

10584 \renewcommand*{\CustomAbbreviationFields}{%
10585   name={\glsxtrshortlongdescname},%
10586   sort={\glsxtrshortlongdescsort},%
10587   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
10588   \protect\glsxtrfullsep{\the\glslabeltok}}%
10589   \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%

```

```

10590   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
10591   \protect\glsxtrfullsep{\the\glslabeltok}%
10592   \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
10593   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
10594   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
10595 }%

```

Unset the regular attribute if it has been set.

```

10596 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10597   \glsasattribute{\the\glslabeltok}{regular}%
10598   {%
10599     \glssetattribute{\the\glslabeltok}{regular}{false}%
10600   }%
10601   {}}%
10602 }%
10603 }%
10604 {%
10605   \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
10606 }

```

short-em

```

10607 \newabbreviationstyle{short-em}%
10608 {%

```

Set accessibility attributes if enabled.

```

10609   \glsxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

10610 \renewcommand*{\CustomAbbreviationFields}{%
10611   name={\glsxtrshortnolongname},
10612   sort={\the\glsshorttok},
10613   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10614   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10615   text={\protect\glsabbrvemfont{\the\glsshorttok}},
10616   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
10617   description={\the\glslongtok}}%
10618 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10619   \glssetattribute{\the\glslabeltok}{regular}{true}}%
10620 }%
10621 {%
10622   \renewcommand*{\abbrvpluralsuffix}{\glsxtrsuffix}%
10623   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10624   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10625   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10626   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

10627 \renewcommand*{\glxtrinlinefullformat}[2]{%
10628   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
10629   \ifglxtrininsertinside##2\fi}%

```

```

10630 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10631 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10632 }%
10633 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10634 \protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}%
10635 \ifglxtrinsertinside##2\fi}%
10636 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10637 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10638 }%

10639 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10640 \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}%
10641 \ifglxtrinsertinside##2\fi}%
10642 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10643 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10644 }%
10645 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10646 \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}%
10647 \ifglxtrinsertinside##2\fi}%
10648 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10649 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10650 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

10651 \renewcommand*{\glxtrfullformat}[2]{%
10652 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10653 \ifglxtrinsertinside\else##2\fi
10654 }%
10655 \renewcommand*{\glxtrfullplformat}[2]{%
10656 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10657 \ifglxtrinsertinside\else##2\fi
10658 }%
10659 \renewcommand*{\Glsxtrfullformat}[2]{%
10660 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
10661 \ifglxtrinsertinside\else##2\fi
10662 }%
10663 \renewcommand*{\Glsxtrfullplformat}[2]{%
10664 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
10665 \ifglxtrinsertinside\else##2\fi
10666 }%
10667 }

```

short-em-nolong

```
10668 \letabbreviationstyle{short-em-nolong}{short-em}
```

short-em-desc

```

10669 \newabbreviationstyle{short-em-desc}%
10670 {}

```

Set accessibility attributes if enabled. The default name includes the long form but `\glxtrshortdesname` could be modified to omit the long form, so include the `nameshortaccess` attribute.

```
10671 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
```

Setup the default fields.

```
10672 \renewcommand*{\CustomAbbreviationFields}{%
10673   name={\glxtrshortdesname},
10674   sort={\the\glsshorttok},
10675   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
10676   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
10677   text={\protect\glssabbrvemfont{\the\glsshorttok}},
10678   plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
10679 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10680   \glsssetattribute{\the\glslabeltok}{regular}{true}}%
10681 }%
10682 {%

10683 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10684 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
10685 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10686 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10687 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short format followed by the long form in parentheses.

```
10688 \renewcommand*{\glxtrinlinefullformat}[2]{%
10689   \glsfirstabbrvemfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10690   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10691   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
10692 }%

10693 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10694   \glsfirstabbrvemfont{\glssaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
10695   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10696   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
10697 }%

10698 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10699   \glsfirstabbrvemfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10700   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10701   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
10702 }%

10703 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10704   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
10705   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10706   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
10707 }%
```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```
10708 \renewcommand*{\glxtrfullformat}[2]{%
10709   \glsfirstabbrvemfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
10710   \ifglxtrininsertinside\else##2\fi
```



```

10711 }%
10712 \renewcommand*{\glxstrfullplformat}[2]{%
10713   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
10714   \ifglxstrinsertinside\else##2\fi
10715 }%
10716 \renewcommand*{\Glsxtrfullformat}[2]{%
10717   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxstrinsertinside##2\fi}%
10718   \ifglxstrinsertinside\else##2\fi
10719 }%
10720 \renewcommand*{\Glsxtrfullplformat}[2]{%
10721   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
10722   \ifglxstrinsertinside\else##2\fi
10723 }%
10724 }

```

-em-nolong-desc

```

10725 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

10726 \newabbreviationstyle{nolong-short-em}%
10727 {%
10728   \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
10729 }%
10730 {%
10731   \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

10732 \renewcommand*{\glxstrinlinefullformat}[2]{%
10733   \protect\glsfirstlongdefaultfont{\glssaccesslong{##1}%
10734     \ifglxstrinsertinside##2\fi}%
10735   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10736   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10737 }%
10738 \renewcommand*{\glxstrinlinefullplformat}[2]{%
10739   \protect\glsfirstlongdefaultfont{\glssaccesslongpl{##1}%
10740     \ifglxstrinsertinside##2\fi}%
10741   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10742   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
10743 }%
10744 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10745   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
10746     \ifglxstrinsertinside##2\fi}%
10747   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10748   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
10749 }%
10750 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10751   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
10752     \ifglxstrinsertinside##2\fi}%
10753   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
10754   \glxstrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%

```

```
10755 }%
10756 }
```

long-noshort-em The short form is explicitly invoked through commands like \glssshort.

```
10757 \newabbreviationstyle{long-noshort-em}%
10758 {%
```

Set accessibility attributes if enabled.

```
10759 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel
```

Setup the default fields.

```
10760 \renewcommand*{\CustomAbbreviationFields}{%
10761   name={\glxtrlongnoshortname},
10762   sort={\the\glssshorttok},
10763   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
10764   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
10765   text={\protect\glslongdefaultfont{\the\glslongtok}},
10766   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
10767   description={\the\glslongtok}%
10768 }%
10769 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10770   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
10771 }%
10772 {%

10773 \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
10774 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10775 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10776 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10777 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
10778 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10779   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10780   \ifglxtrininsertinside \else##2\fi
10781 }%
10782 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10783   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10784   \ifglxtrininsertinside \else##2\fi
10785 }%
10786 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10787   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10788   \ifglxtrininsertinside \else##2\fi
10789 }%
10790 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10791   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10792   \ifglxtrininsertinside \else##2\fi
10793 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
10794 \renewcommand*{\glsxtrinlinefullformat}[2]{%
```

```

10795 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10796 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10797 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10798 }%
10799 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10800 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10801 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10802 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10803 }%
10804 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10805 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10806 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10807 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10808 }%
10809 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10810 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10811 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10812 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10813 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

10814 \renewcommand*{\glxtrfullformat}[2]{%
10815 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10816 \ifglxtrinsertinside\else##2\fi
10817 }%
10818 \renewcommand*{\glxtrfullplformat}[2]{%
10819 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10820 \ifglxtrinsertinside\else##2\fi
10821 }%
10822 \renewcommand*{\Glsxtrfullformat}[2]{%
10823 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10824 \ifglxtrinsertinside\else##2\fi
10825 }%
10826 \renewcommand*{\Glsxtrfullplformat}[2]{%
10827 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10828 \ifglxtrinsertinside\else##2\fi
10829 }%
10830 }

```

long-em Backward compatibility:

```
10831 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

10832 \newabbreviationstyle{long-em-noshort-em}%
10833 {}

```

Set accessibility attributes if enabled.

```
10834 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel
```

Setup the default fields.

```

10835 \renewcommand*{\CustomAbbreviationFields}{%
10836   name={\glxstrlongnoshortname},
10837   sort={\the\glsshorttok},
10838   first={\protect\glxfirstlongemfont{\the\glslongtok}},
10839   firstplural={\protect\glxfirstlongemfont{\the\glslongpltok}},
10840   text={\protect\glslongemfont{\the\glslongtok}},
10841   plural={\protect\glslongemfont{\the\glslongpltok}},%
10842   description={\protect\glslongemfont{\the\glslongtok}}}%
10843 }%
10844 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10845   \glsetattribute{\the\glslabeltok}{regular}{true}}%
10846 }%
10847 {%

10848 \renewcommand*{\abbrvpluralsuffix}{\glxxtremsuffix}%
10849 \renewcommand*{\glxabbrvfont}[1]{\glxabbrvemfont{##1}}%
10850 \renewcommand*{\glxfirstabbrvfont}[1]{\glxfirstabbrvemfont{##1}}%
10851 \renewcommand*{\glxfirstlongfont}[1]{\glxfirstlongemfont{##1}}%
10852 \renewcommand*{\glxlongfont}[1]{\glxlongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10853 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10854   \glxlongemfont{\glxsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10855   \ifglxtrininsertinside \else##2\fi
10856 }%
10857 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10858   \glxlongemfont{\glxsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10859   \ifglxtrininsertinside \else##2\fi
10860 }%
10861 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10862   \glxlongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
10863   \ifglxtrininsertinside \else##2\fi
10864 }%
10865 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10866   \glxlongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
10867   \ifglxtrininsertinside \else##2\fi
10868 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10869 \renewcommand*{\glxtrininlinefullformat}[2]{%
10870   \glxfirstlongemfont{\glxsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10871   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10872   \glxtrparen{\protect\glxfirstabbrvemfont{\glxsaccessshort{##1}}}%
10873 }%
10874 \renewcommand*{\glxtrininlinefullplformat}[2]{%
10875   \glxfirstlongemfont{\glxsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10876   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
10877   \glxtrparen{\protect\glxfirstabbrvemfont{\glxsaccessshortpl{##1}}}%
10878 }%

```

```

10879 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10880   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10881   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10882   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
10883 }%
10884 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10885   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10886   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10887   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
10888 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

10889 \renewcommand*{\glxtrfullformat}[2]{%
10890   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10891   \ifglxtrinsertinside\else##2\fi
10892 }%
10893 \renewcommand*{\glxtrfullplformat}[2]{%
10894   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10895   \ifglxtrinsertinside\else##2\fi
10896 }%
10897 \renewcommand*{\Glsxtrfullformat}[2]{%
10898   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10899   \ifglxtrinsertinside\else##2\fi
10900 }%
10901 \renewcommand*{\Glsxtrfullplformat}[2]{%
10902   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10903   \ifglxtrinsertinside\else##2\fi
10904 }%
10905 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

10906 \newabbreviationstyle{long-em-noshort-em-noreg}%
10907 {}%

```

Set accessibility attributes if enabled.

```

10908 \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

10909 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

10910 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10911   \glshasattribute{\the\glslabeltok}{regular}%
10912   {%
10913     \glissetattribute{\the\glslabeltok}{regular}{false}%
10914   }%
10915   {}%
10916 }%
10917 }%
10918 {}%

```

```

10919 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
10920 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

10921 \newabbreviationstyle{long-noshort-em-desc}%
10922 {%
10923 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
10924 }%
10925 {%

10926 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%
10927 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
10928 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
10929 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
10930 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

10931 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
10932 \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10933 \ifglsxtrininsertinside \else##2\fi
10934 }%
10935 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
10936 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10937 \ifglsxtrininsertinside \else##2\fi
10938 }%
10939 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10940 \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
10941 \ifglsxtrininsertinside \else##2\fi
10942 }%
10943 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10944 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
10945 \ifglsxtrininsertinside \else##2\fi
10946 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

10947 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10948 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10949 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10950 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
10951 }%
10952 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10953 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10954 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10955 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
10956 }%
10957 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10958 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10959 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
10960 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

10961 }%
10962 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10963   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10964   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
10965   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
10966 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

10967 \renewcommand*{\glxtrfullformat}[2]{%
10968   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10969   \ifglxtrinsertinside\else##2\fi
10970 }%
10971 \renewcommand*{\glxtrfullplformat}[2]{%
10972   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10973   \ifglxtrinsertinside\else##2\fi
10974 }%
10975 \renewcommand*{\Glsxtrfullformat}[2]{%
10976   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10977   \ifglxtrinsertinside\else##2\fi
10978 }%
10979 \renewcommand*{\Glsxtrfullplformat}[2]{%
10980   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10981   \ifglxtrinsertinside\else##2\fi
10982 }%
10983 }

```

long-desc-em Backward compatibility:

```

10984 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like `\glxtrshort`. The long form is emphasized. No accessibility attributes need to be set.

```

10985 \newabbreviationstyle{long-em-noshort-em-desc}%
10986 {%
10987   \renewcommand*{\CustomAbbreviationFields}{%
10988     name={\glxtrlongnoshortdescname},
10989     sort={\the\glslongtok},
10990     first={\protect\glsfirstlongemfont{\the\glslongtok}},
10991     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
10992     text={\glslongemfont{\the\glslongtok}},
10993     plural={\glslongemfont{\the\glslongpltok}}}%
10994   }%
10995   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10996     \glssetattribute{\the\glslabeltok}{regular}{true}}%
10997 }%
10998 {%
10999   \renewcommand*{\abbrvpluralsuffix}{\glxtremsuffix}%
11000   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
11001   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%

```

```

11002 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
11003 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

11004 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
11005   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11006   \ifglsxtrininsertinside \else##2\fi
11007 }%
11008 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
11009   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11010   \ifglsxtrininsertinside \else##2\fi
11011 }%
11012 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11013   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
11014   \ifglsxtrininsertinside \else##2\fi
11015 }%
11016 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11017   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
11018   \ifglsxtrininsertinside \else##2\fi
11019 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

11020 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11021   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11022   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11023   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
11024 }%
11025 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
11026   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
11027   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11028   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
11029 }%
11030 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11031   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11032   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11033   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
11034 }%
11035 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11036   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
11037   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11038   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
11039 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

11040 \renewcommand*{\glsxtrfullformat}[2]{%
11041   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
11042   \ifglsxtrininsertinside\else##2\fi
11043 }%
11044 \renewcommand*{\glsxtrfullplformat}[2]{%

```



```

11045 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11046 \ifglxtrinsertinside\else##2\fi
11047 }%
11048 \renewcommand*{\Glsxtrfullformat}[2]{%
11049 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11050 \ifglxtrinsertinside\else##2\fi
11051 }%
11052 \renewcommand*{\Glsxtrfullplformat}[2]{%
11053 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11054 \ifglxtrinsertinside\else##2\fi
11055 }%
11056 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

11057 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
11058 {%
11059 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
    Unset the regular attribute if it has been set.
11060 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11061 \glshasattribute{\the\glslabeltok}{regular}%
11062 {%
11063 \glssetattribute{\the\glslabeltok}{regular}{false}%
11064 }%
11065 {}%
11066 }%
11067 }%
11068 {%
11069 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
11070 }

```

ort-em-footnote

```

11071 \newabbreviationstyle{short-em-footnote}%
11072 {%
    Set accessibility attributes if enabled.
11073 \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel
    Setup the default fields.
11074 \renewcommand*{\CustomAbbreviationFields}{%
11075 name={\glxtrfootnotename},
11076 sort={\the\glsshorttok},
11077 description={\the\glslongtok},%
11078 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
11079 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11080 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
11081 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
11082 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
11083 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
11084 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
11085 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

11086 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11087   \glsssetattribute{\the\glslabelltok}{nohyperfirst}{true}%
11088   \glshasattribute{\the\glslabelltok}{regular}%
11089   {%
11090     \glsssetattribute{\the\glslabelltok}{regular}{false}%
11091   }%
11092   {}%
11093 }%
11094 }%
11095 {%

11096 \renewcommand*{\abbrvpluralsuffix}{\glstxtremsuffix}%
11097 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
11098 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
11099 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
11100 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

11101 \renewcommand*{\glsxtrfullformat}[2]{%
11102   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11103   \ifglsxtrininsertinside\else##2\fi
11104   \protect\glsxtrabbrvfootnote{##1}%
11105   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11106 }%
11107 \renewcommand*{\glsxtrfullplformat}[2]{%
11108   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11109   \ifglsxtrininsertinside\else##2\fi
11110   \protect\glsxtrabbrvfootnote{##1}%
11111   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11112 }%
11113 \renewcommand*{\Glsxtrfullformat}[2]{%
11114   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11115   \ifglsxtrininsertinside\else##2\fi
11116   \protect\glsxtrabbrvfootnote{##1}%
11117   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11118 }%
11119 \renewcommand*{\Glsxtrfullplformat}[2]{%
11120   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11121   \ifglsxtrininsertinside\else##2\fi
11122   \protect\glsxtrabbrvfootnote{##1}%
11123   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11124 }%

```

The first use full form and the inline full form use the short (long) style.

```

11125 \renewcommand*{\glsxtrinlinefullformat}[2]{%
11126   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11127   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11128   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%

```

```

11129 }%
11130 \renewcommand*{\glxstrinlinefullplformat}[2]{%
11131   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
11132   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
11133   \glxstrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
11134 }%
11135 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11136   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
11137   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
11138   \glxstrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
11139 }%
11140 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11141   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
11142   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
11143   \glxstrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
11144 }%
11145 }

```

footnote-em Backward compatibility:

```

11146 \@glxstr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

m-footnote-desc Like short-em-footnote but with user supplied description.

```

11147 \newabbreviationstyle{short-em-footnote-desc}%
11148 {%

```

Set accessibility attributes if enabled.

```

11149 \glxstrAccSuppAbbrSetNameLongAttrs\glscategorylabel

```

Setup the default fields.

```

11150 \renewcommand*{\CustomAbbreviationFields}{%
11151   name={\glxstrfootnotedesname},
11152   sort={\glxstrfootnotedesort},
11153   first={\protect\glsfirstabbrvemfont{\the\glssshorttok}%
11154     \protect\glxstrabbrvfootnote{\the\glslabeltok}%
11155     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
11156   firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}%
11157     \protect\glxstrabbrvfootnote{\the\glslabeltok}%
11158     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
11159   text={\protect\glssabbrvemfont{\the\glssshorttok}},%
11160   plural={\protect\glssabbrvemfont{\the\glssshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

11161 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11162   \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
11163   \glshasattribute{\the\glslabeltok}{regular}%
11164   {%
11165     \glsssetattribute{\the\glslabeltok}{regular}{false}%
11166   }%
11167   {}%

```

```

11168 }%
11169 }%
11170 {%
11171   \GlsXtrUseAbbrStyleFmts{short-em-footnote}%
11172 }

```

em-postfootnote

```

11173 \newabbreviationstyle{short-em-postfootnote}%
11174 {%

```

Set accessibility attributes if enabled.

```

11175   \glxtrAccSuppAbbrSetNoLongAttrs\glscategorylabel

```

Setup the default fields.

```

11176   \renewcommand*{\CustomAbbreviationFields}{%
11177     name={\glxtrfootnotename},
11178     sort={\the\glsshorttok},
11179     description={\the\glslongtok},%
11180     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
11181     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
11182     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
11183     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

11184   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11185     \csdef{glxtrpostlink\glscategorylabel}{%
11186       \glxtrifwasfirstuse
11187     }%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

11188       \glxtrdopostpunc{\protect\glxtrabbrvffootnote{\glslabel}%
11189         {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
11190     }%
11191   }%
11192 }%
11193   \glshasattribute{\the\glslabeltok}{regular}%
11194   {%
11195     \glissetattribute{\the\glslabeltok}{regular}{false}%
11196   }%
11197   {}%
11198 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

11199   \renewcommand*{\glxtrsetupfulldefs}{%
11200     \let\glxtrifwasfirstuse\@secondoftwo
11201   }%
11202 }%
11203 {%

```

```

11204 \renewcommand*\abbrvpluralsuffix{\glxxtremsuffix}%
11205 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
11206 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
11207 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
11208 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

11209 \renewcommand*\glsxtrfullformat[2]{%
11210   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11211   \ifglsxtrininsertinside\else##2\fi
11212 }%
11213 \renewcommand*\glsxtrfullplformat[2]{%
11214   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11215   \ifglsxtrininsertinside\else##2\fi
11216 }%
11217 \renewcommand*\Glsxtrfullformat[2]{%
11218   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11219   \ifglsxtrininsertinside\else##2\fi
11220 }%
11221 \renewcommand*\Glsxtrfullplformat[2]{%
11222   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11223   \ifglsxtrininsertinside\else##2\fi
11224 }%

```

The first use full form and the inline full form use the short (long) style.

```

11225 \renewcommand*\glsxtrininlinefullformat[2]{%
11226   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11227   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11228   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11229 }%
11230 \renewcommand*\glsxtrininlinefullplformat[2]{%
11231   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11232   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11233   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11234 }%
11235 \renewcommand*\Glsxtrininlinefullformat[2]{%
11236   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
11237   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11238   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
11239 }%
11240 \renewcommand*\Glsxtrininlinefullplformat[2]{%
11241   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
11242   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
11243   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
11244 }%
11245 }

```

postfootnote-em Backward compatibility:

```

11246 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

stfootnote-desc Like short-em-postfootnote but with user supplied description.

```
11247 \newabbreviationstyle{short-em-postfootnote-desc}%  
11248 {%
```

Set accessibility attributes if enabled.

```
11249 \glxstrAccSuppAbbrSetNameLongAttrs\glscategorylabel
```

Setup the default fields.

```
11250 \renewcommand*{\CustomAbbreviationFields}{%  
11251   name={\glxstrfootnotedesname},  
11252   sort={\glxstrfootnotedesort},  
11253   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%  
11254   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%  
11255   text={\protect\glsabbrvemfont{\the\glsshorttok}},%  
11256   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```
11257 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
11258   \csdef{glxtrpostlink\glscategorylabel}{%  
11259     \glxstrifwasfirstuse  
11260     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
11261       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%  
11262       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%  
11263     }%  
11264   }%  
11265 }%  
11266 \glshasattribute{\the\glslabeltok}{regular}%  
11267 {%  
11268   \glissetattribute{\the\glslabeltok}{regular}{false}%  
11269   }%  
11270 }%  
11271 }%
```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```
11272 \renewcommand*{\glxtrsetupfulldefs}{%  
11273   \let\glxtrifwasfirstuse\@secondoftwo  
11274 }%  
11275 }%  
11276 {%  
11277   \GlsXtrUseAbbrStyleFmts{short-em-postfootnote}%  
11278 }
```

### 1.7.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

`glxtruserfield` Default is the useri field.

```
11279 \newcommand*{\glxtruserfield}{useri}
```

`glxtruserparen` The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If `\glscurrentfieldvalue` has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```
11280 \ifdef\glscurrentfieldvalue
11281 {
11282   \newcommand*{\glxtruserparen}[2]{%
11283     \glxtrfullsep{#2}%
11284     \glxtrparen
11285     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
11286   }
11287 }
11288 {
11289   \newcommand*{\glxtruserparen}[2]{%
11290     \glxtrfullsep{#2}%
11291     \glxtrparen
11292     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
11293   }
11294 }
```

Font used for short form:

`lsabbrvuserfont`

```
11295 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}
```

Font used for short form on first use:

`stabbrvuserfont`

```
11296 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}
```

Font used for long form:

`glslonguserfont`

```
11297 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

`rstlonguserfont`

```
11298 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

`lsxtrusersuffix`

```
11299 \newcommand*{\lsxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

Description encapsulator.

`userdescription` The first argument is the description. The second argument is the label.

```
11300 \newcommand*{\gluserdescription}[2]{\glslonguserfont{#1}}
```

long-short-user

```
11301 \newabbreviationstyle{long-short-user}%  
11302 {%
```

Set accessibility attributes if enabled.

```
11303 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11304 \renewcommand*{\CustomAbbreviationFields}{%  
11305   name={\glxtrlongshortname},  
11306   sort={\the\glsshorttok},  
11307   first={\protect\glsfirstlonguserfont{\the\glslongtok}%  
11308     \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%  
11309     {\the\glslabeltok}},%  
11310   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%  
11311     \protect\glxtruserparen  
11312     {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%  
  
11313   text={\protect\glssabbrvuserfont{\the\glsshorttok}},%  
11314   plural={\protect\glssabbrvuserfont{\the\glsshortpltok}},%  
11315   description={\protect\glssuserdescription{\the\glslongtok}%  
11316     {\the\glslabeltok}}}%
```

Unset the regular attribute if it has been set.

```
11317 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
11318   \glshasattribute{\the\glslabeltok}{regular}%  
11319   {%  
11320     \glissetattribute{\the\glslabeltok}{regular}{false}%  
11321     }%  
11322   }%  
11323 }%  
11324 }%  
11325 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
11326 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%  
11327 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvuserfont{##1}}%  
11328 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%  
11329 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%  
11330 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
11331 \renewcommand*{\glxtrfullformat}[2]{%  
11332   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%  
11333   \ifglxtrininsertinside\else##2\fi  
11334   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%  
11335 }%  
11336 \renewcommand*{\glxtrfullplformat}[2]{%  
11337   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%  
11338   \ifglxtrininsertinside\else##2\fi  
11339   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
```



```

11340 }%
11341 \renewcommand*{\Glsxtrfullformat}[2]{%
11342   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11343   \ifglxtrinsertinside\else##2\fi
11344   \glxtruserparen{\glsfirstabbrvuserfont{\Glsaccessshort{##1}}{##1}%
11345 }%
11346 \renewcommand*{\Glsxtrfullplformat}[2]{%
11347   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11348   \ifglxtrinsertinside\else##2\fi
11349   \glxtruserparen{\glsfirstabbrvuserfont{\Glsaccessshortpl{##1}}{##1}%
11350 }%
11351 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

11352 \newabbreviationstyle{long-postshort-user}%
11353 {%

```

Set accessibility attributes if enabled.

```

11354 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

11355 \renewcommand*{\CustomAbbreviationFields}{%
11356   name={\glxtrlongshortname},
11357   sort={\the\glsshorttok},
11358   first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11359   firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

11360   text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11361   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
11362   description={\protect\glsuserdescription{\the\glslongtok}%
11363     {\the\glslabeltok}}}%
11364 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11365   \csdef{glxtrpostlink\glscategorylabel}{%
11366     \glxtrifwasfirstuse
11367     {%
11368       \glxtruserparen
11369         {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
11370         {\glslabel}%
11371     }%
11372   }%
11373 }%
11374 \glshasattribute{\the\glslabeltok}{regular}%
11375 {%
11376   \glissetattribute{\the\glslabeltok}{regular}{false}%
11377 }%
11378 }%
11379 }%
11380 }%
11381 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11382 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11383 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11384 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11385 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11386 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11387 \renewcommand*{\glxtrfullformat}[2]{%
11388   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11389   \ifglxtrinsertinside\else##2\fi
11390 }%
11391 \renewcommand*{\glxtrfullplformat}[2]{%
11392   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11393   \ifglxtrinsertinside\else##2\fi
11394 }%
11395 \renewcommand*{\Glsxtrfullformat}[2]{%
11396   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11397   \ifglxtrinsertinside\else##2\fi
11398 }%
11399 \renewcommand*{\Glsxtrfullplformat}[2]{%
11400   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11401   \ifglxtrinsertinside\else##2\fi
11402 }%

```

In-line format:

```

11403 \renewcommand*{\glxtrinlinefullformat}[2]{%
11404   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11405   \ifglxtrinsertinside\else##2\fi
11406   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
11407 }%
11408 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11409   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11410   \ifglxtrinsertinside\else##2\fi
11411   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
11412 }%
11413 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11414   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
11415   \ifglxtrinsertinside\else##2\fi
11416   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
11417 }%
11418 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11419   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
11420   \ifglxtrinsertinside\else##2\fi
11421   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
11422 }%
11423 }

```

ortuserdesname

```

11424 \newcommand*{\glxtrlongshortuserdesname}{%
11425   \protect\glslonguserfont{\the\glslongtok}%

```

```

11426 \protect\glxtruserparen
11427 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
11428 }

```

**short-user-desc** Like long-postshort-user but the user supplies the description.

```

11429 \newabbreviationstyle{long-postshort-user-desc}%
11430 {%
    Set accessibility attributes if enabled.
11431 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
    Setup the default fields.
11432 \renewcommand*{\CustomAbbreviationFields}{%
11433     name={\glxtrlongshortuserdescname},
11434     sort={\the\glslongtok},
11435     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11436     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
11437     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11438     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
11439 }%
11440 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11441 \csdef{glxtrpostlink\glscategorylabel}{%
11442     \glxtrifwasfirstuse
11443     {%
11444         \glxtruserparen
11445         {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
11446         {\glslabel}%
11447     }%
11448     {}%
11449 }%
11450 \glshasattribute{\the\glslabeltok}{regular}%
11451 {%
11452     \glissetattribute{\the\glslabeltok}{regular}{false}%
11453 }%
11454 {}%
11455 }%
11456 }%
11457 {%
11458 \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
11459 }

```

**t-postlong-user** Like short-long-user but defers the parenthetical matter to after the link.

```

11460 \newabbreviationstyle{short-postlong-user}%
11461 {%
    Set accessibility attributes if enabled.
11462 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
    Setup the default fields.
11463 \renewcommand*{\CustomAbbreviationFields}{%

```

```

11464     name={\glxtrshortlongname},
11465     sort={\the\glsshorttok},
11466     first={\protect\glfirstlonguserfont{\the\glslongtok}},%
11467     firstplural={\protect\glfirstlonguserfont{\the\glslongpltok}},%

11468     text={\protect\glabbrvuserfont{\the\glsshorttok}},%
11469     plural={\protect\glabbrvuserfont{\the\glsshortpltok}},%
11470     description={\protect\gluserdescription{\the\glslongtok}%
11471       {\the\glslabeltok}}}%
11472 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11473   \csdef{glxtrpostlink\glscategorylabel}{%
11474     \glxtrifwasfirstuse
11475     {%
11476       \glxtruserparen
11477       {\glfirstlonguserfont{\glentrylong{\glslabel}}}%
11478       {\glslabel}%
11479     }%
11480   }%
11481 }%
11482 \glshasattribute{\the\glslabeltok}{regular}%
11483 {%
11484   \glsetattribute{\the\glslabeltok}{regular}{false}%
11485 }%
11486 {}%
11487 }%
11488 }%
11489 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11490 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
11491 \renewcommand*{\glabbrvfont}[1]{\glabbrvuserfont{##1}}%
11492 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvuserfont{##1}}%
11493 \renewcommand*{\glfirstlongfont}[1]{\glfirstlonguserfont{##1}}%
11494 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

11495 \renewcommand*{\glxtrfullformat}[2]{%
11496   \glfirstabbrvuserfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
11497   \ifglxtrininsertinside\else##2\fi
11498 }%
11499 \renewcommand*{\glxtrfullplformat}[2]{%
11500   \glfirstabbrvuserfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
11501   \ifglxtrininsertinside\else##2\fi
11502 }%
11503 \renewcommand*{\Glsxtrfullformat}[2]{%
11504   \glfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
11505   \ifglxtrininsertinside\else##2\fi
11506 }%
11507 \renewcommand*{\Glsxtrfullplformat}[2]{%
11508   \glfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

11509 \ifglxtrinsertinside\else##2\fi
11510 }%

```

In-line format:

```

11511 \renewcommand*{\glxtrinlinefullformat}[2]{%
11512 \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11513 \ifglxtrinsertinside\else##2\fi
11514 \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
11515 }%
11516 \renewcommand*{\glxtrinlinefullplformat}[2]{%
11517 \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11518 \ifglxtrinsertinside\else##2\fi
11519 \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
11520 }%
11521 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
11522 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11523 \ifglxtrinsertinside\else##2\fi
11524 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
11525 }%
11526 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
11527 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11528 \ifglxtrinsertinside\else##2\fi
11529 \glsxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
11530 }%
11531 }

```

onguserdesname

```

11532 \newcommand*{\glxtrshortlonguserdesname}{%
11533 \protect\glssabbrvuserfont{\the\glssshorttok}%
11534 \protect\glsxtruserparen
11535 {\protect\glslonguserfont{\the\glslongpltok}}}%
11536 {\the\glslabeltok}%
11537 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

11538 \newabbreviationstyle{short-postlong-user-desc}%
11539 {}%

```

Set accessibility attributes if enabled.

```

11540 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11541 \renewcommand*{\CustomAbbreviationFields}{%
11542 name={\glxtrshortlonguserdesname},
11543 sort={\the\glssshorttok},
11544 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
11545 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
11546 text={\protect\glssabbrvuserfont{\the\glssshorttok}},%
11547 plural={\protect\glssabbrvuserfont{\the\glssshortpltok}}}%
11548 }%

```

```

11549 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11550   \csdef{glxstrpostlink\glscategorylabel}{%
11551     \glxstrifwasfirstuse
11552     {%
11553       \glxtruserparen
11554       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
11555       {\glslabel}%
11556     }%
11557   }%
11558 }%
11559 \glshasattribute{\the\glslabeltok}{regular}%
11560 {%
11561   \glissetattribute{\the\glslabeltok}{regular}{false}%
11562 }%
11563 {}%
11564 }%
11565 }%
11566 {%
11567   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
11568 }

```

short-user-desc

```

11569 \newabbreviationstyle{long-short-user-desc}%
11570 {%
  Set accessibility attributes if enabled.
11571   \glxstrAccSuppAbbrSetTextShortAttrs\glscategorylabel
  Setup the default fields.
11572   \renewcommand*{\CustomAbbreviationFields}{%
11573     name={\glxtrlongshortuserdescname},
11574     sort={\glxtrlongshortdescsort},%
11575     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
11576       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
11577       {\the\glslabeltok}},%
11578     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
11579       \protect\glxtruserparen
11580       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
11581     text={\protect\glsabbrvfont{\the\glsshorttok}},%
11582     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
11583   }%
  Unset the regular attribute if it has been set.
11584   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11585     \glshasattribute{\the\glslabeltok}{regular}%
11586     {%
11587       \glissetattribute{\the\glslabeltok}{regular}{false}%
11588     }%
11589     {}%
11590   }%

```

```

11591 }%
11592 {%
11593   \GlsXtrUseAbbrStyleFmts{long-short-user}%
11594 }

```

short-long-user

```

11595 \newabbreviationstyle{short-long-user}%
11596 {%

```

Set accessibility attributes if enabled.

```
11597   \glstrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

\glslonguserfont is used in the description since \glsdesc doesn't set the style. (Now in \glsuserdescription.)

```

11598   \renewcommand*{\CustomAbbreviationFields}{%
11599     name={\glstrshortlongname},
11600     sort={\the\glsshorttok},
11601     description={\protect\glsuserdescription{\the\glslongtok}%
11602       {\the\glslabeltok}},%
11603     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
11604       \protect\glstruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
11605     {\the\glslabeltok}},%
11606     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
11607       \protect\glstruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
11608     {\the\glslabeltok}},%

11609     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
11610     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

11611   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11612     \glshasattribute{\the\glslabeltok}{regular}%
11613     {%
11614       \glissetattribute{\the\glslabeltok}{regular}{false}%
11615     }%
11616   }%
11617 }%
11618 }%
11619 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11620   \renewcommand*{\abbrvpluralsuffix}{\glstrusersuffix}%
11621   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
11622   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
11623   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
11624   \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
11625   \renewcommand*{\glstrfullformat}[2]{%
```

```

11626 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11627 \ifglxtrinsertinside\else##2\fi
11628 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11629 }%
11630 \renewcommand*{\glxtrfullplformat}[2]{%
11631 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11632 \ifglxtrinsertinside\else##2\fi
11633 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11634 }%
11635 \renewcommand*{\Glsxtrfullformat}[2]{%
11636 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
11637 \ifglxtrinsertinside\else##2\fi
11638 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
11639 }%
11640 \renewcommand*{\Glsxtrfullplformat}[2]{%
11641 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
11642 \ifglxtrinsertinside\else##2\fi
11643 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
11644 }%
11645 }

```

-long-user-desc

```

11646 \newabbreviationstyle{short-long-user-desc}%
11647 {%
    Set accessibility attributes if enabled.
11648 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel
    Setup the default fields.
11649 \renewcommand*{\CustomAbbreviationFields}{%
11650 name={\glxtrshortlonguserdescname},
11651 sort={\glxtrshortlongdescsort},%
11652 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
11653 \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
11654 {\the\glslabeltok}},%
11655 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
11656 \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
11657 {\the\glslabeltok}},%
11658 text={\protect\glsabbrvfont{\the\glsshorttok}},%
11659 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
11660 }%
    Unset the regular attribute if it has been set.
11661 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11662 \glshasattribute{\the\glslabeltok}{regular}%
11663 {%
11664 \glissetattribute{\the\glslabeltok}{regular}{false}%
11665 }%
11666 {}%
11667 }%

```



```

11668 }%
11669 {%
11670   \GlsXtrUseAbbrStyleFmts{short-long-user}%
11671 }

```

### 1.7.7 Predefined Styles (Hyphen)

These styles are designed to work with the `keywords` attribute. They check if the inserted material (provided by the final optional argument of commands like `\gls`) starts with a hyphen. If it does, the insert is added to the parenthetical material. Note that commands like `\glsxtrlongset` `\glsinsert` to empty with the entire link-text stored in `\glscustomtext`.

`trifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glsinsert` so check for that and expand.

```

11672 \newrobustcmd*{\glsxtrifhyphenstart}[3]{%
11673   \ifx\glsinsert#1\relax
11674     \expandafter\@glsxtrifhyphenstart#1\relax\relax
11675     \@end@glsxtrifhyphenstart{#2}{#3}%
11676   \else
11677     \@glsxtrifhyphenstart#1\relax\relax\@end@glsxtrifhyphenstart{#2}{#3}%
11678   \fi
11679 }

```

`trifhyphenstart`

```

11680 \def\@glsxtrifhyphenstart#1#2\@end@glsxtrifhyphenstart#3#4{%
11681   \ifx-#1\relax#3\else #4\fi
11682 }

```

`longhyphenshort`

`\glsxtrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}`

The `<long>` and `<short>` arguments may be the plural form. The `<long>` argument may also be the first letter uppercase form.

```

11683 \newcommand*{\glsxtrlonghyphenshort}[4]{%

```

Grouping is needed to localise the redefinitions.

```

11684 {%

```

If `<insert>` starts with a hyphen, redefine `\glsxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glsxtrwordsep` if `<insert>` doesn't start with a hyphen.

```

11685   \glsxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{}%
11686   \glsfirstlonghyphenfont{#2\ifglsxtrininsertinside{#4}\fi}%
11687   \ifglsxtrininsertinside\else{#4}\fi
11688   \glsxtrfullsep{#1}%
11689   \glsxtrparen{\glsfirstabbrvhyphenfont{#3\ifglsxtrininsertinside{#4}\fi}%

```

```

11690 \ifglxtrinsertinside\else{#4}\fi}%
11691 }%
11692 }

```

abbrvhyphenfont

```

11693 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%

```

abbrvhyphenfont

```

11694 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%

```

slonghyphenfont

```

11695 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%

```

tlonghyphenfont

```

11696 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%

```

The default short form suffix:

xtrhyphensuffix

```

11697 \newcommand*{\glsxtrhyphensuffix}{\glsxtrabbrvpluralsuffix}

```

en-short-hyphen Designed for use with the markwords attribute.

```

11698 \newabbreviationstyle{long-hyphen-short-hyphen}%
11699 {%

```

Set accessibility attributes if enabled.

```

11700 \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

11701 \renewcommand*{\CustomAbbreviationFields}{%
11702   name={\glsxtrlongshortname},
11703   sort={\the\glsshorttok},
11704   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
11705   \protect\glsxtrfullsep{\the\glslabeltok}%
11706   \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
11707   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
11708   \protect\glsxtrfullsep{\the\glslabeltok}%
11709   \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11710   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11711   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
11712   description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

11713 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11714   \glsasattribute{\the\glslabeltok}{regular}%
11715   {%
11716     \glssetattribute{\the\glslabeltok}{regular}{false}%
11717   }}%
11718 {}%
11719 }%

```

```

11720 }%
11721 {%
11722   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
11723   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
11724   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
11725   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11726   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

11727   \renewcommand*{\glxtrfullformat}[2]{%
11728     \glxtrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11729   }%
11730   \renewcommand*{\glxtrfullplformat}[2]{%
11731     \glxtrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
11732     {\glsaccessshortpl{##1}}{##2}%
11733   }%
11734   \renewcommand*{\Glsxtrfullformat}[2]{%
11735     \glxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
11736   }%
11737   \renewcommand*{\Glsxtrfullplformat}[2]{%
11738     \glxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
11739     {\glsaccessshortpl{##1}}{##2}%
11740   }%
11741 }

```

**ort-hyphen-desc** Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

11742 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
11743 {%

```

Set accessibility attributes if enabled.

```

11744   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11745   \renewcommand*{\CustomAbbreviationFields}{%
11746     name={\glxtrlongshortdescname},
11747     sort={\glxtrlongshortdescsort},
11748     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
11749     \protect\glxtrfullsep{\the\glslabeltok}%
11750     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
11751     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
11752     \protect\glxtrfullsep{\the\glslabeltok}%
11753     \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
11754     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11755     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
11756   }%

```

Unset the regular attribute if it has been set.

```

11757   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11758     \glshasattribute{\the\glslabeltok}{regular}%
11759     {%
11760       \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

11761 }%
11762 {}%
11763 }%
11764 }%
11765 {%
11766 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
11767 }

```

nghyphennoshort

```
\glxtrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```
11768 \newcommand*{\glxtrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
11769 {%
```

If *<insert>* starts with a hyphen, redefine `\glxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glxtrwordsep` if *<insert>* doesn't start with a hyphen.

```

11770 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
11771 \glslfirstlonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
11772 \ifglxtrininsertinside\else{#3}\fi
11773 }%
11774 }

```

short-desc-noreg This version doesn't show the short form (except explicitly with `\glxtrshort`). Since `\glxtrshort` doesn't support the hyphen switch, the short form just uses the default short-form font command. This style won't work with the regular as the regular form isn't flexible enough. No accessibility attributes need to be set.

```

11775 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
11776 {%
11777 \renewcommand*{\CustomAbbreviationFields}{%
11778   name={\glxtrlongnoshortdescname},
11779   sort={\expandonce\glxtrorglong},
11780   first={\protect\glslfirstlonghyphenfont{\the\glslongtok}},%
11781   firstplural={\protect\glslfirstlonghyphenfont{\the\glslongpltok}},%
11782   text={\protect\glslonghyphenfont{\the\glslongtok}},%
11783   plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
11784 }%

```

Unset the regular attribute if it has been set.

```

11785 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11786   \glshasattribute{\the\glslabeltok}{regular}%
11787   {%
11788     \glissetattribute{\the\glslabeltok}{regular}{false}%
11789   }%
11790 }%

```

```

11791 }%
11792 }%
11793 {%
11794 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

11795 \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
11796 \renewcommand*\glssabbrvfont[1]{\glssabbrvdefaultfont{##1}}%
11797 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
11798 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
11799 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

11800 \renewcommand*\glsxtrsubsequentfmt}[2]{%
11801   \glsxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
11802 }%
11803 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
11804   \glsxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
11805 }%
11806 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
11807   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11808 }%
11809 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
11810   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11811 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

11812 \renewcommand*\glsxtrinilinefullformat}[2]{%
11813   \glsxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
11814   \glsxtrfullsep{##1}%
11815   \glsxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
11816 }%
11817 \renewcommand*\glsxtrinilinefullplformat}[2]{%
11818   \glsxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
11819   \glsxtrfullsep{##1}%
11820   \glsxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
11821 }%
11822 \renewcommand*\Glsxtrinilinefullformat}[2]{%
11823   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11824   \glsxtrfullsep{##1}%
11825   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
11826 }%
11827 \renewcommand*\Glsxtrinilinefullplformat}[2]{%
11828   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11829   \glsxtrfullsep{##1}%
11830   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
11831 }%

```

The first use full form only displays the long form.

```

11832 \renewcommand*\glsxtrfullformat}[2]{%
11833   \glsxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%

```

```

11834 }%
11835 \renewcommand*{\glxtrfullplformat}[2]{%
11836   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
11837 }%
11838 \renewcommand*{\Glsxtrfullformat}[2]{%
11839   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
11840 }%
11841 \renewcommand*{\Glsxtrfullplformat}[2]{%
11842   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
11843 }%
11844 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

11845 \newabbreviationstyle{long-hyphen-noshort-noreg}%
11846 {%

```

Set accessibility attributes if enabled.

```

11847   \glxtrAccSuppAbbrSetNameShortAttrs\glscategorylabel

```

Setup the default fields.

```

11848   \renewcommand*{\CustomAbbreviationFields}{%
11849     name={\glxtrlongnoshortname},
11850     sort={\the\glsshorttok},
11851     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11852     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11853     text={\protect\glslonghyphenfont{\the\glslongtok}},%
11854     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
11855     description={\the\glslongtok}%
11856   }%

```

Unset the regular attribute if it has been set.

```

11857   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11858     \glshasattribute{\the\glslabeltok}{regular}%
11859     {%
11860       \glissetattribute{\the\glslabeltok}{regular}{false}%
11861     }%
11862   }%
11863 }%
11864 }%
11865 {%

11866   \GlsXtrUseAbbrStyleFmts{long-hyphen-noshort-desc-noreg}%
11867 }

```

lsxtrlonghyphen

```
\glxtrlonghyphen{<long>}{<label>}{<insert>}
```

Used by long-hyphen-postshort-hyphen. The <insert> is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```
11868 \newcommand*{\glxtrlonghyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
11869  {%
11870   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
11871   \glxfirstlonghyphenfont{#1}%
11872  }%
11873 }
```

posthyphenshort

```
\glxtrposthyphenshort{<label>}{<insert>}
```

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like \glxtrlonghyphenshort but omits the <long> part. This always uses the singular short form.

```
11874 \newcommand*{\glxtrposthyphenshort}[2]{%
11875  {%
11876   \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
11877   \ifglxtrininsertinside{\glxfirstlonghyphenfont{#2}}\else{#2}\fi
11878   \glxtrfullsep{#1}%
11879   \glxtrparen
11880   {\glxfirstabbrvhyphenfont{\glxentryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
11881   \ifglxtrininsertinside\else{#2}\fi
11882  }%
11883 }%
11884 }
```

phensubsequent

```
\glxtrposthyphensubsequent{<label>}{<insert>}
```

Format in the post-link hook for subsequent use. The label is ignored by default.

```
11885 \newcommand*{\glxtrposthyphensubsequent}[2]{%
11886   \glxabbrvfont{\ifglxtrininsertinside {#2}\fi}%
11887   \ifglxtrininsertinside \else{#2}\fi
11888 }
```

ostshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
11889 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
11890 {%
```

Set accessibility attributes if enabled.

```
11891 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel
```

Setup the default fields.

```
11892 \renewcommand*{\CustomAbbreviationFields}{%
11893   name={\glxtrlongshortname},
11894   sort={\the\glsshorttok},
11895   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11896   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11897   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11898   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
11899   description={\protect\glslonghyphenfont{\the\glslongtok}}}%
11900 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11901   \csdef{glxtrpostlink\glscategorylabel}{%
11902     \glxtrifwasfirstuse
11903     {%
11904       \glxtrposthyphenshort{\glslabel}{\glsinsert}%
11905     }%
11906     {%
```

Put the insertion into the post-link:

```
11907     \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
11908   }%
11909 }%
11910 \glshasattribute{\the\glslabeltok}{regular}%
11911 {%
11912   \glssetattribute{\the\glslabeltok}{regular}{false}%
11913 }%
11914 {}%
11915 }%
11916 }%
11917 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
11918 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
11919 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
11920 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
11921 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
11922 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

Subsequent use needs to omit the insertion:

```
11923 \renewcommand*{\glxtrsubsequentfmt}[2]{%
11924   \glsabbrvfont{\glsaccessshort{##1}}}%
11925 }%
11926 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
11927   \glsabbrvfont{\glsaccessshortpl{##1}}}%
11928 }%
11929 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
11930   \glsabbrvfont{\Glsaccessshort{##1}}}%
11931 }%
11932 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
11933   \Glsabbrvfont{\Glsaccessshortpl{##1}}}%
11934 }
```



```

11933 \glsabbrvfont{\Glsaccessshortpl{##1}}%
11934 }%

```

First use full form:

```

11935 \renewcommand*\glsxtrfullformat}[2]{%
11936 \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
11937 }%
11938 \renewcommand*\glsxtrfullplformat}[2]{%
11939 \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
11940 }%
11941 \renewcommand*\Glsxtrfullformat}[2]{%
11942 \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
11943 }%
11944 \renewcommand*\Glsxtrfullplformat}[2]{%
11945 \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
11946 }%

```

In-line format.

```

11947 \renewcommand*\glsxtrinlinefullformat}[2]{%
11948 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
11949 \ifglsxtrininsertinside{##2}\fi}%
11950 \ifglsxtrininsertinside \else{##2}\fi
11951 }%
11952 \renewcommand*\glsxtrinlinefullplformat}[2]{%
11953 \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}}%
11954 \ifglsxtrininsertinside{##2}\fi}%
11955 \ifglsxtrininsertinside \else{##2}\fi
11956 }%
11957 \renewcommand*\Glsxtrinlinefullformat}[2]{%
11958 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
11959 \ifglsxtrininsertinside{##2}\fi}%
11960 \ifglsxtrininsertinside \else{##2}\fi
11961 }%
11962 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
11963 \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}}%
11964 \ifglsxtrininsertinside{##2}\fi}%
11965 \ifglsxtrininsertinside \else{##2}\fi
11966 }%
11967 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

11968 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
11969 {%

```

Set accessibility attributes if enabled.

```

11970 \glsxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

11971 \renewcommand*\CustomAbbreviationFields}{%
11972 name={\glsxtrlongshortdescname},
11973 sort={\glsxtrlongshortdescsort},%

```

```

11974   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
11975   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
11976   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
11977   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
11978 }%
11979 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
11980   \csdef{glsxtrpostlink\glscategorylabel}{%
11981     \glsxtrifwasfirstuse
11982     {%
11983       \glsxtrposthyphenshort{\glslabel}{\glsinsert}}%
11984     }%
11985     {%

```

Put the insertion into the post-link:

```

11986       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}}%
11987     }%
11988   }%
11989   \glsattribute{\the\glslabeltok}{regular}%
11990   {%
11991     \glssetAttribute{\the\glslabeltok}{regular}{false}%
11992   }%
11993   {}%
11994 }%
11995 }%
11996 {%
11997   \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
11998 }

```

shorthyphenlong

```
\glsxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
11999 \newcommand*{\glsxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
12000 {%
```

If *<insert>* starts with a hyphen, redefine `\glsxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.)

```

12001   \glsxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{%
12002   \glsfirstabbrvhyphenfont{#2\ifglsxtrininsertinside{#4}\fi}%
12003   \ifglsxtrininsertinside\else{#4}\fi
12004   \glsxtrfullsep{#1}%
12005   \glsxtrparen{\glsfirstlonghyphenfont{#3\ifglsxtrininsertinside{#4}\fi}%
12006   \ifglsxtrininsertinside\else{#4}\fi}%
12007 }%

```

12008 }

hen-long-hyphen Designed for use with the markwords attribute.

12009 \newabbreviationstyle{short-hyphen-long-hyphen}%

12010 {%

Set accessibility attributes if enabled.

12011 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

Setup the default fields.

12012 \renewcommand\*{\CustomAbbreviationFields}{%

12013 name={\glxtrshortlongname},

12014 sort={\the\glsshorttok},

12015 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%

12016 \protect\glxtrfullsep{\the\glslabeltok}}%

12017 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%

12018 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%

12019 \protect\glxtrfullsep{\the\glslabeltok}}%

12020 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%

12021 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}}},%

12022 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%

12023 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

Unset the regular attribute if it has been set.

12024 \renewcommand\*{\GlsXtrPostNewAbbreviation}{%

12025 \glshasattribute{\the\glslabeltok}{regular}}%

12026 {%

12027 \glissetattribute{\the\glslabeltok}{regular}{false}}%

12028 }%

12029 {}%

12030 }%

12031 }%

12032 {%

12033 \renewcommand\*{\abbrvpluralsuffix}{\glxtrhyphensuffix}}%

12034 \renewcommand\*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%

12035 \renewcommand\*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%

12036 \renewcommand\*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%

12037 \renewcommand\*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

The first use full form and the inline full form are the same for this style.

12038 \renewcommand\*{\glxtrfullformat}[2]{%

12039 \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}}%

12040 }%

12041 \renewcommand\*{\glxtrfullplformat}[2]{%

12042 \glxtrshorthyphenlong{##1}}%

12043 {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}}%

12044 }%

12045 \renewcommand\*{\Glsxtrfullformat}[2]{%

12046 \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}}%

12047 }%

12048 \renewcommand\*{\Glsxtrfullplformat}[2]{%

```

12049 \glxtrshorthyphenlong{##1}%
12050 {\glssaccessshortpl{##1}}{\glssaccesslongpl{##1}}{##2}%
12051 }%
12052 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

12053 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
12054 {%

```

Set accessibility attributes if enabled.

```

12055 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12056 \renewcommand*{\CustomAbbreviationFields}{%
12057   name={\glxtrshortlongdescname},
12058   sort={\glxtrshortlongdescsort},
12059   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
12060   \protect\glxtrfullsep{\the\glslabeltok}}%
12061   \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
12062   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
12063   \protect\glxtrfullsep{\the\glslabeltok}}%
12064   \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
12065   text={\protect\glssabbrvhyphenfont{\the\glsshorttok}}},%
12066   plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}}}%
12067 }%

```

Unset the regular attribute if it has been set.

```

12068 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12069   \glshasattribute{\the\glslabeltok}{regular}}%
12070   {%
12071     \glsssetAttribute{\the\glslabeltok}{regular}{false}}%
12072   }%
12073   {}%
12074 }%
12075 }%
12076 {%
12077   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
12078 }

```

sxtrshorthyphen

```
\glxtrshorthyphen{<short>}{<label>}{<insert>}
```

Used by short-hyphen-postlong-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

12079 \newcommand*{\glxtrshorthyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

12080 {%

```

```

12081 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
12082 \glsfirstabbrvhyphenfont{#1}%
12083 }%
12084 }

```

rposthyphenlong

`\glxtrposthyphenlong{<label>}{<insert>}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glxtrshorthyphenlong` but omits the *<short>* part. This always uses the singular long form.

```

12085 \newcommand*\glxtrposthyphenlong[2]{%
12086 {%
12087   \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
12088   \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
12089   \glxtrfullsep{#1}%
12090   \glxtrparen
12091   {\glsfirstlonghyphenfont{\gl Sentrylong{#1}\ifglxtrininsertinside{#2}\fi}%
12092   \ifglxtrininsertinside\else{#2}\fi
12093   }%
12094 }%
12095 }

```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

12096 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
12097 {%

```

Set accessibility attributes if enabled.

```

12098 \glxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

12099 \renewcommand*\CustomAbbreviationFields{%
12100   name={\glxtrshortlongname},
12101   sort={\the\glsshorttok},
12102   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12103   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12104   text={\protect\glssabbrvhyphenfont{\the\glsshorttok}},%
12105   plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}},%
12106   description={\protect\glslonghyphenfont{\the\glslongtok}}}%
12107 \renewcommand*\GlsXtrPostNewAbbreviation{%
12108   \csdef{glxtrpostlink\glscategorylabel}{%
12109     \glxtrifwasfirstuse
12110     {%
12111       \glxtrposthyphenlong{\glslabel}{\glinsert}%
12112     }%
12113   }%

```

Put the insertion into the post-link:

```

12114      \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
12115      }%
12116      }%
12117      \glshasattribute{\the\glslabeltok}{regular}%
12118      {%
12119      \glissetattribute{\the\glslabeltok}{regular}{false}%
12120      }%
12121      {}%
12122      }%
12123 }%
12124 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

12125 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
12126 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
12127 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
12128 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
12129 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

12130 \renewcommand*{\glxtrsubsequentfmt}[2]{%
12131   \glsabbrvfont{\glsaccessshort{##1}}%
12132   }%
12133 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
12134   \glsabbrvfont{\glsaccessshortpl{##1}}%
12135   }%
12136 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
12137   \glsabbrvfont{\Glsaccessshort{##1}}%
12138   }%
12139 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
12140   \glsabbrvfont{\Glsaccessshortpl{##1}}%
12141   }%

```

First use full form:

```

12142 \renewcommand*{\glxtrfullformat}[2]{%
12143   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
12144   }%
12145 \renewcommand*{\glxtrfullplformat}[2]{%
12146   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
12147   }%
12148 \renewcommand*{\Glsxtrfullformat}[2]{%
12149   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
12150   }%
12151 \renewcommand*{\Glsxtrfullplformat}[2]{%
12152   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
12153   }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

12154 \renewcommand*{\glxtrinlinefullformat}[2]{%

```

```

12155 \glsfirstabbrvhyphenfont{\glsaccessshort{##1}%
12156 \ifglxtrinsertinside{##2}\fi}%
12157 \ifglxtrinsertinside \else{##2}\fi
12158 }%
12159 \renewcommand*{\glxtrinlinefullplformat}[2]{%
12160 \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}%
12161 \ifglxtrinsertinside{##2}\fi}%
12162 \ifglxtrinsertinside \else{##2}\fi
12163 }%
12164 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12165 \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}%
12166 \ifglxtrinsertinside{##2}\fi}%
12167 \ifglxtrinsertinside \else{##2}\fi
12168 }%
12169 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12170 \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
12171 \ifglxtrinsertinside{##2}\fi}%
12172 \ifglxtrinsertinside \else{##2}\fi
12173 }%
12174 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

12175 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
12176 {%

```

Set accessibility attributes if enabled.

```

12177 \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12178 \renewcommand*{\CustomAbbreviationFields}{%
12179 name={\glxtrshortlongdescname},
12180 sort={\glxtrshortlongdescsort},%
12181 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
12182 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
12183 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
12184 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
12185 }%
12186 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12187 \csdef{glxtrpostlink\glscategorylabel}{%
12188 \glxtrifwasfirstuse
12189 {%
12190 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
12191 }%
12192 {%

```

Put the insertion into the post-link:

```

12193 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
12194 }%
12195 }%
12196 \glshasattribute{\the\glslabeltok}{regular}%

```

```

12197   {%
12198       \glssetattribute{\the\glslabeltok}{regular}{false}%
12199   }%
12200   {}%
12201   }%
12202 }%
12203 {%
12204   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
12205 }

```

### 1.7.8 Predefined Styles (No Short on First Use)

These styles show only the long form on first use and only the short form on subsequent use.

`\lsabbrvonlyfont`

```

12206 \newcommand*{\lsabbrvonlyfont}{\glsabbrvdefaultfont}%

```

`\stabbrvonlyfont`

```

12207 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%

```

`\glslongonlyfont`

```

12208 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

`\rstlongonlyfont`

```

12209 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

`\lsxtronlysuffix`

```

12210 \newcommand*{\lsxtronlysuffix}{\glsxtrabbrvpluralsuffix}

```

`\glsxtronlyname` The default name format for this style.

```

12211 \newcommand*{\glsxtronlyname}{%
12212   \protect\glsabbrvonlyfont{\the\glsshorttok}%
12213 }

```

`only-short-only`

```

12214 \newabbreviationstyle{long-only-short-only}%
12215 {%

```

Set accessibility attributes if enabled.

```

12216   \glsxtrAccSuppAbbrSetFirstLongAttrs\glscategorylabel

```

Setup the default fields.

```

12217   \renewcommand*{\CustomAbbreviationFields}{%
12218     name={\glsxtronlyname},
12219     sort={\the\glsshorttok},
12220     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
12221     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%

```



```

12222   text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
12223   plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
12224   description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

12225   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12226     \glsattribute{\the\glslabeltok}{regular}%
12227     {%
12228       \glssetattribute{\the\glslabeltok}{regular}{false}%
12229     }%
12230   }%
12231 }%
12232 }%
12233 {%

12234   \renewcommand*{\abbrvpluralsuffix}{\glsxtronlysuffix}%
12235   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
12236   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
12237   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
12238   \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

12239   \renewcommand*{\glsxtrfullformat}[2]{%
12240     \glsfirstlongonlyfont{\glsaccesslong{##1}}\ifglsxtrininsertinside##2\fi}%
12241     \ifglsxtrininsertinside\else##2\fi
12242   }%
12243   \renewcommand*{\glsxtrfullplformat}[2]{%
12244     \glsfirstlongonlyfont{\glsaccesslongpl{##1}}\ifglsxtrininsertinside##2\fi}%
12245     \ifglsxtrininsertinside\else##2\fi
12246   }%
12247   \renewcommand*{\Glsxtrfullformat}[2]{%
12248     \glsfirstlongonlyfont{\Glsaccesslong{##1}}\ifglsxtrininsertinside##2\fi}%
12249     \ifglsxtrininsertinside\else##2\fi
12250   }%
12251   \renewcommand*{\Glsxtrfullplformat}[2]{%
12252     \glsfirstlongonlyfont{\Glsaccesslongpl{##1}}\ifglsxtrininsertinside##2\fi}%
12253     \ifglsxtrininsertinside\else##2\fi
12254   }%

```

The inline full form does show the short form.

```

12255   \renewcommand*{\glsxtrininlinefullformat}[2]{%
12256     \glsfirstlongonlyfont{\glsaccesslong{##1}}\ifglsxtrininsertinside##2\fi}%
12257     \ifglsxtrininsertinside\else##2\fi
12258     \glsxtrfullsep{##1}%
12259     \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
12260   }%
12261   \renewcommand*{\glsxtrininlinefullplformat}[2]{%
12262     \glsfirstlongonlyfont{\glsaccesslongpl{##1}}\ifglsxtrininsertinside##2\fi}%
12263     \ifglsxtrininsertinside\else##2\fi
12264     \glsxtrfullsep{##1}%
12265     \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%

```

```

12266 }%
12267 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
12268   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
12269   \ifglxtrinsertinside\else##2\fi
12270   \glxtrfullsep{##1}%
12271   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
12272 }%
12273 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
12274   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
12275   \ifglxtrinsertinside\else##2\fi
12276   \glxtrfullsep{##1}%
12277   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
12278 }%
12279 }

```

xtronlydescsort

```

12280 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

12281 \newcommand*{\glxtronlydescname}{%
12282   \protect\glslongfont{\the\glslongtok}%
12283 }

```

short-only-desc

```

12284 \newabbreviationstyle{long-only-short-only-desc}%
12285 {%

```

Set accessibility attributes if enabled.

```

12286   \glxtrAccSuppAbbrSetTextShortAttrs\glscategorylabel

```

Setup the default fields.

```

12287   \renewcommand*{\CustomAbbreviationFields}{%
12288     name={\glxtronlydescname},
12289     sort={\glxtronlydescsort},%
12290     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
12291     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
12292     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
12293     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}%
12294   }%

```

Unset the regular attribute if it has been set.

```

12295   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
12296     \glshasattribute{\the\glslabeltok}{regular}%
12297     {%
12298       \glissetattribute{\the\glslabeltok}{regular}{false}%
12299     }%
12300   }%
12301 }%
12302 }%
12303 {%

```

```
12304 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
12305 }
```

## 1.8 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glstryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glssabrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref's \texorpdfstring` which can simply use the expandable command in the PDF string case. The  $\TeX$  string case can now use `\glstrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that `glossaries` automatically loads `textcase`, so the label can be protected from case change with `textcase's \NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
12306 \let\@glstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
12307 \renewcommand*{\markright}[1]{%
12308 \glstrmarkhook
12309 \@glstr@org@markright{\@glstrinmark#1\@glstrnotinmark}%
12310 \glstrrestoremarkhook
12311 }
```

`\markboth` Save original definition:

```
12312 \let\@glstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
12313 \renewcommand*{\markboth}[2]{%
12314 \glstrmarkhook
12315 \@glstr@org@markboth
12316 {\@glstrinmark#1\@glstrnotinmark}%
12317 {\@glstrinmark#2\@glstrnotinmark}%
```

```

12318 \glsxtrrestoremarkhook
12319 }

```

Also do this for \@starttoc

\@starttoc Save original definition:

```

12320 \let\@glsxtr@org@@starttoc\@starttoc

```

Redefine:

```

12321 \renewcommand*{\@starttoc}[1]{%
12322 \glsxtrmarkhook
12323 \@glsxtrinmark
12324 \@glsxtr@org@@starttoc{#1}%
12325 \@glsxtrnotinmark
12326 \glsxtrrestoremarkhook
12327 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

12328 \newcommand*{\glsxtrRevertMarks}{%
12329 \let\markright\@glsxtr@org@markright
12330 \let\markboth\@glsxtr@org@markboth
12331 \let\@starttoc\@glsxtr@org@@starttoc
12332 }

```

rRevertTocMarks Just restores \@starttoc.

```

12333 \newcommand*{\glsxtrRevertTocMarks}{%
12334 \let\@starttoc\@glsxtr@org@@starttoc
12335 }

```

\glsxtrifinmark

```

12336 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

12337 \newrobustcmd*{\@glsxtrinmark}{%
12338 \let\glsxtrifinmark\@firstoftwo
12339 }

```

glsxtrnotinmark

```

12340 \newrobustcmd*{\@glsxtrnotinmark}{%
12341 \let\glsxtrifinmark\@secondoftwo
12342 }

```

eorpdfforheading

```

12343 \ifdef\texorpdfstring
12344 {
12345 \newcommand*{\glsxtrtitleorpdfforheading}[3]{\texorpdfstring{#1}{#2}}
12346 }

```

```

12347 {
12348   \newcommand*{\glxstrtitleorpdforheading}[3]{#1}
12349 }

```

\glxstrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

12350 \newcommand*{\glxstrmarkhook}{%

```

Save current definitions:

```

12351 \let\@glxstr@org@MakeUppercase\MakeUppercase
12352 \let\@glxstr@org@glxstrtitleorpdforheading\glxstrtitleorpdforheading
12353 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
12354 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
12355 \let\@glxstr@org@Glsxstrtitleshort\Glsxstrtitleshort
12356 \let\@glxstr@org@Glsxstrtitleshortpl\Glsxstrtitleshortpl
12357 \let\@glxstr@org@glxstrtitlename\glxstrtitlename
12358 \let\@glxstr@org@Glsxstrtitlename\Glsxstrtitlename
12359 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
12360 \let\@glxstr@org@Glsxstrtitletext\Glsxstrtitletext
12361 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
12362 \let\@glxstr@org@Glsxstrtitleplural\Glsxstrtitleplural
12363 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
12364 \let\@glxstr@org@Glsxstrtitlefirst\Glsxstrtitlefirst
12365 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
12366 \let\@glxstr@org@Glsxstrtitlefirstplural\Glsxstrtitlefirstplural
12367 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
12368 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
12369 \let\@glxstr@org@Glsxstrtitlelong\Glsxstrtitlelong
12370 \let\@glxstr@org@Glsxstrtitlelongpl\Glsxstrtitlelongpl
12371 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
12372 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
12373 \let\@glxstr@org@Glsxstrtitlefull\Glsxstrtitlefull
12374 \let\@glxstr@org@Glsxstrtitlefullpl\Glsxstrtitlefullpl

```

New definitions

```

12375 \let\glxstrifinmark\@firstoftwo
12376 \let\MakeUppercase\MakeTextUppercase
12377 \let\glxstrtitleorpdforheading\@thirdofthree
12378 \let\glxstrtitleshort\glxstrheadshort
12379 \let\glxstrtitleshortpl\glxstrheadshortpl
12380 \let\Glsxstrtitleshort\Glsxstrheadshort
12381 \let\Glsxstrtitleshortpl\Glsxstrheadshortpl
12382 \let\glxstrtitlename\glxstrheadname
12383 \let\Glsxstrtitlename\Glsxstrheadname
12384 \let\glxstrtitletext\glxstrheadtext
12385 \let\Glsxstrtitletext\Glsxstrheadtext
12386 \let\glxstrtitleplural\glxstrheadplural
12387 \let\Glsxstrtitleplural\Glsxstrheadplural
12388 \let\glxstrtitlefirst\glxstrheadfirst
12389 \let\Glsxstrtitlefirst\Glsxstrheadfirst

```

```

12390 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
12391 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
12392 \let\glsxtrtitlelong\glsxtrheadlong
12393 \let\glsxtrtitlelongpl\glsxtrheadlongpl
12394 \let\Glsxtrtitlelong\Glsxtrheadlong
12395 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
12396 \let\glsxtrtitlefull\glsxtrheadfull
12397 \let\glsxtrtitlefullpl\glsxtrheadfullpl
12398 \let\Glsxtrtitlefull\Glsxtrheadfull
12399 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
12400 }

```

`restoremarkhook` Hook used in new definition of `\markboth` and `\markright` to restore the modified definitions. (This is in case the original `\markboth` and `\markright` shouldn't be grouped for some reason. There already is some grouping within those original definitions, but some of the code lies outside that grouping, and possibly there's a reason for it.)

```

12401 \newcommand*{\glsxtrrestoremarkhook}{%
12402   \let\glsxtrinfinmark\@secondoftwo
12403   \let\MakeUppercase\@glsxtr@org@MakeUppercase
12404   \let\glsxtrtitleorpdforheading\@glsxtr@org@glsxtrtitleorpdforheading
12405   \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
12406   \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
12407   \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
12408   \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
12409   \let\glsxtrtitlename\@glsxtr@org@glsxtrtitlename
12410   \let\Glsxtrtitlename\@glsxtr@org@Glsxtrtitlename
12411   \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
12412   \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
12413   \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
12414   \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
12415   \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
12416   \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
12417   \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
12418   \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
12419   \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
12420   \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
12421   \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
12422   \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
12423   \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
12424   \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
12425   \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
12426   \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
12427 }

```

Instead of using one document-wide conditional, use `headuc` attribute to determine whether or not to use the all upper case form.

`glsxtrheadshort` Command used to display short form in the page header.

```

12428 \newcommand*{\glsxtrheadshort}[1]{%

```

```

12429 \protect\NoCaseChange
12430 {%
12431   \glsifattribute{#1}{headuc}{true}%
12432   {%
12433     \GLSxtrshort[noindex,hyper=false]{#1}[]%
12434   }%
12435   {%
12436     \GLSxtrshort[noindex,hyper=false]{#1}[]%
12437   }%
12438 }%
12439 }

```

**lsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents.

```

12440 \newrobustcmd*{\lsxtrtitleshort}[1]{%
12441   \GLSxtrshort[noindex,hyper=false]{#1}[]%
12442 }

```

**sxtrheadshortpl** Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

12443 \newcommand*{\lsxtrheadshortpl}[1]{%
12444   \protect\NoCaseChange
12445   {%
12446     \glsifattribute{#1}{headuc}{true}%
12447     {%
12448       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12449     }%
12450     {%
12451       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12452     }%
12453   }%
12454 }

```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents.

```

12455 \newrobustcmd*{\lsxtrtitleshortpl}[1]{%
12456   \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12457 }

```

**Glsxtrheadshort** Command used to display short form in the page header with the first letter converted to upper case.

```

12458 \newcommand*{\Glsxtrheadshort}[1]{%
12459   \protect\NoCaseChange
12460   {%
12461     \glsifattribute{#1}{headuc}{true}%
12462     {%
12463       \GLSxtrshort[noindex,hyper=false]{#1}[]%
12464     }%
12465     {%

```

```

12466 \Glsxtrshort[noindex,hyper=false]{#1}[]%
12467 }%
12468 }%
12469 }

```

**lsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12470 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
12471 \Glsxtrshort[noindex,hyper=false]{#1}[]%
12472 }

```

**LSxtrtitleshort** Command to display short form of abbreviation in section title and table of contents in all upper case.

```

12473 \newrobustcmd*{\GLSxtrtitleshort}[1]{%
12474 \GLSxtrshort[noindex,hyper=false]{#1}[]%
12475 }

```

**sxtrheadshortpl** Command used to display plural short form in the page header with the first letter converted to upper case.

```

12476 \newcommand*{\Glsxtrheadshortpl}[1]{%
12477 \protect\NoCaseChange
12478 {%
12479 \glsifattribute{#1}{headuc}{true}%
12480 {%
12481 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12482 }%
12483 {%
12484 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12485 }%
12486 }%
12487 }

```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12488 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
12489 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
12490 }

```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents in all upper case.

```

12491 \newrobustcmd*{\GLSxtrtitleshortpl}[1]{%
12492 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
12493 }

```

**\glsxtrheadname** As above but for the name value.

```

12494 \newcommand*{\glsxtrheadname}[1]{%
12495 \protect\NoCaseChange
12496 {%

```



```

12497 \glsifattribute{#1}{headuc}{true}%
12498 {%
12499 \GLSname[noindex,hyper=false]{#1}[]%
12500 }%
12501 {%
12502 \glsname[noindex,hyper=false]{#1}[]%
12503 }%
12504 }%
12505 }

```

`glsxtrtitlename` Command to display name value in section title and table of contents.

```

12506 \newrobustcmd*{\glsxtrtitlename}[1]{%
12507 \glsname[noindex,hyper=false]{#1}[]%
12508 }

```

`\Glsxtrheadname` First letter converted to upper case

```

12509 \newcommand*{\Glsxtrheadname}[1]{%
12510 \protect\NoCaseChange
12511 {%
12512 \glsifattribute{#1}{headuc}{true}%
12513 {%
12514 \GLSname[noindex,hyper=false]{#1}[]%
12515 }%
12516 {%
12517 \Glsname[noindex,hyper=false]{#1}[]%
12518 }%
12519 }%
12520 }

```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```

12521 \newrobustcmd*{\Glsxtrtitlename}[1]{%
12522 \Glsname[noindex,hyper=false]{#1}[]%
12523 }

```

`GLSxtrtitlename` Command to display name value in section title and table of contents in all upper case.

```

12524 \newrobustcmd*{\GLSxtrtitlename}[1]{%
12525 \GLSname[noindex,hyper=false]{#1}[]%
12526 }

```

`\glsxtrheadtext` As above but for the text value.

```

12527 \newcommand*{\glsxtrheadtext}[1]{%
12528 \protect\NoCaseChange
12529 {%
12530 \glsifattribute{#1}{headuc}{true}%
12531 {%
12532 \GLStext[noindex,hyper=false]{#1}[]%
12533 }%

```

```

12534   {%
12535       \glstext[noindex,hyper=false]{#1}[]%
12536   }%
12537 }%
12538 }

```

**glsxtrtitletext** Command to display text value in section title and table of contents.

```

12539 \newrobustcmd*{\glsxtrtitletext}[1]{%
12540   \glstext[noindex,hyper=false]{#1}[]%
12541 }

```

**\Glsxtrheadtext** First letter converted to upper case

```

12542 \newcommand*{\Glsxtrheadtext}[1]{%
12543   \protect\NoCaseChange
12544   {%
12545     \glsifattribute{#1}{headuc}{true}%
12546     {%
12547       \GLStext[noindex,hyper=false]{#1}[]%
12548     }%
12549     {%
12550       \Glstext[noindex,hyper=false]{#1}[]%
12551     }%
12552   }%
12553 }

```

**Glsxtrtitletext** Command to display text value in section title and table of contents with the first letter changed to upper case.

```

12554 \newrobustcmd*{\Glsxtrtitletext}[1]{%
12555   \GLStext[noindex,hyper=false]{#1}[]%
12556 }

```

**GLSxtrtitletext** Command to display text value in section title and table of contents in all upper case.

```

12557 \newrobustcmd*{\GLSxtrtitletext}[1]{%
12558   \GLStext[noindex,hyper=false]{#1}[]%
12559 }

```

**lsxtrheadplural** As above but for the plural value.

```

12560 \newcommand*{\lsxtrheadplural}[1]{%
12561   \protect\NoCaseChange
12562   {%
12563     \glsifattribute{#1}{headuc}{true}%
12564     {%
12565       \GLSplural[noindex,hyper=false]{#1}[]%
12566     }%
12567     {%
12568       \glsplural[noindex,hyper=false]{#1}[]%
12569     }%
12570   }%
12571 }

```

**sxtrtitleplural** Command to display plural value in section title and table of contents.

```
12572 \newrobustcmd*{\glxsxtrtitleplural}[1]{%
12573   \glsplural[noindex,hyper=false]{#1}[]%
12574 }
```

**lsxtrheadplural** Convert first letter to upper case.

```
12575 \newcommand*{\Glsxtrheadplural}[1]{%
12576   \protect\NoCaseChange
12577   {%
12578     \glsifattribute{#1}{headuc}{true}%
12579     {%
12580       \GLSplural[noindex,hyper=false]{#1}[]%
12581     }%
12582     {%
12583       \Glsplural[noindex,hyper=false]{#1}[]%
12584     }%
12585   }%
12586 }
```

**sxtrtitleplural** Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
12587 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
12588   \Glsplural[noindex,hyper=false]{#1}[]%
12589 }
```

**Sxtrtitleplural** Command to display plural value in section title and table of contents in all upper case.

```
12590 \newrobustcmd*{\GLSxtrtitleplural}[1]{%
12591   \GLSplural[noindex,hyper=false]{#1}[]%
12592 }
```

**glxtrheadfirst** As above but for the first value.

```
12593 \newcommand*{\glxtrheadfirst}[1]{%
12594   \protect\NoCaseChange
12595   {%
12596     \glsifattribute{#1}{headuc}{true}%
12597     {%
12598       \GLSfirst[noindex,hyper=false]{#1}[]%
12599     }%
12600     {%
12601       \glsfirst[noindex,hyper=false]{#1}[]%
12602     }%
12603   }%
12604 }
```

**lsxtrtitlefirst** Command to display first value in section title and table of contents.

```
12605 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
12606   \glsfirst[noindex,hyper=false]{#1}[]%
12607 }
```

Glsxtrheadfirst First letter converted to upper case

```
12608 \newcommand*{\Glsxtrheadfirst}[1]{%
12609   \protect\NoCaseChange
12610   {%
12611     \glsifattribute{#1}{headuc}{true}%
12612     {%
12613       \GLSfirst[noindex,hyper=false]{#1}[]%
12614     }%
12615     {%
12616       \Glsfirst[noindex,hyper=false]{#1}[]%
12617     }%
12618   }%
12619 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```
12620 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
12621   \GLSfirst[noindex,hyper=false]{#1}[]%
12622 }
```

LSxtrtitlefirst Command to display first value in section title and table of contents in all upper case.

```
12623 \newrobustcmd*{\GLSxtrtitlefirst}[1]{%
12624   \GLSfirst[noindex,hyper=false]{#1}[]%
12625 }
```

headfirstplural As above but for the firstplural value.

```
12626 \newcommand*{\glsxtrheadfirstplural}[1]{%
12627   \protect\NoCaseChange
12628   {%
12629     \glsifattribute{#1}{headuc}{true}%
12630     {%
12631       \GLSfirstplural[noindex,hyper=false]{#1}[]%
12632     }%
12633     {%
12634       \glsfirstplural[noindex,hyper=false]{#1}[]%
12635     }%
12636   }%
12637 }
```

itlefirstplural Command to display firstplural value in section title and table of contents.

```
12638 \newrobustcmd*{\glsxtrtitlefirstplural}[1]{%
12639   \glsfirstplural[noindex,hyper=false]{#1}[]%
12640 }
```

headfirstplural First letter converted to upper case

```
12641 \newcommand*{\Glsxtrheadfirstplural}[1]{%
12642   \protect\NoCaseChange
12643   {%
```

```

12644 \glsifattribute{#1}{headuc}{true}%
12645 {%
12646 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12647 }%
12648 {%
12649 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12650 }%
12651 }%
12652 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

12653 \newrobustcmd*{\GLSxtrtitlefirstplural}[1]{%
12654 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12655 }

```

`\titlefirstplural` Command to display first value in section title and table of contents in all upper case.

```

12656 \newrobustcmd*{\GLSxtrtitlefirstplural}[1]{%
12657 \GLSfirstplural[noindex,hyper=false]{#1}[]%
12658 }

```

`\glsxtrheadlong` Command used to display long form in the page header.

```

12659 \newcommand*{\glsxtrheadlong}[1]{%
12660 \protect\NoCaseChange
12661 {%
12662 \glsifattribute{#1}{headuc}{true}%
12663 {%
12664 \GLSxtrlong[noindex,hyper=false]{#1}[]%
12665 }%
12666 {%
12667 \glsxtrlong[noindex,hyper=false]{#1}[]%
12668 }%
12669 }%
12670 }

```

`\glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

12671 \newrobustcmd*{\glsxtrtitlelong}[1]{%
12672 \glsxtrlong[noindex,hyper=false]{#1}[]%
12673 }

```

`\lsxtrheadlongpl` Command used to display plural long form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrlongpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

12674 \newcommand*{\glsxtrheadlongpl}[1]{%
12675 \protect\NoCaseChange
12676 {%
12677 \glsifattribute{#1}{headuc}{true}%
12678 {%

```

```

12679      \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12680  }%
12681  {%
12682      \glsxtrlongpl[noindex,hyper=false]{#1}[]%
12683  }%
12684 }%
12685 }

```

**sxtrtitlelongpl** Command to display plural long form of abbreviation in section title and table of contents.

```

12686 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
12687   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
12688 }

```

**\Glsxtrheadlong** Command used to display long form in the page header with the first letter converted to upper case.

```

12689 \newcommand*{\Glsxtrheadlong}[1]{%
12690   \protect\NoCaseChange
12691   {%
12692     \glsifattribute{#1}{headuc}{true}%
12693     {%
12694       \GLSxtrlong[noindex,hyper=false]{#1}[]%
12695     }%
12696     {%
12697       \Glsxtrlong[noindex,hyper=false]{#1}[]%
12698     }%
12699   }%
12700 }

```

**Glsxtrtitlelong** Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12701 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
12702   \Glsxtrlong[noindex,hyper=false]{#1}[]%
12703 }

```

**GLSxtrtitlelong** Command to display long form of abbreviation in section title and table of contents in all upper case.

```

12704 \newrobustcmd*{\GLSxtrtitlelong}[1]{%
12705   \GLSxtrlong[noindex,hyper=false]{#1}[]%
12706 }

```

**lSxtrheadlongpl** Command used to display plural long form in the page header with the first letter converted to upper case.

```

12707 \newcommand*{\Glsxtrheadlongpl}[1]{%
12708   \protect\NoCaseChange
12709   {%
12710     \glsifattribute{#1}{headuc}{true}%
12711     {%
12712       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%

```

```

12713 }%
12714 {%
12715     \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
12716 }%
12717 }%
12718 }

```

**sxtrtitlelongpl** Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12719 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
12720     \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
12721 }

```

**Sxtrtitlelongpl** Command to display plural long form of abbreviation in section title and table of contents in all upper case.

```

12722 \newrobustcmd*{\GLSxtrtitlelongpl}[1]{%
12723     \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
12724 }

```

**\glsxtrheadfull** Command used to display full form in the page header.

```

12725 \newcommand*{\glsxtrheadfull}[1]{%
12726     \protect\NoCaseChange
12727     {%
12728         \gl@ifattribute{#1}{headuc}{true}%
12729         {%
12730             \GLSxtrfull[noindex,hyper=false]{#1}[]%
12731         }%
12732         {%
12733             \glsxtrfull[noindex,hyper=false]{#1}[]%
12734         }%
12735     }%
12736 }

```

**glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents.

```

12737 \newrobustcmd*{\glsxtrtitlefull}[1]{%
12738     \glsxtrfull[noindex,hyper=false]{#1}[]%
12739 }

```

**lxsxtrheadfullpl** Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

12740 \newcommand*{\glsxtrheadfullpl}[1]{%
12741     \protect\NoCaseChange
12742     {%
12743         \gl@ifattribute{#1}{headuc}{true}%
12744         {%
12745             \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12746         }%

```

```

12747   {%
12748   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
12749   }%
12750 }%
12751 }

```

**sxtrtitlefullpl** Command to display plural full form of abbreviation in section title and table of contents.

```

12752 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
12753   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
12754 }

```

**\Glsxtrheadfull** Command used to display full form in the page header with the first letter converted to upper case.

```

12755 \newcommand*{\Glsxtrheadfull}[1]{%
12756   \protect\NoCaseChange
12757   {%
12758     \glsifattribute{#1}{headuc}{true}%
12759     {%
12760       \GLSxtrfull[noindex,hyper=false]{#1}[]%
12761     }%
12762     {%
12763       \Glsxtrfull[noindex,hyper=false]{#1}[]%
12764     }%
12765   }%
12766 }

```

**Glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12767 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
12768   \Glsxtrfull[noindex,hyper=false]{#1}[]%
12769 }

```

**GLSxtrtitlefull** Command to display full form of abbreviation in section title and table of contents in all upper case.

```

12770 \newrobustcmd*{\GLSxtrtitlefull}[1]{%
12771   \GLSxtrfull[noindex,hyper=false]{#1}[]%
12772 }

```

**lsxtrheadfullpl** Command used to display plural full form in the page header with the first letter converted to upper case.

```

12773 \newcommand*{\Glsxtrheadfullpl}[1]{%
12774   \protect\NoCaseChange
12775   {%
12776     \glsifattribute{#1}{headuc}{true}%
12777     {%
12778       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12779     }%
12780     {%

```



```

12781 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
12782 }%
12783 }%
12784 }

```

`\Sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

12785 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
12786 \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
12787 }

```

`\Sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents in all upper case.

```

12788 \newrobustcmd*{\GLSxtrtitlefullpl}[1]{%
12789 \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
12790 }

```

`\glsfmtshort` Provide a way of using the formatted short form in section headings. If `hyperref` has been loaded, use `\texorpdfstring` for convenience in PDF bookmarks.

```

12791 \ifdef\texorpdfstring
12792 {
12793 \newcommand*{\glsfmtshort}[1]{%
12794 \texorpdfstring
12795 {\glsxtrtitleshort{#1}}%
12796 {\glsentryshort{#1}}%
12797 }
12798 }
12799 {
12800 \newcommand*{\glsfmtshort}[1]{%
12801 \glsxtrtitleshort{#1}}
12802 }

```

Similarly for the plural version.

```

\glsfmtshortpl
12803 \ifdef\texorpdfstring
12804 {
12805 \newcommand*{\glsfmtshortpl}[1]{%
12806 \texorpdfstring
12807 {\glsxtrtitleshortpl{#1}}%
12808 {\glsentryshortpl{#1}}%
12809 }
12810 }
12811 {
12812 \newcommand*{\glsfmtshortpl}[1]{%
12813 \glsxtrtitleshortpl{#1}}
12814 }

```

The case-changing version isn't suitable for PDF bookmarks, so the PDF alternative uses the non-case-changing version.

`\Glsfmtshort` Singular form (first letter uppercase).

```
12815 \ifdef\teorpdfstring
12816 {
12817   \newcommand*\Glsfmtshort}[1]{%
12818     \teorpdfstring
12819       {\Glsxtrtitleshort{#1}}%
12820       {\glsentryshort{#1}}%
12821   }
12822 }
12823 {
12824   \newcommand*\Glsfmtshort}[1]{%
12825     \Glsxtrtitleshort{#1}}
12826 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```
12827 \ifdef\teorpdfstring
12828 {
12829   \newcommand*\Glsfmtshortpl}[1]{%
12830     \teorpdfstring
12831       {\Glsxtrtitleshortpl{#1}}%
12832       {\glsentryshortpl{#1}}%
12833   }
12834 }
12835 {
12836   \newcommand*\Glsfmtshortpl}[1]{%
12837     \Glsxtrtitleshortpl{#1}}
12838 }
```

`\glsfmtname` As above but for the name value.

```
12839 \ifdef\teorpdfstring
12840 {
12841   \newcommand*\glsfmtname}[1]{%
12842     \teorpdfstring
12843       {\glsxtrtitlename{#1}}%
12844       {\glsentryname{#1}}%
12845   }
12846 }
12847 {
12848   \newcommand*\glsfmtname}[1]{%
12849     \glsxtrtitlename{#1}}
12850 }
```

`\Glsfmtname` First letter converted to upper case.

```
12851 \ifdef\teorpdfstring
12852 {
12853   \newcommand*\Glsfmtname}[1]{%
12854     \teorpdfstring
12855       {\Glsxtrtitlename{#1}}%
12856       {\glsentryname{#1}}%
```

```

12857 }
12858 }
12859 {
12860   \newcommand*{\Glsfmtname}[1]{%
12861     \Glsxtrtitlename{#1}}
12862 }

```

`\Glsfmtname` All upper case.

```

12863 \ifdef\teorpdfstring
12864 {
12865   \newcommand*{\Glsfmtname}[1]{%
12866     \teorpdfstring
12867       {\Glsxtrtitlename{#1}}%
12868     {\glsentryname{#1}}%
12869   }
12870 }
12871 {
12872   \newcommand*{\Glsfmtname}[1]{%
12873     \Glsxtrtitlename{#1}}
12874 }

```

`\glsfmttext` As above but for the text value.

```

12875 \ifdef\teorpdfstring
12876 {
12877   \newcommand*{\glsfmttext}[1]{%
12878     \teorpdfstring
12879       {\glsxtrtitletext{#1}}%
12880     {\glsentrytext{#1}}%
12881   }
12882 }
12883 {
12884   \newcommand*{\glsfmttext}[1]{%
12885     \glsxtrtitletext{#1}}
12886 }

```

`\Glsfmttext` First letter converted to upper case.

```

12887 \ifdef\teorpdfstring
12888 {
12889   \newcommand*{\Glsfmttext}[1]{%
12890     \teorpdfstring
12891       {\Glsxtrtitletext{#1}}%
12892     {\glsentrytext{#1}}%
12893   }
12894 }
12895 {
12896   \newcommand*{\Glsfmttext}[1]{%
12897     \Glsxtrtitletext{#1}}
12898 }

```

`\GLSfmttext` All upper case.

```
12899 \ifdef\texorpdfstring
12900 {
12901   \newcommand*\GLSfmttext}[1]{%
12902     \texorpdfstring
12903     {\GLSxtrtitletext{#1}}%
12904     {\glentrytext{#1}}%
12905   }
12906 }
12907 {
12908   \newcommand*\GLSfmttext}[1]{%
12909     \GLSxtrtitletext{#1}}
12910 }
```

`\glsfmtplural` As above but for the plural value.

```
12911 \ifdef\texorpdfstring
12912 {
12913   \newcommand*\glsfmtplural}[1]{%
12914     \texorpdfstring
12915     {\glsxtrtitleplural{#1}}%
12916     {\glentryplural{#1}}%
12917   }
12918 }
12919 {
12920   \newcommand*\glsfmtplural}[1]{%
12921     \glsxtrtitleplural{#1}}
12922 }
```

`\Glsfmtplural` First letter converted to upper case.

```
12923 \ifdef\texorpdfstring
12924 {
12925   \newcommand*\Glsfmtplural}[1]{%
12926     \texorpdfstring
12927     {\Glsxtrtitleplural{#1}}%
12928     {\glentryplural{#1}}%
12929   }
12930 }
12931 {
12932   \newcommand*\Glsfmtplural}[1]{%
12933     \Glsxtrtitleplural{#1}}
12934 }
```

`\GLSfmtplural` All upper case.

```
12935 \ifdef\texorpdfstring
12936 {
12937   \newcommand*\GLSfmtplural}[1]{%
12938     \texorpdfstring
12939     {\GLSxtrtitleplural{#1}}%
12940     {\glentryplural{#1}}%
```

```

12941 }
12942 }
12943 {
12944   \newcommand*{\GLSfmtplural}[1]{%
12945     \GLSxtrtitleplural{#1}}
12946 }

```

`\glsfmtfirst` As above but for the first value.

```

12947 \ifdef\teorpdfstring
12948 {
12949   \newcommand*{\glsfmtfirst}[1]{%
12950     \teorpdfstring
12951     {\glsxtrtitlefirst{#1}}%
12952     {\glsentryfirst{#1}}%
12953   }
12954 }
12955 {
12956   \newcommand*{\glsfmtfirst}[1]{%
12957     \glsxtrtitlefirst{#1}}
12958 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

12959 \ifdef\teorpdfstring
12960 {
12961   \newcommand*{\Glsfmtfirst}[1]{%
12962     \teorpdfstring
12963     {\Glsxtrtitlefirst{#1}}%
12964     {\glsentryfirst{#1}}%
12965   }
12966 }
12967 {
12968   \newcommand*{\Glsfmtfirst}[1]{%
12969     \Glsxtrtitlefirst{#1}}
12970 }

```

`\GLSfmtfirst` All upper case.

```

12971 \ifdef\teorpdfstring
12972 {
12973   \newcommand*{\GLSfmtfirst}[1]{%
12974     \teorpdfstring
12975     {\GLSxtrtitlefirst{#1}}%
12976     {\glsentryfirst{#1}}%
12977   }
12978 }
12979 {
12980   \newcommand*{\GLSfmtfirst}[1]{%
12981     \GLSxtrtitlefirst{#1}}
12982 }

```

```
\glsfmtfirstpl As above but for the firstplural value.
12983 \ifdef\teorpdfstring
12984 {
12985   \newcommand*\glsfmtfirstpl}[1]{%
12986     \teorpdfstring
12987     {\glstrtitlefirstplural{#1}}%
12988     {\glstentryfirstplural{#1}}%
12989   }
12990 }
12991 {
12992   \newcommand*\glsfmtfirstpl}[1]{%
12993     \glstrtitlefirstplural{#1}}
12994 }
```

```
\Glsfmtfirstpl First letter converted to upper case.
12995 \ifdef\teorpdfstring
12996 {
12997   \newcommand*\Glsfmtfirstpl}[1]{%
12998     \teorpdfstring
12999     {\Glsxtrtitlefirstplural{#1}}%
13000     {\glstentryfirstplural{#1}}%
13001   }
13002 }
13003 {
13004   \newcommand*\Glsfmtfirstpl}[1]{%
13005     \Glsxtrtitlefirstplural{#1}}
13006 }
```

```
\GLSfmtfirstpl All upper case.
13007 \ifdef\teorpdfstring
13008 {
13009   \newcommand*\GLSfmtfirstpl}[1]{%
13010     \teorpdfstring
13011     {\GLSxtrtitlefirstplural{#1}}%
13012     {\glstentryfirstplural{#1}}%
13013   }
13014 }
13015 {
13016   \newcommand*\GLSfmtfirstpl}[1]{%
13017     \GLSxtrtitlefirstplural{#1}}
13018 }
```

```
\glsfmtlong As above but for the long value.
13019 \ifdef\teorpdfstring
13020 {
13021   \newcommand*\glsfmtlong}[1]{%
13022     \teorpdfstring
13023     {\glstrtitlelong{#1}}%
13024     {\glstentrylong{#1}}%
```

```

13025 }
13026 }
13027 {
13028   \newcommand*{\glsfmtlong}[1]{%
13029     \glsxtrtitlelong{#1}}
13030 }

```

`\Glsfmtlong` First letter converted to upper case.

```

13031 \ifdef\txorpdfstring
13032 {
13033   \newcommand*{\Glsfmtlong}[1]{%
13034     \txorpdfstring
13035       {\Glsxtrtitlelong{#1}}%
13036       {\glsentrylong{#1}}%
13037   }
13038 }
13039 {
13040   \newcommand*{\Glsfmtlong}[1]{%
13041     \Glsxtrtitlelong{#1}}
13042 }

```

`\GLSfmtlong` All upper case.

```

13043 \ifdef\txorpdfstring
13044 {
13045   \newcommand*{\GLSfmtlong}[1]{%
13046     \txorpdfstring
13047       {\GLSxtrtitlelong{#1}}%
13048       {\glsentrylong{#1}}%
13049   }
13050 }
13051 {
13052   \newcommand*{\GLSfmtlong}[1]{%
13053     \GLSxtrtitlelong{#1}}
13054 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

13055 \ifdef\txorpdfstring
13056 {
13057   \newcommand*{\glsfmtlongpl}[1]{%
13058     \txorpdfstring
13059       {\glsxtrtitlelongpl{#1}}%
13060       {\glsentrylongpl{#1}}%
13061   }
13062 }
13063 {
13064   \newcommand*{\glsfmtlongpl}[1]{%
13065     \glsxtrtitlelongpl{#1}}
13066 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```
13067 \ifdef\txorpdfstring
13068 {
13069   \newcommand*\Glsfmtlongpl}[1]{%
13070     \txorpdfstring
13071     {\Glsxtrtitlelongpl{#1}}%
13072     {\glsentrylongpl{#1}}%
13073   }
13074 }
13075 {
13076   \newcommand*\Glsfmtlongpl}[1]{%
13077     \Glsxtrtitlelongpl{#1}}
13078 }
```

`\GLSfmtlongpl` All upper case.

```
13079 \ifdef\txorpdfstring
13080 {
13081   \newcommand*\GLSfmtlongpl}[1]{%
13082     \txorpdfstring
13083     {\GLSxtrtitlelongpl{#1}}%
13084     {\glsentrylongpl{#1}}%
13085   }
13086 }
13087 {
13088   \newcommand*\GLSfmtlongpl}[1]{%
13089     \GLSxtrtitlelongpl{#1}}
13090 }
```

`\glspdffmtfull` Can't use `\glsxtrinlinefullformat` in PDF bookmarks as it's not fully expandable. This command is for the PDF part of `\txorpdfstring` for the full form.

```
13091 \newcommand*\glspdffmtfull}[1]{\glsentrylong{#1} (\glsentryshort{#1})}%
```

`glspdffmtfullpl` Likewise for plural.

```
13092 \newcommand*\glspdffmtfullpl}[1]{\glsentrylongpl{#1} (\glsentryshortpl{#1})}%
```

`\glsfmtfull` In-line full format.

```
13093 \ifdef\txorpdfstring
13094 {
13095   \newcommand*\glsfmtfull}[1]{%
13096     \txorpdfstring
13097     {\glsxtrtitlefull{#1}}%
13098     {\glspdffmtfull{#1}}%
13099   }
13100 }
13101 {
13102   \newcommand*\glsfmtfull}[1]{%
13103     \glsxtrtitlefull{#1}}
13104 }
```



`\Glsfmtfull` First letter converted to upper case.

```
13105 \ifdef\textorpdfstring
13106 {
13107   \newcommand*\Glsfmtfull}[1]{%
13108     \textorpdfstring
13109     {\Glsxtrtitlefull{#1}}%
13110     {\glspdffmtfull{#1}{}}%
13111   }
13112 }
13113 {
13114   \newcommand*\Glsfmtfull}[1]{%
13115     \Glsxtrtitlefull{#1}}
13116 }
```

`\GLSfmtfull` All upper case.

```
13117 \ifdef\textorpdfstring
13118 {
13119   \newcommand*\GLSfmtfull}[1]{%
13120     \textorpdfstring
13121     {\GLSxtrtitlefull{#1}}%
13122     {\glspdffmtfull{#1}}%
13123   }
13124 }
13125 {
13126   \newcommand*\GLSfmtfull}[1]{%
13127     \GLSxtrtitlefull{#1}}
13128 }
```

`\glsfmtfullpl` In-line full plural format.

```
13129 \ifdef\textorpdfstring
13130 {
13131   \newcommand*\glsfmtfullpl}[1]{%
13132     \textorpdfstring
13133     {\glsxtrtitlefullpl{#1}}%
13134     {\glspdffmtfullpl{#1}}%
13135   }
13136 }
13137 {
13138   \newcommand*\glsfmtfullpl}[1]{%
13139     \glsxtrtitlefullpl{#1}}
13140 }
```

`\Glsfmtfullpl` First letter converted to upper case.

```
13141 \ifdef\textorpdfstring
13142 {
13143   \newcommand*\Glsfmtfullpl}[1]{%
13144     \textorpdfstring
13145     {\Glsxtrtitlefullpl{#1}}%
13146     {\glspdffmtfullpl{#1}{}}%
13147   }
13148 }
```

```

13147 }
13148 }
13149 {
13150   \newcommand*{\GLSfmtfullpl}[1]{%
13151     \GLSxtrtitlefullpl{#1}}
13152 }

```

`\GLSfmtfullpl` All upper case.

```

13153 \ifdef\texorpdfstring
13154 {
13155   \newcommand*{\GLSfmtfullpl}[1]{%
13156     \texorpdfstring
13157       {\GLSxtrtitlefullpl{#1}}%
13158       {\glspdfdfmtfullpl{#1}{}}%
13159   }
13160 }
13161 {
13162   \newcommand*{\GLSfmtfullpl}[1]{%
13163     \GLSxtrtitlefullpl{#1}}
13164 }

```

## 1.9 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

13165 \newcommand*{\RequireGlossariesExtraLang}[1]{%
13166   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
13167 }

```

`sariesExtraLang`

```

13168 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
13169   \ProvidesFile{glossariesxtr-#1.ldf}%
13170 }

```

Load any required language modules that are available. This doesn't generate any warning if none are found, since they're not essential. (The only command that really needs defining for the document is `\abbreviationsname`, which can simply be redefined. However, with `bib2gls` it might be useful to provide custom rules for a particular locale.)

`xtr@loaddialect` The dialect label should be stored in `\this@dialect` before using this command.

```

13171 \newcommand{\glxtr@loaddialect}{%
13172   \IfTrackedLanguageFileExists{\this@dialect}%
13173   {glossariesxtr-}% prefix
13174   {.ldf}%
13175   {%

```

```

13176 \RequireGlossariesExtraLang{\CurrentTrackedTag}%
13177 }%
13178 {}% not found

```

If glossaries-extra-bib2gls has been loaded, \@glsxtrdialecthook will check for the associated script, otherwise it will do nothing.

```

13179 \@glsxtrdialecthook
13180 }

```

```

13181 \@ifpackageloaded{tracklang}
13182 {%
13183 \AnyTrackedLanguages
13184 {%
13185 \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
13186 }%
13187 {}%
13188 }
13189 {}

```

Load glossaries-extra-stylemods if required.

```

13190 \@glsxtr@redefstyles
    and set the style:
13191 \@glsxtr@do@style

```

## 1.10 glossaries-extra-bib2gls.sty

This package provides additional support for bib2gls and is automatically loaded by the record option.

```

13192 \NeedsTeXFormat{LaTeX2e}
13193 \ProvidesPackage{glossaries-extra-bib2gls}[2021/11/04 v1.47 (NLCT)]

```

Provide convenient shortcut commands for predefined glossary types.

printunsrtacronyms

```

13194 \ifglsacronym
13195 \providecommand*\printunsrtacronyms[1][]{%
13196 \printunsrtglossary[type=\acronymtype,#1]}%
13197 \fi

```

printunsrtindex

```

13198 \ifglossaryexists{index}
13199 {
13200 \providecommand*\printunsrtindex[1][]{%
13201 \printunsrtglossary[type=index,#1]}%
13202 }{}

```

printunsrtsymbols

```

13203 \ifglossaryexists{symbols}

```

```

13204 {
13205   \providecommand*\printunsrtsymbols}[1] [] {%
13206     \printunsrtglossary[type=symbols,#1]}%
13207 }{}

```

intunsrtnumbers

```

13208 \ifglossaryexists{numbers}
13209 {
13210   \providecommand*\printunsrtnumbers}[1] [] {%
13211     \printunsrtglossary[type=numbers,#1]}%
13212 }{}

```

rtaabbreviations

```

13213 \ifglossaryexists{abbreviations}
13214 {
13215   \providecommand*\printunsrtaabbreviations}[1] [] {%
13216     \printunsrtglossary[type=abbreviations,#1]}%
13217 }{}

```

splaynumberlist Allow \glsdisplaynumberlist and make it robust.

```

13218 \renewcommand*\glsdisplaynumberlist}[1] {%
13219   \glsdoifexists{#1}%
13220   {%
13221     {\let\bibglsglN\glsnumlistsep
13222      \let\bibglsglLastDelimN\glsnumlistlastsep
13223      \glsxtrusefield{#1}{location}}%
13224   }%
13225   }%
13226 }
13227 \robustify\glsdisplaynumberlist

```

entrynumberlist

```

13228 \renewcommand*\glsentrynumberlist}[1]{\glsxtrusefield{#1}{location}}

```

These are some convenient macros for use with custom rules.

\glsheX

```

13229 \newcommand*\glsheX{\string\u}

```

lscapturedgroup

```

13230 \newcommand*\glsapturedgroup{\string\$}

```

nZeroChildCount For use with bib2gls's save-child-count resource option.

```

13231 \newcommand*\GlsXtrIfHasNonZeroChildCount{%
13232   \@ifstar\s@GlsXtrIfHasNonZeroChildCount\@GlsXtrIfHasNonZeroChildCount
13233 }

```

nZeroChildCount

```
13234 \newcommand*{\@GlsXtrIfHasNonZeroChildCount}[3]{%
13235   \@GlsXtrIfFieldNonZero{childcount}{#1}{#2}{#3}%
13236 }
```

nZeroChildCount

```
13237 \newcommand*{\s@GlsXtrIfHasNonZeroChildCount}[3]{%
13238   \s@GlsXtrIfFieldNonZero{childcount}{#1}{#2}{#3}%
13239 }
```

rprovidecommand For use in @preamble, this behaves like \providecommand in the document but like \renewcommand in bib2gls.

```
13240 \newcommand*{\glstrprovidecommand}{\providecommand}
```

glstrenewcommand Like \renewcommand but only generates a warning rather than an error if the command isn't defined.

```
13241 \newcommand*{\glstrenewcommand}{\@star@or@long\glstr@renewcommand}
```

tr@renewcommand

```
13242 \newcommand*{\glstr@renewcommand}[1]{%
13243   \begingroup \escapechar\m@ne\xdef\@gtempa{\string#1}\endgroup
13244   \expandafter\@ifundefined\@gtempa
13245   {%
13246     \GlossariesExtraWarning{can't redefine \noexpand#1(not already defined)}%
13247   }%
13248   \relax
13249   \relax
13250   \let\@ifdefinable\@rc@ifdefinable
13251   \new@command#1%
13252 }
```

lossarylocation For use with indexcounter and bib2gls.

```
13253 \newcommand*{\glstr@wrglossarylocation}[2]{#1}
```

indexCounterLink

`\GlsXtrIndexCounterLink{<text>}{<label>}`

For use with indexcounter and bib2gls.

```
13254 \ifdef\hyperref
13255 {%
13256   \newcommand*{\GlsXtrIndexCounterLink}[2]{%
13257     \glstrifhasfield{indexcounter}{#2}%
13258     {\hyperref[wrglossary.\glscurrentfieldvalue]{#1}}%
13259     {#1}%
13260   }
```

```

13261 }
13262 {
13263   \newcommand*{\GlsXtrIndexCounterLink}[2]{#1}
13264 }

```

GlsXtrDualField

`\GlsXtrDualField`

The internal field used to store the dual label. The dual-field defaults to dual if no value is supplied so that's used as the default.

```

13265 \newcommand*{\GlsXtrDualField}{dual}

```

XtrDualBackLink

`\GlsXtrDualBackLink{<text>}{<label>}`

Adds a hyperlink to the dual entry.

```

13266 \newcommand*{\GlsXtrDualBackLink}[2]{%
13267   \glstrifhasfield{\GlsXtrDualField}{#2}%
13268   {\glshyperlink[#1]{\glscurrentfieldvalue}}%
13269   {#2}%
13270 }

```

TeXEntryAliases Convenient shortcut for use with entry-type-aliases to alias standard BibTeX entry types to @bibtexentry.

```

13271 \newcommand*{\GlsXtrBibTeXEntryAliases}{%
13272   article=bibtexentry,
13273   book=bibtexentry,
13274   booklet=bibtexentry,
13275   conference=bibtexentry,
13276   inbook=bibtexentry,
13277   incollection=bibtexentry,
13278   inproceedings=bibtexentry,
13279   manual=bibtexentry,
13280   mastersthesis=bibtexentry,
13281   misc=bibtexentry,
13282   phdthesis=bibtexentry,
13283   proceedings=bibtexentry,
13284   techreport=bibtexentry,
13285   unpublished=bibtexentry
13286 }

```

ideBibTeXFields Convenient shortcut to define the standard BibTeX fields.

```

13287 \newcommand*{\GlsXtrProvideBibTeXFields}{%
13288   \glsaddstoragekey{address}{\glstrbibaddress}%

```

```

13289 \glsaddstoragekey{author}{\glsxtrbibauthor}%
13290 \glsaddstoragekey{booktitle}{\glsxtrbibbooktitle}%
13291 \glsaddstoragekey{chapter}{\glsxtrbibchapter}%
13292 \glsaddstoragekey{edition}{\glsxtrbibedition}%
13293 \glsaddstoragekey{howpublished}{\glsxtrbibhowpublished}%
13294 \glsaddstoragekey{institution}{\glsxtrbibinstitution}%
13295 \glsaddstoragekey{journal}{\glsxtrbibjournal}%
13296 \glsaddstoragekey{month}{\glsxtrbibmonth}%
13297 \glsaddstoragekey{note}{\glsxtrbibnote}%
13298 \glsaddstoragekey{number}{\glsxtrbibnumber}%
13299 \glsaddstoragekey{organization}{\glsxtrbiborganization}%
13300 \glsaddstoragekey{pages}{\glsxtrbibpages}%
13301 \glsaddstoragekey{publisher}{\glsxtrbibpublisher}%
13302 \glsaddstoragekey{school}{\glsxtrbibschooll}%
13303 \glsaddstoragekey{series}{\glsxtrbibseries}%
13304 \glsaddstoragekey{title}{\glsxtrbibtitle}%
13305 \glsaddstoragekey{bibtextype}{\glsxtrbibtype}%
13306 \glsaddstoragekey{volume}{\glsxtrbibvolume}%
13307 }

```

Multiple supplementary references are only supported with bib2gls.

**ltisuppllocation** This is like `\glsxtrsupphypernumber` but the second argument is the external file name (which isn't obtained from the `externallocation` attribute). The third argument is the formatting (encap) control sequence *name*. This is ignored by default, but is set by `bib2gls` to the original encap in case it's required.

```

13308 \newcommand*\glsxtrmultisuppllocation[3]{%
13309   {%
13310     \def\glsxtrsuppllocationurl{#2}%
13311     \glshypernumber{#1}%
13312   }%
13313 }

```

**rdisplaysupplloc**

`\glsxtrdisplaysupplloc{<prefix>}{<counter>}{<format>}{<src>}{<location>}`

This is like `\glsnoidxdisplayloc` but is used for supplementary locations and so requires an extra argument.

```

13314 \newcommand*\glsxtrdisplaysupplloc[5]{%
13315   \setentrycounter{#1}{#2}%
13316   \glsxtrmultisuppllocation{#5}{#4}{#3}%
13317 }

```

**splaylocnameref** `\glsxtrdisplaylocnameref{<prefix>}{<counter>}{<format>}{<location>}{<name>}{<href>}{<hcounter>}{<external file>}` Used with the `[nameref]` record package option. The `<href>` argument was obtained from `\@currentHref` and the `<hcounter>` argument was obtained from

`\theHentrycounter`, which is more reliable. If `hyperref` hasn't been loaded, this just behaves like `\glsnoidxdisplayloc`.

```
13318 \ifundef\hyperlink
13319 {
13320   \newcommand*\glsxtrdisplaylocnameref}[8]{%
13321     \glsnoidxdisplayloc{#1}{#2}{#3}{#4}%
13322   }
13323 }
13324 {
```

Default action uses `\hcounter`. Equations and pages typically don't have a title, so check the counter name (otherwise the title may section or chapter title, which may be confusing). As from v1.42, this now checks if the control sequence `\glsxtr<counter>locfmt` is defined.

```
13325   \newcommand*\glsxtrdisplaylocnameref}[8]{%
13326     \ifcsdef{glsxtr#2locfmt}%
13327     {\glsxtrnamerefink{#3}{\csuse{glsxtr#2locfmt}{#4}{#5}}{#2.#7}{#8}}%
13328     {%
13329       \ifstrempy{#5}%
13330       {%
```

No title, so just use the location as the link text.

```
13331     \glsxtrnamerefink{#3}{#4}{#2.#7}{#8}%
13332   }%
13333   {%
13334     \ifstrequal{#2}{page}%
13335     {\glsxtrnamerefink{#3}{#4}{#2.#7}{#8}}%
13336     {\glsxtrnamerefink{#3}{#5}{#2.#7}{#8}}%
13337   }%
13338   }%
13339 }
13340 }
```

`requationlocfmt`

`\glsxtrequationlocfmt{<location>}{<title>}`

```
13341 \newcommand*\glsxtrequationlocfmt}[2]{(#1)}
```

`sxtrnamerefink`

`\glsxtrfmtnamerefink{<format>}{<title>}{<href>}{<external file>}`

```
13342 \newcommand*\glsxtrnamerefink}[4]{%
```

Locally change `\glshypernumber` to `\@firstofone` to remove the normal location hyperlink.

```
13343   \begingroup
13344     \let\glshypernumber\@firstofone
```



If the *<external file>* argument is empty, an internal link is used, otherwise an external one is needed.

```

13345 \ifstrempy{#4}%
13346 {\glstrfmtinternalnameref{#3}{#1}{#2}}%
13347 {\glstrfmtexternalnameref{#3}{#1}{#2}{#4}}%
13348 \endgroup
13349 }

```

sxtrnameloclink

`\glsxtrnamerefloclink{<prefix>}{<counter>}{<format>}{<location>}{<text>}{<external file>}`

Like `\@gls@numberlink`, this creates a hyperlink to the target obtained from the prefix, counter and location but uses *<text>* as the hyperlink text. As with regular indexing, this will fail if the target name can't be formed by prefixing the location value.

```

13350 \newcommand{\glsxtrnameloclink}[6]{%
13351 \begingroup
13352 \setentrycounter[#1]{#2}%
13353 \def\glxtr@locationhypertext{#5}%
13354 \let\glshypernumber\@firstofone
13355 \def\@glsnumberformat{#3}%
13356 \def\glxtrsupplocationurl{#6}%
13357 \toks@={}%
13358 \@glxtr@bibgls@removespaces#4 \@nil
13359 \endgroup
13360 }

```

ls@removespaces

```

13361 \def\@glxtr@bibgls@removespaces#1 #2\@nil{%
13362 \toks@=\expandafter{\the\toks@#1}%
13363 \ifx\\#2\\%
13364 \edef\@glo@tmp{\the\toks@}%
13365 \ifx\@glo@tmp\empty
13366 \else
13367 \protected@edef\@glo@tmp{\glstrycounter\@glo@counterprefix\the\toks@}%
13368 \ifdefvoid\glxtrsupplocationurl
13369 {%
13370 \expandafter\glstrfmtinternalnameref\expandafter{\@glo@tmp}%
13371 {\@glsnumberformat}{\glxtr@locationhypertext}%
13372 }%
13373 {%
13374 \expandafter\glstrfmtexternalnameref\expandafter{\@glo@tmp}%
13375 {\@glsnumberformat}{\glxtr@locationhypertext}{\glxtrsupplocationurl}%
13376 }%
13377 \fi

```

```

13378 \else
13379   \@gls@ReturnAfterFi{%
13380     \@glsxtr@bibgls@removespaces#2\@nil
13381   }%
13382 \fi
13383 }

```

internalnameref

```
\glsxtrfmtinternalnameloc{<target>}{<format>}{<title>}
```

```

13384 \newcommand*{\glsxtrfmtinternalnameref}[3]{%
13385   \csuse{#2}{\glsdohyperlink{#1}{#3}}%
13386 }

```

externalnameref

```
\glsxtrfmtexternalnameloc{<target>}{<format>}{<title>}{<file>}
```

```

13387 \newcommand*{\glsxtrfmtexternalnameref}[4]{%
13388   \csuse{#2}{\hyperref{#4}{#1}{#3}}%
13389 }

```

glsxtrSetWidest

```
\glsxtrSetWidest{<type>}{<level>}{<text>}
```

As from **bib2gls** v1.8, this is used by the set-widest resource option for the altree and the styles provided by the glossary-longextra package.

```
13390 \newcommand*{\glsxtrSetWidest}[3]{%
```

Check which style options have been provided. (The style packages may not have been loaded.)

```

13391 \ifdef\glsupdatewidest
13392 {%
13393   \ifdef\glslongextraUpdateWidest
13394   {%

```

Relevant style packages all loaded. If the *<type>* has been given, append to glossary preamble.

```

13395   \ifstrempy{#1}
13396   {%
13397     \glsupdatewidest[#2]{#3}%
13398     \ifnum#2=0\relax
13399     \glslongextraUpdateWidest{#3}%
13400   \else

```

```

13401         \glslongextraUpdateWidestChild{#2}{#3}%
13402     \fi
13403 }%
13404 {%
13405     \apptoglossarypreamble[#1]{\glsupdatewidest[#2]{#3}}%
13406     \ifnum#2=0\relax
13407         \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13408     \else
13409         \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
13410     \fi
13411 }%
13412 }%
13413 {%

```

Only alttree.

```

13414     \ifstrempy{#1}
13415     {%
13416         \glsupdatewidest[#2]{#3}%
13417     }%
13418     {%
13419         \apptoglossarypreamble[#1]{\glsupdatewidest[#2]{#3}}%
13420     }%
13421 }%
13422 }%
13423 {%

```

\glsupdatewidest hasn't been defined. This could just mean that the glossaries-extra-stylemods package hasn't been loaded.

```

13424     \ifdef\glssetwidest
13425     {%
13426         \ifdef\glslongextraUpdateWidest
13427         {%

```

Relevant glossary-tree and glossary-longextra have been loaded. If the *<type>* has been given, append to glossary preamble.

```

13428     \ifstrempy{#1}
13429     {%
13430         \glssetwidest[#2]{#3}%
13431         \ifnum#2=0\relax
13432             \glslongextraUpdateWidest{#3}%
13433         \else
13434             \glslongextraUpdateWidestChild{#2}{#3}%
13435         \fi
13436     }%
13437     {%
13438         \apptoglossarypreamble[#1]{\glssetwidest[#2]{#3}}%
13439         \ifnum#2=0\relax
13440             \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13441         \else
13442             \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%

```

```

13443         \fi
13444     }%
13445 }%
13446 {%
    Only alttree.
13447     \ifstrempy{#1}
13448     {%
13449         \glsetwidest[#2]{#3}%
13450     }%
13451     {%
13452         \apptoglossarypreamble[#1]{\glsetwidest[#2]{#3}}%
13453     }%
13454 }%
13455 }%
13456 {%
13457     \ifdef\glslongextraUpdateWidest
13458     {%
        glossary-longextra has been loaded.
13459         \ifstrempy{#1}
13460         {%
13461             \ifnum#2=0\relax
13462                 \glslongextraUpdateWidest{#3}%
13463             \else
13464                 \glslongextraUpdateWidestChild{#2}{#3}%
13465             \fi
13466         }%
13467         {%
13468             \ifnum#2=0\relax
13469                 \apptoglossarypreamble[#1]{\glslongextraUpdateWidest{#3}}%
13470             \else
13471                 \apptoglossarypreamble[#1]{\glslongextraUpdateWidestChild{#2}{#3}}%
13472             \fi
13473         }%
13474     }%
    Neither glossary-tree nor glossary-longextra have been loaded. Do nothing.
13475 }%
13476 }%
13477 }%
13478 }

```

tWidestFallback

```
\glxtrSetWidestFallback{<max depth>}{<list>}
```

Used when **bib2gls** can't determine the widest name. The *<list>* argument is a comma-separated list of glossary labels. The *<max depth>* refers to the maximum hierarchical depth.

This will either be 0 (only top-level entries) or 2 (up to two child-levels).

```

13479 \newcommand*{\glxtrSetWidestFallback}[2]{%
13480   \ifnum#1=0\relax
13481   \ifdef\glxFindWidestTopLevelName
13482   {%
13483     \glxFindWidestTopLevelName[#2]%
13484   }%
13485   {%
13486     \GlossariesExtraWarning{You need stylemods={tree} to
13487       provide a fallback for set-widest}%
13488   }%
13489   \else
13490   \ifdef\glxFindWidestLevelTwo
13491   {%
13492     \glxFindWidestLevelTwo[#2]%
13493     \ifdef\glxlongextraUpdateWidestChild
13494     {%
13495       \glxlongextraUpdateWidestChild{#1}{\csuse{@glswidestnamei}}%
13496       \glxlongextraUpdateWidestChild{#1}{\csuse{@glswidestnameii}}%
13497     }%
13498     {}%
13499   }%
13500   {%
13501     \GlossariesExtraWarning{You need stylemods={tree} to
13502       provide a fallback for set-widest}%
13503   }%
13504   \fi
13505 }
```

`r@labelprefixes` List of label prefixes.

```

13506 \newcommand*{\@glxtr@labelprefixes}{}
```

`arlabelprefixes` List of label prefixes.

```

13507 \newcommand*{\glxtrclearlabelprefixes}{%
13508   \renewcommand*{\@glxtr@labelprefixes}{}%
13509 }
```

`raddlabelprefix` Add prefix to the list. These should be added in the order of precedence with the last one as a fallback. This doesn't check against duplicates as it may be useful to replicate a prefix at the end as the fallback.

```

13510 \newcommand*{\glxtraddlabelprefix}[1]{%
13511   \ifstrempy{#1}%
13512   {\glxtraddlabelprefix{\empty}}%
13513   {%
13514     \ifdefempty\@glxtr@labelprefixes
13515     {\def\@glxtr@labelprefixes{#1}}%
13516     {\appto\@glxtr@labelprefixes{,#1}}%
13517   }%
13518 }
```

pendlabelprefix Inserts at the start of the list.

```
13519 \newcommand*\glxtrprependlabelprefix}[1]{%
13520   \ifstrempy{#1}%
13521   {\glxtrprependlabelprefix{\empty}}%
13522   {%
13523     \ifdefempty\@glxtr@labelprefixes
13524     {\def\@glxtr@labelprefixes{#1}}%
13525     {\preto\@glxtr@labelprefixes{#1,}}%
13526   }%
13527 }
```

labelprefixlist

`\glxtrifinlabelprefixlist{<prefix>}{<true>}{<false>}`

Test if the given prefix is in the list.

```
13528 \newcommand*\glxtrifinlabelprefixlist}[3]{%
13529   \ifstrempy{#1}%
13530   {\glxtrifinlabelprefixlist{\empty}{#2}{#3}}%
13531   {%
13532     \DTLifinlist{#1}{\@glxtr@labelprefixes}{#2}{#3}%
13533   }%
13534 }
```

prefixlabellist This is provided for the benefit of **bib2gls**. It's possible that the user may add more prefixes after the start of the document, but that can lead to inconsistencies. The final element of the list (the fallback) is the only prefix of interest for **bib2gls**.

```
13535 \AtBeginDocument{%
13536   \protected@write\@auxout{}\string\providecommand{\string\@glxtr@prefixlabellist}[1]{}%
13537   \protected@write\@auxout{}\string\@glxtr@prefixlabellist{\@glxtr@labelprefixes}%
13538 }
```

t@prefixedlabel Iterate through all the prefixes and find the first prefix and label combination that exists. If none found, this could mean that it's the first  $\LaTeX$  run, so the last prefix in the list needs to be the fallback one. Grouping is used in case of a nested for loop.

```
13539 \newcommand*\@glxtr@get@prefixedlabel}[1]{%
13540   \begingroup
```

Initialise to the unprefixed label in the event that the list is empty.

```
13541   \protected@edef\@gls@thislabel{#1}%
13542   \@for\@glxtr@prefix:=\@glxtr@labelprefixes\do
13543   {%
13544     \protected@edef\@gls@thislabel{\@glxtr@prefix#1}%
13545     \ifglsentryexists{\@gls@thislabel}{\@endfortrue}{}%
13546   }%

13547   \edef\@glo@tmp{\endgroup\noexpand\def\noexpand\@gls@thislabel{\@gls@thislabel}}\@glo@tmp
13548 }
```

`\dgl` Like `\gls` but tries the prefixes. (Can't use `\pgls` as that's provided by glossaries-prefix.) Since this command is designed for bib2gls's dual entry system, the "d" stands for "dual".

```
13549 \newrobustcmd*{\dgl}{\@gls@hyp@opt\dgl}
```

`\@dgl`

```
13550 \newcommand*{\@dgl}[2][]{%
13551   \@glsxtr@get@prefixedlabel{#2}%
13552   \new@ifnextchar[{\@gls@{#1}\@gls@thislabel}]{\@gls@{#1}\@gls@thislabel}[]}%
13553 }
```

`\dglsp`

```
13554 \newrobustcmd*{\dglsp}{\@gls@hyp@opt\dglsp}
```

`\@dglsp`

```
13555 \newcommand*{\@dglsp}[2][]{%
13556   \@glsxtr@get@prefixedlabel{#2}%
13557   \new@ifnextchar[{\@glspl@{#1}\@gls@thislabel}]{\@glspl@{#1}\@gls@thislabel}[]}%
13558 }
```

`\dGls`

```
13559 \newrobustcmd*{\dGls}{\@gls@hyp@opt\dGls}
```

`\@dGls`

```
13560 \newcommand*{\@dGls}[2][]{%
13561   \@glsxtr@get@prefixedlabel{#2}%
13562   \new@ifnextchar[{\@Gls@{#1}\@gls@thislabel}]{\@Gls@{#1}\@gls@thislabel}[]}%
13563 }
```

`\dGlspl`

```
13564 \newrobustcmd*{\dGlspl}{\@gls@hyp@opt\dGlspl}
```

`\@dGlspl`

```
13565 \newcommand*{\@dGlspl}[2][]{%
13566   \@glsxtr@get@prefixedlabel{#2}%
13567   \new@ifnextchar[{\@Glspl@{#1}\@gls@thislabel}]{\@Glspl@{#1}\@gls@thislabel}[]}%
13568 }
```

`\dGLS`

```
13569 \newrobustcmd*{\dGLS}{\@gls@hyp@opt\dGLS}
```

`\@dGLS`

```
13570 \newcommand*{\@dGLS}[2][]{%
13571   \@glsxtr@get@prefixedlabel{#2}%
13572   \new@ifnextchar[{\@GLS@{#1}\@gls@thislabel}]{\@GLS@{#1}\@gls@thislabel}[]}%
13573 }
```

`\dGLSpl`

```
13574 \newrobustcmd*{\dGLSpl}{\@gls@hyp@opt\dGLSpl}
```

`\dGLSp1`

```
13575 \newcommand*{\dGLSp1}[2] [] {%
13576   \@glsxtr@get@prefixedlabel{#2}%
13577   \new@ifnextchar[{\@GLSp1@{#1}{\@gls@thislabel}}{\@GLSp1@{#1}{\@gls@thislabel}}[]}%
13578 }
```

`\dglslink` Like `\glslink` but tries the prefixes.

```
13579 \newrobustcmd*{\dglslink}[3] [] {%
13580   \@glsxtr@get@prefixedlabel{#2}%
13581   \glslink[#1]{\@gls@thislabel}{#3}%
13582 }
```

`\dglstdisp` Like `\glsdisp` but tries the prefixes.

```
13583 \newrobustcmd*{\dglstdisp}[3] [] {%
13584   \@glsxtr@get@prefixedlabel{#2}%
13585   \glsdisp[#1]{\@gls@thislabel}{#3}%
13586 }
```

Provide missing Greek letters for use in maths mode. These commands are recognised by `bib2gls` and will be mapped to the Mathematical Greek Italic letters. This ensures that the Greek letters that have the same shape as Latin letters are kept with the other mathematical Greek letters for sorting purposes. The  $\TeX$  version of these commands (provided here) use an upright font for capitals and italic for lower case to provide a better match with the other Greek symbols provided by the kernel.

`\Alpha`

```
13587 \providecommand*{\Alpha}{\mathrm{A}}
```

`\Beta`

```
13588 \providecommand*{\Beta}{\mathrm{B}}
```

`\Epsilon`

```
13589 \providecommand*{\Epsilon}{\mathrm{E}}
```

`\Zeta`

```
13590 \providecommand*{\Zeta}{\mathrm{Z}}
```

`\Eta`

```
13591 \providecommand*{\Eta}{\mathrm{H}}
```

`\Iota`

```
13592 \providecommand*{\Iota}{\mathrm{I}}
```

`\Kappa`

```
13593 \providecommand*{\Kappa}{\mathrm{K}}
```

`\Mu`

```
13594 \providecommand*{\Mu}{\mathrm{M}}
```



```

\Nu
13595 \providecommand*\Nu{\mathrm{N}}

\Omicron
13596 \providecommand*\Omicron{\mathrm{O}}

\Rho
13597 \providecommand*\Rho{\mathrm{P}}

\Tau
13598 \providecommand*\Tau{\mathrm{T}}

\Chi
13599 \providecommand*\Chi{\mathrm{X}}

\Digamma
13600 \providecommand*\Digamma{\mathrm{F}}

\omicron
13601 \providecommand*\omicron{\mathit{o}}

        Provide corresponding upright characters if upgreek has been loaded. (The upper case
        characters are the same as above.)
13602 \@ifpackageloaded{upgreek}%
13603 {

\Upalpha
13604   \providecommand*\Upalpha{\mathrm{A}}

\Upbeta
13605   \providecommand*\Upbeta{\mathrm{B}}

\Upsilon
13606   \providecommand*\Upsilon{\mathrm{E}}

\Upzeta
13607   \providecommand*\Upzeta{\mathrm{Z}}

\Upeta
13608   \providecommand*\Upeta{\mathrm{H}}

\Upiota
13609   \providecommand*\Upiota{\mathrm{I}}

\Upkappa
13610   \providecommand*\Upkappa{\mathrm{K}}

```

```

\Upmu
13611 \providecommand*\Upmu{\mathrm{M}}

\Upnu
13612 \providecommand*\Upnu{\mathrm{N}}

\Upomicron
13613 \providecommand*\Upomicron{\mathrm{O}}

\Uprho
13614 \providecommand*\Uprho{\mathrm{P}}

\Uptau
13615 \providecommand*\Uptau{\mathrm{T}}

\Upchi
13616 \providecommand*\Upchi{\mathrm{X}}

\upomicron
13617 \providecommand*\upomicron{\mathrm{o}}

13618}%
13619{}% upgreek.sty not loaded

```

This package provides some basic rules, but it's not intended for complete coverage of all locales. The CLDR should provide the appropriate locale-sensitive rules. These macros are primarily to help construct custom rules to include, for example, Greek maths symbols mixed with Latin. For the full rule syntax, see the Java API for [RuleBaseCollator](#)

If you want to provide a rule-block for a particular locale to allow for customization within that locale, create a file called `glossariesxtr-⟨tag⟩.ldf` (where `⟨tag⟩` identifies the locale) and add similar commands. See the description of `\IfTrackedLanguageFileExists` in the `tracklang` manual for the allowed forms of `⟨tag⟩`. The simplest is to just use the root language label or ISO code. The file will then be automatically loaded by `glossaries-extra` if the document has support for that language.

When combining these blocks of rules, remember to separate them with the appropriate character. For example:

```

sort-rule={\glxtrcontrolrules
;\glxtrspacerules
;\glxtrnonprintablerules
;\glxtrcombiningdiacriticrules
,\glxtrhyphenrules
<\glxtrgeneralpuncrules
<\glxtrdigitrules
<\glxtrfractionrules
<\glxtrGeneralLatinIVrules
<\glxtrMathItalicGreekIrules
}

```

xtrcontrolrules These are control characters that are usually placed at the start of a rule in the ‘ignored characters’ section. These control characters are unlikely to appear in any entry fields but are provided for completeness. \string is used for punctuation characters in case they’ve been made active.

```
13620 \newcommand*{\glxtrcontrolrules}{%
13621 \string'\glshex 200B\string'\string=\glshex 200C\string=\glshex 200D
13622 \string=\glshex 200E\string=\glshex 200F\string=\glshex 0000\string=\glshex 0001
13623 \string=\glshex 0002\string=\glshex 0003\string=\glshex 0004\string=\glshex 0005
13624 \string=\glshex 0006\string=\glshex 0007\string=\glshex 0008
13625 \string=\string'\glshex 0009\string'\string=\string'\glshex 000B\string'
13626 \string=\glshex 000E\string=\glshex 000F\string=\string'\glshex
13627 0010\string'\string=\glshex 0011
13628 \string=\glshex 0012\string=\glshex 0013\string=\glshex 0014\string=\glshex 0015
13629 \string=\glshex 0016\string=\glshex 0017\string=\glshex 0018\string=\glshex 0019
13630 \string=\glshex 001A\string=\glshex 001B\string=\glshex 001C\string=\glshex 001D
13631 \string=\glshex 001E\string=\glshex 001F\string=\glshex 007F\string=\glshex 0080
13632 \string=\glshex 0081\string=\glshex 0082\string=\glshex 0083\string=\glshex 0084
13633 \string=\glshex 0085\string=\glshex 0086\string=\glshex 0087\string=\glshex 0088
13634 \string=\glshex 0089\string=\glshex 008A\string=\glshex 008B\string=\glshex 008C
13635 \string=\glshex 008D\string=\glshex 008E\string=\glshex 008F\string=\glshex 0090
13636 \string=\glshex 0091\string=\glshex 0092\string=\glshex 0093\string=\glshex 0094
13637 \string=\glshex 0095\string=\glshex 0096\string=\glshex 0097\string=\glshex 0098
13638 \string=\glshex 0099\string=\glshex 009A\string=\glshex 009B\string=\glshex 009C
13639 \string=\glshex 009D\string=\glshex 009E\string=\glshex 009F
13640 }
```

lsxtrspacerules These are space characters.

```
13641 \newcommand*{\glsxtrspacerules}{%
13642 \string' \string'\string;
13643 \string'\glshex 00A0\string'\string;
13644 \string'\glshex 2000\string'\string;
13645 \string'\glshex 2001\string'\string;
13646 \string'\glshex 2002\string'\string;
13647 \string'\glshex 2003\string'\string;
13648 \string'\glshex 2004\string'\string;
13649 \string'\glshex 2005\string'\string;
13650 \string'\glshex 2006\string'\string;
13651 \string'\glshex 2007\string'\string;
13652 \string'\glshex 2008\string'\string;
13653 \string'\glshex 2009\string'\string;
13654 \string'\glshex 200A\string'\string;
13655 \string'\glshex 3000\string'
13656 }
```

nprintablerules These are non-printable characters (BOM, tabs, line feed and carriage return).

```
13657 \newcommand*{\glxtrnonprintablerules}{%
13658 \string'\glshex FEFF\string'\string;
13659 \string'\glshex 000A\string'\string;
13660 \string'\glshex 0009\string'\string;
```

```

13661 \string'\glshex 000C\string'\string;
13662 \string'\glshex 000B\string'
13663 }

```

gdiacriticrules Combining diacritic marks. This is split into multiple macros.

```

13664 \newcommand*{\glxtrcombiningdiacriticrules}{%
13665 \glxtrcombiningdiacriticIrules\string;
13666 \glxtrcombiningdiacriticIIrules\string;
13667 \glxtrcombiningdiacriticIIIrules\string;
13668 \glxtrcombiningdiacriticIVrules
13669 }

```

diacriticIrules First set of combining diacritic marks.

```

13670 \newcommand*{\glxtrcombiningdiacriticIrules}{%
13671 \glshex 0301\string;% combining acute
13672 \glshex 0300\string;% combining grave
13673 \glshex 0306\string;% combining breve
13674 \glshex 0302\string;% combining circumflex
13675 \glshex 030C\string;% combining caron
13676 \glshex 030A\string;% combining ring
13677 \glshex 030D\string;% combining vertical line above
13678 \glshex 0308\string;% combining diaeresis
13679 \glshex 030B\string;% combining double acute
13680 \glshex 0303\string;% combining tilde
13681 \glshex 0307\string;% combining dot above
13682 \glshex 0304% combining macron
13683 }

```

diacriticIIrules Second set of combining diacritic marks.

```

13684 \newcommand*{\glxtrcombiningdiacriticIIrules}{%
13685 \glshex 0337\string;% combining short solidus overlay
13686 \glshex 0327\string;% combining cedilla
13687 \glshex 0328\string;% combining ogonek
13688 \glshex 0323\string;% combining dot below
13689 \glshex 0332\string;% combining low line
13690 \glshex 0305\string;% combining overline
13691 \glshex 0309\string;% combining hook above
13692 \glshex 030E\string;% combining double vertical line above
13693 \glshex 030F\string;% combining double grave accent
13694 \glshex 0310\string;% combining candrabindu
13695 \glshex 0311\string;% combining inverted breve
13696 \glshex 0312\string;% combining turned comma above
13697 \glshex 0313\string;% combining comma above
13698 \glshex 0314\string;% combining reversed comma above
13699 \glshex 0315\string;% combining comma above right
13700 \glshex 0316\string;% combining grave accent below
13701 \glshex 0317% combining acute accent below
13702 }

```

acriticIIIrules Third set of combining diacritic marks.

```
13703 \newcommand*{\glxtrcombiningdiacriticIIIrules}{%
13704 \glshex 0318\string;% combining left tack below
13705 \glshex 0319\string;% combining right tack below
13706 \glshex 031A\string;% combining left angle above
13707 \glshex 031B\string;% combining horn
13708 \glshex 031C\string;% combining left half ring below
13709 \glshex 031D\string;% combining up tack below
13710 \glshex 031E\string;% combining down tack below
13711 \glshex 031F\string;% combining plus sign below
13712 \glshex 0320\string;% combining minus sign below
13713 \glshex 0321\string;% combining palatalized hook below
13714 \glshex 0322\string;% combining retroflex hook below
13715 \glshex 0324\string;% combining diaeresis below
13716 \glshex 0325\string;% combining ring below
13717 \glshex 0326\string;% combining comma below
13718 \glshex 0329\string;% combining vertical line below
13719 \glshex 032A\string;% combining bridge below
13720 \glshex 032B\string;% combining inverted double arch below
13721 \glshex 032C\string;% combining caron below
13722 \glshex 032D\string;% combining circumflex accent below
13723 \glshex 032E\string;% combining breve below
13724 \glshex 032F\string;% combining inverted breve below
13725 \glshex 0330\string;% combining tilde below
13726 \glshex 0331\string;% combining macron below
13727 \glshex 0333\string;% combining double low line
13728 \glshex 0334\string;% combining tilde overlay
13729 \glshex 0335\string;% combining short stroke overlay
13730 \glshex 0336\string;% combining long stroke overlay
13731 \glshex 0338\string;% combining long solidus overlay
13732 \glshex 0339\string;% combining combining right half ring below
13733 \glshex 033A\string;% combining inverted bridge below
13734 \glshex 033B\string;% combining square below
13735 \glshex 033C\string;% combining seagull below
13736 \glshex 033D\string;% combining x above
13737 \glshex 033E\string;% combining vertical tilde
13738 \glshex 033F\string;% combining double overline
13739 \glshex 0342\string;% combining Greek perispomeni
13740 \glshex 0344\string;% combining Greek dialytika tonos
13741 \glshex 0345\string;% combining Greek ypogegrammeni
13742 \glshex 0360\string;% combining double tilde
13743 \glshex 0361\string;% combining double inverted breve
13744 \glshex 0483\string;% combining Cyrillic titlo
13745 \glshex 0484\string;% combining Cyrillic palatalization
13746 \glshex 0485\string;% combining Cyrillic dasia pneumata
13747 \glshex 0486% combining Cyrillic psili pneumata
13748 }
```

iacriticIVrules Fourth set of combining diacritic marks.

```

13749 \newcommand*{\glxtrcombiningdiacriticIVrules}{%
13750 \glshex 20D0\string;% combining left harpoon above
13751 \glshex 20D1\string;% combining right harpoon above
13752 \glshex 20D2\string;% combining long vertical line overlay
13753 \glshex 20D3\string;% combining short vertical line overlay
13754 \glshex 20D4\string;% combining anticlockwise arrow above
13755 \glshex 20D5\string;% combining clockwise arrow above
13756 \glshex 20D6\string;% combining left arrow above
13757 \glshex 20D7\string;% combining right arrow above
13758 \glshex 20D8\string;% combining ring overlay
13759 \glshex 20D9\string;% combining clockwise ring overlay
13760 \glshex 20DA\string;% combining anticlockwise ring overlay
13761 \glshex 20DB\string;% combining three dots above
13762 \glshex 20DC\string;% combining four dots above
13763 \glshex 20DD\string;% combining enclosing circle
13764 \glshex 20DE\string;% combining enclosing square
13765 \glshex 20DF\string;% combining enclosing diamond
13766 \glshex 20E0\string;% combining enclosing circle backslash
13767 \glshex 20E1% combining left right arrow above
13768 }

```

#### sxtrhyphenrules Hyphens.

```

13769 \newcommand*{\glxtrhyphenrules}{%
13770 \string'\string-\string'\string;% ASCII hyphen
13771 \glshex 00AD\string;% soft hyphen
13772 \glshex 2010\string;% hyphen
13773 \glshex 2011\string;% non-breaking hyphen
13774 \glshex 2012\string;% figure dash
13775 \glshex 2013\string;% en dash
13776 \glshex 2014\string;% em dash
13777 \glshex 2015\string;% horizontal bar
13778 \glshex 2212\string=\glshex 207B\string=\glshex 208B% minus sign
13779 }

```

#### eneralpuncrules General punctuation.

```

13780 \newcommand*{\glxtrgeneralpuncrules}{%
13781 \glxtrgeneralpuncIrules
13782 \string<\glxtrcurrencyrules
13783 \string<\glxtrgeneralpuncIIrules
13784 }

```

#### eneralpuncIrules First set of general punctuation.

```

13785 \newcommand*{\glxtrgeneralpuncIrules}{%
13786 \string'\glshex 005F\string'% underscore
13787 \string<\glshex 00AF% macron
13788 \string<\string'\glshex 002C\string'% comma
13789 \string<\string'\glshex 003B\string'% semi-colon
13790 \string<\string'\glshex 003A\string'% colon
13791 \string<\string'\glshex 0021\string'% exclamation mark

```

13792 \string<\glshex 00A1% inverted exclamation mark  
 13793 \string<\string'\glshex 003F\string'% question mark  
 13794 \string<\glshex 00BF% inverted question mark  
 13795 \string<\string'\glshex 002F\string'% solidus  
 13796 \string<\string'\glshex 002E\string'% full stop  
 13797 \string<\glshex 00B4% acute accent  
 13798 \string<\string'\glshex 0060\string'% grave accent  
 13799 \string<\string'\glshex 005E\string'% circumflex accent  
 13800 \string<\glshex 00A8% diaeresis  
 13801 \string<\string'\glshex 007E\string'% tilde  
 13802 \string<\glshex 00B7% middle dot  
 13803 \string<\glshex 00B8% cedilla  
 13804 \string<\string'\glshex 0027\string'% straight apostrophe  
 13805 \string<\string'\glshex 0022\string'% straight double quote  
 13806 \string<\glshex 00AB% left guillemet  
 13807 \string<\glshex 00BB% right guillemet  
 13808 \string<\string'\glshex 0028\string'% left parenthesis  
 13809 \string=>\glshex 207D\string=>\glshex 208D% super/subscript left parenthesis  
 13810 \string<\string'\glshex 0029\string'% right parenthesis  
 13811 \string=>\glshex 207E\string=>\glshex 208E% super/subscript right parenthesis  
 13812 \string<\string'\glshex 005B\string'% left square bracket  
 13813 \string<\string'\glshex 005D\string'% right square bracket  
 13814 \string<\string'\glshex 007B\string'% left curly bracket  
 13815 \string<\string'\glshex 007D\string'% right curly bracket  
 13816 \string<\glshex 00A7% section sign  
 13817 \string<\glshex 00B6% pilcrow sign  
 13818 \string<\glshex 00A9% copyright sign  
 13819 \string<\glshex 00AE% registered sign  
 13820 \string<\string'\glshex 0040\string'% at sign  
 13821 }

trcurrencyrules General punctuation.

13822 \newcommand\*{\glxtrcurrencyrules}{%  
 13823 \glshex 00A4% currency sign  
 13824 \string<\glshex 0E3F% Thai currency symbol baht  
 13825 \string<\glshex 00A2% cent sign  
 13826 \string<\glshex 20A1% colon sign  
 13827 \string<\glshex 20A2% cruzeiro sign  
 13828 \string<\string'\glshex 0024\string'% dollar sign  
 13829 \string<\glshex 20AB% dong sign  
 13830 \string<\glshex 20AC% euro sign  
 13831 \string<\glshex 20A3% French franc sign  
 13832 \string<\glshex 20A4% lira sign  
 13833 \string<\glshex 20A5% mill sign  
 13834 \string<\glshex 20A6% naira sign  
 13835 \string<\glshex 20A7% peseta sign  
 13836 \string<\glshex 00A3% pound sign  
 13837 \string<\glshex 20A8% rupee sign  
 13838 \string<\glshex 20AA% new sheqel sign

```

13839 \string<\glshex 20A9% won sign
13840 \string<\glshex 00A5% yen sign
13841 }

```

eralpuncIIrules Second set of general punctuation.

```

13842 \newcommand*{\glxtrgeneralpuncIIrules}{%
13843 \string'\glshex 002A\string'% asterisk
13844 \string<\string'\glshex 005C\string'% backslash
13845 \string<\string'\glshex 0026\string'% ampersand
13846 \string<\string'\glshex 0023\string'% hash sign
13847 \string<\string'\glshex 0025\string'% percent sign
13848 \string<\string'\glshex 002B\string'% plus sign
13849 \string=\glshex 207A\string=\glshex 208A% super/subscript plus sign
13850 \string<\glshex 00B1% plus-minus sign
13851 \string<\glshex 00F7% division sign
13852 \string<\glshex 00D7% multiplication sign
13853 \string<\string'\glshex 003C\string'% less-than sign
13854 \string<\string'\glshex 003D\string'% equals sign
13855 \string<\string'\glshex 003E\string'% greater-than sign
13856 \string<\glshex 00AC% not sign
13857 \string<\string'\glshex 007C\string'% vertical bar (pipe)
13858 \string<\glshex 00A6% broken bar
13859 \string<\glshex 00B0% degree sign
13860 \string<\glshex 00B5% micron sign
13861 }

```

eralLatinIrules Basic Latin alphabet.

```

13862 \newcommand*{\glxtrGeneralLatinIrules}{%
13863 \glxtrLatinA
13864 \string<b,B%
13865 \string<c,C%
13866 \string<d,D%
13867 \string<\glxtrLatinE
13868 \string<f,F%
13869 \string<g,G%
13870 \string<\glxtrLatinH
13871 \string<\glxtrLatinI
13872 \string<j,J%
13873 \string<\glxtrLatinK
13874 \string<\glxtrLatinL
13875 \string<\glxtrLatinM
13876 \string<\glxtrLatinN
13877 \string<\glxtrLatinO
13878 \string<\glxtrLatinP
13879 \string<q,Q%
13880 \string<r,R%
13881 \string<\glxtrLatinS
13882 \string<\glxtrLatinT
13883 \string<u,U%

```



```

13884 \string<v,V%
13885 \string<w,W%
13886 \string<\glxtrLatinX
13887 \string<y,Y%
13888 \string<z,Z
13889 }

```

**alLatinIIrules** General Latin alphabet (eth between D and E, ß treated as SS).

```

13890 \newcommand*{\glxtrGeneralLatinIIrules}{%
13891 \glxtrLatinA
13892 \string<b,B%
13893 \string<c,C%
13894 \string<d,D%
13895 \string<\glxtrLatinEth
13896 \string<\glxtrLatinE
13897 \string<f,F%
13898 \string<g,G%
13899 \string<\glxtrLatinH
13900 \string<\glxtrLatinI
13901 \string<j,J%
13902 \string<\glxtrLatinK
13903 \string<\glxtrLatinL
13904 \string<\glxtrLatinM
13905 \string<\glxtrLatinN
13906 \string<\glxtrLatinO
13907 \string<\glxtrLatinP
13908 \string<q,Q%
13909 \string<r,R%
13910 \string<\glxtrLatinS
13911 \string& SS \string, \glxtrLatinEszettSs
13912 \string<\glxtrLatinT
13913 \string<u,U%
13914 \string<v,V%
13915 \string<w,W%
13916 \string<\glxtrLatinX
13917 \string<y,Y%
13918 \string<z,Z%
13919 }

```

**alLatinIIIrules** General Latin alphabet (eth between D and E, ß treated as SZ).

```

13920 \newcommand*{\glxtrGeneralLatinIIIrules}{%
13921 \glxtrLatinA
13922 \string<b,B%
13923 \string<c,C%
13924 \string<d,D%
13925 \string<\glxtrLatinEth
13926 \string<\glxtrLatinE
13927 \string<f,F%
13928 \string<g,G%

```

```

13929 \string<\glxtrLatinH
13930 \string<\glxtrLatinI
13931 \string<j,J%
13932 \string<\glxtrLatinK
13933 \string<\glxtrLatinL
13934 \string<\glxtrLatinM
13935 \string<\glxtrLatinN
13936 \string<\glxtrLatinO
13937 \string<\glxtrLatinP
13938 \string<q,Q%
13939 \string<r,R%
13940 \string<\glxtrLatinS
13941 \string& SZ, \glxtrLatinEszettSz
13942 \string<\glxtrLatinT
13943 \string<u,U%
13944 \string<v,V%
13945 \string<w,W%
13946 \string<\glxtrLatinX
13947 \string<y,Y%
13948 \string<z,Z%
13949 }

```

`\glxtrGeneralLatinIVrules` General Latin alphabet (Æ treated as AE and Ætreated as OE, Þtreated as TH, ß treated as SS, eth between D and E).

```

13950 \newcommand*{\glxtrGeneralLatinIVrules}{%
13951 \glxtrLatinA
13952 \string& AE , \glxtrLatinAELigature
13953 \string<b,B%
13954 \string<c,C%
13955 \string<d,D%
13956 \string<\glxtrLatinEth
13957 \string<\glxtrLatinE
13958 \string<f,F%
13959 \string<g,G%
13960 \string<\glxtrLatinH
13961 \string<\glxtrLatinI
13962 \string<j,J%
13963 \string<\glxtrLatinK
13964 \string<\glxtrLatinL
13965 \string<\glxtrLatinM
13966 \string<\glxtrLatinN
13967 \string<\glxtrLatinO
13968 \string& OE , \glxtrLatinOELigature
13969 \string<\glxtrLatinP
13970 \string<q,Q%
13971 \string<r,R%
13972 \string<\glxtrLatinS
13973 \string& SS , \glxtrLatinEszettSs
13974 \string<\glxtrLatinT

```

```

13975 \string& th =\glshex 00DE
13976 \string& TH =\glshex 00FE
13977 \string<u,U%
13978 \string<v,V%
13979 \string<w,W%
13980 \string<\glxtrLatinX
13981 \string<y,Y%
13982 \string<z,Z%
13983 }

```

**eralLatinVrules** General Latin alphabet (eth between D and E, ß treated as SS, Þ treated as TH).

```

13984 \newcommand*{\glxtrGeneralLatinVrules}{%
13985 \glxtrLatinA
13986 \string<b,B%
13987 \string<c,C%
13988 \string<d,D%
13989 \string<\glxtrLatinEth
13990 \string<\glxtrLatinE
13991 \string<f,F%
13992 \string<g,G%
13993 \string<\glxtrLatinH
13994 \string<\glxtrLatinI
13995 \string<j,J%
13996 \string<\glxtrLatinK
13997 \string<\glxtrLatinL
13998 \string<\glxtrLatinM
13999 \string<\glxtrLatinN
14000 \string<\glxtrLatinO
14001 \string<\glxtrLatinP
14002 \string<q,Q%
14003 \string<r,R%
14004 \string<\glxtrLatinS
14005 \string& SS , \glxtrLatinEszettSs
14006 \string<\glxtrLatinT
14007 \string& th =\glshex 00DE
14008 \string& TH =\glshex 00FE
14009 \string<u,U%
14010 \string<v,V%
14011 \string<w,W%
14012 \string<\glxtrLatinX
14013 \string<y,Y%
14014 \string<z,Z%
14015 }

```

**ralLatinVirules** General Latin alphabet (eth between D and E, ß treated as SZ, Þ treated as TH).

```

14016 \newcommand*{\glxtrGeneralLatinVirules}{%
14017 \glxtrLatinA
14018 \string<b,B%
14019 \string<c,C%

```

```

14020 \string<d,D%
14021 \string<\glxtrLatinEth
14022 \string<\glxtrLatinE
14023 \string<f,F%
14024 \string<g,G%
14025 \string<\glxtrLatinH
14026 \string<\glxtrLatinI
14027 \string<j,J%
14028 \string<\glxtrLatinK
14029 \string<\glxtrLatinL
14030 \string<\glxtrLatinM
14031 \string<\glxtrLatinN
14032 \string<\glxtrLatinO
14033 \string<\glxtrLatinP
14034 \string<q,Q%
14035 \string<r,R%
14036 \string<\glxtrLatinS
14037 \string& SZ , \glxtrLatinEszettSz
14038 \string<\glxtrLatinT
14039 \string& th =\glshex 00DE
14040 \string& TH =\glshex 00FE
14041 \string<u,U%
14042 \string<v,V%
14043 \string<w,W%
14044 \string<\glxtrLatinX
14045 \string<y,Y%
14046 \string<z,Z%
14047 }

```

allLatinVIIrules General Latin alphabet (Æ between A and B, eth between D and E, insular G as G, Œ between O and P, long S equivalent to S, Þ between T and U and wynn as W).

```

14048 \newcommand*{\glxtrGeneralLatinVIIrules}{%
14049 \glxtrLatinA
14050 \string<\glxtrLatinAELigature
14051 \string<b,B%
14052 \string<c,C%
14053 \string<d,D%
14054 \string<\glxtrLatinEth
14055 \string<\glxtrLatinE
14056 \string<f,F%
14057 \string<\glxtrLatinInsularG
14058 \string<\glxtrLatinH
14059 \string<\glxtrLatinI
14060 \string<j,J%
14061 \string<\glxtrLatinK
14062 \string<\glxtrLatinL
14063 \string<\glxtrLatinM
14064 \string<\glxtrLatinN
14065 \string<\glxtrLatinO

```

```

14066 \string<\glxtrLatinOELigature
14067 \string<\glxtrLatinP
14068 \string<q,Q%
14069 \string<r,R%
14070 \string<\glshex 017F=\glxtrLatinS % s and long s
14071 \string<\glxtrLatinT
14072 \string<\glxtrLatinThorn
14073 \string<u,U%
14074 \string<v,V%
14075 \string< w\string=\glshex 01BF, W\string=\glshex 01F7
14076 \string<\glxtrLatinX
14077 \string<y,Y%
14078 \string<z,Z%
14079 }

```

1LatinVIIIrules General Latin alphabet (Æ treated as AE and Ætreated as OE, Þtreated as TH, ß treated as SS, eth treated as D, Ø treated as O, Ł treated as L).

```

14080 \newcommand*{\glxtrGeneralLatinVIIIrules}{%
14081 \glxtrLatinA
14082 \string& AE , \glxtrLatinAELigature
14083 \string<b,B%
14084 \string<c,C%
14085 \string<\glshex 00F0\string;d,\glshex 00D0\string;D% D and eth
14086 \string<\glxtrLatinE
14087 \string<f,F%
14088 \string<g,G%
14089 \string<\glxtrLatinH
14090 \string<\glxtrLatinI
14091 \string<j,J%
14092 \string<\glxtrLatinK
14093 \string<\glshex 0142\string=\glxtrLatinL\string=\glshex 0141% L and \L
14094 \string<\glxtrLatinM
14095 \string<\glxtrLatinN
14096 \string<\glshex 00F8\string=\glxtrLatinO\string=\glshex 00D8% O and \O
14097 \string& OE , \glxtrLatinOELigature
14098 \string<\glxtrLatinP
14099 \string<q,Q%
14100 \string<r,R%
14101 \string<\glxtrLatinS
14102 \string& SS , \glxtrLatinEszettSs
14103 \string<\glxtrLatinT
14104 \string& th =\glshex 00DE
14105 \string& TH =\glshex 00FE
14106 \string<u,U%
14107 \string<v,V%
14108 \string<w,W%
14109 \string<\glxtrLatinX
14110 \string<y,Y%
14111 \string<z,Z%

```

14112 }

\glsxtrLatinA

14113 \newcommand\*{\glsxtrLatinA}{%

14114   a\string=\glshex 00AA\string=\glshex 2090,A

14115 }

\glsxtrLatinE

14116 \newcommand\*{\glsxtrLatinE}{%

14117   e\string=\glshex 2091,E

14118 }

\glsxtrLatinH

14119 \newcommand\*{\glsxtrLatinH}{%

14120   h\string=\glshex 2095,H

14121 }

\glsxtrLatinI

14122 \newcommand\*{\glsxtrLatinI}{%

14123   i\string=\glshex 2071,I

14124 }

\glsxtrLatinK

14125 \newcommand\*{\glsxtrLatinK}{%

14126   k\string=\glshex 2096,K

14127 }

\glsxtrLatinL

14128 \newcommand\*{\glsxtrLatinL}{%

14129   l\string=\glshex 2097,L

14130 }

\glsxtrLatinM

14131 \newcommand\*{\glsxtrLatinM}{%

14132   m\string=\glshex 2098,M

14133 }

\glsxtrLatinN

14134 \newcommand\*{\glsxtrLatinN}{%

14135   n\string=\glshex 207F\string=\glshex 2099,N

14136 }

\glsxtrLatinO

14137 \newcommand\*{\glsxtrLatinO}{%

14138   o\string=\glshex 00BA\string=\glshex 2092,O

14139 }

```

\glxtrLatinP
14140 \newcommand*{\glxtrLatinP}{%
14141   p\string=\glshex 209A,P
14142 }

\glxtrLatinS
14143 \newcommand*{\glxtrLatinS}{%
14144   s\string=\glshex 209B,S
14145 }

\glxtrLatinT
14146 \newcommand*{\glxtrLatinT}{%
14147   t\string=\glshex 209C,T
14148 }

\glxtrLatinX
14149 \newcommand*{\glxtrLatinX}{%
14150   x\string=\glshex 2093,X
14151 }

lsxtrLatinSchwa  Latin schwa (lower case, subscript and upper case).
14152 \newcommand*{\glxtrLatinSchwa}{%
14153   \glshex 0259\string=\glshex 2094,\glshex 018F
14154 }

trLatinEszettSs
14155 \newcommand*{\glxtrLatinEszettSs}{%
14156   \glshex 00DF% eszett
14157   \string=\glshex 017Fs % long S s
14158 }

trLatinEszettSz
14159 \newcommand*{\glxtrLatinEszettSz}{%
14160   \glshex 00DF% eszett
14161   \string= \glshex 017Fz % long S z
14162 }

\glxtrLatinEth
14163 \newcommand*{\glxtrLatinEth}{%
14164   \glshex 00F0,\glshex 00D0% eth
14165 }

lsxtrLatinThorn
14166 \newcommand*{\glxtrLatinThorn}{%
14167   \glshex 00FE,\glshex 00DE% thorn
14168 }

```

LatinAELigature

```
14169 \newcommand*{\glxtrLatinAELigature}{%
14170   \glshex 00E6,\glshex 00C6% AE-ligature
14171 }
```

LatinOELigature

```
14172 \newcommand*{\glxtrLatinOELigature}{%
14173   \glshex 0153,\glshex 0152% OE-ligature
14174 }
```

\glxtrLatinAA

```
14175 \newcommand*{\glxtrLatinAA}{%
14176   \glshex 00E5=a\glshex 030A,% \aa
14177   \glshex 00C5=A\glshex 030A% \AA
14178 }
```

glxtrLatinWynn

```
14179 \newcommand*{\glxtrLatinWynn}{%
14180   \glshex 01BF,\glshex 01F7% wynn
14181 }
```

trLatinInsularG

```
14182 \newcommand*{\glxtrLatinInsularG}{%
14183   \glshex 1D79,\glshex A77D% insular G
14184   \string; g, G
14185 }
```

sxtrLatinOslash

```
14186 \newcommand*{\glxtrLatinOslash}{%
14187   \glshex 00F8,\glshex 00D8% \o, \O
14188 }
```

sxtrLatinLslash

```
14189 \newcommand*{\glxtrLatinLslash}{%
14190   \glshex 0142,\glshex 0141% \l, \L
14191 }
```

thUpGreekIrules Includes digamma between epsilon and zeta.

```
14192 \newcommand*{\glxtrMathUpGreekIrules}{%
14193   \glxtrUpAlpha
14194   \string<\glxtrUpBeta
14195   \string<\glxtrUpGamma
14196   \string<\glxtrUpDelta
14197   \string<\glxtrUpEpsilon
14198   \string<\glxtrUpDigamma
14199   \string<\glxtrUpZeta
14200   \string<\glxtrUpEta
14201   \string<\glxtrUpTheta
```



```

14202 \string<\glxtrUpIota
14203 \string<\glxtrUpKappa
14204 \string<\glxtrUpLambda
14205 \string<\glxtrUpMu
14206 \string<\glxtrUpNu
14207 \string<\glxtrUpXi
14208 \string<\glxtrUpOmicron
14209 \string<\glxtrUpPi
14210 \string<\glxtrUpRho
14211 \string<\glxtrUpSigma
14212 \string<\glxtrUpTau
14213 \string<\glxtrUpUpsilon
14214 \string<\glxtrUpPhi
14215 \string<\glxtrUpChi
14216 \string<\glxtrUpPsi
14217 \string<\glxtrUpOmega
14218 }

```

**hUpGreekIIrules** Doesn't include digamma.

```

14219 \newcommand*{\glxtrMathUpGreekIIrules}{%
14220 \glxtrUpAlpha
14221 \string<\glxtrUpBeta
14222 \string<\glxtrUpGamma
14223 \string<\glxtrUpDelta
14224 \string<\glxtrUpEpsilon
14225 \string<\glxtrUpZeta
14226 \string<\glxtrUpEta
14227 \string<\glxtrUpTheta
14228 \string<\glxtrUpIota
14229 \string<\glxtrUpKappa
14230 \string<\glxtrUpLambda
14231 \string<\glxtrUpMu
14232 \string<\glxtrUpNu
14233 \string<\glxtrUpXi
14234 \string<\glxtrUpOmicron
14235 \string<\glxtrUpPi
14236 \string<\glxtrUpRho
14237 \string<\glxtrUpSigma
14238 \string<\glxtrUpTau
14239 \string<\glxtrUpUpsilon
14240 \string<\glxtrUpPhi
14241 \string<\glxtrUpChi
14242 \string<\glxtrUpPsi
14243 \string<\glxtrUpOmega
14244 }

```

**alicGreekIrules** Includes (upright) digamma between epsilon and zeta (there isn't an italic digamma), so don't mix with `\glxtrMathUpGreekIrules` or there may be unexpected results.

```

14245 \newcommand*{\glxtrMathItalicGreekIrules}{%

```

```

14246 \glxtrMathItalicAlpha
14247 \string<\glxtrMathItalicBeta
14248 \string<\glxtrMathItalicGamma
14249 \string<\glxtrMathItalicDelta
14250 \string<\glxtrMathItalicEpsilon
14251 \string<\glxtrUpDigamma
14252 \string<\glxtrMathItalicZeta
14253 \string<\glxtrMathItalicEta
14254 \string<\glxtrMathItalicTheta
14255 \string<\glxtrMathItalicIota
14256 \string<\glxtrMathItalicKappa
14257 \string<\glxtrMathItalicLambda
14258 \string<\glxtrMathItalicMu
14259 \string<\glxtrMathItalicNu
14260 \string<\glxtrMathItalicXi
14261 \string<\glxtrMathItalicOmicron
14262 \string<\glxtrMathItalicPi
14263 \string<\glxtrMathItalicRho
14264 \string<\glxtrMathItalicSigma
14265 \string<\glxtrMathItalicTau
14266 \string<\glxtrMathItalicUpsilon
14267 \string<\glxtrMathItalicPhi
14268 \string<\glxtrMathItalicChi
14269 \string<\glxtrMathItalicPsi
14270 \string<\glxtrMathItalicOmega
14271 }

```

licGreekIIrules Doesn't include digamma.

```

14272 \newcommand*{\glxtrMathItalicGreekIIrules}{%
14273 \glxtrMathItalicAlpha
14274 \string<\glxtrMathItalicBeta
14275 \string<\glxtrMathItalicGamma
14276 \string<\glxtrMathItalicDelta
14277 \string<\glxtrMathItalicEpsilon
14278 \string<\glxtrMathItalicZeta
14279 \string<\glxtrMathItalicEta
14280 \string<\glxtrMathItalicTheta
14281 \string<\glxtrMathItalicIota
14282 \string<\glxtrMathItalicKappa
14283 \string<\glxtrMathItalicLambda
14284 \string<\glxtrMathItalicMu
14285 \string<\glxtrMathItalicNu
14286 \string<\glxtrMathItalicXi
14287 \string<\glxtrMathItalicOmicron
14288 \string<\glxtrMathItalicPi
14289 \string<\glxtrMathItalicRho
14290 \string<\glxtrMathItalicSigma
14291 \string<\glxtrMathItalicTau
14292 \string<\glxtrMathItalicUpsilon

```

```

14293 \string<\glxtrMathItalicPhi
14294 \string<\glxtrMathItalicChi
14295 \string<\glxtrMathItalicPsi
14296 \string<\glxtrMathItalicOmega
14297 }

```

**UpperGreekIrules** Upper case only (includes upright digamma).

```

14298 \newcommand*{\glxtrMathItalicUpperGreekIrules}{%
14299 \glshex 1D6E2% upper case alpha (maths italic)
14300 \string<\glshex 1D6E3% upper case beta (maths italic)
14301 \string<\glshex 1D6E4% upper case gamma (maths italic)
14302 \string<\glshex 1D6E5% upper case delta (maths italic)
14303 \string<\glshex 1D6E6% upper case epsilon (maths italic)
14304 \string<\glshex 03DC% upper case digamma
14305 \string<\glshex 1D6E7% upper case zeta (maths italic)
14306 \string<\glshex 1D6E8% upper case eta (maths italic)
14307 \string<\glshex 1D6E9% upper case theta (maths italic)
14308 \string<\glshex 1D6F3% upper case theta variant (maths italic)
14309 \string<\glshex 1D6EA% upper case iota (maths italic)
14310 \string<\glshex 1D6EB% upper case kappa (maths italic)
14311 \string<\glshex 1D6EC% upper case lambda (maths italic)
14312 \string<\glshex 1D6ED% upper case mu (maths italic)
14313 \string<\glshex 1D6EE% upper case nu (maths italic)
14314 \string<\glshex 1D6EF% upper case xi (maths italic)
14315 \string<\glshex 1D6F0% upper case omicron (maths italic)
14316 \string<\glshex 1D6F1% upper case pi (maths italic)
14317 \string<\glshex 1D6F2% upper case rho (maths italic)
14318 \string<\glshex 1D6F4% upper case sigma (maths italic)
14319 \string<\glshex 1D6F5% upper case tau (maths italic)
14320 \string<\glshex 1D6F6% upper case upsilon (maths italic)
14321 \string<\glshex 1D6F7% upper case phi (maths italic)
14322 \string<\glshex 1D6F8% upper case chi (maths italic)
14323 \string<\glshex 1D6F9% upper case psi (maths italic)
14324 \string<\glshex 1D6FA% upper case omega (maths italic)
14325 }

```

**UpperGreekIIrules** Upper case only (doesn't include upright digamma).

```

14326 \newcommand*{\glxtrMathItalicUpperGreekIIrules}{%
14327 \glshex 1D6E2% upper case alpha (maths italic)
14328 \string<\glshex 1D6E3% upper case beta (maths italic)
14329 \string<\glshex 1D6E4% upper case gamma (maths italic)
14330 \string<\glshex 1D6E5% upper case delta (maths italic)
14331 \string<\glshex 1D6E6% upper case epsilon (maths italic)
14332 \string<\glshex 1D6E7% upper case zeta (maths italic)
14333 \string<\glshex 1D6E8% upper case eta (maths italic)
14334 \string<\glshex 1D6E9% upper case theta (maths italic)
14335 \string<\glshex 1D6F3% upper case theta variant (maths italic)
14336 \string<\glshex 1D6EA% upper case iota (maths italic)
14337 \string<\glshex 1D6EB% upper case kappa (maths italic)

```

```

14338 \string<\glshex 1D6EC% upper case lambda (maths italic)
14339 \string<\glshex 1D6ED% upper case mu (maths italic)
14340 \string<\glshex 1D6EE% upper case nu (maths italic)
14341 \string<\glshex 1D6EF% upper case xi (maths italic)
14342 \string<\glshex 1D6F0% upper case omicron (maths italic)
14343 \string<\glshex 1D6F1% upper case pi (maths italic)
14344 \string<\glshex 1D6F2% upper case rho (maths italic)
14345 \string<\glshex 1D6F4% upper case sigma (maths italic)
14346 \string<\glshex 1D6F5% upper case tau (maths italic)
14347 \string<\glshex 1D6F6% upper case upsilon (maths italic)
14348 \string<\glshex 1D6F7% upper case phi (maths italic)
14349 \string<\glshex 1D6F8% upper case chi (maths italic)
14350 \string<\glshex 1D6F9% upper case psi (maths italic)
14351 \string<\glshex 1D6FA% upper case omega (maths italic)
14352 }

```

LowerGreekIrules Lower case only (includes upright digamma).

```

14353 \newcommand*{\glstrMathItalicLowerGreekIrules}{%
14354 \glshex 1D6FC% lower case alpha (maths italic)
14355 \string<\glshex 1D6FD% lower case beta (maths italic)
14356 \string<\glshex 1D6FE% lower case gamma (maths italic)
14357 \string<\glshex 1D6FF% lower case delta (maths italic)
14358 \string<\glshex 1D700% lower case epsilon (maths italic)
14359 \string=<\glshex 1D716% lower case epsilon variant (maths italic)
14360 \string<\glshex 03DD% lower case digamma
14361 \string<\glshex 1D701% lower case zeta (maths italic)
14362 \string<\glshex 1D702% lower case eta (maths italic)
14363 \string<\glshex 1D703% lower case theta (maths italic)
14364 \string=<\glshex 1D717% lower case theta variant (maths italic)
14365 \string<\glshex 1D704% lower case iota (maths italic)
14366 \string<\glshex 1D705% lower case kappa (maths italic)
14367 \string=<\glshex 1D718% lower case kappa variant (maths italic)
14368 \string<\glshex 1D706% lower case lambda (maths italic)
14369 \string<\glshex 1D707% lower case mu (maths italic)
14370 \string<\glshex 1D708% lower case nu (maths italic)
14371 \string<\glshex 1D709% lower case xi (maths italic)
14372 \string<\glshex 1D70A% lower case omicron (maths italic)
14373 \string<\glshex 1D70B% lower case pi (maths italic)
14374 \string=<\glshex 1D71B% lower case pi variant (maths italic)
14375 \string<\glshex 1D70C% lower case rho (maths italic)
14376 \string=<\glshex 1D71A% lower case rho variant (maths italic)
14377 \string<\glshex 1D70D% lower case final sigma (maths italic)
14378 \string=<\glshex 1D70E% lower case sigma (maths italic)
14379 \string<\glshex 1D70F% lower case tau (maths italic)
14380 \string<\glshex 1D710% lower case upsilon (maths italic)
14381 \string<\glshex 1D711% lower case phi (maths italic)
14382 \string=<\glshex 1D719% lower case phi variant (maths italic)
14383 \string<\glshex 1D712% lower case chi (maths italic)
14384 \string<\glshex 1D713% lower case psi (maths italic)

```

```

14385 \string<\glshex 1D714% lower case omega (maths italic)
14386 }

```

**LowerGreeklIrules** Lower case only (doesn't includes upright digamma).

```

14387 \newcommand*{\glxtrMathItalicLowerGreeklIrules}{%
14388 \glshex 1D6FC% lower case alpha (maths italic)
14389 \string<\glshex 1D6FD% lower case beta (maths italic)
14390 \string<\glshex 1D6FE% lower case gamma (maths italic)
14391 \string<\glshex 1D6FF% lower case delta (maths italic)
14392 \string<\glshex 1D700% lower case epsilon (maths italic)
14393 \string=<\glshex 1D716% lower case epsilon variant (maths italic)
14394 \string<\glshex 1D701% lower case zeta (maths italic)
14395 \string<\glshex 1D702% lower case eta (maths italic)
14396 \string<\glshex 1D703% lower case theta (maths italic)
14397 \string=<\glshex 1D717% lower case theta variant (maths italic)
14398 \string<\glshex 1D704% lower case iota (maths italic)
14399 \string<\glshex 1D705% lower case kappa (maths italic)
14400 \string=<\glshex 1D718% lower case kappa variant (maths italic)
14401 \string<\glshex 1D706% lower case lambda (maths italic)
14402 \string<\glshex 1D707% lower case mu (maths italic)
14403 \string<\glshex 1D708% lower case nu (maths italic)
14404 \string<\glshex 1D709% lower case xi (maths italic)
14405 \string<\glshex 1D70A% lower case omicron (maths italic)
14406 \string<\glshex 1D70B% lower case pi (maths italic)
14407 \string=<\glshex 1D71B% lower case pi variant (maths italic)
14408 \string<\glshex 1D70C% lower case rho (maths italic)
14409 \string=<\glshex 1D71A% lower case rho variant (maths italic)
14410 \string<\glshex 1D70D% lower case final sigma (maths italic)
14411 \string=<\glshex 1D70E% lower case sigma (maths italic)
14412 \string<\glshex 1D70F% lower case tau (maths italic)
14413 \string<\glshex 1D710% lower case upsilon (maths italic)
14414 \string<\glshex 1D711% lower case phi (maths italic)
14415 \string=<\glshex 1D719% lower case phi variant (maths italic)
14416 \string<\glshex 1D712% lower case chi (maths italic)
14417 \string<\glshex 1D713% lower case psi (maths italic)
14418 \string<\glshex 1D714% lower case omega (maths italic)
14419 }

```

**MathGreeklIrules** Includes both upright and italic with digamma between epsilon and zeta.

```

14420 \newcommand*{\glxtrMathGreeklIrules}{%
14421 \glxtrMathItalicAlpha
14422 \string;\glxtrUpAlpha
14423 \string<\glxtrMathItalicBeta
14424 \string;\glxtrUpBeta
14425 \string<\glxtrMathItalicGamma
14426 \string;\glxtrUpGamma
14427 \string<\glxtrMathItalicDelta
14428 \string;\glxtrUpDelta
14429 \string<\glxtrMathItalicEpsilon

```

```

14430 \string;\glxtrUpEpsilon
14431 \string<\glxtrUpDigamma
14432 \string<\glxtrMathItalicZeta
14433 \string;\glxtrUpZeta
14434 \string<\glxtrMathItalicEta
14435 \string;\glxtrUpEta
14436 \string<\glxtrMathItalicTheta
14437 \string;\glxtrUpTheta
14438 \string<\glxtrMathItalicIota
14439 \string;\glxtrUpIota
14440 \string<\glxtrMathItalicKappa
14441 \string;\glxtrUpKappa
14442 \string<\glxtrMathItalicLambda
14443 \string;\glxtrUpLambda
14444 \string<\glxtrMathItalicMu
14445 \string;\glxtrUpMu
14446 \string<\glxtrMathItalicNu
14447 \string;\glxtrUpNu
14448 \string<\glxtrMathItalicXi
14449 \string;\glxtrUpXi
14450 \string<\glxtrMathItalicOmicron
14451 \string;\glxtrUpOmicron
14452 \string<\glxtrMathItalicPi
14453 \string;\glxtrUpPi
14454 \string<\glxtrMathItalicRho
14455 \string;\glxtrUpRho
14456 \string<\glxtrMathItalicSigma
14457 \string;\glxtrUpSigma
14458 \string<\glxtrMathItalicTau
14459 \string;\glxtrUpTau
14460 \string<\glxtrMathItalicUpsilon
14461 \string;\glxtrUpUpsilon
14462 \string<\glxtrMathItalicPhi
14463 \string;\glxtrUpPhi
14464 \string<\glxtrMathItalicChi
14465 \string;\glxtrUpChi
14466 \string<\glxtrMathItalicPsi
14467 \string;\glxtrUpPsi
14468 \string<\glxtrMathItalicOmega
14469 \string;\glxtrUpOmega
14470 }

```

**mathGreekIIrules** Includes both upright and italic (digamma not included).

```

14471 \newcommand*{\glxtrMathGreekIIrules}{%
14472 \glxtrMathItalicAlpha
14473 \string;\glxtrUpAlpha
14474 \string<\glxtrMathItalicBeta
14475 \string;\glxtrUpBeta
14476 \string<\glxtrMathItalicGamma

```

```

14477 \string;\glsxtrUpGamma
14478 \string<\glsxtrMathItalicDelta
14479 \string;\glsxtrUpDelta
14480 \string<\glsxtrMathItalicEpsilon
14481 \string;\glsxtrUpEpsilon
14482 \string<\glsxtrMathItalicZeta
14483 \string;\glsxtrUpZeta
14484 \string<\glsxtrMathItalicEta
14485 \string;\glsxtrUpEta
14486 \string<\glsxtrMathItalicTheta
14487 \string;\glsxtrUpTheta
14488 \string<\glsxtrMathItalicIota
14489 \string;\glsxtrUpIota
14490 \string<\glsxtrMathItalicKappa
14491 \string;\glsxtrUpKappa
14492 \string<\glsxtrMathItalicLambda
14493 \string;\glsxtrUpLambda
14494 \string<\glsxtrMathItalicMu
14495 \string;\glsxtrUpMu
14496 \string<\glsxtrMathItalicNu
14497 \string;\glsxtrUpNu
14498 \string<\glsxtrMathItalicXi
14499 \string;\glsxtrUpXi
14500 \string<\glsxtrMathItalicOmicron
14501 \string;\glsxtrUpOmicron
14502 \string<\glsxtrMathItalicPi
14503 \string;\glsxtrUpPi
14504 \string<\glsxtrMathItalicRho
14505 \string;\glsxtrUpRho
14506 \string<\glsxtrMathItalicSigma
14507 \string;\glsxtrUpSigma
14508 \string<\glsxtrMathItalicTau
14509 \string;\glsxtrUpTau
14510 \string<\glsxtrMathItalicUpsilon
14511 \string;\glsxtrUpUpsilon
14512 \string<\glsxtrMathItalicPhi
14513 \string;\glsxtrUpPhi
14514 \string<\glsxtrMathItalicChi
14515 \string;\glsxtrUpChi
14516 \string<\glsxtrMathItalicPsi
14517 \string;\glsxtrUpPsi
14518 \string<\glsxtrMathItalicOmega
14519 \string;\glsxtrUpOmega
14520 }

```

`\glsxtrUpAlpha`

```

14521 \newcommand*{\glsxtrUpAlpha}{%
14522 \glshex 03B1,% lower case alpha
14523 \glshex 0391% upper case alpha

```

14524 }

\glxtrUpBeta

```
14525 \newcommand*{\glxtrUpBeta}{%
14526   \glshex 03B2,% lower case beta
14527   \glshex 0392% upper case beta
14528 }
```

\glxtrUpGamma

```
14529 \newcommand*{\glxtrUpGamma}{%
14530   \glshex 03B3,% lower case gamma
14531   \glshex 0393% upper case gamma
14532 }
```

\glxtrUpDelta

```
14533 \newcommand*{\glxtrUpDelta}{%
14534   \glshex 03B4,% lower case delta
14535   \glshex 0394% upper case delta
14536 }
```

glxtrUpEpsilon

```
14537 \newcommand*{\glxtrUpEpsilon}{%
14538   \glshex 03B5% lower case epsilon
14539   \string=\glshex 03F5,% lower case epsilon variant
14540   \glshex 0395% upper case epsilon
14541 }
```

glxtrUpDigamma

```
14542 \newcommand*{\glxtrUpDigamma}{%
14543   \glshex 03DD,% lower case digamma
14544   \glshex 03DC% upper case digamma
14545 }
```

\glxtrUpZeta

```
14546 \newcommand*{\glxtrUpZeta}{%
14547   \glshex 03B6,% lower case zeta
14548   \glshex 0396% upper case zeta
14549 }
```

\glxtrUpEta

```
14550 \newcommand*{\glxtrUpEta}{%
14551   \glshex 03B7,% lower case eta
14552   \glshex 0397% upper case eta
14553 }
```

\glxtrUpTheta

```
14554 \newcommand*{\glxtrUpTheta}{%
14555   \glshex 03B8% lower case theta
```



```

14556 \string=\glshex 03D1,% lower case theta variant
14557 \glshex 0398% upper case theta
14558 }

```

\glxtrUpIota

```

14559 \newcommand*\glxtrUpIota}{%
14560 \glshex 03B9,% lower case iota
14561 \glshex 0399% upper case iota
14562 }

```

\glxtrUpKappa

```

14563 \newcommand*\glxtrUpKappa}{%
14564 \glshex 03BA,% lower case kappa
14565 \string=\glshex 03F0,% lower case kappa variant
14566 \glshex 039A% upper case kappa
14567 }

```

\glxtrUpLambda

```

14568 \newcommand*\glxtrUpLambda}{%
14569 \glshex 03BB,% lower lambda
14570 \glshex 039B% upper case lambda
14571 }

```

\glxtrUpMu

```

14572 \newcommand*\glxtrUpMu}{%
14573 \glshex 03BC,% lower case mu
14574 \glshex 039C% upper case mu
14575 }

```

\glxtrUpNu

```

14576 \newcommand*\glxtrUpNu}{%
14577 \glshex 03BD,% lower case nu
14578 \glshex 039D% upper case nu
14579 }

```

\glxtrUpXi

```

14580 \newcommand*\glxtrUpXi}{%
14581 \glshex 03BE,% lower case xi
14582 \glshex 039E% upper case xi
14583 }

```

\glxtrUpOmicron

```

14584 \newcommand*\glxtrUpOmicron}{%
14585 \glshex 03BF,% lower case omicron
14586 \glshex 039F% upper case omicron
14587 }

```

\glxtrUpPi

```
14588 \newcommand*{\glxtrUpPi}{%
14589 \glshex 03C0% lower case pi
14590 \string=\glshex 03D6,% lower case pi variant
14591 \glshex 03A0% upper case pi
14592 }
```

\glxtrUpRho

```
14593 \newcommand*{\glxtrUpRho}{%
14594 \glshex 03C1% lower case rho
14595 \string=\glshex 03F1,% lower case rho variant
14596 \glshex 03A1% upper case rho
14597 }
```

\glxtrUpSigma

```
14598 \newcommand*{\glxtrUpSigma}{%
14599 \glshex 03C2% lower case sigma
14600 \string=\glshex 03C3,% lower case sigma
14601 \glshex 03A3% upper case sigma
14602 }
```

\glxtrUpTau

```
14603 \newcommand*{\glxtrUpTau}{%
14604 \glshex 03C4,% lower case tau
14605 \glshex 03A4% upper case tau
14606 }
```

\glxtrUpUpsilon

```
14607 \newcommand*{\glxtrUpUpsilon}{%
14608 \glshex 03C5,% lower case upsilon
14609 \glshex 03A5% upper case upsilon
14610 }
```

\glxtrUpPhi

```
14611 \newcommand*{\glxtrUpPhi}{%
14612 \glshex 03C6% lower case phi
14613 \string=\glshex 03D5,% lower case phi variant
14614 \glshex 03A6% upper case phi
14615 }
```

\glxtrUpChi

```
14616 \newcommand*{\glxtrUpChi}{%
14617 \glshex 03C7,% lower case chi
14618 \glshex 03A7% upper case chi
14619 }
```

\glxtrUpPsi

```
14620 \newcommand*{\glxtrUpPsi}{%
```

```

14621 \glshex 03C8,% lower case psi
14622 \glshex 03A8% upper case psi
14623 }

```

$\backslash\mathrm{glxtrUpOmega}$

```

14624 \newcommand*{\glxtrUpOmega}{%
14625 \glshex 03C9,% lower case omega
14626 \glshex 03A9% upper case omega
14627 }

```

$\mathrm{MathItalicAlpha}$

```

14628 \newcommand*{\glxtrMathItalicAlpha}{%
14629 \glshex 1D6FC,% lower case alpha (maths italic)
14630 \glshex 1D6E2% upper case alpha (maths italic)
14631 }

```

$\mathrm{rMathItalicBeta}$

```

14632 \newcommand*{\glxtrMathItalicBeta}{%
14633 \glshex 1D6FD,% lower case beta (maths italic)
14634 \glshex 1D6E3% upper case beta (maths italic)
14635 }

```

$\mathrm{MathItalicGamma}$

```

14636 \newcommand*{\glxtrMathItalicGamma}{%
14637 \glshex 1D6FE,% lower case gamma (maths italic)
14638 \glshex 1D6E4% upper case gamma (maths italic)
14639 }

```

$\mathrm{MathItalicDelta}$

```

14640 \newcommand*{\glxtrMathItalicDelta}{%
14641 \glshex 1D6FF,% lower case delta (maths italic)
14642 \glshex 1D6E5% upper case delta (maths italic)
14643 }

```

$\mathrm{thItalicEpsilon}$

```

14644 \newcommand*{\glxtrMathItalicEpsilon}{%
14645 \glshex 1D700% lower case epsilon (maths italic)
14646 \string=\glshex 1D716,% lower case epsilon variant (maths italic)
14647 \glshex 1D6E6% upper case epsilon (maths italic)
14648 }

```

$\mathrm{rMathItalicZeta}$

```

14649 \newcommand*{\glxtrMathItalicZeta}{%
14650 \glshex 1D701,% lower case zeta (maths italic)
14651 \glshex 1D6E7% upper case zeta (maths italic)
14652 }

```

trMathItalicEta

```
14653 \newcommand*{\glxtrMathItalicEta}{%
14654   \glshex 1D702,% lower case eta (maths italic)
14655   \glshex 1D6E8% upper case eta (maths italic)
14656 }
```

MathItalicTheta

```
14657 \newcommand*{\glxtrMathItalicTheta}{%
14658   \glshex 1D703% lower case theta (maths italic)
14659   \string=\glshex 1D717,% lower case theta variant (maths italic)
14660   \glshex 1D6E9% upper case theta (maths italic)
14661   \string=\glshex 1D6F3% upper case theta variant (maths italic)
14662 }
```

rMathItalicIota

```
14663 \newcommand*{\glxtrMathItalicIota}{%
14664   \glshex 1D704,% lower case iota (maths italic)
14665   \glshex 1D6EA% upper case iota (maths italic)
14666 }
```

MathItalicKappa

```
14667 \newcommand*{\glxtrMathItalicKappa}{%
14668   \glshex 1D705% lower case kappa (maths italic)
14669   \string=\glshex 1D718,% lower case kappa variant (maths italic)
14670   \glshex 1D6EB% upper case kappa (maths italic)
14671 }
```

athItalicLambda

```
14672 \newcommand*{\glxtrMathItalicLambda}{%
14673   \glshex 1D706,% lower case lambda (maths italic)
14674   \glshex 1D6EC% upper case lambda (maths italic)
14675 }
```

xtrMathItalicMu

```
14676 \newcommand*{\glxtrMathItalicMu}{%
14677   \glshex 1D707,% lower case mu (maths italic)
14678   \glshex 1D6ED% upper case mu (maths italic)
14679 }
```

xtrMathItalicNu

```
14680 \newcommand*{\glxtrMathItalicNu}{%
14681   \glshex 1D708,% lower case nu (maths italic)
14682   \glshex 1D6EE% upper case nu (maths italic)
14683 }
```

xtrMathItalicXi

```
14684 \newcommand*{\glxtrMathItalicXi}{%
14685   \glshex 1D709,% lower case xi (maths italic)
```

```

14686 \glshex 1D6EF% upper case xi (maths italic)
14687 }

thItalicOmicron

14688 \newcommand*{\glxtrMathItalicOmicron}{%
14689 \glshex 1D70A,% lower case omicron (maths italic)
14690 \glshex 1D6F0% upper case omicron (maths italic)
14691 }

xtrMathItalicPi

14692 \newcommand*{\glxtrMathItalicPi}{%
14693 \glshex 1D70B% lower case pi (maths italic)
14694 \string=\glshex 1D71B,% lower case pi variant (maths italic)
14695 \glshex 1D6F1% upper case pi (maths italic)
14696 }

trMathItalicRho

14697 \newcommand*{\glxtrMathItalicRho}{%
14698 \glshex 1D70C% lower case rho (maths italic)
14699 \string=\glshex 1D71A,% lower case rho variant (maths italic)
14700 \glshex 1D6F2% upper case rho (maths italic)
14701 }

MathItalicSigma

14702 \newcommand*{\glxtrMathItalicSigma}{%
14703 \glshex 1D70D% lower case final sigma (maths italic)
14704 \string=\glshex 1D70E,% lower case sigma (maths italic)
14705 \glshex 1D6F4% upper case sigma (maths italic)
14706 }

trMathItalicTau

14707 \newcommand*{\glxtrMathItalicTau}{%
14708 \glshex 1D70F,% lower case tau (maths italic)
14709 \glshex 1D6F5% upper case tau (maths italic)
14710 }

thItalicUpsilon

14711 \newcommand*{\glxtrMathItalicUpsilon}{%
14712 \glshex 1D710,% lower case upsilon (maths italic)
14713 \glshex 1D6F6% upper case upsilon (maths italic)
14714 }

trMathItalicPhi

14715 \newcommand*{\glxtrMathItalicPhi}{%
14716 \glshex 1D711% lower case phi (maths italic)
14717 \string=\glshex 1D719,% lower case phi variant (maths italic)
14718 \glshex 1D6F7% upper case phi (maths italic)
14719 }

```

trMathItalicChi

```
14720 \newcommand*{\glxtrMathItalicChi}{%
14721   \glshex 1D712,% lower case chi (maths italic)
14722   \glshex 1D6F8% upper case chi (maths italic)
14723 }
```

trMathItalicPsi

```
14724 \newcommand*{\glxtrMathItalicPsi}{%
14725   \glshex 1D713,% lower case psi (maths italic)
14726   \glshex 1D6F9% upper case psi (maths italic)
14727 }
```

MathItalicOmega

```
14728 \newcommand*{\glxtrMathItalicOmega}{%
14729   \glshex 1D714,% lower case omega (maths italic)
14730   \glshex 1D6FA% upper case omega (maths italic)
14731 }
```

thItalicPartial

```
14732 \newcommand*{\glxtrMathItalicPartial}{%
14733   \glshex 1D715% partial differential (maths italic)
14734 }
```

MathItalicNabla

```
14735 \newcommand*{\glxtrMathItalicNabla}{%
14736   \glshex 1D6FB% nabla (maths italic)
14737 }
```

lsxtrdigitrules Digits from the Basic Latin set and subscript and superscript digit rules.

```
14738 \newcommand*{\glxtrdigitrules}{%
14739   0\string=\glshex 2080\string=\glshex 2070
14740   \string<1\string=\glshex 2081\string=\glshex 00B9
14741   \string<2\string=\glshex 2082\string=\glshex 00B2
14742   \string<3\string=\glshex 2083\string=\glshex 00B3
14743   \string<4\string=\glshex 2084\string=\glshex 2074
14744   \string<5\string=\glshex 2085\string=\glshex 2075
14745   \string<6\string=\glshex 2086\string=\glshex 2076
14746   \string<7\string=\glshex 2087\string=\glshex 2077
14747   \string<8\string=\glshex 2088\string=\glshex 2078
14748   \string<9\string=\glshex 2089\string=\glshex 2079
14749 }
```

BasicDigitrules Digits from the Basic Latin set.

```
14750 \newcommand*{\glxtrBasicDigitrules}{%
14751   0\string<1\string<2\string<3\string<4%
14752   \string<5\string<6\string<7\string<8\string<9%
14753 }
```

criptDigitrules Subscript digits.

```
14754 \newcommand*{\glxtrSubScriptDigitrules}{%
14755 \glshex 2080% subscript 0
14756 \string<\glshex 2081% subscript 1
14757 \string<\glshex 2082% subscript 2
14758 \string<\glshex 2083% subscript 3
14759 \string<\glshex 2084% subscript 4
14760 \string<\glshex 2085% subscript 5
14761 \string<\glshex 2086% subscript 6
14762 \string<\glshex 2087% subscript 7
14763 \string<\glshex 2088% subscript 8
14764 \string<\glshex 2089% subscript 9
14765 }
```

criptDigitrules Superscript digits.

```
14766 \newcommand*{\glxtrSuperScriptDigitrules}{%
14767 \glshex 2070% superscript 0
14768 \string<\glshex 00B9% superscript 1
14769 \string<\glshex 00B2% superscript 2
14770 \string<\glshex 00B3% superscript 3
14771 \string<\glshex 2074% superscript 4
14772 \string<\glshex 2075% superscript 5
14773 \string<\glshex 2076% superscript 6
14774 \string<\glshex 2077% superscript 7
14775 \string<\glshex 2078% superscript 8
14776 \string<\glshex 2079% superscript 9
14777 }
```

trfractionrules Vulgar fractions.

```
14778 \newcommand*{\glxtrfractionrules}{%
14779 \glshex 215F% fraction numerator one (1/)
14780 \string<\glshex 2189% zero thirds (0/3 = 0)
14781 \string<\glshex 2152% one tenth (1/10 = 0.1)
14782 \string<\glshex 2151% one ninth (1/9 ~ 0.111)
14783 \string<\glshex 215B% one eighth (1/8 = 0.125)
14784 \string<\glshex 2150% one seventh (1/7 ~ 0.143)
14785 \string<\glshex 2159% one sixth (1/6 ~ 0.167)
14786 \string<\glshex 2155% one fifth (1/5 = 0.2)
14787 \string<\glshex 00BC% one quarter (1/4 = 0.25)
14788 \string<\glshex 2153% one third (1/3 ~ 0.333)
14789 \string<\glshex 215C% three eighths (3/8 = 0.375)
14790 \string<\glshex 2156% two fifths (2/5 = 0.4)
14791 \string<\glshex 00BD% one half (1/2 = 0.5)
14792 \string<\glshex 2157% three fifths (3/5 = 0.6)
14793 \string<\glshex 215D% five eighths (5/8 = 0.625)
14794 \string<\glshex 2154% two thirds (2/3 ~ 0.667)
14795 \string<\glshex 00BE% three quarters (3/4 = 0.75)
14796 \string<\glshex 2158% four fifths (4/5 = 0.8)
14797 \string<\glshex 215A% five sixths (5/6 ~ 0.833)
```

```

14798 \string<\glshex 215E% seven eighths (7/8 = 0.875)
14799 }

```

sxtrdialecthook Check for scripts associated with the document dialects.

```

14800 \renewcommand{\@glsxtrdialecthook}{%
14801   \ifundef\CurrentTrackedScript
14802   {%
14803     \TrackLangIfHasDefaultScript{\CurrentTrackedLanguage}%
14804     {%
14805       \edef\CurrentTrackedScript{%
14806         \TrackLangGetDefaultScript\CurrentTrackedLanguage}%
14807     }%
14808   }%
14809 }%
14810 {}%
14811 \ifdef\CurrentTrackedScript
14812 {%
14813   \let\gls@orgTrackLangRequireDialectPrefix\TrackLangRequireDialectPrefix
14814   \def\TrackLangRequireDialectPrefix{glossariesxtr-}%
14815   \let\CurrentTrackedTag\CurrentTrackedScript
14816   \IfFileExists{\TrackLangRequireDialectPrefix\CurrentTrackedTag.ldf}
14817   {\RequireGlossariesExtraLang{\CurrentTrackedTag}}%
14818   {}%
14819   \let\TrackLangRequireDialectPrefix\gls@orgTrackLangRequireDialectPrefix
14820 }%
14821 {}%
14822 }

```

If `\glsxtr@loaddialect` has been defined, then `glossaries-extra-bib2gls` has been loaded after `glossaries-extra`. (For example, through `\glossariesextrasetup`.) Not recommended, but if this has been done try to find the associated language resources.

```

14823 \ifdef\glsxtr@loaddialect
14824 {%
14825   \@ifpackageloaded{tracklang}
14826   {%
14827     \AnyTrackedLanguages
14828     {%
14829       \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
14830     }%
14831   }%
14832 }
14833 {}
14834 }
14835 {}

```



## 2 Style Adjustments (glossaries-extra-stylemods.sty)

This package adjusts the predefined styles so that they include the post description hook. Also, some other minor adjustments may be made to make existing styles more flexible.

### 2.1 Package Initialisation

First identify package:

```
14836 \NeedsTeXFormat{LaTeX2e}
14837 \ProvidesPackage{glossaries-extra-stylemods}[2021/11/04 v1.47 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-⟨option⟩.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

`sxtr@loadstyles`

```
14838 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
14839 \DeclareOption{all}{%
14840   \appto\@glxtr@loadstyles{%
14841     \RequirePackage{glossary-inline}%
14842     \RequirePackage{glossary-list}%
14843     \RequirePackage{glossary-tree}%
14844     \RequirePackage{glossary-mcols}%
14845     \RequirePackage{glossary-long}%
14846     \RequirePackage{glossary-longragged}%
14847     \RequirePackage{glossary-longbooktabs}%
14848     \RequirePackage{glossary-super}%
14849     \RequirePackage{glossary-superragged}%
14850     \RequirePackage{glossary-bookindex}%
14851     \RequirePackage{glossary-longextra}%
14852     \RequirePackage{glossary-topic}%
14853   }
14854 }

14855 \DeclareOption*{%
14856   \IfFileExists{glossary-\CurrentOption.sty}
14857   {\eappto\@glxtr@loadstyles{%
14858     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
14859   }%
```

```

14860   {%
14861       \PackageError{glossaries-extra-styles}%
14862       {Unknown option ‘\CurrentOption’}{}%
14863   }%
14864 }

```

Process the package options:

```
14865 \ProcessOptions
```

Load the required packages:

```
14866 \@glsxtr@loadstyles
```

Adjust the styles so that they all have the post description hook. Also, instead of having a hard-coded \space before the location, use:

sxtrprelocation This uses \providecommand as the same command is also provided by glossary-bookindex.

```
14867 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

ewglossarystyle

```

14868 \providecommand{\renewglossarystyle}[2]{%
14869   \ifcsundef{@glsstyle@#1}%
14870   {%
14871       \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
14872   }%
14873   {%
14874       \csdef{@glsstyle@#1}{#2}%
14875   }%
14876 }

```

## 2.2 List-Like Styles

The list-like styles mostly already use the post description hook. Only the listdotted style need modifying to add this.

```

14877 \ifdef{\@glsstyle@listdotted}
14878 {%
14879   \renewglossarystyle{listdotted}{%
14880     \setglossarystyle{list}%
14881     \renewcommand*{\glossentry}[2]{%
14882       \item[]\makebox[\glslistdottedwidth][l]{%
14883         \glsentryitem{##1}%
14884         \glstarget{##1}{\glossentryname{##1}}%
14885         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
14886       \glossentrydesc{##1}\glspostdescription}%
14887     \renewcommand*{\subglossentry}[3]{%
14888       \item[]\makebox[\glslistdottedwidth][l]{%
14889         \glssubentryitem{##2}%
14890         \glstarget{##2}{\glossentryname{##2}}%
14891         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%

```

```

14892     \glossentrydesc{##2}\glspostdescription}%
14893 }
14894 }
14895 {%

```

Assume the style isn't required if it hasn't already been defined.

```

14896 }

```

The sublistdotted style doesn't display the description for top-level entries. Sub-level entries use the listdottedstyle.

The other list styles would be easier to adapt if the space before the number list wasn't hard coded.

```

14897 \ifdef{\@glsstyle@list}
14898 {%

```

listprelocation Space before number list for top-level entries.

```

14899 \newcommand{\glslistprelocation}{\glxstrprelocation}

```

childprelocation Space before number list for child entries.

```

14900 \newcommand{\glslistchildprelocation}{\glslistprelocation}

```

childpostlocation Full stop after number list.

```

14901 \newcommand{\glslistchildpostlocation}{.}

```

\glslistdesc

```

14902 \newcommand{\glslistdesc}[1]{\glossentrydesc{#1}\glspostdescription}

```

glslistgroupskip

```

14903 \newcommand{\glslistgroupskip}{\nobreak\indexspace\nobreak}

```

\glslistitem

```

14904 \newcommand{\glslistitem}[1]{%
14905   \item[\glsentryitem{#1}%
14906     \glstarget{#1}{\glossentryname{#1}}}%
14907 }

```

\glslistinit This command was only added to glossary-list v4.48 so provide it if it hasn't been defined:

```

14908 \providecommand{\glslistinit}{%
14909   \ifdef\GetTitleStringDisableCommands
14910     {%
14911       \GetTitleStringSetup{expand}%
14912       \GetTitleStringDisableCommands{%
14913         \let\glsentryitem\@gobble
14914         \let\glstarget\@secondoftwo
14915         \let\glossentryname\glslistexpandedname
14916         \let\glslistgroupheaderfmt\@firstofone
14917         \let\glsgetgrouptitle\@firstofone
14918         \let\glsnavhypertarget\@secondoftwo
14919         \let\glsnavigation\relax

```

```

14920     }%
14921     }%
14922     {}%
14923 }

```

istexpandedname This command was only added to glossary-list v4.48 so provide it if it hasn't been defined. The original definition uses \glsunexpandedfieldvalue which was added to glossaries v4.48 (so if \glslistexpandedname hasn't been defined then neither will \glsunexpandedfieldvalue).

```

14924 \providecommand{\glslistexpandedname}[1]{%
14925   \ifcsname glo@\glsdetoklabel{#1}@name\endcsname
14926     \expandafter\expandonce\csname glo@\glsdetoklabel{#1}@name\expandafter\endcsname
14927   \fi
14928 }

```

Redefine list to use these commands.

```

14929 \renewglossarystyle{list}{%
14930   \renewenvironment{theglossary}%
14931     {\glslistinit\begin{description}}{\end{description}}%
14932   \renewcommand*\glossaryheader{}%
14933   \renewcommand*\glsgroupheading[1]{}%
14934   \renewcommand*\glossentry[2]{%
14935     \glslistitem{##1}\glslistdesc{##1}\glslistprelocation ##2}%
14936   \renewcommand*\subglossentry[3]{%
14937     \glssubentryitem{##2}%
14938     \glstarget{##2}{\strut}\space
14939     \glslistdesc{##2}%
14940     \glslistchildprelocation ##3\glslistchildpostlocation}%
14941   \renewcommand*\glsgroupskip{\ifglsnogroupskip\else\glslistgroupskip\fi}%
14942 }
14943 }
14944 {}

```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```

14945 \ifdef{\@glsstyle@altlist}
14946 {}

```

\glsaltlistitem

```

14947 \newcommand{\glsaltlistitem}[1]{%
14948   \glslistitem{#1}%
14949   \mbox{}\par\nobreak\@afterheading
14950 }

14951 \renewglossarystyle{altlist}{%
14952   \setglossarystyle{list}%
14953   \renewcommand*\glossentry[2]{%
14954     \glsaltlistitem{##1}%
14955     \glslistdesc{##1}\glslistprelocation ##2}%
14956   \renewcommand*\subglossentry[3]{%
14957     \par

```

```

14958     \glssubentryitem{##2}%
14959     \glstarget{##2}{\strut}\glslistdesc{##2}%
14960     \glslistchildprelocation ##3}%
14961 }
14962 }
14963 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

14964 \ifdef{\@glsstyle@listgroup}
14965 {%

```

roupafterheader

```

14966   \newcommand{\glslistgroupheaderitem}[2]{\item[{#2}]}
14967   \newcommand{\glslistgroupafterheader}{%
14968     \mbox{}\par\nobreak\@afterheading
14969   }

14970   \renewglossarystyle{listgroup}{%
14971     \setglossarystyle{list}%
14972     \renewcommand*\{\glsgroupheading}[1]{%
14973       \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
14974       \glslistgroupafterheader
14975     }%
14976   }
14977 }
14978 {}

```

Similarly for listhypergroup.

```

14979 \ifdef{\@glsstyle@listhypergroup}
14980 {%
14981   \renewglossarystyle{listhypergroup}{%
14982     \setglossarystyle{list}%
14983     \renewcommand*\{\glossaryheader}{%
14984       \glslistnavigationitem{\glsnavigation}}%
14985     \renewcommand*\{\glsgroupheading}[1]{%
14986       \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt
14987         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}}%
14988       \glslistgroupafterheader
14989     }%
14990   }
14991 }
14992 {}

```

Similarly for altlistgroup.

```

14993 \ifdef{\@glsstyle@altlistgroup}
14994 {%
14995   \renewglossarystyle{altlistgroup}{%
14996     \setglossarystyle{altlist}%
14997     \renewcommand*\{\glsgroupheading}[1]{%
14998       \glslistgroupheaderitem{##1}%

```

```

14999         {\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
15000     \glslistgroupafterheader
15001 }%
15002 }
15003 }
15004 {}

```

Similarly for `altlisthypergroup`.

```

15005 \ifdef{\@glsstyle@altlisthypergroup}
15006 {%
15007     \renewglossarystyle{altlisthypergroup}{%
15008         \setglossarystyle{altlist}%
15009         \renewcommand*\glossaryheader{%
15010             \glslistnavigationitem{\glsnavigation}}%
15011         \renewcommand*\glsgroupheading}[1]{%
15012             \glslistgroupheaderitem{##1}{\glslistgroupheaderfmt
15013                 {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}}%
15014             \glslistgroupafterheader
15015         }%
15016     }
15017 }
15018 {}

```

## 2.3 Longtable Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glstrprelocation`.

```

15019 \ifcsdef{@glsstyle@long}
15020 {%
15021     \renewglossarystyle{long}{%
15022         \renewenvironment{theglossary}%
15023             {\begin{longtable}{lp{\glsdescwidth}}}%
15024             {\end{longtable}}%
15025         \renewcommand*\glossaryheader{}%
15026         \renewcommand*\glsgroupheading}[1]{%
15027             \renewcommand{\glossentry}[2]{%
15028                 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15029                 \glossentrydesc{##1}\glspostdescription
15030                 \glstrprelocation ##2\tabularnewline
15031             }%
15032             \renewcommand{\subglossentry}[3]{%
15033                 &
15034                 \glssubentryitem{##2}%
15035                 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
15036                 \glstrprelocation ##3\tabularnewline
15037             }%
15038             \ifglsnogroupskip
15039                 \renewcommand*\glsgroupskip{}%

```

```

15040 \else
15041 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
15042 \fi
15043 }
15044 }
15045 {}

```

Three column style:

```

15046 \ifcsdef{@glsstyle@long3col}
15047 {%
15048 \renewglossarystyle{long3col}{%
15049 \renewenvironment{theglossary}%
15050 {\begin{longtable}\lp{\glsdescwidth}p{\glspagelistwidth}}}%
15051 {\end{longtable}}}%
15052 \renewcommand*{\glossaryheader}{}%
15053 \renewcommand*{\glsgroupheading}[1]{}%
15054 \renewcommand{\glossentry}[2]{%
15055 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15056 \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
15057 }%
15058 \renewcommand{\subglossentry}[3]{%
15059 &
15060 \glssubentryitem{##2}%
15061 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15062 ##3\tabularnewline
15063 }%

```

Conditional needs to be outside of `\glsgroupskip` otherwise it can cause “Incomplete `\iftrue`” errors.

```

15064 \ifglsnogroupskip
15065 \renewcommand*{\glsgroupskip}{}%
15066 \else
15067 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
15068 \fi
15069 }
15070 }
15071 {}

```

Four column style:

```

15072 \ifcsdef{@glsstyle@long4col}
15073 {%
15074 \renewglossarystyle{long4col}{%
15075 \renewenvironment{theglossary}%
15076 {\begin{longtable}{l111}}}%
15077 {\end{longtable}}}%
15078 \renewcommand*{\glossaryheader}{}%
15079 \renewcommand*{\glsgroupheading}[1]{}%
15080 \renewcommand{\glossentry}[2]{%
15081 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15082 \glossentrydesc{##1}\glspostdescription &
15083 \glossentrysymbol{##1} &

```

```

15084     ##2\tabularnewline
15085 }%
15086 \renewcommand{\subglossentry}[3]{%
15087     &
15088     \glssubentryitem{##2}%
15089     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15090     \glossentrysymbol{##2} & ##3\tabularnewline
15091 }%

15092 \ifglsgroupskip
15093     \renewcommand*{\glsgroupskip}{}%
15094 \else
15095     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
15096 \fi
15097 }
15098 }
15099 {}

```

The styles in glossary-longbooktabs are all based on the styles in glossary-long, so no adjustments are needed for that package.

## 2.4 Long Ragged Styles

The three and four column styles require adjustment for the post-description hook, but not the two column styles. However, the two-column styles need to have `\space` replaced with `\glxtrprelocation`.

```

15100 \ifcsdef{@glstyle@longragged}
15101 {%
15102     \renewglossarystyle{longragged}{%
15103         \renewenvironment{theglossary}%
15104             {\begin{longtable}{l>{\raggedright}p{\glstdescwidth}}}%
15105             {\end{longtable}}%
15106         \renewcommand*{\glossaryheader}{}%
15107         \renewcommand*{\glsgroupheading}[1]{}%
15108         \renewcommand{\glossentry}[2]{%
15109             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15110             \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
15111             \tabularnewline
15112         }%
15113         \renewcommand{\subglossentry}[3]{%
15114             &
15115             \glssubentryitem{##2}%
15116             \glstarget{##2}{\strut}\glossentrydesc{##2}%
15117             \glspostdescription\glxtrprelocation ##3%
15118             \tabularnewline
15119         }%
15120         \ifglsgroupskip
15121             \renewcommand*{\glsgroupskip}{}%
15122         \else

```



```

15123     \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
15124     \fi
15125   }
15126 }
15127 {}

```

Three and four column styles don't use `\glsxtrprelocation` since the number list is in its own column.

```

15128 \ifcsdef{@glsstyle@longragged3col}
15129 {%
15130   \renewglossarystyle{longragged3col}{%
15131     \renewenvironment{theglossary}%
15132       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
15133         >{\raggedright}p{\glspagelistwidth}}}%
15134       {\end{longtable}}}%
15135     \renewcommand*{\glossaryheader}{}%
15136     \renewcommand*{\glsgroupheading}[1]{}%
15137     \renewcommand{\glossentry}[2]{%
15138       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15139       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
15140     }%
15141     \renewcommand{\subglossentry}[3]{%
15142       &
15143       \glssubentryitem{##2}%
15144       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15145       ##3\tabularnewline
15146     }%
15147     \ifglsnogroupskip
15148       \renewcommand*{\glsgroupskip}{}%
15149     \else
15150       \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
15151     \fi
15152   }
15153 }
15154 {}

```

Four column style:

```

15155 \ifcsdef{@glsstyle@altlongragged4col}
15156 {%
15157   \renewglossarystyle{altlongragged4col}{%
15158     \renewenvironment{theglossary}%
15159       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
15160         >{\raggedright}p{\glspagelistwidth}}}%
15161       {\end{longtable}}}%
15162     \renewcommand*{\glossaryheader}{}%
15163     \renewcommand*{\glsgroupheading}[1]{}%
15164     \renewcommand{\glossentry}[2]{%
15165       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15166       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &

```

```

15167     ##2\tabularnewline
15168 }%
15169 \renewcommand{\subglossentry}[3]{%
15170     &
15171     \glssubentryitem{##2}%
15172     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15173     \glossentrysymbol{##2} & ##3\tabularnewline
15174 }%

15175 \ifglsgroupskip
15176     \renewcommand*\{glsgroupskip}{}%
15177 \else
15178     \renewcommand*\{glsgroupskip}{& & \tabularnewline}%
15179 \fi
15180 }
15181 }
15182 {}

```

## 2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded \space changed to \glxtrprelocation.

```

15183 \ifcsdef{@glstyle@super}
15184 {%
15185     \renewglossarystyle{super}{%
15186         \renewenvironment{theglossary}%
15187             {\tablehead{}\tabletail}{%
15188                 \begin{supertabular}{lp{\glsdescwidth}}%
15189                 {\end{supertabular}}%
15190         \renewcommand*\{glossaryheader}{}%
15191         \renewcommand*\{glsgroupheading}[1]{}%
15192         \renewcommand{\glossentry}[2]{%
15193             \glssentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15194             \glossentrydesc{##1}\glspostdescription
15195             \glxtrprelocation ##2\tabularnewline
15196         }%
15197         \renewcommand{\subglossentry}[3]{%
15198             &
15199             \glssubentryitem{##2}%
15200             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
15201             \glxtrprelocation ##3\tabularnewline
15202         }%
15203         \ifglsgroupskip
15204             \renewcommand*\{glsgroupskip}{}%
15205         \else
15206             \renewcommand*\{glsgroupskip}{& \tabularnewline}%
15207         \fi
15208     }

```

```
15209 }
15210 {}
```

Three column style:

```
15211 \ifcsdef{@glsstyle@super3col}
15212 {%
15213   \renewglossarystyle{super3col}{%
15214     \renewenvironment{theglossary}%
15215       {\tablehead{}\tabletail}%
15216       \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
15217     {\end{supertabular}}%
15218     \renewcommand*{\glossaryheader}{}%
15219     \renewcommand*{\glsgroupheading}[1]{}%
15220     \renewcommand{\glossentry}[2]{%
15221       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15222       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
15223     }%
15224     \renewcommand{\subglossentry}[3]{%
15225       &
15226       \glssubentryitem{##2}%
15227       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15228       ##3\tabularnewline
15229     }%

15230   \ifglsnogroupskip
15231     \renewcommand*{\glsgroupskip}{}%
15232   \else
15233     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
15234   \fi
15235 }
15236 }
15237 {}
```

Four column styles:

```
15238 \ifcsdef{@glsstyle@super4col}
15239 {%
15240   \renewglossarystyle{super4col}{%
15241     \renewenvironment{theglossary}%
15242       {\tablehead{}\tabletail}%
15243       \begin{supertabular}{llll}}}%
15244     \end{supertabular}}%
15245     \renewcommand*{\glossaryheader}{}%
15246     \renewcommand*{\glsgroupheading}[1]{}%
15247     \renewcommand{\glossentry}[2]{%
15248       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15249       \glossentrydesc{##1}\glspostdescription &
15250       \glossentrysymbol{##1} & ##2\tabularnewline
15251     }%
15252     \renewcommand{\subglossentry}[3]{%
15253       &
```

```

15254     \glssubentryitem{##2}%
15255     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15256     \glossentrysymbol{##2} & ##3\tabularnewline
15257 }%

15258 \ifglsnogroupskip
15259     \renewcommand*{\glsgroupskip}{}%
15260 \else
15261     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
15262 \fi
15263 }
15264 }
15265 {}

```

## 2.6 Super Ragged Styles

The three and four column styles require adjustment for the post-description hook, but not the two column styles. However, the two-column styles need to have `\space` replaced with `\glxtrprelocation`.

```

15266 \ifcsdef{@glstyle@superragged}
15267 {%
15268     \renewglossarystyle{superragged}%
15269     \renewenvironment{theglossary}%
15270         {\tablehead{}\tabletail}%
15271         {\begin{supertabular}{l>{\raggedright}p{\glsdescwidth}}}%
15272         {\end{supertabular}}%
15273     \renewcommand*{\glossaryheader}{}%
15274     \renewcommand*{\glsgroupheading}[1]{}%
15275     \renewcommand{\glossentry}[2]{%
15276         \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15277         \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
15278         \tabularnewline
15279     }%
15280     \renewcommand{\subglossentry}[3]{%
15281         &
15282         \glssubentryitem{##2}%
15283         \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
15284         \glxtrprelocation ##3%
15285         \tabularnewline
15286     }%
15287     \ifglsnogroupskip
15288         \renewcommand*{\glsgroupskip}{}%
15289     \else
15290         \renewcommand*{\glsgroupskip}{& \tabularnewline}%
15291     \fi
15292 }
15293 }
15294 {}

```

Three column style:

```
15295 \ifcsdef{@glsstyle@superragged3col}
15296 {%
15297   \renewglossarystyle{superragged3col}{%
15298     \renewenvironment{theglossary}%
15299       {\tablehead{}\tabletail{}}%
15300       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}%
15301         >{\raggedright}p{\glspagelistwidth}}}%
15302       {\end{supertabular}}}%
15303   \renewcommand*{\glossaryheader}{}%
15304   \renewcommand*{\glsgroupheading}[1]{}%
15305   \renewcommand{\glossentry}[2]{%
15306     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15307     \glossentrydesc{##1}\glspostdescription &
15308     ##2\tabularnewline
15309   }%
15310   \renewcommand{\subglossentry}[3]{%
15311     &
15312     \glssubentryitem{##2}%
15313     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15314     ##3\tabularnewline
15315   }%

15316   \ifglsnogroupskip
15317     \renewcommand*{\glsgroupskip}{}%
15318   \else
15319     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
15320   \fi
15321 }
15322 }
15323 {}
```

Four columns:

```
15324 \ifcsdef{@glsstyle@altsuperragged4col}
15325 {%
15326   \renewglossarystyle{altsuperragged4col}{%
15327     \renewenvironment{theglossary}%
15328       {\tablehead{}\tabletail{}}%
15329       \begin{supertabular}{1>{\raggedright}p{\glsdescwidth}1%
15330         >{\raggedright}p{\glspagelistwidth}}}%
15331       {\end{supertabular}}}%
15332   \renewcommand*{\glossaryheader}{}%
15333   \renewcommand{\glossentry}[2]{%
15334     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
15335     \glossentrydesc{##1}\glspostdescription &
15336     \glossentrysymbol{##1} & ##2\tabularnewline
15337   }%
15338   \renewcommand{\subglossentry}[3]{%
15339     &
15340     \glssubentryitem{##2}%

```

```

15341      \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
15342      \glossentrysymbol{##2} & ##3\tabularnewline
15343  }%

15344  \ifglsnogroupskip
15345      \renewcommand*{\glsgroupskip}{}%
15346  \else
15347      \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
15348  \fi
15349  }
15350 }
15351 {}

```

## 2.7 Inline Style

The inline style is dealt with slightly differently. The `\glspostdescription` hook is actually in `\glspostinline`, which is called at the end of the glossary. The original definition of `\glspostinline` also includes a space, which is unnecessary. Here, instead of redefining the inline style, just redefine `\glspostinline` and `\glsinlinedescformat`.

```

15352 \ifdef{\@glsstyle@inline}
15353 {%
15354     \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}

    Just use \glxtrpostdescription instead of \glspostdescription.

15355     \renewcommand*{\glsinlinedescformat}[3]{%
15356         \space#1\glxtrpostdescription}
15357     \renewcommand*{\glsinlinesubdescformat}[3]{%
15358         #1\glxtrpostdescription}

```

The default settings don't show the location lists, so there's no adjustment for `\glxtrprelocation`.

```

15359 }
15360 {}

```

## 2.8 Tree Styles

Redefine both `\glstreenamefmt` and `\glstreegroupheaderfmt` in terms of `\glstreedefaultnamefmt` to make it easier to change both at the same time or only change one without affecting the other.

```

15361 \ifdef\glstreenamefmt
15362 {

```

```

\glstreedefaultnamefmt

```

```

15363     \newcommand{\glstreedefaultnamefmt}[1]{\textbf{#1}}

```

```

\glstreenamefmt

```

```

15364     \renewcommand{\glstreenamefmt}[1]{\glstreedefaultnamefmt{#1}}

```

egrouphaderfmt This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
15365 \def\glstreegroupheaderfmt#1{\glstreedefaultnamefmt{#1}}
```

eenavigationfmt This command was only introduced to glossary-tree v4.22, so it may not be defined.

```
15366 \def\glstreenavigationfmt#1{\glstreedefaultnamefmt{#1}}
```

lstreetPreHeader Takes the label as the first argument and title as the second argument so this can be modified to add a bookmark.

```
15367 \newcommand{\glstreetPreHeader}[2]{}

15368 }
15369 {}

    The index style is redefined so that the space before the number list isn't hard coded.
15370 \ifdef{\@glsstyle@index}
15371 {

treeprelocation The space before the number list for top-level entries. This is shared by the other tree styles.
```

```
15372 \newcommand*{\glstreetprelocation}{\glstxtrprelocation}
```

childprelocation The space before the number list for child entries. This is shared by the other tree styles.

```
15373 \newcommand*{\glstreetchildprelocation}{\glstreetprelocation}
```

Don't prohibit a page break at the start of a new group if there's no header.

lstreetgroupskip

```
15374 \newcommand{\glstreetgroupskip}{\indexspace}
```

groupheaderskip This doesn't include \@afterheading as it can cause interference with some styles.

```
15375 \newcommand{\glstreetgroupheaderskip}{\nopagebreak\glstreetgroupskip\nobreak}
```

Modify the index style.

```
15376 \renewglossarystyle{index}{%
15377   \renewenvironment{theglossary}%
15378     {\setlength{\parindent}{0pt}%
15379     \setlength{\parskip}{0pt plus 0.3pt}%
15380     \let\item\glstreetitem
15381     \let\subitem\glstreetsubitem
15382     \let\subsubitem\glstreetsubsubitem
15383   }%
15384   {\par}%
15385   \renewcommand*{\glossaryheader}{}%
15386   \renewcommand*{\glsgroupheading}[1]{}%
15387   \renewcommand*{\glossentry}[2]{%
15388     \item\glsentryitem{##1}%
15389     \glstreetnamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15390     \glstreetsymbol{##1}%
15391     \glstreetDescLoc{##1}{##2}%

```

```

15392 }%
15393 \renewcommand{\subglossentry}[3]{%
15394   \ifcase##1\relax
15395     \item
15396   \or
15397     \subitem
15398     \glssubentryitem{##2}%
15399   \else
15400     \subsubitem
15401   \fi
15402   \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
15403   \glstreechildsymbol{##2}%
15404   \glstreeChildDescLoc{##2}{##3}%
15405 }%
15406 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15407 }
15408 }
15409 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

15410 \ifdef{\@glsstyle@indexgroup}
15411 {%
15412   \renewglossarystyle{indexgroup}{%
15413     \setglossarystyle{index}%
15414     \renewcommand*{\glsgroupheading}[1]{%
15415       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
15416       \glstreePreHeader{##1}{\glxtr@grptitle}%
15417       \item\glstreegroupheaderfmt{\glxtr@grptitle}%
15418       \glstreegroupheaderskip\@afterheading
15419     }%
15420   }
15421 }
15422 {}

```

Similarly for indexhypergroup.

```

15423 \ifdef{\@glsstyle@indexhypergroup}
15424 {%
15425   \renewglossarystyle{indexhypergroup}{%
15426     \setglossarystyle{index}%
15427     \renewcommand*{\glossaryheader}{%
15428       \item\glstreenavigationfmt{\glsnavigation}%
15429       \glstreegroupheaderskip\@afterheading}%
15430     \renewcommand*{\glsgroupheading}[1]{%
15431       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
15432       \glstreePreHeader{##1}{\glxtr@grptitle}%
15433       \item\glstreegroupheaderfmt
15434         {\glsnavhypertarget{##1}{\glxtr@grptitle}}%
15435       \glstreegroupheaderskip\@afterheading}%
15436   }%
15437 }

```



```
15438 {}
```

Adjust tree style to remove hard coded space before number list.

```
15439 \ifdef{\@glsstyle@tree}
```

```
15440 {%
```

The original alttree style doesn't use \glstreepredesc but since v1.42 the modified style (below) has switched to using \glstreeDescLoc so provide an alternative that can be used with alttree.

sxtrtreepredesc

```
15441 \newcommand{\glsxtrtreepredesc}{\glstreepredesc}
```

reechildpredesc

```
15442 \newcommand{\glsxtrtreechildpredesc}{\glstreechildpredesc}
```

Provide a command for use with the tree styles that displays the pre-description separator, the description and post-description hook.

\glstreedesc

```
15443 \newcommand{\glstreedesc}[1]{%
```

```
15444 \glsxtrtreepredesc\glossentrydesc{#1}\glspostdescription
```

```
15445 }
```

\glstreeDescLoc

`\glstreeDescLoc{<label>}{<location>}`

This checks for the description and symbol. If both are missing, a different separator may be required. For example, a comma and space if there's no description or symbol but just a space if either of those fields are present.

```
15446 \newcommand{\glstreeDescLoc}[2]{%
```

```
15447 \ifglshasdesc{#1}%
```

```
15448 {\glstreedesc{#1}\glstreeprelocation}%
```

```
15449 {\ifglshassymbol{#1}{\glstreeprelocation}{\glstreeNoDescSymbolPreLocation}}%
```

```
15450 #2%
```

```
15451 }
```

mbolPreLocation

`\glstreeNoDescSymbolPreLocation`

```
15452 \newcommand{\glstreeNoDescSymbolPreLocation}{\space}
```

Similarly for the symbol.

\glstreesymbol

```
15453 \newcommand{\glstreesymbol}[1]{%
15454   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}%
15455 }%
```

And for the child entries:

lstreechilddesc

```
15456 \newcommand{\glstreechilddesc}[1]{%
15457   \glstxttreechildpredesc\glossentrydesc{#1}\glspostdescription
15458 }%
```

reeChildDescLoc

```
15459 \newcommand{\glstreeChildDescLoc}[2]{%
15460   \ifglshasdesc{#1}%
15461   {\glstreechilddesc{#1}\glstreechildprelocation}%
15462   {\ifglshassymbol{#1}{\glstreechildprelocation}%
15463    {\glstreeNoDescSymbolPreLocation}%
15464   }%
15465   #2%
15466 }%
```

treechildsymbol This just behaves in the same way as the top-level.

```
15467 \newcommand{\glstreechildsymbol}[1]{%
15468   \glstreesymbol{#1}%
15469 }%

15470 \renewglossarystyle{tree}{%
15471   \renewenvironment{theglossary}%
15472     {\setlength{\parindent}{0pt}%
15473     \setlength{\parskip}{0pt plus 0.3pt}}%
15474   {}%
15475   \renewcommand*{\glossaryheader}{}%
15476   \renewcommand*{\glsgroupheading}[1]{}%
15477   \renewcommand{\glossentry}[2]{%
15478     \hangindent0pt\relax
15479     \parindent0pt\relax
15480     \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15481     \glstreesymbol{##1}%
15482     \glstreeDescLoc{##1}{##2}\par
15483   }%
15484   \renewcommand{\subglossentry}[3]{%
15485     \hangindent##1\glstreeindent\relax
15486     \parindent##1\glstreeindent\relax
15487     \ifnum##1=1\relax
15488       \glssubentryitem{##2}%
15489     \fi
15490     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
15491     \glstreechildsymbol{##2}%

```

```

15492      \glstreeChildDescLoc{##2}{##3}\par
15493    }%
15494    \renewcommand*{\glsgroupskip}{\ifglsgnognogroupskip\else\glstreegroupskip\fi}%
15495  }%
15496 }
15497 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

15498 \ifdef{\@glsstyle@treegroup}
15499 {%
15500   \renewglossarystyle{treegroup}{%
15501     \setglossarystyle{tree}%
15502     \renewcommand{\glsgroupheading}[1]{%
15503       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15504       \glstreePreHeader{##1}{\glstr@grptitle}%
15505       \par\noindent\glstreegroupheaderfmt{\glstr@grptitle}%
15506       \glstreegroupheaderskip\@afterheading}%
15507   }
15508 }
15509 {}

```

Similarly for treehypergroup

```

15510 \ifdef{\@glsstyle@treehypergroup}
15511 {%
15512   \renewglossarystyle{treehypergroup}{%
15513     \setglossarystyle{tree}%
15514     \renewcommand*{\glossaryheader}{%
15515       \par\noindent\glstreenavigationfmt{\glsnavigation}%
15516       \glstreegroupheaderskip\@afterheading}%
15517     \renewcommand*{\glsgroupheading}[1]{%
15518       \glstrgetgrouptitle{##1}{\glstr@grptitle}%
15519       \glstreePreHeader{##1}{\glstr@grptitle}%
15520       \par\noindent
15521       \glstreegroupheaderfmt
15522       {\glsnavhypertarget{##1}{\glstr@grptitle}}}%
15523     \glstreegroupheaderskip\@afterheading}%
15524   }
15525 }
15526 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

15527 \ifdef{\@glsstyle@treenoname}
15528 {%

```

Provide a command for use with the treenoname styles that displays the pre-description separator, the description and post-description hook.

streenonamedesc

```

15529   \newcommand{\glstreenonamedesc}[1]{%
15530     \glstreepredesc\glossentrydesc{#1}\glspostdescription
15531   }%

```

Similarly for the symbol.

treenonamesymbol

```
15532 \newcommand{\glstreenonamesymbol}[1]{%
15533   \ifglshassymbol{#1}{\space(\glossentrysymbol{#1})}{}}%
15534 }%
```

treenonameDescLoc

```
15535 \newcommand{\glstreenonameDescLoc}[2]{%
15536   \glstreenonamedesc{#1}\glstreeprelocation#2%
15537 }
```

nonamechilddesc The child entry doesn't have the pre-description separator as the name isn't displayed.

```
15538 \newcommand{\glstreenonamechilddesc}[1]{%
15539   \glossentrydesc{#1}\glspostdescription
15540 }%
```

nameChildDescLoc

```
15541 \newcommand{\glstreenonameChildDescLoc}[2]{%
15542   \glstreenonamechilddesc{#1}\glstreechildprelocation#2%
15543 }

15544 \renewglossarystyle{treenoname}{%
15545   \renewenvironment{theglossary}%
15546     {\setlength{\parindent}{0pt}%
15547     \setlength{\parskip}{0pt plus 0.3pt}}%
15548   {%
15549     \renewcommand*{\glossaryheader}{}%
15550     \renewcommand*{\glsgroupheading}[1]{}%
15551     \renewcommand{\glossentry}[2]{%
15552       \hangindent0pt\relax
15553       \parindent0pt\relax
15554       \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
15555       \glstreenonamesymbol{##1}%

15556       \glstreenonameDescLoc{##1}{##2}\par
15557     }%
15558     \renewcommand{\subglossentry}[3]{%
15559       \hangindent##1\glstreeindent\relax
15560       \parindent##1\glstreeindent\relax
15561       \ifnum##1=1\relax
15562         \glssubentryitem{##2}%
15563       \fi
15564       \glstarget{##2}{\strut}%
15565       \glstreenonameChildDescLoc{##2}{##3}\par
15566     }%
15567     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\glstreegroupskip\fi}%
15568   }
15569 }
15570 {}
```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

15571 \ifdef{\@glsstyle@treenonamegroup}
15572 {%
15573   \renewglossarystyle{treenonamegroup}{%
15574     \setglossarystyle{treenoname}%
15575     \renewcommand{\glsgroupheading}[1]{%
15576       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
15577       \glstreePreHeader{##1}{\glxtr@grptitle}%
15578       \par\noindent\glstreegroupheaderfmt{\glxtr@grptitle}%
15579       \glstreegroupheaderskip\@afterheading
15580     }%
15581   }
15582 }
15583 {}

```

Similarly for `treenonamehypergroup`

```

15584 \ifdef{\@glsstyle@treenonamehypergroup}
15585 {%
15586   \renewglossarystyle{treenonamehypergroup}{%
15587     \setglossarystyle{treenoname}%
15588     \renewcommand*\{\glossaryheader}{%
15589       \par\noindent\glstreenavigationfmt{\glsnavigation}%
15590       \glstreegroupheaderskip\@afterheading}%
15591     \renewcommand*\{\glsgroupheading}[1]{%
15592       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
15593       \glstreePreHeader{##1}{\glxtr@grptitle}%
15594       \par\noindent
15595       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}}%
15596       \glstreegroupheaderskip\@afterheading}%
15597   }
15598 }
15599 {}

```

The `almtree` style is redefined to make it easier to make minor adjustments.

```

15600 \ifdef{\@glsstyle@almtree}
15601 {%

```

Only redefine this style if it's already been defined.

`salttreepredesc`

```

15602 \newcommand{\glsalttreepredesc}{}

```

`treechildpredesc`

```

15603 \newcommand{\glsalttreechildpredesc}{\glsalttreepredesc}

```

`bolDescLocation`

<code>\glxtralmtreeSymbolDescLocation{&lt;label&gt;}{&lt;location list&gt;}</code>
--

Layout the symbol, description and location for top-level entries.

```

15604 \newcommand{\glxstralttreeSymbolDescLocation}[2]{%
15605   {%
15606     \let\par\glxstrAltTreePar

15607     \let\glxstrtreepredesc\glsalttreepredesc
15608     \let\glxstrtreechildpredesc\glsalttreechildpredesc
15609     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%

15610     \glstreeDescLoc{#1}{#2}\par
15611   }%
15612 }

```

**trAltTreeIndent** Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```
15613 \newlength\glxstrAltTreeIndent
```

**lsxstrAltTreePar** Multi-paragraph descriptions need to keep the hanging indent.

```

15614 \newcommand{\glxstrAltTreePar}{%
15615   \@@par
15616   \glxstrAltTreeSetHangIndent
15617   \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
15618 }

```

**bolDescLocation**

`\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

15619 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
15620   \glxstralttreeSymbolDescLocation{#2}{#3}%
15621 }

```

**trtreetopindent** The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```
15622 \newlength\glxstrtreetopindent
```

**sxstralttreeInit** User-level initialisation for the alttree style.

```

15623 \newcommand*{\glxstralttreeInit}{%
15624   \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
15625   \glxstrAltTreeIndent=\parindent
15626 }

```

**\gglsetwidest** The original `\glssetwidest` only uses `\def`. This uses `\gdef`.

```

15627 \newcommand*{\gglsetwidest}[2][0]{%
15628   \csgdef{@glswidestname\romannumeral#1}{#2}%
15629 }

```

`\eglssetwidest` The original `\glssetwidest` only uses `\def`. This uses `\protected@csedef`.

```
15630 \newcommand*{\eglssetwidest}[2][0]{%
15631   \protected@csedef{\glswidestname\romannumeral#1}{#2}%
15632 }
```

`\xglssetwidest` Like the above but uses `\protected@csxdef`.

```
15633 \newcommand*{\xglssetwidest}[2][0]{%
15634   \protected@csxdef{\glswidestname\romannumeral#1}{#2}%
15635 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
15636 \newcommand*{\glsupdatewidest}[2][0]{%
15637   \ifcsundef{\glswidestname\romannumeral#1}%
15638   {\csdef{\glswidestname\romannumeral#1}{#2}}%
15639   {%
15640     \settowidth{\dimen@}{\csuse{\glswidestname\romannumeral#1}}%
15641     \settowidth{\dimen@ii}{#2}%
15642     \ifdim\dimen@ii>\dimen@
15643     \csdef{\glswidestname\romannumeral#1}{#2}%
15644     \fi
15645   }%
15646 }
```

`glsupdatewidest` As above but global definition.

```
15647 \newcommand*{\gglsupdatewidest}[2][0]{%
15648   \ifcsundef{\glswidestname\romannumeral#1}%
15649   {\csgdef{\glswidestname\romannumeral#1}{#2}}%
15650   {%
15651     \settowidth{\dimen@}{\csuse{\glswidestname\romannumeral#1}}%
15652     \settowidth{\dimen@ii}{#2}%
15653     \ifdim\dimen@ii>\dimen@
15654     \csgdef{\glswidestname\romannumeral#1}{#2}%
15655     \fi
15656   }%
15657 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
15658 \newcommand*{\eglsupdatewidest}[2][0]{%
15659   \ifcsundef{\glswidestname\romannumeral#1}%
15660   {\protected@csedef{\glswidestname\romannumeral#1}{#2}}%
15661   {%
15662     \settowidth{\dimen@}{\csuse{\glswidestname\romannumeral#1}}%
15663     \settowidth{\dimen@ii}{#2}%
15664     \ifdim\dimen@ii>\dimen@
15665     \protected@csedef{\glswidestname\romannumeral#1}{#2}%
15666     \fi
15667   }%
15668 }
```

`glsupdatewidest` As above but global.

```
15669 \newcommand*{\xglsupdatewidest}[2][0]{%
15670   \ifcsundef{@glswidestname\romannumeral#1}%
15671   {\protected@csxdef{@glswidestname\romannumeral#1}{#2}}%
15672   {%
15673     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
15674     \settowidth{\dimen@ii}{#2}%
15675     \ifdim\dimen@ii>\dimen@
15676     \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
15677     \fi
15678   }%
15679 }
```

`lsgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```
15680 \newcommand*{\glsgetwidestname}{\@glswidestname}
```

`etwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```
15681 \newcommand*{\glsgetwidestsubname}[1]{%
15682   \ifcsundef{@glswidestname\romannumeral#1}%
15683   {\@glswidestname}%
15684   {\csuse{@glswidestname\romannumeral#1}}%
15685 }
```

`estTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glsfindwidesttoplevelname`

```
15686 \let\glsFindWidestTopLevelName\glsfindwidesttoplevelname
```

`sedTopLevelName` Like `\glsfindwidesttoplevelname` but has an additional check that the entry has been used. Only useful if the glossaries occur at the end of the document, in which case this command should go at the start of the glossary. Alternatively, place at the end of the document and save for the next run.

```
15687 \newrobustcmd*{\glsFindWidestUsedTopLevelName}[1][\@glo@types]{%
15688   \dimen@=0pt\relax
15689   \gls@tmplen=0pt\relax
15690   \forallglossaries[#1]{\@gls@type}%
15691   {%
15692     \forglentries[\@gls@type]{\@glo@label}%
15693     {%
15694       \ifglsused{\@glo@label}%
15695       {%
15696         \ifglshasparent{\@glo@label}%
15697         {}%
15698         {%
15699           \settowidth{\dimen@}%
15700             {\glstreenamfmt{\glstryname{\@glo@label}}}%
15701           \ifdim\dimen@>\gls@tmplen
15702             \gls@tmplen=\dimen@
15703             \eglssetwidest{\glstryname{\@glo@label}}%
15704           \fi

```



```

15705         }%
15706     }%
15707     {}%
15708     }%
15709     }%
15710 }

```

**destUsedAnyName** Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

15711 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
15712     \dimen@=0pt\relax
15713     \gls@tmplen=0pt\relax
15714     \foralllglossaries[#1]{\@gls@type}%
15715     {%
15716         \forallgsentries[\@gls@type]{\@glo@label}%
15717         {%
15718             \ifglsused{\@glo@label}%
15719             {%
15720                 \settowidth{\dimen@}%
15721                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15722                 \ifdim\dimen@>\gls@tmplen
15723                     \gls@tmplen=\dimen@
15724                     \eglssetwidest{\glsentryname{\@glo@label}}%
15725             }%
15726         }%
15727     }%
15728 }%
15729 }%
15730 }

```

**ndWidestAnyName** Like the above but doesn't check is the entry has been used.

```

15731 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
15732     \dimen@=0pt\relax
15733     \gls@tmplen=0pt\relax
15734     \foralllglossaries[#1]{\@gls@type}%
15735     {%
15736         \forallgsentries[\@gls@type]{\@glo@label}%
15737         {%
15738             \settowidth{\dimen@}%
15739             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15740             \ifdim\dimen@>\gls@tmplen
15741                 \gls@tmplen=\dimen@
15742                 \eglssetwidest{\glsentryname{\@glo@label}}%
15743             }%
15744         }%
15745     }%
15746 }

```

estUsedLevelTwo This is like \glsFindWidestUsedTopLevelName but also sets the first two sub-levels as well.

Any entry that has a great-grandparent is ignored.

```

15747 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
15748   \dimen@=0pt\relax
15749   \dimen@i=0pt\relax
15750   \dimen@ii=0pt\relax
15751   \forallglossaries[#1]{\@gls@type}%
15752   {%
15753     \forallglsentries[\@gls@type]{\@glo@label}%
15754     {%
15755       \ifglsused{\@glo@label}%
15756       {%
15757         \ifglsashasparent{\@glo@label}%
15758         {%
15759           \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel}{\@glo@label}@parent}}%
15760           \ifglsashasparent{\@glo@parent}%
15761           {%
15762             \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel}{\@glo@parent}@parent}}%
15763             \ifglsashasparent{\@glo@parent}%
15764             {%
15765               {%
15766                 \settowidth{\gls@tmplen}%
15767                   {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15768                 \ifdim\gls@tmplen>\dimen@ii
15769                   \dimen@ii=\gls@tmplen
15770                   \eglssetwidest[2]{\glsentryname{\@glo@label}}%
15771                 \fi
15772               }%
15773             }%
15774           {%
15775             \settowidth{\gls@tmplen}%
15776               {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15777             \ifdim\gls@tmplen>\dimen@i
15778               \dimen@i=\gls@tmplen
15779               \eglssetwidest[1]{\glsentryname{\@glo@label}}%
15780             \fi
15781           }%
15782         }%
15783       {%
15784         \settowidth{\gls@tmplen}%
15785           {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15786         \ifdim\gls@tmplen>\dimen@
15787           \dimen@=\gls@tmplen
15788           \eglssetwidest{\glsentryname{\@glo@label}}%
15789         \fi
15790       }%
15791     }%
15792   }%

```

```

15793     }%
15794     }%
15795 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

15796 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
15797     \dimen@=0pt\relax
15798     \dimen@i=0pt\relax
15799     \dimen@ii=0pt\relax
15800     \foralllglossaries[#1]{\@gls@type}%
15801     {%
15802         \forglsentries[\@gls@type]{\@glo@label}%
15803         {%
15804             \ifglshasparent{\@glo@label}%
15805             {%
15806                 \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
15807                 \ifglshasparent{\@glo@parent}%
15808                 {%
15809                     \protected@edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
15810                     \ifglshasparent{\@glo@parent}%
15811                     {}%
15812                     {%
15813                         \settowidth{\gls@tmplen}%
15814                             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15815                         \ifdim\gls@tmplen>\dimen@ii
15816                             \dimen@ii=\gls@tmplen
15817                             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
15818                         \fi
15819                     }%
15820                 }%
15821             }%
15822             \settowidth{\gls@tmplen}%
15823                 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15824             \ifdim\gls@tmplen>\dimen@i
15825                 \dimen@i=\gls@tmplen
15826                 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
15827             \fi
15828         }%
15829     }%
15830     {%
15831         \settowidth{\gls@tmplen}%
15832             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15833         \ifdim\gls@tmplen>\dimen@
15834             \dimen@=\gls@tmplen
15835             \eglssetwidest{\glsentryname{\@glo@label}}%
15836         \fi
15837     }%
15838 }%
15839 }%

```

15840 }

edAnyNameSymbol Like the \glsFindWidestUsedAnyName but also measures the symbol. The length of the widest symbol is stored in the second argument should be a length register.

```
15841 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
15842   \dimen@=0pt\relax
15843   \gls@tmplen=0pt\relax
15844   #2=0pt\relax
15845   \forallglossaries[#1]{\@gls@type}%
15846   {%
15847     \forallglsentries[\@gls@type]{\@glo@label}%
15848     {%
15849       \ifglsused{\@glo@label}%
15850       {%
15851         \settowidth{\dimen@}%
15852           {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15853         \ifdim\dimen@>\gls@tmplen
15854           \gls@tmplen=\dimen@
15855           \eglssetwidest{\glsentryname{\@glo@label}}%
15856         \fi
15857         \settowidth{\dimen@}%
15858           {\glsentrysymbol{\@glo@label}}%
15859         \ifdim\dimen@>#2\relax
15860           #2=\dimen@
15861         \fi
15862       }%
15863     }%
15864   }%
15865 }%
15866 }
```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```
15867 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
15868   \dimen@=0pt\relax
15869   \gls@tmplen=0pt\relax
15870   #2=0pt\relax
15871   \forallglossaries[#1]{\@gls@type}%
15872   {%
15873     \forallglsentries[\@gls@type]{\@glo@label}%
15874     {%
15875       \settowidth{\dimen@}%
15876         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
15877       \ifdim\dimen@>\gls@tmplen
15878         \gls@tmplen=\dimen@
15879         \eglssetwidest{\glsentryname{\@glo@label}}%
15880       \fi
15881       \settowidth{\dimen@}%
15882         {\glsentrysymbol{\@glo@label}}%
15883       \ifdim\dimen@>#2\relax
```

```

15884         #2=\dimen@
15885     \fi
15886 }%
15887 }%
15888 }

```

**eSymbolLocation** Like the `\glsFindWidestUsedAnyNameSymbol` but also measures the location list. This requires `\glsentrynumberlist`. The length of the widest symbol is stored in the second argument should be a length register. The length of the widest location list is stored in the third argument, which should also be a length register.

```

15889 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
15890     \dimen@=0pt\relax
15891     \gls@tmplen=0pt\relax
15892     #2=0pt\relax
15893     #3=0pt\relax
15894     \foralllglossaries[#1]{\@gls@type}%
15895     {%
15896         \forglseentries[\@gls@type]{\@glo@label}%
15897         {%
15898             \ifglused{\@glo@label}%
15899             {%
15900                 \settowidth{\dimen@}%
15901                 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15902                 \ifdim\dimen@>\gls@tmplen
15903                     \gls@tmplen=\dimen@
15904                     \eglssetwidest{\glsentryname{\@glo@label}}%
15905                 \fi
15906                 \settowidth{\dimen@}%
15907                 {\glsentrysymbol{\@glo@label}}%
15908                 \ifdim\dimen@>#2\relax
15909                     #2=\dimen@
15910                 \fi
15911                 \settowidth{\dimen@}%
15912                 {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
15913                 \ifdim\dimen@>#3\relax
15914                     #3=\dimen@
15915                 \fi
15916             }%
15917         }%
15918     }%
15919 }%
15920 }

```

**eSymbolLocation** Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

15921 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
15922     \dimen@=0pt\relax
15923     \gls@tmplen=0pt\relax
15924     #2=0pt\relax
15925     #3=0pt\relax

```

```

15926 \forall glossaries[#1]{\@gls@type}%
15927 {%
15928   \forall sentries[\@gls@type]{\@glo@label}%
15929   {%
15930     \settowidth{\dimen@}%
15931     {\glstreenamfmt{\glstryname{\@glo@label}}}%
15932     \ifdim\dimen@>\gls@tmplen
15933       \gls@tmplen=\dimen@
15934       \eglssetwidest{\glstryname{\@glo@label}}%
15935     \fi
15936     \settowidth{\dimen@}%
15937     {\glstryymbol{\@glo@label}}%
15938     \ifdim\dimen@>#2\relax
15939       #2=\dimen@
15940     \fi
15941     \settowidth{\dimen@}%
15942     {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
15943     \ifdim\dimen@>#3\relax
15944       #3=\dimen@
15945     \fi
15946   }%
15947 }%
15948 }

```

**AnyNameLocation** Like the `\glsFindWidestUsedAnyNameSymbolLocation` but doesn't measure the symbol. The length of the widest location list is stored in the second argument, which should be a length register.

```

15949 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
15950   \dimen@=0pt\relax
15951   \gls@tmplen=0pt\relax
15952   #2=0pt\relax
15953   \forall glossaries[#1]{\@gls@type}%
15954   {%
15955     \forall sentries[\@gls@type]{\@glo@label}%
15956     {%
15957       \ifglsused{\@glo@label}%
15958       {%
15959         \settowidth{\dimen@}%
15960         {\glstreenamfmt{\glstryname{\@glo@label}}}%
15961         \ifdim\dimen@>\gls@tmplen
15962           \gls@tmplen=\dimen@
15963           \eglssetwidest{\glstryname{\@glo@label}}%
15964         \fi
15965         \settowidth{\dimen@}%
15966         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
15967         \ifdim\dimen@>#2\relax
15968           #2=\dimen@
15969         \fi
15970       }%
15971     }%
15972   }%

```

```

15971      {}%
15972    }%
15973  }%
15974 }

```

**AnyNameLocation** Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

15975 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
15976   \dimen@=0pt\relax
15977   \gls@tmplen=0pt\relax
15978   #2=0pt\relax
15979   \forallglossaries[#1]{\@gls@type}%
15980   {%
15981     \forglsentries[\@gls@type]{\@glo@label}%
15982     {%
15983       \settowidth{\dimen@}%
15984         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
15985       \ifdim\dimen@>\gls@tmplen
15986         \gls@tmplen=\dimen@
15987         \eglssetwidest{\glsentryname{\@glo@label}}%
15988       \fi
15989       \settowidth{\dimen@}%
15990         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
15991       \ifdim\dimen@>#2\relax
15992         #2=\dimen@
15993       \fi
15994     }%
15995   }%
15996 }

```

**computeTreeIndent** Compute the value of `\glstreeindent`. Argument is the entry label. (Ignored in default definition, but this command may be redefined to take the particular entry into account.)  
Note that the sub-levels modify `\glstreeindent`.

```

15997 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
15998   \glstreeindent=\glsxtrtreetopindent\relax
15999 }

```

**teTreeSubIndent**

`\glsxtrComputeTreeSubIndent{<level>}{<label>}{<register>}`

Compute the indent for the sub-entries. The first argument is the level, the second argument is the entry label and the third argument is the length register used to store the computed indent.

```

16000 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
16001   \ifcsundef{@glswidestname\romannumeral#1}%
16002   {%
16003     \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%

```

```

16004 }%
16005 {%
16006     \settowidth{#3}{\glstreenamefmt{%
16007         \csname @glswidestname\romannumeral#1\endcsname\space}}%
16008 }%
16009 }

```

eeSetHangIndent Set \hangindent for top-level entries:

```

16010 \newcommand*{\glxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```

16011 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}

```

Redefine alttree:

```

16012 \renewglossarystyle{alttree}{%
16013     \renewenvironment{theglossary}%
16014     {%
16015         \glxtralttreeInit
16016         \def\@gls@prevlevel{-1}%
16017         \mbox{}\par}%
16018     {\par}%
16019     \renewcommand*{\glossaryheader}{}%
16020     \renewcommand*{\glsgroupheading}[1]{}%
16021     \renewcommand{\glossentry}[2]{%
16022         \ifnum\@gls@prevlevel=0\relax
16023         \else
16024             \glxtrComputeTreeIndent{##1}%
16025         \fi
16026         \parindent\glstreeindent
16027         \glxtrAltTreeSetHangIndent
16028         \makebox[0pt][r]%
16029         {%
16030             \glstreenamebox{\glstreeindent}%
16031             {%
16032                 \glssentryitem{##1}%
16033                 \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
16034             }%
16035         }%
16036         \glxtralttreeSymbolDescLocation{##1}{##2}%
16037         \def\@gls@prevlevel{0}%
16038     }
16039     \renewcommand{\subglossentry}[3]{%
16040         \ifnum##1=1\relax
16041             \glssubentryitem{##2}%
16042         \fi
16043         \ifnum\@gls@prevlevel=##1\relax
16044         \else
16045             \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
16046             \ifnum\@gls@prevlevel<##1\relax

```



```

16047         \setlength\glstreeindent\gls@tmplen
16048         \addtolength\glstreeindent\parindent
16049         \parindent\glstreeindent
16050     \else
16051         \ifnum\@gls@prevlevel=0\relax
16052             \glstrComputeTreeIndent{##2}%
16053         \else
16054             \glstrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
16055         \fi
16056         \addtolength\parindent{-\glstreeindent}%
16057         \setlength\glstreeindent\parindent
16058     \fi
16059 \fi
16060 \glstrAltTreeSetSubHangIndent{##1}%
16061 \makebox[0pt][r]{\glstreenamebox{\gls@tmplen}{%
16062     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
16063 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
16064 \def\@gls@prevlevel{##1}%
16065 }%
16066 \renewcommand*\@glsgroupskip{\ifglsnogroupskip\else\glstreegroupskip\fi}%
16067 }
16068 }%
16069 {%
16070 }

```

Redefine `alttreegroup` so that it discourages a break after group headings.

```

16071 \ifdef{\@glsstyle@alttreegroup}
16072 {%
16073     \renewglossarystyle{alttreegroup}{%
16074         \setglossarystyle{alttree}%
16075         \renewcommand{\glsgroupheading}[1]{\par
16076             \def\@gls@prevlevel{-1}%
16077             \hangindent0pt\relax
16078             \parindent0pt\relax
16079             \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16080             \glstreePreHeader{##1}{\glstr@grptitle}%
16081             \glstreegroupheaderfmt{\glstr@grptitle}%

```

Can't use `\@afterheading` here as it messes with the first item of the group.

```

16082         \glstreegroupheaderskip
16083     }%
16084 }%
16085 }%
16086 {%
16087 }

```

Similarly for `alttreehypergroup`.

```

16088 \ifdef{\@glsstyle@alttreehypergroup}
16089 {%
16090     \renewglossarystyle{alttreehypergroup}{%
16091         \setglossarystyle{alttree}%

```

```

16092 \renewcommand*{\glossaryheader}{%
16093 \par
16094 \def\@gls@prevlevel{-1}%
16095 \hangindent0pt\relax
16096 \parindent0pt\relax
16097 \glstreenavigationfmt{\glsnavigation}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

16098 \glstreegroupheaderskip
16099 }%
16100 \renewcommand*{\glsgroupheading}[1]{%
16101 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16102 \glstreePreHeader{##1}{\glstr@grptitle}%
16103 \par
16104 \def\@gls@prevlevel{-1}%
16105 \hangindent0pt\relax
16106 \parindent0pt\relax
16107 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glstr@grptitle}}%

```

Can't use \@afterheading here as it messes with the first item of the group.

```

16108 \glstreegroupheaderskip
16109 }%
16110 }
16111 }%
16112 {%
16113 }

```

## 2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

16114 \ifdef{\@glsstyle@mcolindexgroup}
16115 {%
16116 \renewglossarystyle{mcolindexgroup}{%
16117 \setglossarystyle{mcolindex}%
16118 \renewcommand*{\glsgroupheading}[1]{%
16119 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16120 \glstreePreHeader{##1}{\glstr@grptitle}%
16121 \item\glstreegroupheaderfmt{\glstr@grptitle}%
16122 \glstreegroupheaderskip\@afterheading
16123 }%
16124 }
16125 }%
16126 {%
16127 }

```

Similarly for mcolindexhypergroup.

```

16128 \ifdef{\@glsstyle@mcolindexhypergroup}
16129 {%
16130 \renewglossarystyle{mcolindexhypergroup}{%

```

```

16131 \setglossarystyle{mcolindex}%
16132 \renewcommand*{\glossaryheader}{%
16133 \item\glstreenavigationfmt{\glsnavigation}%

16134 \glstreegroupheaderskip\@afterheading
16135 }%
16136 \renewcommand*{\glsgroupheading}[1]{%
16137 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16138 \glstreePreHeader{##1}{\glstr@grptitle}%
16139 \item\glstreegroupheaderfmt
16140 {\glsnavhypertarget{##1}{\glstr@grptitle}}}%
16141 \glstreegroupheaderskip\@afterheading
16142 }%
16143 }
16144 }%
16145 {%
16146 }

```

Similarly for mcolindexspannav.

```

16147 \ifdef{\@glsstyle@mcolindexspannav}
16148 {%
16149 \renewglossarystyle{mcolindexspannav}{%
16150 \setglossarystyle{index}%
16151 \renewenvironment{theglossary}%
16152 {%
16153 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]}%
16154 \setlength{\parindent}{0pt}%
16155 \setlength{\parskip}{0pt plus 0.3pt}%
16156 \let\item\glstreeitem}%
16157 {\end{multicols}}}%
16158 \renewcommand*{\glsgroupheading}[1]{%
16159 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16160 \glstreePreHeader{##1}{\glstr@grptitle}%
16161 \item\glstreegroupheaderfmt
16162 {\glsnavhypertarget{##1}{\glstr@grptitle}}}%
16163 \glstreegroupheaderskip\@afterheading
16164 }%
16165 }
16166 }%
16167 {%
16168 }

```

Similarly for mcoltreegroup.

```

16169 \ifdef{\@glsstyle@mcoltreegroup}
16170 {%
16171 \renewglossarystyle{mcoltreegroup}{%
16172 \setglossarystyle{mcoltree}%
16173 \renewcommand{\glsgroupheading}[1]{%
16174 \glstrgetgrouptitle{##1}{\glstr@grptitle}%
16175 \glstreePreHeader{##1}{\glstr@grptitle}%

```

```

16176     \par\noindent\glstreegroupheaderfmt{\glxtr@grptitle}%
16177     \glstreegroupheaderskip\@afterheading
16178 }%
16179 }
16180 }%
16181 {%
16182 }

```

Similarly for mcoltreehypergroup.

```

16183 \ifdef{\@glsstyle@mcoltreehypergroup}
16184 {%
16185   \renewglossarystyle{mcoltreehypergroup}{%
16186     \setglossarystyle{mcoltree}%
16187     \renewcommand*\glossaryheader{%
16188       \par\noindent\glstreenavigationfmt{\glsnavigation}%
16189       \glstreegroupheaderskip
16190     }%
16191     \renewcommand*\glsgroupheading}[1]{%
16192       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16193       \glstreePreHeader{##1}{\glxtr@grptitle}%
16194       \par\noindent
16195       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16196       \glstreegroupheaderskip\@afterheading
16197     }%
16198   }
16199 }%
16200 {%
16201 }

```

Similarly for mcoltreespannav.

```

16202 \ifdef{\@glsstyle@mcoltreespannav}
16203 {%
16204   \renewglossarystyle{mcoltreespannav}{%
16205     \setglossarystyle{tree}%
16206     \renewenvironment{theglossary}%
16207     {%
16208       \begin{multicols}{\glsmcols}%
16209       [\noindent\glstreenavigationfmt{\glsnavigation}]%
16210       \setlength{\parindent}{0pt}%
16211       \setlength{\parskip}{0pt plus 0.3pt}%
16212     }%
16213     {\end{multicols}}%
16214     \renewcommand*\glsgroupheading}[1]{%
16215       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16216       \glstreePreHeader{##1}{\glxtr@grptitle}%
16217       \par\noindent
16218       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16219       \glstreegroupheaderskip\@afterheading
16220     }%
16221   }

```

```

16222 }%
16223 {%
16224 }

```

Similarly for mcoltreenonamegroup.

```

16225 \ifdef{\@glsstyle@mcoltreenonamegroup}
16226 {%
16227   \renewglossarystyle{mcoltreenonamegroup}{%
16228     \setglossarystyle{mcoltreenoname}%
16229     \renewcommand{\glsgroupheading}[1]{%
16230       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16231       \glstreePreHeader{##1}{\glxtr@grptitle}%
16232       \par\noindent\glstreegroupheaderfmt{\glxtr@grptitle}%
16233       \glstreegroupheaderskip\@afterheading
16234     }%
16235   }
16236 }%
16237 {%
16238 }

```

Similarly for mcoltreenonamehypergroup.

```

16239 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
16240 {%
16241   \renewglossarystyle{mcoltreenonamehypergroup}{%
16242     \setglossarystyle{mcoltreenoname}%
16243     \renewcommand*\{\glossaryheader}{%
16244       \par\noindent\glstreenavigationfmt{\glsnavigation}%
16245       \glstreegroupheaderskip
16246     }%
16247     \renewcommand*\{\glsgroupheading}[1]{%
16248       \glxtrgetgrouptitle{##1}{\glxtr@grptitle}%
16249       \glstreePreHeader{##1}{\glxtr@grptitle}%
16250       \par\noindent
16251       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glxtr@grptitle}}%
16252       \glstreegroupheaderskip\@afterheading}%
16253   }
16254 }%
16255 {%
16256 }

```

Similarly for mcoltreenonamespannav.

```

16257 \ifdef{\@glsstyle@mcoltreenonamespannav}
16258 {%
16259   \renewglossarystyle{mcoltreenonamespannav}{%
16260     \setglossarystyle{treenoname}%
16261     \renewenvironment{theglossary}%
16262     {%
16263       \begin{multicols}{\glsmcols}%
16264       [\noindent\glstreenavigationfmt{\glsnavigation}]%
16265       \setlength{\parindent}{0pt}%
16266       \setlength{\parskip}{0pt plus 0.3pt}%

```

```

16267 }%
16268 {\end{multicols}}}%
16269 \renewcommand*{\glsgroupheading}[1]{%
16270   \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
16271   \glstreePreHeader{##1}{\glsxtr@grptitle}%
16272   \par\noindent
16273   \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsxtr@grptile}}}%
16274   \glstreegroupheaderskip\@afterheading}%
16275 }
16276 }%
16277 {%
16278 }

```

`mcolalmtree` needs adjusting so that it uses `\glsxtralmtreeInit` This doesn't use `\mbox{}`\par which would unbalance the top of the columns.

```

16279 \ifdef{\@glsstyle@mcolalmtree}
16280 {%
16281   \renewglossarystyle{mcolalmtree}{%
16282     \setglossarystyle{almtree}%
16283     \renewenvironment{theglossary}%
16284     {%
16285       \glsxtralmtreeInit
16286       \def\@gls@prevlevel{-1}%
16287       \begin{multicols}{\glscols}%
16288     }%
16289     {\par\end{multicols}}}%
16290   }
16291 }%
16292 {%
16293 }

```

Redefine `mcolalmtreegroup` to discourage page breaks after the group headings.

```

16294 \ifdef{\@glsstyle@mcolalmtreegroup}
16295 {%
16296   \renewglossarystyle{mcolalmtreegroup}{%
16297     \setglossarystyle{mcolalmtree}%
16298     \renewcommand{\glsgroupheading}[1]{%
16299       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
16300       \glstreePreHeader{##1}{\glsxtr@grptitle}%
16301       \par
16302       \def\@gls@prevlevel{-1}%
16303       \hangindentOpt\relax
16304       \parindentOpt\relax
16305       \glstreegroupheaderfmt{\glsxtr@grptitle}%
16306       \glstreegroupheaderskip
16307     }%
16308   }
16309 }%
16310 {%
16311 }

```

Similarly for mcolalttreehypergroup.

```

16312 \ifdef{\@glsstyle@mcolalttreehypergroup}
16313 {%
16314   \renewglossarystyle{mcolalttreehypergroup}{%
16315     \setglossarystyle{mcolalttree}%
16316     \renewcommand*{\glossaryheader}{%
16317       \par
16318       \def\@gls@prevlevel{-1}%
16319       \hangindentOpt\relax
16320       \parindentOpt\relax
16321       \glstreenavigationfmt{\glsnavigation}%
16322       \glstreegroupheaderskip
16323     }%
16324     \renewcommand*{\glsgroupheading}[1]{%
16325       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
16326       \glstreePreHeader{##1}{\glsxtr@grptitle}%
16327       \par
16328       \def\@gls@prevlevel{-1}%
16329       \hangindentOpt\relax
16330       \parindentOpt\relax
16331       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsxtr@grptitle}}%
16332       \glstreegroupheaderskip
16333     }%
16334   }
16335 }%
16336 {%
16337 }

```

Similarly for mcolalttreespannav.

```

16338 \ifdef{\@glsstyle@mcolalttreespannav}
16339 {%
16340   \renewglossarystyle{mcolalttreespannav}{%
16341     \setglossarystyle{alttree}%
16342     \renewenvironment{theglossary}%
16343     {%
16344       \glxtralttreeInit
16345       \def\@gls@prevlevel{-1}%
16346       \begin{multicols}{\glsmcols}%
16347         [\noindent\glstreenavigationfmt{\glsnavigation}]%
16348       }%
16349       {\par\end{multicols}}%
16350     \renewcommand*{\glsgroupheading}[1]{%
16351       \glsxtrgetgrouptitle{##1}{\glsxtr@grptitle}%
16352       \glstreePreHeader{##1}{\glsxtr@grptitle}%
16353       \par
16354       \def\@gls@prevlevel{-1}%
16355       \hangindentOpt\relax
16356       \parindentOpt\relax
16357       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsxtr@grptitle}}%
16358       \glstreegroupheaderskip

```

```

16359     }%
16360   }
16361 }%
16362 {%
16363 }

      Reset the default style
16364 \ifx\@glossary@default@style\relax
16365 \else
16366   \setglossarystyle{\@glxtr@current@style}
16367 \fi

```



## 3 bookindex style (glossary-bookindex.sty)

### 3.1 Package Initialisation and Options

```
16368 \NeedsTeXFormat{LaTeX2e}
16369 \ProvidesPackage{glossary-bookindex}[2021/11/04 v1.47 (NLCT)]
```

Load required packages.

```
16370 \RequirePackage{multicol}
16371 \RequirePackage{glossary-tree}
```

trbookindexcols    Number of columns.

```
16372 \newcommand{\glstrbookindexcols}{2}
```

trbookindexname    Format used for top-level entries. (Argument is the label.)

```
16373 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

ookindexsubname    Format used for sub entries.

```
16374 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

xsxtrprelocation    Provide in case glossaries-stylemods isn't loaded.

```
16375 \providecommand*{\glsxtrprelocation}{\space}
```

ndexprelocation    Separator used before location list for top-level entries. Version 1.22 has removed the  
                    \ifglsnopostdot check since this style doesn't display the description.

```
16376 \newcommand*{\glsxtrbookindexprelocation}[1]{%
16377    \glsxtrifhasfield{location}{#1}%
16378    {,\glsxtrprelocation}%
16379    {\glsxtrprelocation}%
16380 }
```

xsubprelocation    Separator used before location list for sub-entries.

```
16381 \newcommand*{\glsxtrbookindexsubprelocation}[1]{%
16382    \glsxtrbookindexprelocation{#1}%
16383 }
```

okindexlocation

```
\glsxtrbookindexlocation{\label}{\location}
```

Displays the location.

```
16384 \newcommand*{\glsxtrbookindexlocation}[2]{#2}
```

ndexsublocation

`\glxtrbookindexlocation{<label>}{<location>}`

Displays the location for sub-entries.

```
16385 \newcommand*{\glxtrbookindexsublocation}{\glxtrbookindexlocation}
```

xparentchildsep Separator used between top-level parent and child entry.

```
16386 \newcommand{\glxtrbookindexparentchildsep}{\nopagebreak}
```

rentsubchildsep Separator used between sub-level parent and child entry.

```
16387 \newcommand{\glxtrbookindexparentsubchildsep}{\glxtrbookindexparentchildsep}
```

ookindexbetween Between two top-level entries identified by the labels in the arguments.

```
16388 \newcommand{\glxtrbookindexbetween}[2]{}%
```

indexsubbetween Between two level 1 entries identified by the labels in the arguments.

```
16389 \newcommand{\glxtrbookindexsubbetween}[2]{}%
```

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.

```
16390 \newcommand{\glxtrbookindexsubsubbetween}[2]{}%
```

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.

```
16391 \newcommand{\glxtrbookindexatendgroup}[1]{}%
```

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.

```
16392 \newcommand{\glxtrbookindexsubatendgroup}[1]{}%
```

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.

```
16393 \newcommand{\glxtrbookindexsubsubatendgroup}[1]{}%
```

kindexgroupskip Group separator.

```
16394 \newcommand{\glxtrbookindexgroupskip}{\ifglxnogroupskip\else\indexspace\fi}
```

Format group title.

dexformatheader Group separator.

```
16395 \newcommand*{\glxtrbookindexformatheader}[1]{%
16396   \par{\centering\glstreegroupheaderfmt{#1}\par}%
16397 }
```

okindexbookmark Book mark group heading if supported.

```
16398 \ifdef\pdfbookmark
16399 {%
16400   \newcommand*{\glxtrbookindexbookmark}[2]{%
16401     \ifdefstring{@@glossarysec}{chapter}%
16402     {\pdfbookmark[1]{#1}{#2}}%
16403   }%
16404 }
```

```

16403     {\pdfbookmark[2]{#1}{#2}}%
16404   }
16405 }
16406 {%
16407   \newcommand*{\glxtrbookindexbookmark}[2]{%
16408 }

```

**bookmarkprefix** Make the bookmark label prefix used for letter groups depend on the glossary label (instead of original hardcoded “index.”).

```

16409 \newcommand*{\glxtrbookindexbookmarkprefix}{\currentglossary.}

```

**indexcolspread**

```

16410 \newcommand*{\glxtrbookindexcolspread}{}

```

**dexmulticolseenv**

```

16411 \newcommand*{\glxtrbookindexmulticolseenv}{multicols}

```

Define the style.

```

16412 \newglossarystyle{bookindex}{%
16413   \setglossarystyle{index}%
16414   \renewenvironment{theglossary}%
16415   {%
16416     \ifnum\glxtrbookindexcols>1\relax
16417       \ifdefempty\glxtrbookindexcolspread
16418       {%
16419         \edef\glxtr@beginbookindex{%
16420           \noexpand\begin{\glxtrbookindexmulticolseenv}
16421             {\glxtrbookindexcols}%
16422         }%
16423       }%
16424     {%
16425       \edef\glxtr@beginbookindex{%
16426         \noexpand\begin{\glxtrbookindexmulticolseenv}%
16427           {\glxtrbookindexcols}{\glxtrbookindexcolspread}%
16428       }%
16429     }%
16430   \else
16431     \def\glxtr@beginbookindex{}%
16432   \fi
16433   \glxtr@beginbookindex
16434   \setlength{\parindent}{0pt}%
16435   \setlength{\parskip}{0pt plus 0.3pt}%
16436   \let\@glxtr@bookindex@sep\glxtrbookindexparentchildsep
16437   \let\@glxtr@bookindex@subsep\glxtrbookindexparentsubchildsep
16438   \let\@glxtr@bookindex@between\@gobble
16439   \let\@glxtr@bookindex@subbetween\@gobble
16440   \let\@glxtr@bookindex@subsubbetween\@gobble
16441   \let\@glxtr@bookindex@atendgroup\relax
16442   \let\@glxtr@bookindex@subatendgroup\relax

```

```

16443 \let\@glsxtr@bookindex@subsubatendgroup\relax
16444 \let\@glsxtr@bookindexgroupskip\relax
16445 }%
16446 {%

```

Do end group hooks.

```

16447 \@glsxtr@bookindex@subsubatendgroup
16448 \@glsxtr@bookindex@subatendgroup
16449 \@glsxtr@bookindex@atendgroup

```

End multicol environment.

```

16450 \ifnum\glsxtrbookindexcols>1\relax
16451 \edef\glsxtr@endbookindex{%
16452 \noexpand\end{\glsxtrbookindexmulticolenv}%
16453 }%
16454 \else
16455 \def\glsxtr@endbookindex{%
16456 \fi
16457 \glsxtr@endbookindex
16458 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

16459 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

16460 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

16461 \@glsxtr@bookindex@between{##1}%

```

Update separators.

```

16462 \let\@glsxtr@bookindex@sep\glsxtrbookindexparentchildsep
16463 \let\@glsxtr@bookindex@subsep\glsxtrbookindexparentsubchildsep
16464 \let\@glsxtr@bookindex@subbetween\@gobble
16465 \let\@glsxtr@bookindex@subsubbetween\@gobble
16466 \edef\@glsxtr@bookindex@between{%
16467 \noexpand\glsxtrbookindexbetween{##1}%
16468 }%
16469 \edef\@glsxtr@bookindex@atendgroup{%
16470 \noexpand\glsxtrbookindexatendgroup{##1}%
16471 }%
16472 \let\@glsxtr@bookindex@subatendgroup\relax
16473 \let\@glsxtr@bookindex@subsubatendgroup\relax

```

Format entry.

```

16474 \glstreeitem
16475 \glsentryitem{##1}%
16476 \glstarget{##1}{\glsxtrbookindexname{##1}}%
16477 \glsxtrbookindexprelocation{##1}%
16478 \glsxtrbookindexlocation{##1}{##2}%
16479 }%
16480 \renewcommand{\subglossentry}[3]{%
16481 \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```
16482      \glstreeitem
16483      \or
```

Level 1.

```
16484      \@glstr@bookindex@sep
16485      \@glstr@bookindex@subbetween{##2}%
16486      \let\@glstr@bookindex@sep\relax
```

Update separators.

```
16487      \let\@glstr@bookindex@subsubbetween\@gobble
16488      \let\@glstr@bookindex@subsep\glstrbookindexparentsubchildsep
16489      \edef\@glstr@bookindex@subbetween{%
16490          \noexpand\glstrbookindexsubbetween{##2}%
16491      }%
16492      \edef\@glstr@bookindex@atsubendgroup{%
16493          \noexpand\glstrbookindexatsubendgroup{##1}%
16494      }%
```

Start sub-item.

```
16495      \glstreesubitem
16496      \glssubentryitem{##2}%
16497      \else
```

All other levels.

```
16498      \@glstr@bookindex@subsep
16499      \@glstr@bookindex@subsubbetween{##2}%
```

Update separators.

```
16500      \let\@glstr@bookindex@subsep\relax
16501      \edef\@glstr@bookindex@subsubbetween{%
16502          \noexpand\glstrbookindexsubsubbetween{##2}%
16503      }%
16504      \edef\@glstr@bookindex@atsubsubendgroup{%
16505          \noexpand\glstrbookindexatsubsubendgroup{##1}%
16506      }%
```

Start sub-sub-item.

```
16507      \glstreesubsubitem
16508      \fi
```

Format entry.

```
16509      \glstarget{##2}{\glstrbookindexsubname{##2}}%
16510      \glstrbookindexsubprelocation{##2}%
16511      \glstrbookindexsublocation{##2}{##3}%
16512      }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
16513      \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
16514      \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
16515 \glxtr@bookindex@subsubatendgroup
16516 \glxtr@bookindex@subatendgroup
16517 \glxtr@bookindex@atendgroup
16518 \glxtr@bookindexgroupskip
```

Update separators.

```
16519 \let\glxtr@bookindexgroupskip\glxtrbookindexgroupskip
16520 \let\glxtr@bookindex@between\@gobble
16521 \let\glxtr@bookindex@atendgroup\relax
16522 \let\glxtr@bookindex@subatendgroup\relax
16523 \let\glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
16524 \glxtrgetgrouptitle{##1}{\glxtrcurrentgrptitle}%
```

Do the PDF bookmark if supported.

```
16525 \glxtrbookindexbookmark{\glxtrcurrentgrptitle}{\glxtrbookindexbookmarkprefix##1}%
```

Format the group title.

```
16526 \glxtrbookindexformatheader{\glxtrcurrentgrptitle}%
16527 \nopagebreak\indexspace\nopagebreak\@afterheading
16528 }%
16529 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glxtrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`\glxtrbookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
16530 \newcommand{\glxtrbookindexthepage}{%
16531 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
16532 }
```

`\glxtrbookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
16533 \newcommand*{\glxtrbookindexmarkentry}[1]{%
16534 \protected@write\@auxout
16535 {\let\glxtrbookindexthepage\relax}%
16536 {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
16537 }
```

`\glxtr@setbookindexmark`

```
16538 \newcommand*{\glxtr@setbookindexmark}[2]{%
16539 \ifcsundef\glxtr@idxfirstmark@#1}%
16540 {\csgdef\glxtr@idxfirstmark@#1}{#2}}%
16541 {}%
16542 \csgdef\glxtr@idxlastmark@#1}{#2}%
16543 }
```

dexfirstmarkfmt

```
16544 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
16545   \glstryname{#1}%
16546 }
```

kindexfirstmark

```
16547 \newcommand*{\glxtrbookindexfirstmark}{%
16548   \letcs{\glxtr@label}{\glxtr@idxfirstmark@\glxtrbookindexthepage}%
16549   \ifdef\glxtr@label
16550     {\glxtrbookindexfirstmarkfmt{\glxtr@label}}%
16551   {}%
16552 }
```

ndexlastmarkfmt

```
16553 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
16554   \glstryname{#1}%
16555 }
```

okindexlastmark

```
16556 \newcommand*{\glxtrbookindexlastmark}{%
16557   \letcs{\glxtr@label}{\glxtr@idxlastmark@\glxtrbookindexthepage}%
16558   \ifdef\glxtr@label
16559     {\glxtrbookindexlastmarkfmt{\glxtr@label}}%
16560   {}%
16561 }
```

## 4 longextra styles (glossary-longextra.sty)

### 4.1 Package Initialisation and Options

Provides additional long styles.

```
16562 \NeedsTeXFormat{LaTeX2e}  
16563 \ProvidesPackage{glossary-longextra}[2021/11/04 v1.47 (NLCT)]
```

Load required packages.

```
16564 \RequirePackage{glossary-longbooktabs}
```

ongextraNameFmt

```
\glslongextraNameFmt{<label>}
```

Governs the way the name is displayed.

```
16565 \newcommand{\glslongextraNameFmt}[1]{%  
16566   \glsentryitem{#1}\glstarget{#1}{\glossentryname{#1}}%  
16567 }
```

ongextraDescFmt

```
\glslongextraDescFmt{<label>}
```

Governs the way the description is displayed.

```
16568 \newcommand{\glslongextraDescFmt}[1]{%  
16569   \glossentrydesc{#1}\glspostdescription  
16570 }
```

gextraSymbolFmt

```
\glslongextraSymbolFmt{<label>}
```

Governs the way the symbol is displayed.

```
16571 \newcommand{\glslongextraSymbolFmt}[1]{\glossentrysymbol{#1}}
```



extraLocationFmt

```
\glslongextraLocationFmt{<label>}{<location list>}
```

Governs the way the location is displayed.

```
16572 \newcommand{\glslongextraLocationFmt}[2]{#2}
```

extraSubNameFmt

```
\glslongextraSubNameFmt{<level>}{<label>}
```

Governs the way the child name is displayed. Just does the sub-entry counter, if enabled, and the target.

```
16573 \newcommand{\glslongextraSubNameFmt}[2]{%
16574   \glssubentryitem{#2}\glstarget{#2}{\strut}%
16575 }
```

extraSubDescFmt

```
\glslongextraSubDescFmt{<level>}{<label>}
```

Governs the way the child description is displayed.

```
16576 \newcommand{\glslongextraSubDescFmt}[2]{%
16577   \glslongextraDescFmt{#2}%
16578 }
```

extraSubSymbolFmt

```
\glslongextraSubSymbolFmt{<level>}{<label>}
```

Governs the way the child symbol is displayed.

```
16579 \newcommand{\glslongextraSubSymbolFmt}[2]{%
16580   \glslongextraSymbolFmt{#2}%
16581 }
```

extraSubLocationFmt

```
\glslongextraSubLocationFmt{<level>}{<label>}{<location list>}
```

Governs the way the child location list is displayed.

```
16582 \newcommand{\glslongextraSubLocationFmt}[3]{#3}
```

`gextraNameAlign` Alignment for the name column.

```
16583 \newcommand{\glslongextraNameAlign}{l}
```

`gextraDescAlign` Alignment for the description column.

```
16584 \newcommand{\glslongextraDescAlign}{>{\raggedright}p{\glsdescwidth}}
```

`extraSymbolAlign` Alignment for the symbol column.

```
16585 \newcommand{\glslongextraSymbolAlign}{c}
```

`raLocationAlign` Alignment for the location column.

```
16586 \newcommand{\glslongextraLocationAlign}{>{\raggedright}p{\glspagelistwidth}}
```

`extraGroupHeading` Used to format the letter group headings. The first argument is the number of columns in the table. The second is the group *label* (not the title).

```
16587 \newcommand{\glslongextraGroupHeading}[2]{}
```

`extraHeaderFormat` Format for the column headers.

```
16588 \newcommand{\glslongextraHeaderFmt}[1]{\textbf{#1}}
```

`aNameDescHeader`

```
16589 \newcommand{\glslongextraNameDescHeader}{%
16590 \glslongextraNameDescTabularHeader\endhead
16591 \glslongextraNameDescTabularFooter\endfoot
16592 }
```

`scTabularHeader`

```
16593 \newcommand{\glslongextraNameDescTabularHeader}{%
16594 \toprule
16595 \glslongextraHeaderFmt\entryname &
16596 \glslongextraHeaderFmt\descriptionname\tabularnewline
16597 \midrule
16598 }
```

`scTabularFooter`

```
16599 \newcommand{\glslongextraNameDescTabularFooter}{%
16600 \bottomrule
16601 }
```

Unlike the `alttree` style, there aren't different widths for the hierarchical levels.

`gextraSetWidest` Provide in case the tree styles haven't been loaded.

```
16602 \newcommand*{\glslongextraSetWidest}[1]{%
16603 \def\@glslongextrawidestname{#1}%
16604 }
```

`extrawidestname` Pick up the widest name from the `alttree` style if it has been set. (Will expand to nothing otherwise.)

```
16605 \newcommand*{\@glslongextrawidestname}{\csuse{@glswidestname}}
```

traUpdateWidest

```

16606 \newcommand*{\glslongextraUpdateWidest}[1]{%
16607   \ifundef\@glslongextrawidestname
16608   {\def\@glslongextrawidestname{#1}}%
16609   {%
16610     \settowidth{\dimen@}{\@glslongextrawidestname}%
16611     \settowidth{\dimen@ii}{#1}%
16612     \ifdim\dimen@ii>\dimen@
16613     \def\@glslongextrawidestname{#1}%
16614     \fi
16615   }%
16616 }
```

dateWidestChild

`\glslongextraUpdateWidestChild{<level>}{<text>}`

Used by `\glsxtrSetWidest` in `glossaries-extra-bib2gls`. Does nothing by default, since the default action in these styles is to omit the child name. If the child name should be displayed, then this needs to be redefined to use `\glslongextraUpdateWidest`.

```

16617 \newcommand*{\glslongextraUpdateWidestChild}[2]{}
```

traSetDescWidth    Computes the value of `\glsdescwidth` for the styles that only have name and description columns.

```

16618 \newcommand{\glslongextraSetDescWidth}{%
16619   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\entryname}%

   Has the widest name been set.

16620   \settowidth{\dimen@}{\glsnamefont{\@glslongextrawidestname}}%
16621   \ifdim\dimen@>\gls@tmplen
16622     \gls@tmplen=\dimen@
16623   \fi
```

Description width is `\linewidth` less `4\tabcolsep` less the width of the name column.

```

16624   \setlength{\glsdescwidth}{\dimexpr\linewidth-4\tabcolsep-\gls@tmplen}%
16625 }
```

SymSetDescWidth    Computes the value of `\glsdescwidth` for the styles that only have name, symbol and description columns.

```

16626 \newcommand{\glslongextraSymSetDescWidth}{%
```

Work out the size for just the name and description style.

```

16627   \glslongextraSetDescWidth
```

Now work out the symbol column width. This is assuming that the column title will be the widest text in the column.

```

16628   \settowidth{\gls@tmplen}{\glslongextraHeaderFmt\symbolname}%
```

Subtract 2\tabcolsep and the symbol header width.

```

16629 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\gls@tmplen}%
16630 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that only have name, location and description columns.

```

16631 \newcommand{\glslongextraLocSetDescWidth}{%
  Work out the size for just the name and description style.
16632 \glslongextraSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
16633 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
16634 }

```

LocSetDescWidth Computes the value of \glsdescwidth for the styles that have name, symbol, location and description columns.

```

16635 \newcommand{\glslongextraSymLocSetDescWidth}{%
  Work out the size for just the name, symbol and description style.
16636 \glslongextraSymSetDescWidth
  Subtract 2\tabcolsep and the location list column width.
16637 \setlength{\glsdescwidth}{\dimexpr\glsdescwidth-2\tabcolsep-\glspagelistwidth}%
16638 }

```

ExtraUseTabular If true use tabular instead of longtable. Obviously only intended for short glossaries that can fit into a single page.

```

16639 \newif\ifGlsLongExtraUseTabular
16640 \GlsLongExtraUseTabularfalse

```

raTabularVAlign Only used with the tabular setting.

```

16641 \newcommand*{\glslongextraTabularVAlign}{c}

```

long-name-desc Two column style with multi-lined descriptions and header. This is similar to the longragged-booktabs style.

```

16642 \newglossarystyle{long-name-desc}%
16643 {%
16644 \ifGlsLongExtraUseTabular
16645 \renewenvironment{theglossary}%
16646 {%
16647 \glslongextraSetDescWidth
16648 \edef\@glslongextra@begintab{%
16649 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
16650 \expandonce\glslongextraNameAlign
16651 \expandonce\glslongextraDescAlign}}%
16652 \@glslongextra@begintab
16653 }%
16654 {%

```

```

16655     \glslongextraNameDescTabularFooter
16656     \end{tabular}%
16657 }%
16658 \renewcommand*{\glossaryheader}{\glslongextraNameDescTabularHeader}%
16659 \else
16660 \renewenvironment{theglossary}%
16661 {%
16662     \glspatchLToutput
16663     \glslongextraSetDescWidth
16664     \edef\@glslongextra@begintab{%
16665         \noexpand\begin{longtable}{%
16666             \expandonce\glslongextraNameAlign
16667             \expandonce\glslongextraDescAlign}}%
16668     \@glslongextra@begintab
16669 }%
16670 {\end{longtable}}%
16671 \renewcommand*{\glossaryheader}{\glslongextraNameDescHeader}%
16672 \fi
16673 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
16674 \renewcommand{\glossentry}[2]{%
16675     \glslongextraNameFmt{##1} &
16676     \glslongextraDescFmt{##1}\tabularnewline
16677 }%
16678 \renewcommand{\subglossentry}[3]{%
16679     \glslongextraSubNameFmt{##1}{##2}
16680     &
16681     \glslongextraSubDescFmt{##1}{##2}%
16682     \tabularnewline
16683 }%
16684 \ifglsgroupskip
16685     \renewcommand*{\glsgroupskip}{}%
16686 \else
16687     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16688 \fi
16689 }

```

#### cLocationHeader

```

16690 \newcommand{\glslongextraNameDescLocationHeader}{%
16691     \glslongextraNameDescLocationTabularHeader\endhead
16692     \glslongextraNameDescLocationTabularFooter\endfoot
16693 }

```

#### onTabularHeader

```

16694 \newcommand{\glslongextraNameDescLocationTabularHeader}{%
16695     \toprule
16696     \glslongextraHeaderFmt\entryname &
16697     \glslongextraHeaderFmt\descriptionname &
16698     \glslongextraHeaderFmt\pagelistname\tabularnewline
16699     \midrule

```

16700 }

onTabularFooter

```
16701 \newcommand{\glslongextraNameDescLocationTabularFooter}{%
16702   \bottomrule
16703 }
```

g-name-desc-loc Three columns: name, description and location list.

```
16704 \newglossarystyle{long-name-desc-loc}%
16705 {%
16706   \ifGlsLongExtraUseTabular
16707     \renewenvironment{theglossary}%
16708       {%
16709         \glslongextraLocSetDescWidth
16710         \edef\@glslongextra@begintab{%
16711           \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16712             \expandonce\glslongextraNameAlign
16713             \expandonce\glslongextraDescAlign
16714             \expandonce\glslongextraLocationAlign
16715           }}%
16716         \@glslongextra@begintab
16717       }%
16718       {%
16719         \glslongextraNameDescLocationTabularFooter
16720         \end{tabular}%
16721       }%
16722     \renewcommand*\glossaryheader{\glslongextraNameDescLocationTabularHeader}%
16723   \else
16724     \renewenvironment{theglossary}%
16725       {%
16726         \glspatchLToutput
16727         \glslongextraLocSetDescWidth
16728         \edef\@glslongextra@begintab{%
16729           \noexpand\begin{longtable}%
16730             \expandonce\glslongextraNameAlign
16731             \expandonce\glslongextraDescAlign
16732             \expandonce\glslongextraLocationAlign
16733           }}%
16734         \@glslongextra@begintab
16735       }%
16736       {\end{longtable}}%
16737     \renewcommand*\glossaryheader{\glslongextraNameDescLocationHeader}%
16738   \fi
16739   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
16740   \renewcommand{\glossentry}[2]{%
16741     \glslongextraNameFmt{##1} &
16742     \glslongextraDescFmt{##1} &
16743     \glslongextraLocationFmt{##1}{##2}\tabularnewline
16744   }%
```

```

16745 \renewcommand{\subglossentry}[3]{%
16746     \glslongextraSubNameFmt{##1}{##2}&
16747     \glslongextraSubDescFmt{##1}{##2}&
16748     \glslongextraSubLocationFmt{##1}{##2}{##3}%
16749     \tabularnewline
16750 }%
16751 \ifglsgroupskip
16752 \renewcommand*{\glsgroupskip}{}%
16753 \else
16754 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16755 \fi
16756 }

```

aDescNameHeader

```

16757 \newcommand{\glslongextraDescNameHeader}{%
16758 \glslongextraDescNameTabularHeader\endhead
16759 \glslongextraDescNameTabularFooter\endfoot
16760 }

```

meTabularHeader

```

16761 \newcommand{\glslongextraDescNameTabularHeader}{%
16762 \toprule
16763 \glslongextraHeaderFmt\descriptionname&
16764 \glslongextraHeaderFmt\entryname \tabularnewline
16765 \midrule
16766 }

```

meTabularFooter

```

16767 \newcommand{\glslongextraDescNameTabularFooter}{%
16768 \bottomrule
16769 }

```

long-desc-name Like name-desc but swaps the columns.

```

16770 \newglossarystyle{long-desc-name}%
16771 {%
16772 \ifGlsLongExtraUseTabular
16773 \renewenvironment{theglossary}%
16774 {%
16775     \glslongextraSetDescWidth
16776     \edef\@glslongextra@begintab{%
16777         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16778             \expandonce\glslongextraDescAlign
16779             \expandonce\glslongextraNameAlign}}%
16780     \@glslongextra@begintab
16781 }%
16782 {%
16783     \glslongextraDescNameTabularFooter
16784     \end{tabular}%
16785 }%

```

```

16786 \renewcommand*{\glossaryheader}{\glslongextraDescNameTabularHeader}%
16787 \else
16788 \renewenvironment{theglossary}%
16789 {%
16790 \glspatchLToutput
16791 \glslongextraSetDescWidth
16792 \edef\@glslongextra@begintab{%
16793 \noexpand\begin{longtable}{%
16794 \expandonce\glslongextraDescAlign
16795 \expandonce\glslongextraNameAlign}}%
16796 \@glslongextra@begintab
16797 }%
16798 {\end{longtable}}%
16799 \renewcommand*{\glossaryheader}{\glslongextraDescNameHeader}%
16800 \fi
16801 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{2}{##1}}%
16802 \renewcommand{\glossentry}[2]{%
16803 \glslongextraDescFmt{##1} &
16804 \glslongextraNameFmt{##1}\tabularnewline
16805 }%
16806 \renewcommand{\subglossentry}[3]{%
16807 \glslongextraSubDescFmt{##1}{##2} &
16808 \glslongextraSubNameFmt{##1}{##2}\tabularnewline
16809 }%
16810 \ifglsgroupskip
16811 \renewcommand*{\glsgroupskip}{}%
16812 \else
16813 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16814 \fi
16815 }

```

nDescNameHeader

```

16816 \newcommand{\glslongextraLocationDescNameHeader}{%
16817 \glslongextraLocationDescNameTabularHeader\endhead
16818 \glslongextraLocationDescNameTabularFooter\endfoot
16819 }

```

meTabularHeader

```

16820 \newcommand{\glslongextraLocationDescNameTabularHeader}{%
16821 \toprule
16822 \glslongextraHeaderFmt\pagelistname&
16823 \glslongextraHeaderFmt\descriptionname&
16824 \glslongextraHeaderFmt\entryname \tabularnewline
16825 \midrule
16826 }

```

meTabularFooter

```

16827 \newcommand{\glslongextraLocationDescNameTabularFooter}{%
16828 \bottomrule

```



16829 }

g-loc-desc-name Three columns: location, description and name.

```
16830 \newglossarystyle{long-loc-desc-name}%
16831 {%
16832   \ifGlsLongExtraUseTabular
16833     {%
16834       \glslongextraLocSetDescWidth
16835       \edef\@glslongextra@begintab{%
16836         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16837           \expandonce\glslongextraLocationAlign
16838           \expandonce\glslongextraDescAlign
16839           \expandonce\glslongextraNameAlign}}%
16840       \@glslongextra@begintab
16841     }%
16842     {%
16843       \glslongextraLocationDescNameTabularFooter
16844       \end{tabular}%
16845     }%
16846     \renewcommand*\glossaryheader{\glslongextraLocationDescNameTabularHeader}%
16847   \else
16848     \renewenvironment{theglossary}%
16849     {%
16850       \glspatchLToutput
16851       \glslongextraLocSetDescWidth
16852       \edef\@glslongextra@begintab{%
16853         \noexpand\begin{longtable}{%
16854           \expandonce\glslongextraLocationAlign
16855           \expandonce\glslongextraDescAlign
16856           \expandonce\glslongextraNameAlign}}%
16857       \@glslongextra@begintab
16858     }%
16859     {\end{longtable}}%
16860     \renewcommand*\glossaryheader{\glslongextraLocationDescNameHeader}%
16861   \fi
16862   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{3}{##1}}%
16863   \renewcommand{\glossentry}[2]{%
16864     \glslongextraLocationFmt{##1}{##2} &
16865     \glslongextraDescFmt{##1} &
16866     \glslongextraNameFmt{##1}\tabularnewline
16867   }%
16868   \renewcommand{\subglossentry}[3]{%
16869     \glslongextraSubLocationFmt{##1}{##2}{##3} &
16870     \glslongextraSubDescFmt{##1}{##2} &
16871     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
16872   }%
16873   \ifglsgnogroupskip
16874     \renewcommand*\glsgroupskip{}%
16875   \else
```

```

16876 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16877 \fi
16878 }

```

meDescSymHeader

```

16879 \newcommand{\glslongextraNameDescSymHeader}{%
16880 \glslongextraNameDescSymTabularHeader\endhead
16881 \glslongextraNameDescSymTabularFooter\endfoot
16882 }

```

ymTabularHeader

```

16883 \newcommand{\glslongextraNameDescSymTabularHeader}{%
16884 \toprule
16885 \glslongextraHeaderFmt\entryname &
16886 \glslongextraHeaderFmt\descriptionname &
16887 \glslongextraHeaderFmt\symbolname\tabularnewline
16888 \midrule
16889 }

```

ymTabularFooter

```

16890 \newcommand{\glslongextraNameDescSymTabularFooter}{%
16891 \bottomrule
16892 }

```

g-name-desc-sym Three column style with symbol in the third column.

```

16893 \newglossarystyle{long-name-desc-sym}%
16894 {%
16895 \ifGlsLongExtraUseTabular
16896 \renewenvironment{theglossary}%
16897 {%
16898 \glslongextraSymSetDescWidth
16899 \edef\@glslongextra@begintab{%
16900 \noexpand\begin{tabular}{\glslongextraTabularVAlign}{%
16901 \expandonce\glslongextraNameAlign
16902 \expandonce\glslongextraDescAlign
16903 \expandonce\glslongextraSymbolAlign
16904 }}%
16905 \@glslongextra@begintab
16906 }%
16907 {%
16908 \glslongextraNameDescSymTabularFooter
16909 \end{tabular}%
16910 }%
16911 \renewcommand*{\glossaryheader}{\glslongextraNameDescSymTabularHeader}%
16912 \else
16913 \renewenvironment{theglossary}%
16914 {%
16915 \glspatchLToutput
16916 \glslongextraSymSetDescWidth

```

```

16917 \edef\@glslongextra@begintab{%
16918 \noexpand\begin{longtable}{%
16919 \expandonce\glslongextraNameAlign
16920 \expandonce\glslongextraDescAlign
16921 \expandonce\glslongextraSymbolAlign
16922 }}%
16923 \@glslongextra@begintab
16924 }%
16925 {\end{longtable}}}%
16926 \renewcommand*{\glossaryheader}{\glslongextraNameDescSymHeader}%
16927 \fi
16928 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
16929 \renewcommand{\glossentry}[2]{%
16930 \glslongextraNameFmt{##1} &
16931 \glslongextraDescFmt{##1} &
16932 \glslongextraSymbolFmt{##1}\tabularnewline
16933 }%
16934 \renewcommand{\subglossentry}[3]{%
16935 \glslongextraSubNameFmt{##1}{##2} &
16936 \glslongextraSubDescFmt{##1}{##2} &
16937 \glslongextraSubSymbolFmt{##1}{##2}%
16938 \tabularnewline
16939 }%
16940 \ifglsgnognroupskip
16941 \renewcommand*{\glsgroupskip}{}%
16942 \else
16943 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
16944 \fi
16945 }

```

mLocationHeader

```

16946 \newcommand{\glslongextraNameDescSymLocationHeader}{%
16947 \glslongextraNameDescSymLocationTabularHeader\endhead
16948 \glslongextraNameDescSymLocationTabularFooter\endfoot
16949 }

```

onTabularHeader

```

16950 \newcommand{\glslongextraNameDescSymLocationTabularHeader}{%
16951 \toprule
16952 \glslongextraHeaderFmt\entryname &
16953 \glslongextraHeaderFmt\descriptionname &
16954 \glslongextraHeaderFmt\symbolname &
16955 \glslongextraHeaderFmt\pagelistname\tabularnewline
16956 \midrule
16957 }

```

onTabularFooter

```

16958 \newcommand{\glslongextraNameDescSymLocationTabularFooter}{%
16959 \bottomrule

```

16960 }

me-desc-sym-loc Four columns: name, description and location

```
16961 \newglossarystyle{long-name-desc-sym-loc}%
16962 {%
16963   \ifGlsLongExtraUseTabular
16964     \renewenvironment{theglossary}%
16965     {%
16966       \glslongextraSymLocSetDescWidth
16967       \edef\@glslongextra@begintab{%
16968         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
16969           \expandonce\glslongextraNameAlign
16970           \expandonce\glslongextraDescAlign
16971           \expandonce\glslongextraSymbolAlign
16972           \expandonce\glslongextraLocationAlign
16973         }}%
16974       \@glslongextra@begintab
16975     }%
16976     {%
16977       \glslongextraNameDescSymLocationTabularFooter
16978       \end{tabular}%
16979     }%
16980     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationTabularHeader}%
16981   \else
16982     \renewenvironment{theglossary}%
16983     {%
16984       \glspatchLToutput
16985       \glslongextraSymLocSetDescWidth
16986       \edef\@glslongextra@begintab{%
16987         \noexpand\begin{longtable}{%
16988           \expandonce\glslongextraNameAlign
16989           \expandonce\glslongextraDescAlign
16990           \expandonce\glslongextraSymbolAlign
16991           \expandonce\glslongextraLocationAlign
16992         }}%
16993       \@glslongextra@begintab
16994     }%
16995     {\end{longtable}}%
16996     \renewcommand*\glossaryheader{\glslongextraNameDescSymLocationHeader}%
16997   \fi
16998   \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{4}{##1}}%
16999   \renewcommand{\glossentry}[2]{%
17000     \glslongextraNameFmt{##1} &
17001     \glslongextraDescFmt{##1} &
17002     \glslongextraSymbolFmt{##1}&
17003     \glslongextraLocationFmt{##1}{##2}\tabularnewline
17004   }%
17005   \renewcommand{\subglossentry}[3]{%
17006     \glslongextraSubNameFmt{##1}{##2} &
```

```

17007 \glslongextraSubDescFmt{##1}{##2} &
17008 \glslongextraSubSymbolFmt{##1}{##2}&
17009 \glslongextraSubLocationFmt{##1}{##2}{##3}%
17010 \tabularnewline
17011 }%
17012 \ifglsnogroupskip
17013 \renewcommand*{\glsgroupskip}{}%
17014 \else
17015 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17016 \fi
17017 }

```

meSymDescHeader

```

17018 \newcommand{\glslongextraNameSymDescHeader}{%
17019 \glslongextraNameSymDescTabularHeader\endhead
17020 \glslongextraNameSymDescTabularFooter\endfoot
17021 }

```

scTabularHeader

```

17022 \newcommand{\glslongextraNameSymDescTabularHeader}{%
17023 \toprule
17024 \glslongextraHeaderFmt\entryname &
17025 \glslongextraHeaderFmt\symbolname &
17026 \glslongextraHeaderFmt\descriptionname\tabularnewline
17027 \midrule
17028 }

```

scTabularFooter

```

17029 \newcommand{\glslongextraNameSymDescTabularFooter}{%
17030 \bottomrule
17031 }

```

g-name-sym-desc Three column style with symbol in the second column.

```

17032 \newglossarystyle{long-name-sym-desc}%
17033 {%
17034 \ifGlsLongExtraUseTabular
17035 \renewenvironment{theglossary}%
17036 {%
17037 \glslongextraSymSetDescWidth
17038 \edef\@glslongextra@begintab{%
17039 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17040 \expandonce\glslongextraNameAlign
17041 \expandonce\glslongextraSymbolAlign
17042 \expandonce\glslongextraDescAlign
17043 }}%
17044 \@glslongextra@begintab
17045 }%
17046 {%
17047 \glslongextraNameSymDescTabularFooter

```

```

17048     \end{tabular}%
17049   }%
17050   \renewcommand*{\glossaryheader}{\glslongextraNameSymDescTabularHeader}%
17051   \else
17052     \renewenvironment{theglossary}%
17053     {%
17054       \glspatchLToutput
17055       \glslongextraSymSetDescWidth
17056       \edef\@glslongextra@begintab{%
17057         \noexpand\begin{longtable}{%
17058           \expandonce\glslongextraNameAlign
17059           \expandonce\glslongextraSymbolAlign
17060           \expandonce\glslongextraDescAlign
17061         }}%
17062       \@glslongextra@begintab
17063     }%
17064     {\end{longtable}}%
17065     \renewcommand*{\glossaryheader}{\glslongextraNameSymDescHeader}%
17066   \fi
17067   \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
17068   \renewcommand{\glossentry}[2]{%
17069     \glslongextraNameFmt{##1} &
17070     \glslongextraSymbolFmt{##1} &
17071     \glslongextraDescFmt{##1}\tabularnewline
17072   }%
17073   \renewcommand{\subglossentry}[3]{%
17074     \glslongextraSubNameFmt{##1}{##2} &
17075     \glslongextraSubSymbolFmt{##1}{##2} &
17076     \glslongextraSubDescFmt{##1}{##2}\tabularnewline
17077   }%
17078   \ifglsgroupskip
17079     \renewcommand*{\glsgroupskip}{}%
17080   \else
17081     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17082   \fi
17083 }

```

#### cLocationHeader

```

17084 \newcommand{\glslongextraNameSymDescLocationHeader}{%
17085   \glslongextraNameSymDescLocationTabularHeader\endhead
17086   \glslongextraNameSymDescLocationTabularFooter\endfoot
17087 }

```

#### onTabularHeader

```

17088 \newcommand{\glslongextraNameSymDescLocationTabularHeader}{%
17089   \toprule
17090   \glslongextraHeaderFmt\entryname &
17091   \glslongextraHeaderFmt\symbolname &
17092   \glslongextraHeaderFmt\descriptionname &

```

```

17093 \glslongextraHeaderFmt\pagelistname\tabularnewline
17094 \midrule
17095 }

```

onTabularFooter

```

17096 \newcommand{\glslongextraNameSymDescLocationTabularFooter}{%
17097 \bottomrule
17098 }

```

me-sym-desc-loc Four column style with symbol in the second column.

```

17099 \newglossarystyle{long-name-sym-desc-loc}%
17100 {%
17101 \ifGlsLongExtraUseTabular
17102 \renewenvironment{theglossary}%
17103 {%
17104 \glslongextraSymLocSetDescWidth
17105 \edef\@glslongextra@begintab{%
17106 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17107 \expandonce\glslongextraNameAlign
17108 \expandonce\glslongextraSymbolAlign
17109 \expandonce\glslongextraDescAlign
17110 \expandonce\glslongextraLocationAlign
17111 }}%
17112 \@glslongextra@begintab
17113 }%
17114 {%
17115 \glslongextraNameSymDescLocationTabularFooter
17116 \end{tabular}}%
17117 }%
17118 \renewcommand*\glossaryheader{\glslongextraNameSymDescLocationTabularHeader}%
17119 \else
17120 \renewenvironment{theglossary}%
17121 {%
17122 \glspatchLToutput
17123 \glslongextraSymLocSetDescWidth
17124 \edef\@glslongextra@begintab{%
17125 \noexpand\begin{longtable}{%
17126 \expandonce\glslongextraNameAlign
17127 \expandonce\glslongextraSymbolAlign
17128 \expandonce\glslongextraDescAlign
17129 \expandonce\glslongextraLocationAlign
17130 }}%
17131 \@glslongextra@begintab
17132 }%
17133 {\end{longtable}}%
17134 \renewcommand*\glossaryheader{\glslongextraNameSymDescLocationHeader}%
17135 \fi
17136 \renewcommand*\glsgroupheading[1]{\glslongextraGroupHeading{4}{##1}}%
17137 \renewcommand{\glossentry}[2]{%

```

```

17138 \glslongextraNameFmt{##1} &
17139 \glslongextraSymbolFmt{##1} &
17140 \glslongextraDescFmt{##1} &
17141 \glslongextraLocationFmt{##1}{##2}\tabularnewline
17142 }%
17143 \renewcommand{\subglossentry}[3]{%
17144 \glslongextraSubNameFmt{##1}{##2} &
17145 \glslongextraSubSymbolFmt{##1}{##2} &
17146 \glslongextraSubDescFmt{##1}{##2} &
17147 \glslongextraSubLocationFmt{##1}{##2}{##3}\tabularnewline
17148 }%
17149 \ifglsgnogroupskip
17150 \renewcommand*{\glsgroupskip}{}%
17151 \else
17152 \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17153 \fi
17154 }

```

mDescNameHeader

```

17155 \newcommand{\glslongextraSymDescNameHeader}{%
17156 \glslongextraSymDescNameTabularHeader\endhead
17157 \glslongextraSymDescNameTabularFooter\endfoot
17158 }

```

meTabularHeader

```

17159 \newcommand{\glslongextraSymDescNameTabularHeader}{%
17160 \toprule
17161 \glslongextraHeaderFmt\symbolname &
17162 \glslongextraHeaderFmt\descriptionname &
17163 \glslongextraHeaderFmt\entryname\tabularnewline
17164 \midrule
17165 }

```

meTabularFooter

```

17166 \newcommand{\glslongextraSymDescNameTabularFooter}{%
17167 \bottomrule
17168 }

```

g-sym-desc-name Three column style with symbol in the first column, description in the second and name in the third.

```

17169 \newglossarystyle{long-sym-desc-name}%
17170 {%
17171 \ifGlsLongExtraUseTabular
17172 \renewenvironment{theglossary}%
17173 {%
17174 \glslongextraSymSetDescWidth
17175 \edef\@glslongextra@begintab{%
17176 \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17177 \expandonce\glslongextraSymbolAlign

```



```

17178         \expandonce\glslongextraDescAlign
17179         \expandonce\glslongextraNameAlign
17180     } }%
17181     \@glslongextra@begintab
17182 }%
17183 {%
17184     \glslongextraSymDescNameTabularFooter
17185     \end{tabular}%
17186 }%
17187 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameTabularHeader}%
17188 \else
17189 \renewenvironment{theglossary}%
17190 {%
17191     \glspatchLToutput
17192     \glslongextraSymSetDescWidth
17193     \edef\@glslongextra@begintab{%
17194         \noexpand\begin{longtable}{%
17195             \expandonce\glslongextraSymbolAlign
17196             \expandonce\glslongextraDescAlign
17197             \expandonce\glslongextraNameAlign
17198         } }%
17199     \@glslongextra@begintab
17200 }%
17201     {\end{longtable}}%
17202 \renewcommand*{\glossaryheader}{\glslongextraSymDescNameHeader}%
17203 \fi
17204 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{3}{##1}}%
17205 \renewcommand{\glossentry}[2]{%
17206     \glslongextraSymbolFmt{##1} &
17207     \glslongextraDescFmt{##1} &
17208     \glslongextraNameFmt{##1}\tabularnewline
17209 }%
17210 \renewcommand{\subglossentry}[3]{%
17211     \glslongextraSubSymbolFmt{##1}{##2} &
17212     \glslongextraSubDescFmt{##1}{##2} &
17213     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17214 }%
17215 \ifglsgroupskip
17216     \renewcommand*{\glsgroupskip}{}%
17217 \else
17218     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17219 \fi
17220 }

```

mDescNameHeader

```

17221 \newcommand{\glslongextraLocationSymDescNameHeader}{%
17222     \glslongextraLocationSymDescNameTabularHeader\endhead
17223     \glslongextraLocationSymDescNameTabularFooter\endfoot
17224 }

```

meTabularHeader

```

17225 \newcommand{\glslongextraLocationSymDescNameTabularHeader}{%
17226   \toprule
17227   \glslongextraHeaderFmt\pagelistname &
17228   \glslongextraHeaderFmt\symbolname &
17229   \glslongextraHeaderFmt\descriptionname &
17230   \glslongextraHeaderFmt\entryname\tabularnewline
17231   \midrule
17232 }

```

meTabularFooter

```

17233 \newcommand{\glslongextraLocationSymDescNameTabularFooter}{%
17234   \bottomrule
17235 }

```

c-sym-desc-name Four column style with location list, symbol, description and name.

```

17236 \newglossarystyle{long-loc-sym-desc-name}%
17237 {%
17238   \ifGlsLongExtraUseTabular
17239     \renewenvironment{theglossary}%
17240     {%
17241       \glslongextraSymLocSetDescWidth
17242       \edef\@glslongextra@begintab{%
17243         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17244           \expandonce\glslongextraLocationAlign
17245           \expandonce\glslongextraSymbolAlign
17246           \expandonce\glslongextraDescAlign
17247           \expandonce\glslongextraNameAlign
17248         }}%
17249       \@glslongextra@begintab
17250     }%
17251     {%
17252       \glslongextraLocationSymDescNameTabularFooter
17253       \end{tabular}%
17254     }%
17255     \renewcommand*\glossaryheader{\glslongextraLocationSymDescNameTabularHeader}%
17256   \else
17257     \renewenvironment{theglossary}%
17258     {%
17259       \glspatchLToutput
17260       \glslongextraSymLocSetDescWidth
17261       \edef\@glslongextra@begintab{%
17262         \noexpand\begin{longtable}{%
17263           \expandonce\glslongextraLocationAlign
17264           \expandonce\glslongextraSymbolAlign
17265           \expandonce\glslongextraDescAlign
17266           \expandonce\glslongextraNameAlign
17267         }}%
17268       \@glslongextra@begintab

```

```

17269 }%
17270 {\end{longtable}}}%
17271 \renewcommand*{\glossaryheader}{\glslongextraLocationSymDescNameHeader}%
17272 \fi
17273 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
17274 \renewcommand{\glossentry}[2]{%
17275   \glslongextraLocationFmt{##1}{##2} &
17276   \glslongextraSymbolFmt{##1} &
17277   \glslongextraDescFmt{##1} &
17278   \glslongextraNameFmt{##1}\tabularnewline
17279 }%
17280 \renewcommand{\subglossentry}[3]{%
17281   \glslongextraSubLocationFmt{##1}{##2}{##3} &
17282   \glslongextraSubSymbolFmt{##1}{##2} &
17283   \glslongextraSubDescFmt{##1}{##2} &
17284   \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17285 }%
17286 \ifglsnogroupskip
17287   \renewcommand*{\glsgroupskip}{}%
17288 \else
17289   \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17290 \fi
17291 }

```

#### scSymNameHeader

```

17292 \newcommand{\glslongextraDescSymNameHeader}{%
17293   \glslongextraDescSymNameTabularHeader\endhead
17294   \glslongextraDescSymNameTabularFooter\endfoot
17295 }

```

#### meTabularHeader

```

17296 \newcommand{\glslongextraDescSymNameTabularHeader}{%
17297   \toprule
17298   \glslongextraHeaderFmt\descriptionname &
17299   \glslongextraHeaderFmt\symbolname &
17300   \glslongextraHeaderFmt\entryname\tabularnewline
17301   \midrule
17302 }

```

#### meTabularFooter

```

17303 \newcommand{\glslongextraDescSymNameTabularFooter}{%
17304   \bottomrule
17305 }

```

g-desc-sym-name Three column style with description in the first column, symbol in the second and name in the third.

```

17306 \newglossarystyle{long-desc-sym-name}%
17307 {%
17308   \ifGlsLongExtraUseTabular

```

```

17309 \renewenvironment{theglossary}%
17310 {%
17311     \glslongextraSymSetDescWidth
17312     \edef\@glslongextra@begintab{%
17313         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17314             \expandonce\glslongextraDescAlign
17315             \expandonce\glslongextraSymbolAlign
17316             \expandonce\glslongextraNameAlign
17317         }}%
17318     \@glslongextra@begintab
17319 }%
17320 {%
17321     \glslongextraDescSymNameTabularFooter
17322     \end{tabular}%
17323 }%
17324 \renewcommand*\glossaryheader{\glslongextraDescSymNameTabularHeader}%
17325 \else
17326 \renewenvironment{theglossary}%
17327 {%
17328     \glspatchLToutput
17329     \glslongextraSymSetDescWidth
17330     \edef\@glslongextra@begintab{%
17331         \noexpand\begin{longtable}{%
17332             \expandonce\glslongextraDescAlign
17333             \expandonce\glslongextraSymbolAlign
17334             \expandonce\glslongextraNameAlign
17335         }}%
17336     \@glslongextra@begintab
17337 }%
17338 {\end{longtable}}%
17339 \renewcommand*\glossaryheader{\glslongextraDescSymNameHeader}%
17340 \fi
17341 \renewcommand*\glsgroupeheading[1]{\glslongextraGroupHeading{3}{##1}}%
17342 \renewcommand{\glossentry}[2]{%
17343     \glslongextraDescFmt{##1} &
17344     \glslongextraSymbolFmt{##1} &
17345     \glslongextraNameFmt{##1}\tabularnewline
17346 }%
17347 \renewcommand{\subglossentry}[3]{%
17348     \glslongextraSubDescFmt{##1}{##2} &
17349     \glslongextraSubSymbolFmt{##1}{##2} &
17350     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17351 }%
17352 \ifglsgroupskip
17353     \renewcommand*\glsgroupskip{}%
17354 \else
17355     \renewcommand*\glsgroupskip{\glspenaltygroupskip}%
17356 \fi
17357 }

```

scSymNameHeader

```
17358 \newcommand{\glslongextraLocationDescSymNameHeader}{%
17359   \glslongextraLocationDescSymNameTabularHeader\endhead
17360   \glslongextraLocationDescSymNameTabularFooter\endfoot
17361 }
```

meTabularHeader

```
17362 \newcommand{\glslongextraLocationDescSymNameTabularHeader}{%
17363   \toprule
17364   \glslongextraHeaderFmt\pagelistname &
17365   \glslongextraHeaderFmt\descriptionname &
17366   \glslongextraHeaderFmt\symbolname &
17367   \glslongextraHeaderFmt\entryname\tabularnewline
17368   \midrule
17369 }
```

meTabularFooter

```
17370 \newcommand{\glslongextraLocationDescSymNameTabularFooter}{%
17371   \bottomrule
17372 }
```

c-desc-sym-name Four column style with location list, description, symbol and name.

```
17373 \newglossarystyle{long-loc-desc-sym-name}%
17374 {%
17375   \ifGlsLongExtraUseTabular
17376     \renewenvironment{theglossary}%
17377     {%
17378       \glslongextraSymLocSetDescWidth
17379       \edef\@glslongextra@begintab{%
17380         \noexpand\begin{tabular}[\glslongextraTabularVAlign]{%
17381           \expandonce\glslongextraLocationAlign
17382           \expandonce\glslongextraDescAlign
17383           \expandonce\glslongextraSymbolAlign
17384           \expandonce\glslongextraNameAlign
17385         }}%
17386       \@glslongextra@begintab
17387     }%
17388     {%
17389       \glslongextraLocationDescSymNameTabularFooter
17390       \end{tabular}%
17391     }%
17392     \renewcommand*{\glossaryheader}{\glslongextraLocationDescSymNameTabularHeader}%
17393   \else
17394     \renewenvironment{theglossary}%
17395     {%
17396       \glspatchLToutput
17397       \glslongextraSymLocSetDescWidth
17398       \edef\@glslongextra@begintab{%
17399         \noexpand\begin{longtable}{%

```

```

17400         \expandonce\glslongextraLocationAlign
17401         \expandonce\glslongextraDescAlign
17402         \expandonce\glslongextraSymbolAlign
17403         \expandonce\glslongextraNameAlign
17404     } }%
17405     \@glslongextra@begintab
17406 }%
17407 {\end{longtable}}%
17408 \renewcommand*{\glossaryheader}{\glslongextraLocationDescSymNameHeader}%
17409 \fi
17410 \renewcommand*{\glsgroupheading}[1]{\glslongextraGroupHeading{4}{##1}}%
17411 \renewcommand{\glossentry}[2]{%
17412     \glslongextraLocationFmt{##1}{##2} &
17413     \glslongextraDescFmt{##1} &
17414     \glslongextraSymbolFmt{##1} &
17415     \glslongextraNameFmt{##1}\tabularnewline
17416 }%
17417 \renewcommand{\subglossentry}[3]{%
17418     \glslongextraSubLocationFmt{##1}{##2}{##3} &
17419     \glslongextraSubDescFmt{##1}{##2} &
17420     \glslongextraSubSymbolFmt{##1}{##2} &
17421     \glslongextraSubNameFmt{##1}{##2}\tabularnewline
17422 }%
17423 \ifglsnogroupskip
17424     \renewcommand*{\glsgroupskip}{}%
17425 \else
17426     \renewcommand*{\glsgroupskip}{\glspenaltygroupskip}%
17427 \fi
17428 }

```

## 5 topic styles (glossary-topic.sty)

### 5.1 Package Initialisation and Options

Provides “topic” styles where top-level entries are considered a topic.

```
17429 \NeedsTeXFormat{LaTeX2e}
```

```
17430 \ProvidesPackage{glossary-topic}[2021/11/04 v1.47 (NLCT)]
```

Load required package.

```
17431 \RequirePackage{multicol}
```

The top-level entries act like headers. If the top-level entry has a description it’s placed below the name.

topic

```
17432 \newglossarystyle{topic}{%
```

```
17433   \renewenvironment{theglossary}{%
```

```
17434     {%
```

```
17435       \glstopicInit
```

```
17436       \def\glstopic@prechildren{}}%
```

```
17437       \def\glstopic@prevlevel{-1}}%
```

```
17438     }%
```

```
17439   {\par}}%
```

```
17440   \renewcommand*{\glossaryheader}{}}%
```

```
17441   \renewcommand*{\glsgroupheading}[1]{%
```

```
17442     \def\glstopic@prevlevel{-1}}%
```

```
17443     \glstopicGroupHeading{##1}}%
```

```
17444   }%
```

```
17445   \renewcommand{\glossentry}[2]{%
```

```
17446     \hangindent0pt\relax
```

```
17447     \parindent\glstopicParIndent\relax
```

```
17448     \glstopicItem{##1}{##2}}%
```

If there isn’t a description, penalise a page break.

```
17449     \ifglshasdesc{##1}}%
```

```
17450     {%
```

```
17451       \def\glstopic@prechildren{}}%
```

```
17452     }%
```

```
17453     {%
```

```
17454       \def\glstopic@prechildren{\nopagebreak}}%
```

```
17455     }%
```

```
17456   }%
```

```
17457   \renewcommand{\subglossentry}[3]{%
```

```
17458     \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
```

```
17459     \def\glstopic@prevlevel{##1}}%
```

Grouping is added to scope the effect of `\everypar`.

```

17460   \begingroup
17461   \glstopicAssignSubIndent{##1}%
17462   \glstopicSubItem{##1}{##2}{##3}%

17463   \par
17464   \endgroup
17465 }%
17466 \renewcommand*{\glsgroupskip}{}%
17467 }
```

`\picGroupHeading`

`\glstopicGroupHeading{<group label>}`

May be redefined if letter group headings are required. For example:

```

\renewcommand*{\glstopicGroupHeading}[1]{%
  \glstrgetgrouptitle{#1}{\thisgrptitle}%
  \section*{\thisgrptitle}%
}

17468 \newcommand*{\glstopicGroupHeading}[1]{}
```

`\glstopicItem`

`\glstopicItem{<label>}{<location list>}`

```

17469 \newcommand*{\glstopicItem}[2]{%
17470   \glsparextrglstopicPreSkip\glsparextrnoindent
17471   \glstopicMarker{#1}%
17472   \glstopicTitleFont
17473   {%
17474     \glstryitem{#1}\glstarget{#1}{\glstopicTitle{#1}}%
17475   }%
17476   \ifglshasdesc{#1}%
17477   {\glsparextrnobreak\glstopicMidSkip\glsparextrnobreak
17478     \@afterheading\glstopicDesc{#1}\glsparextrglstopicPostSkip}%
17479   {\glsparextrnobreak\glstopicPostSkip}%
17480   \glstopicLoc{#1}{#2}%
17481 }
```

`\glstopicMarker` May be used to insert a bookmark etc if required.

```

17482 \newcommand*{\glstopicMarker}[1]{}
```

`\glstopicName`

```

17483 \newcommand*{\glstopicTitle}[1]{\Glossentryname{#1}%
17484   \ifglshassymbol{#1}{\space\glossentrysymbol{#1}}{}}%
17485 }
```



stopicTitleFont

```
17486 \newcommand*{\glstopicTitleFont}[1]{\textbf{\large #1}}
```

\glstopicDesc

```
17487 \newcommand*{\glstopicDesc}[1]{\Glossentrydesc{#1}\glspostdescription}
```

\glstopicLoc

```
17488 \newcommand*{\glstopicLoc}[2]{}
```

stopicParIndent

```
17489 \newlength\glstopicParIndent
```

```
17490 \setlength\glstopicParIndent{20pt}
```

stopicSubIndent

```
17491 \newlength\glstopicSubIndent
```

```
17492 \setlength\glstopicSubIndent{20pt}
```

\glstopicInit

```
17493 \newcommand{\glstopicInit}{}%
```

AssignSubIndent

```
\glstopicAssignSubIndent{<level>}
```

Used to set the indentation for sub-levels.

```
17494 \newcommand*{\glstopicAssignSubIndent}[1]{%
```

```
17495   \par
```

```
17496   \parindent\dimexpr#1\glstopicSubIndent-\glstopicSubIndent\relax
```

```
17497   \glstopicAssignWidest{#1}%
```

```
17498   \glstopicsubitemhangindent=\dimexpr\parindent+\glstopicwidest\relax
```

```
17499   \hangindent\glstopicsubitemhangindent\relax
```

```
17500   \everypar{\hangindent\glstopicsubitemhangindent\relax
```

```
17501     \parindent\dimexpr\glstopicSubItemParIndent+\glstopicsubitemhangindent\relax}%
```

```
17502 }%
```

bitemhangindent

```
17503 \newlength\glstopicsubitemhangindent
```

ubItemParIndent

```
17504 \newlength\glstopicSubItemParIndent
```

```
17505 \glstopicSubItemParIndent\parindent
```

\glstopicwidest

```
17506 \newlength\glstopicwidest
```

picAssignWidest

`\glstopicAssignWidest{<level>}`

Used in the definition of `\glstopicAssignSubIndent` to set the indentation from the widest name for the given level. This will require `glossary-tree` to set the values.

```
17507 \newcommand*{\glstopicAssignWidest}[1]{%
17508   \ifcsundef{@glswidestlength\romannumeral#1}%
17509   {%
17510     \ifcsdef{@glswidestname\romannumeral#1}%
17511     {%
17512       \settowidth{\glstopicwidest}{%
17513         \glstopicSubNameFont{\csuse{@glswidestname\romannumeral#1}}%
17514         \glstopicSubItemSep
17515       }%
17516     }%
17517     {\setlength{\glstopicwidest}{0pt}}%
```

Save the value so that it doesn't have to keep being recalculated.

```
17518   \csedef{@glswidestlength\romannumeral#1}{\the\glstopicwidest}%
17519 }%
17520 {\setlength{\glstopicwidest}{\csuse{@glswidestlength\romannumeral#1}}}%
17521 }
```

glstopicPreSkip

```
17522 \newcommand*{\glstopicPreSkip}{\medskip}
```

glstopicMidSkip

```
17523 \newcommand*{\glstopicMidSkip}{\smallskip}
```

lstopicPostSkip

```
17524 \newcommand*{\glstopicPostSkip}{\smallskip}
```

glstopicSubItem

`\glstopicSubItem{<level>}{<label>}{<location list>}`

```
17525 \newcommand*{\glstopicSubItem}[3]{%
17526   \glstopicSubItemBox{#1}{\glstopicSubNameFont{\glentryitem{#2}}%
17527     \glstarget{#2}{\glossentryname{#2}}}%
17528   \glstopicSubItemSep
17529 }%
17530 \ifglshassymbol{#2}{(\glossentrysymbol{#2})\space}{}%
17531 \ifglshasdesc{#2}%
17532   {\glossentrydesc{#2}\glspostdescription\glstopicSubPreLocSep}{}%
17533   \glstopicSubLoc{#2}{#3}%
17534 }
```

topicSubItemSep

```
17535 \newcommand*{\glstopicSubItemSep}{\quad}
```

topicSubItemBox

```
\glstopicSubItemBox{<level>}{<text>}
```

```
17536 \newcommand*{\glstopicSubItemBox}[2]{%
17537   \ifdim\glstopicwidest>0pt\relax\makebox[\glstopicwidest][l]{#2}\else#2\fi
17538 }
```

opicSubNameFont

```
17539 \newcommand*{\glstopicSubNameFont}[1]{\textbf{#1}}
```

picSubPreLocSep

```
17540 \newcommand*{\glstopicSubPreLocSep}{\space}
```

\glstopicSubLoc

```
17541 \newcommand*{\glstopicSubLoc}[2]{#2}
```

\glstopicCols

```
17542 \newcommand*{\glstopicCols}{2}
```

glstopicColsEnv

```
17543 \newcommand*{\glstopicColsEnv}{multicols}
```

topicmcols

```
17544 \newglossarystyle{topicmcols}{%
17545   \renewenvironment{theglossary}%
17546   {%
17547     \glstopicInit
17548     \def\glstopic@prechildren{}%
17549     \def\glstopic@postchildren{}%
17550     \def\glstopic@prevlevel{-1}%
17551   }%
17552   {%
17553     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
17554     \par
17555   }%
17556   \renewcommand*{\glossaryheader}{}%
17557   \renewcommand*{\glsgroupheading}[1]{%
17558     \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
17559     \def\glstopic@prevlevel{-1}%
17560     \glstopicGroupHeading{##1}%
17561   }%
17562   \renewcommand{\glossentry}[2]{%
```

```

17563 \ifnum\glstopic@prevlevel>0\relax\glstopic@postchildren\fi
17564 \def\glstopic@prevlevel{0}%
17565 \hangindentOpt\relax
17566 \parindent\glstopicParIndent\relax
17567 \glstopicItem{##1}{##2}%
17568 \ifnum\glstopicCols>1\relax
    If there isn't a description, penalise a page break.
17569 \ifglshasdesc{##1}%
17570 {%
17571 \edef\glstopic@prechildren{%
17572 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
17573 }%
17574 }%
17575 {%
17576 \edef\glstopic@prechildren{%
17577 \noexpand\nopagebreak
17578 \noexpand\begin{\glstopicColsEnv}{\glstopicCols}%
17579 }%
17580 }%
17581 \edef\glstopic@postchildren{\noexpand\end{\glstopicColsEnv}}%
17582 \fi
17583 }%
17584 \renewcommand{\subglossentry}[3]{%
17585 \ifnum\glstopic@prevlevel=0\relax\glstopic@prechildren\fi
17586 \def\glstopic@prevlevel{##1}%
17587 \glstopicAssignSubIndent{##1}%
17588 \glstopicSubItem{##1}{##2}{##3}%
17589 }%
17590 \renewcommand*{\glsgroupskip}{}%
17591 }

```

# Glossary

**bib2gls** A command line Java application that selects entries from a .bib file and converts them to glossary definitions (like bibtex but also performs hierarchical sorting and collation, thus omitting the need for **xindy** or **makeindex**). Further details at: <http://www.dickimaw-books.com/software/bib2gls/..>

**First use** The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **First use flag** & **First use text**

**First use flag** A conditional that determines whether or not the entry has been used according to the rules of **first use**.

**First use text** The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

**makeindex** An indexing application.

**xindy** An flexible indexing application with multilingual support written in Perl.

# Change History

0.1 (2015-11-22)		
General: Initial experimental release	.....	5
0.2 (2015-11-30)		
\Glsfmtshort: new	.....	370
\glsfmtshort: new	.....	369
\Glsfmtshortpl: new	.....	370
\glsfmtshortpl: new	.....	369
short: switched inline full form to short		
(long)	.....	261
0.3 (2015-12-02)		
\@ACRlong: added redefinition	.....	95
\@ACRlongpl: added redefinition	.....	96
\@ACRshort: added redefinition	.....	93
\@ACRshortpl: added redefinition	.....	94
\@Acrlong: added redefinition	.....	95
\@Acrlongpl: added redefinition	.....	96
\@Acrshort: added redefinition	.....	92
\@Acrshortpl: added redefinition	.....	94
\@GLSdesc: added redefinition	.....	88
\@GLSdescplural: added redefinition	.	89
\@GLSfirst: added redefinition	.....	86
\@GLSfirstplural: added redefinition		87
\@GLSname: added redefinition	.....	88
\@GLSplural: added redefinition	.....	86
\@GLSSymbol: added redefinition	.....	89
\@GLSSymbolplural: added		
redefinition	.....	90
\@GLStext: added redefinition	.....	85
\@GLSuseri: added redefinition	.....	90
\@GLSuserii: added redefinition	.....	90
\@GLSuseriii: added redefinition	....	91
\@GLSuseriv: added redefinition	.....	91
\@GLSuseriv: added redefinition	.....	91
\@GLSuseriv: added redefinition	.....	91
\@GLSdesc: added redefinition	.....	88
\@GLSdescplural: added redefinition	.	88
\@GLSfirst: added redefinition	.....	85
\@GLSfirstplural: added redefinition		87
\@GLSname: added redefinition	.....	87
\@GLSplural: added redefinition	.....	86
\@GLSSymbol: added redefinition	.....	89
\@GLSSymbolplural: added		
redefinition	.....	89
\@GLSxtr@defaultnoglossarywarning:		
new	.....	154
\@GLSxtr@field@linkdefs: new	.....	84
\@GLSxtr@insertdots: new	.....	225
\@print@glossary: added redefinition		150
\glsabbrvdefaultfont: renamed from		
\abbrvdefaultfont	.....	231
\glsaccessdesc: new	.....	184
\glsaccessdescplural: new	.....	185
\glsaccessfirst: new	.....	182
\glsaccessfirstplural: new	.....	182
\glsaccesslong: new	.....	187
\glsaccesslong: new	.....	186
\glsaccessname: new	.....	180
\glsaccessplural: new	.....	181
\glsaccessshort: new	.....	185
\glsaccessshort: new	.....	185
\glsaccessshortpl: new	.....	186

\glsaccessshortpl: new .....	186	\cGLSpl: new .....	123
\glsaccesssymbol: new .....	183	\cGLSpl@: new .....	123
\glsaccesssymbolplural: new .....	183	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new .....	181	new .....	118
\glstentryfmt: added check for short ..	75	\cGLS: new .....	123
\glslongpltok: new .....	225	\cGLSformat: new .....	123
\glsshortpltok: new .....	225	\cGLSpl: new .....	123
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new .....	124
name in \setkeys .....	227	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new .....	19
for plural .....	222	\glsenableentrycount: new .....	119
\GLSxtrlongpl: new .....	242	\glsfirstabbrvdefaultfont: new ..	231
\Glsxtrlongpl: new .....	241	\glsfirstlongdefaultfont: new ...	232
\glsxtrlongpl: new .....	241	\Glsfmtfirst: new .....	373
\glsxtrNoGlossaryWarning: new ....	25	\glsfmtfirst: new .....	373
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new .....	374
new .....	222	\glsfmtfirstpl: new .....	374
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new .....	372
new .....	222	\glsfmtplural: new .....	372
\glsxtrpostlinkendsentence: new ..	221	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new .....	240	\Glsxtrtitleshort .....	370
\Glsxtrshortpl: new .....	240	renamed from \glstentryfmtshort ..	370
\glsxtrshortpl: new .....	239	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort .....	369
\glslabeltok .....	253	renamed from \glstentryfmtshort ..	369
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok .....	251	\Glsxtrtitleshortpl .....	370
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl .....	370
redefinition of \acronymtype .....	21	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl .....	369
\Glsxtrshort .....	370	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl .....	369
\glsxtrshort .....	369	\Glsfmttext: new .....	371
\Glsfmtshortpl: changed to use		\glsfmttext: new .....	371
\glsxtrshortpl .....	370	\glshasattribute: new .....	199
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	198
\glsxtrshortpl .....	369	\glsxtremsuffix: new .....	302
\glsxtrifemptyglossary: new .....	33	\GlsXtrEnableEntryCounting: new ..	118
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new .....	121
argument .....	202	\glsxtrscfont: new .....	268
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new .....	269
argument .....	202	\glsxtrsmfont: new .....	285
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new .....	285
default type to \acronymtype ....	133	short-em: new .....	310
\newterm: fixed name argument .....	201	short-em-desc: new .....	311
0.5 (2015-12-07)		short-em-footnote: new .....	321
\cGLS: new .....	123	short-em-long: new .....	306
\cGLS@: new .....	123	short-em-long-desc: new .....	307

short-em-postfootnote: new .....	324	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new .....	280	attribute .....	359
short-sc-postfootnote: new .....	282	\Glsxtrheadtext: now uses headuc	
short-sm: new .....	289	attribute .....	362
short-sm-desc: new .....	291	\glxtrheadtext: now uses headuc	
short-sm-footnote: new .....	296	attribute .....	361
short-sm-long: new .....	287	short-em-footnote: switch off regular	
short-sm-long-desc: new .....	289	attribute if set .....	322
short-sm-postfootnote: new .....	299	short-em-footnote-desc: switch off	
long-noshort-em: new .....	314	regular attribute if set .....	323
long-noshort-em-desc: new .....	318	short-long: switch off regular attribute	
long-noshort-sm: new .....	293	if set .....	252
long-noshort-sm-desc: new .....	295	short-long-desc: switch off regular	
long-short-em: new .....	302	attribute if set .....	253
long-short-em-desc: new .....	303	short-postfootnote-desc: switch off	
long-short-sm: new .....	286	regular attribute if set .....	259
long-short-sm-desc: new .....	287	short-sc-footnote: switch off regular	
0.5.1 (2015-12-02)		attribute if set .....	280
\Glsaccessstext: new .....	181	short-sc-footnote-desc: switch off	
0.5.1 (2015-12-07)		regular attribute if set .....	282
\@glxtr@doaccsupp: new .....	24	short-sm-footnote: switch off regular	
General: removed \ifglxtruseuchead	358	attribute if set .....	297
\Glsaccessdesc: new .....	184	short-sm-footnote-desc: switch off	
\Glsaccessdescplural: new .....	185	regular attribute if set .....	299
\Glsaccessfirst: new .....	182	long-short: switch off regular attribute	
\Glsaccessfirstplural: new .....	183	if set .....	250
\Glsaccessname: new .....	180	long-short-desc: switch off regular	
\Glsaccessplural: new .....	181	attribute if set .....	251
\Glsaccesssymbol: new .....	183	long-short-sc-desc: switch off regular	
\Glsaccesssymbolplural: new .....	184	attribute if set .....	270
\Glsxtrheadfirst: now uses headuc		footnote: switch off regular attribute if	
attribute .....	364	set .....	255
\glxtrheadfirst: now uses headuc		postfootnote: switch off regular	
attribute .....	363	attribute if set .....	257
\Glsxtrheadfirstplural: now uses		0.5.2 (2015-12-08)	
headuc attribute .....	364	\@GLSdesc@: added accessibility support	88
\glxtrheadfirstplural: now uses		\@GLSdescplural@: added accessibility	
headuc attribute .....	364	support .....	89
\Glsxtrheadplural: now uses headuc		\@GLSfirst@: added accessibility	
attribute .....	363	support .....	86
\glxtrheadplural: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute .....	362	support .....	87
\Glsxtrheadshort: now uses headuc		\@GLSname@: added accessibility support	88
attribute .....	359	\@GLSplural@: added accessibility	
\glxtrheadshort: now uses headuc		support .....	86
attribute .....	358	\@GLSsymbol@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support .....	89
attribute .....	360	\@GLSsymbolplural@: added	
		accessibility support .....	90



\@GLStext@: added accessibility support	85	\GLSaccesslongpl: new	187, 197
\@Glsdesc@: added accessibility support	88	\Glsaccesslongpl: new	187
\@Glsdescplural@: added accessibility support	88	\glsaccesslongpl: new	187
\@Glsfirst@: added accessibility support	85	\GLSaccessname: new	180, 194
\@Glsfirstplural@: added accessibility support	87	\GLSaccessplural: new	182, 194
\@Glsname@: add accessibility support	87	\GLSaccessshort: new	186, 196
\@Glsplural@: added accessibility support	86	\GLSaccessshortpl: new	186, 196
\@Glsymbol@: added accessibility support	89	\GLSaccesssymbol: new	183, 195
\@Glsymbolplural@: added accessibility support	89	\GLSaccesssymbolplural: new	184, 195
\@Glstext@: added accessibility support	85	\GLSaccessstext: new	181, 194
\@Glsdesc@: added accessibility support	88	\glsentryfmt: moved	
\@Glsdescplural@: added accessibility support	88	\glssetabbrvfmt from	
\@Glsfirst@: added accessibility support	85	\glsxtrabbrvfmt to here	75
\@Glsfirstplural@: added accessibility support	87	\GlsXtrEnableInitialTagging: new	218
\@Glsname@: added accessibility support	87	\glsxtrfieldtitlecase: new	203
\@Glsplural@: added accessibility support	86	\GlsXtrFormatLocationList: new	73
\@Glsymbol@: added accessibility support	89	\glsxtrnewabbrevpresetkeyhook:	
\@Glsymbolplural@: added accessibility support	89	new	230
\@Glstext@: added accessibility support	85	\glsxtrtagfont: new	219
\@Glsdesc@: added accessibility support	88	\KV@printgloss@nonumberlist: added	75
\@Glsdescplural@: added accessibility support	88	\mfu@checkword@do: added	218
\@Glsfirst@: added accessibility support	85	\setabbreviationstyle: added check	
\@Glsfirstplural@: added accessibility support	87	for post-definition style switch	246
\@Glsname@: added accessibility support	87	0.5.3 (2015-12-09)	
\@Glsplural@: added accessibility support	86	\@glsxtr@autoindex@at: new	214
\@Glsymbol@: added accessibility support	89	\@glsxtr@autoindex@encap: new	215
\@Glsymbolplural@: added accessibility support	89	\@glsxtr@autoindex@esc: new	215
\@Glstext@: added accessibility support	85	\@glsxtr@autoindex@level: new	215
\@glsxtr@activate@initialtagging:		\@glsxtr@autoindex@setname: new	213
new	219	\@glsxtr@doabbreviationsdef: new	21
\@glsxtr@do@titlecaps@warn: new	219	General: removed	
\@glsxtr@tag: new	219	\GlsXtrNoGlsWarningNoAutoMakeMain	153
General: fixed typo in glossaries-accsupp and tidied up code to use just one		\glsdescwidth: added	72
\@ifpackageloaded	180	\glspagelistwidth: added	73
removed \glsxtrabbrvfmt	243	\glsxtrdoautoindexname: new	212
\glossaryentrynumbers: added	73	\glsxtrpostnamehook: new	209
\Glossentrydesc: added	216	\if@glsxtr@format@override: new	211
\Glossentryname: added	208	\ProvidesGlossariesExtraLang: new	378
\Glossentrysymbol: added	217	\RequireGlossariesExtraLang: new	378
\glossentrysymbol: added	216	0.5.4 (2015-12-15)	
\GLSaccessdesc: new	184, 195	\@newglossaryentry@defunitcounters:	
\GLSaccessdescplural: new	185, 196	new	124
\GLSaccessfirst: new	182, 194	\@GLSxtr@p@acrlong@: new	110
\GLSaccessfirstplural: new	183, 195	\@GLSxtr@p@acrlongpl@: new	110
\GLSaccesslong: new	187, 196	\@GLSxtr@p@acrshort@: new	109
		\@GLSxtr@p@acrshortpl@: new	109
		\@GLSxtr@p@long@: new	109
		\@GLSxtr@p@longpl@: new	109

\@GLSxtr@p@plural@: new .....	108	\@sGlsXtrEnableOnTheFly: new .....	68
\@GLSxtr@p@short@: new .....	108	\cGlsformat: added .....	124
\@GLSxtr@p@shortpl@: new .....	108	\cGlsformat: added .....	124
\@GLSxtr@p@text@: new .....	107	\cGlsplformat: added .....	124
\@GlsXtrEnableOnTheFly: new .....	69	\cGlsplformat: added .....	124
\@Glsxtr: new .....	69	\gl:disablehyper: added .....	106
\@Glsxtr@p@acrlong@: new .....	110	\gl:donohyperlink: added .....	106
\@Glsxtr@p@acrlongpl@: new .....	110	\gl:enableentryunitcount: new ...	126
\@Glsxtr@p@acrshort@: new .....	109	\gl:shasattribute: added check for	
\@Glsxtr@p@acrshortpl@: new .....	109	entry's existence .....	199
\@Glsxtr@p@long@: new .....	109	\gl:sifattribute: added check for	
\@Glsxtr@p@longpl@: new .....	109	entry's existence .....	199
\@Glsxtr@p@plural@: new .....	108	\gl:postlinkhook: added existence	
\@Glsxtr@p@short@: new .....	108	check .....	221
\@Glsxtr@p@shortpl@: new .....	108	\Glsxtr: new .....	69
\@Glsxtr@p@text@: new .....	107	\gl:xtr: new .....	69
\@Glsxtrpl: new .....	70	\gl:xtrcat: new .....	69
\@alt@gl:shyp@opt: new .....	103	\gl:xtrdohyperlink: added .....	104
\@gl:alt@hyp@opt: new .....	103	\gl:xtrdowrglossaryhook: new ....	103
\@gl:alt@hyp@opt@char: new .....	103	\GlsXtrEnableEntryUnitCounting:	
\@gl:alt@hyp@opt@keys: new .....	103	new .....	130
\@gl:increment@currunitcount:		\GlsXtrEnableOnTheFly: new .....	68
new .....	125	\Glsxtrpl: new .....	70
\@gl:local@increment@currunitcount:		\gl:xtrpl: new .....	70
new .....	126	\gl:xtrpostlocalreset: new .....	117
\@gl:setdefault@gl:link@opts:		\gl:xtrpostlocalunset: new .....	117
new .....	100	\gl:xtrpostreset: new .....	117
\@gl:xtr: new .....	69	\gl:xtrpostunset: new .....	116
\@gl:xtr@addunitcounter: new ....	125	\gl:xtrprotectlinks: new .....	107
\@gl:xtr@currunitcount: new ....	126	\GlsXtrSetAltModifier: new .....	103
\@gl:xtr@ifunitcounter: new ....	125	\GlsXtrSetDefaultGlsOpts: new ...	102
\@gl:xtr@p@acrlong@: new .....	109	\gl:xtrstarflywarn: new .....	69
\@gl:xtr@p@acrlongpl@: new .....	110	\GlsXtrWarning: new .....	70
\@gl:xtr@p@acrshort@: new .....	109	\MakeAcronymsAbbreviations: now	
\@gl:xtr@p@acrshortpl@: new .....	109	disables \setacronymstyle .....	133
\@gl:xtr@p@long@: new .....	109	1.0 (2016-01-24)	
\@gl:xtr@p@longpl@: new .....	109	\@gl:xtr@autoindexcrossrefs: new .	18
\@gl:xtr@p@plural@: new .....	107	\@gl:xtr@idx@displaynumberlist:	
\@gl:xtr@p@short@: new .....	108	new .....	144
\@gl:xtr@p@shortpl@: new .....	108	\@gl:xtr@idx@entrynumberlist: new	145
\@gl:xtr@p@text@: new .....	107	\@gl:xtr@noidx@displaynumberlist:	
\@gl:xtr@prevunitcount: new ....	126	new .....	144
\@gl:xtr@setentryunitcountunsetattr:		\@gl:xtr@noidx@entrynumberlist:	
new .....	130	new .....	145
\@gl:xtr@unitcountlist: new ....	125	\@gl:xtr@noidx@numberlistloop:	
\@gl:xtrpl: new .....	70	new .....	145
\@newglossaryentryposthook: added		\@gl:xtr@reg@glosslist: new ....	135
empty see value if not set and added		\makeglossaries: new .....	135
'see' to field key map .....	56		

1.01 (2016-02-02)	
\glxtrdiscardperiod: added check for first use .....	222
short-desc: fixed typo in \glxtrininlinefullformat and added missing second argument ...	262
1.02 (2016-04-25)	
\@glxtr@current@style: new .....	71
\Glsfmtfull: new .....	377
\Glsfmtfull: new .....	376
\Glsfmtfullpl: new .....	377
\Glsfmtfullpl: new .....	377
\Glsfmtlong: new .....	375
\Glsfmtlong: new .....	374
\Glsfmtlongpl: new .....	376
\Glsfmtlongpl: new .....	375
\Glsxtrheadfull: new .....	368
\Glsxtrheadfullpl: new .....	368
\Glsxtrheadfullpl: new .....	367
\Glsxtrheadlong: new .....	366
\Glsxtrheadlong: new .....	365
\Glsxtrheadlongpl: new .....	366
\Glsxtrheadlongpl: new .....	365
\Glsxtrtitlefull: new .....	368
\Glsxtrtitlefull: new .....	367
\Glsxtrtitlefullpl: new .....	369
\Glsxtrtitlefullpl: new .....	368
\Glsxtrtitlelong: new .....	366
\Glsxtrtitlelong: new .....	365
\Glsxtrtitlelongpl: new .....	367
\Glsxtrtitlelongpl: new .....	366
short-postfootnote-desc: added redef of \glxtrsetupfulldefs ..	260
\ifglxtrininsertinside: new .....	249
postfootnote: added redef of \glxtrsetupfulldefs .....	258
stylemods: new .....	25
1.03 (2016-04-27)	
\@GLSfirstplural@: bug fix: misspelt cs name .....	87
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	86
\@GLSfirstplural@: bug fix: misspelt cs name .....	87
\@GLsplural@: fixed bug \@GLsplural@ should be redefined not \@GLsplural	86
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	86
\glxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	366
\glxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	359
1.04 (2015-04-30)	
short-em-footnote: renamed from “footnote-em” .....	321
1.04 (2016-05-02)	
\@@glxtrpostloctag: new .....	75
\@GLSdesc@: set abbreviation and regular format .....	88
\@GLSdescplural@: set abbreviation and regular format .....	89
\@GLSfirst@: set abbreviation format ..	86
\@GLSfirstplural@: set abbreviation and regular format .....	87
\@GLSname@: set abbreviation and regular format .....	88
\@GLSplural@: set abbreviation and regular format .....	86
\@GLSsymbol@: set regular format .....	89
\@GLSsymbolplural@: set regular format	90
\@GLStext@: set abbreviation and regular format .....	85
\@GLSuseri@: set regular format .....	90
\@GLSuserii@: set regular format .....	90
\@GLSuseriii@: set regular format .....	91
\@GLSuseriv@: set regular format .....	91
\@GLSuservi@: set regular format .....	92
\@GLSdesc@: set abbreviation and regular format .....	88
\@GLSdescplural@: set abbreviation and regular format .....	88
\@GLSfirst@: set abbreviation and regular format .....	85
\@GLSfirstplural@: set abbreviation and regular format .....	87
\@GLSname@: set abbreviation and regular format .....	87
\@GLsplural@: set abbreviation and regular format .....	86
\@GLssymbol@: set regular format .....	89
\@GLssymbolplural@: set regular format	89
\@GLStext@: set abbreviation and regular format .....	85
\@GLSuseri@: set regular format .....	90
\@GLSuserii@: set regular format .....	90
\@GLSuseriii@: set regular format .....	91

\@Glsuseriv@: set regular format	91	\glslongfont: new	231
\@Glsuserv@: set regular format	91	\glslonguserfont: new	327
\@Glsuservi@: set regular format	91	\glstrassignfieldfont: new	84
\@gls@preglossaryhook: added check for entry's existence	219	\GlsXtrEnablePreLocationTag: new	74
\@glsdesc@: set abbreviation and regular format	88	\glstrfirstscfont: new	269
\@glsdescplural@: set abbreviation and regular format	88	\glstrfirstsmfont: new	285
\@glsfirst@: set abbreviation and regular format	85	\glstrlongshortdescsort: new	250
\@glsfirstplural@: set abbreviation and regular format	87	\glstrpostnamehook: added category check	209
\@glsname@: set abbreviation and regular format	87	\glstrregularfont: new	76
\@glsplural@: set abbreviation and regular format	86	\glstruserfield: new	327
\@glsymbol@: set regular format	89	\glstruserparen: new	327
\@glsymbolplural@: set regular format	89	\glstrusersuffix: new	327
\@glstext@: set abbreviation and regular format	85	\GlsXtrWarnDeprecatedAbbrStyle: new	248
\@glstr@deprecated@abbrstyle: new	248	short-em-long-em: new	308
\@glstr@do@style: new	26	short-em-long-em-desc: new	309
\@glstr@dolocktag: new	75	short-em-nolong: new	311
\@glstr@idx@entrynumberlist: switched from \let to \newcommand	145	short-em-nolong-desc: new	313
\@glstr@pagetag: new	74	short-em-postfootnote: renamed from "postfootnote-em"	324
\@glstr@pagetag: new	74	short-footnote: new	256
\@glstr@prelocktag: new	75	short-long-user: new	335
\@glstr@postlocktag: new	75	short-long-user-desc: new	336
\@glstr@prelocktag: new	74	short-nolong: new	262
\glossentrydesc: added glossdescfont attribute check	204	short-nolong-desc: new	264
\Glossentryname: added glossnamefont attribute check	208	short-postfootnote: new	259
\glossentryname: added glossnamefont attribute check	205	short-sc-footnote: renamed from "footnote-sc"	280
moved post name hook inside condition	208	short-sc-nolong: new	274
\glsabbrvmfont: new	302	short-sc-nolong-desc: new	276
\glsabbrvuserfont: new	327	short-sc-postfootnote: renamed from "postfootnote-sc"	282
\glsfirstabbrvmfont: new	302	short-sm-footnote: renamed from "footnote-sm"	296
\glsfirstabbrvuserfont: new	327	short-sm-nolong: new	291
\glsfirstlongemfont: new	302	short-sm-nolong-desc: new	292
\glsfirstlonguserfont: new	327	short-sm-postfootnote: renamed from "postfootnote-sm"	299
\glsifnotregularcategory: new	200	\letabbreviationstyle: new	248
\glslongdefaultfont: new	232	\newabbreviationstyle: bug fix: corrected test for existence	247
\glslongemfont: new	302	long-em-noshort-em: new	315
		long-em-noshort-em-desc: new	319
		long-em-short-em: new	304
		long-em-short-em-desc: new	305
		long-noshort: new	268
		long-noshort-desc: new	267

long-noshort-em: renamed from “long-em” .....	314	\glxtrComputeTreeIndent: new ...	455
long-noshort-em-desc: renamed from “long-desc-em” .....	318	\glxtrComputeTreeSubIndent: new	455
long-noshort-sc: renamed from “long-sc” .....	276	\glxtrtreetopindent: new .....	446
long-noshort-sc-desc: renamed from “long-desc-sc” .....	278	short-em-long: fixed incorrect font used by long form .....	307
long-noshort-sm: renamed from “long-sm” .....	293	\xglsssetwidest: new .....	447
long-noshort-sm-desc: renamed from \long-desc-sm .....	295	1.06 (2016-06-18)	
long-short-user: new .....	328	\@glstdoifexistsorwarn: new .....	18
long-short-user-desc: new .....	334	\@glxtr@docdefval: new .....	17
\renewabbreviationstyle: new ....	247	\@glxtr@usesee: new .....	57
style: new .....	26	General: disabled docdef key at the start of the document .....	32
1.05 (2016-06-10)		docdef option changed to choice .....	17
\eglssetwidest: new .....	447	\glxtr@usesee: new .....	57
\glsFindWidestAnyName: new .....	449	\glxtrusesee: new .....	57
\glsFindWidestAnyNameLocation: new .....	455	\glxtruseseeformat: new .....	57
\glsFindWidestAnyNameSymbol: new	452	\if@glxtrdocdefrestricted: new ..	18
\glsFindWidestAnyNameSymbolLocation: new .....	453	1.07 (2016-08-15)	
\glsFindWidestLevelTwo: new ....	451	\@@glxtrp: new .....	110
\glsFindWidestUsedAnyName: new ..	449	\@GLSfirst@: added check for nohyperfirst attribute .....	86
\glsFindWidestUsedAnyNameLocation: new .....	454	\@GLSfirstplural@: added check for nohyperfirst attribute .....	87
\glsFindWidestUsedAnyNameSymbol: new .....	452	\@GLSxtrp: new .....	111
\glsFindWidestUsedAnyNameSymbolLocation: new .....	453	\@Glsfirst@: added check for nohyperfirst attribute .....	86
\glsFindWidestUsedLevelTwo: new .	449	\@Glsfirstplural@: added check for nohyperfirst attribute .....	87
\glsFindWidestUsedTopLevelName: new .....	448	\@Glsxtrp: new .....	111
\glsfirstlongfootnotefont: new ..	254	\@gls@preglossaryhook: added \glossxtrsetpopts .....	220
\glsgetwidestname: new .....	448	\@glsfirst@: added check for nohyperfirst attribute .....	85
\glsgetwidestsubname: new .....	448	\@glsfirstplural@: added check for nohyperfirst attribute .....	87
\glslongfootnotefont: new .....	254	\@glxtrinmark: new .....	356
\glxtrAltTreeIndent: new .....	446	\@glxtrnotinmark: new .....	356
\glxtralttreeInit: new .....	446	\@glxtrp: new .....	110
\glxtrAltTreePar: new .....	446	\@glxtrp@opt: new .....	110
\glxtrAltTreeSetHangIndent: new	456	\glossxtrsetpopts: new .....	110
\glxtrAltTreeSetSubHangIndent: new .....	456	\glsp: new .....	113
\glxtralttreeSubSymbolDescLocation: new .....	446	\glsp: new .....	113
\glxtralttreeSymbolDescLocation: new .....	445	\glxtr@entry@p: new .....	111
		\glxtrabbrvfootnote: new .....	254
		\glxtrchecknohyperfirst: new ....	85
		\glxtrfieldtitlecasecs: new ....	203
		\glxtrifinmark: new .....	356
		\GLSxtrp: new .....	114
		\Glsxtrp: new .....	113

\glxtrp: new .....	112	\glxtrassignfieldfont: added check for existence .....	84
\glxtrsetpopts: new .....	110	\glxtrresourcefile: new .....	155
short-long-desc: added text key ....	253	\printunsrtglossaries: new .....	162
fixed misspelling of \glsabbrvfont in plural key .....	253	\printunsrtglossary: new .....	161
long-short-desc: added missing text key .....	251	1.09 (2016-12-16)	
fixed misspelling of \glsabbrvfont .	251	\@glxtr@gettype: new .....	143
footnote: changed first forms to use \glsfirstlongfootnotefont ...	254	\@glxtr@mixed@assign@sortkey: new .....	144
postfootnote: removed \footnote from first keys .....	257	\@printglossary: redefined to save options .....	142
switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	258	\glxtr@makeglossaries: new .....	143
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glxtrifwasfirstuse .....	134	1.10 (2016-12-17)	
1.08 (2016-12-13)		\@GLSpl@: fixed bug caused by typo in command name .....	77
\@@glxtr@record: new .....	8	1.11 (2017-01-19)	
\@GLS@: added \@glxtr@record .....	77	\@glxtr@do@redef@forglsentries: new .....	6
\@GLSpl@: added \@glxtr@record ...	77	\@glxtr@noidx@do: new .....	169
\@Gls@: added \@glxtr@record .....	77	\@glxtr@redef@forglsentries: new .	6
\@GLspl@: added \@glxtr@record ...	77	\@glxtr@shortcutsval: new .....	23
\@gls@: added \@glxtr@record .....	77	\@glxtr@unsrt@getgrouptitle: new	168
\@gls@link@: added \@glxtr@record .....	78	\@print@noidx@glossary: added redefinition .....	147
\@gls@field@link: added \@glxtr@record .....	76	\glxtr@addloclistfield: added group key .....	14
\@gls@saveentrycounter: new .....	31	added location key .....	14
\@glsdisp: added \@glxtr@record ..	77	\glxtr@fields: new .....	157
\@glspl@: added \@glxtr@record ...	77	\glxtr@linkprefix: new .....	157
\@glxtr@dorecord: new .....	10	\glxtr@org@newignoredglossary: new .....	52
\@glxtr@err@undefaction: new .....	6	\glxtr@s@newignoredglossary: new	52
\@glxtr@record: new .....	7	\glxtr@shortcutsval: new .....	157
\@glxtr@warn@onexistsordo: new ...	6	\glxtr@texencoding: new .....	157
\@glxtr@warn@undefaction: new ....	6	\glxtr@writefields: new .....	157
\@print@unsrt@glossary: new .....	162	\GlsXtrLoadResources: new .....	156
record: added record package option ...	16	\glxtrpageref: new .....	48
\glsadd: added \@glxtr@record ....	83	\glxtrresourcefile: changed extension to .glstex .....	155
\glsdoifexists: now defines \glslabel .....	55	\newignoredglossary: added starred version .....	52
\glxtr@@do@wrglossary: new .....	31	1.12 (2017-02-03)	
\glxtr@addloclistfield: new .....	13	\@@glxtr@recordcounter: new .....	13
\glxtr@indexonly@saveentrycounter: new .....	13	\@gls@preglossaryhook: check for definition .....	219
\glxtr@record: new .....	159	\@glxtr@counterrecordhook: new .	159
\glxtr@resource: new .....	156	\@glxtr@display@loc: new .....	148
\glxtr@saveentrycounter: new ....	31	\@glxtr@docounterrecord: new ...	159
\glxtr@setup@record: new .....	13		

\@glxstr@longnewglossaryentry: new .....	51	\glxstrfieldifinlist: new .....	38
\@glxstr@noop@recordcounter: new .	13	\glxstrfieldlistadd: new .....	37
\@glxstr@op@recordcounter: new ...	13	\glxstrfieldlistadd: new .....	37
\@glxstr@provide@storagekey: new .	34	\glxstrfieldlistgadd: new .....	37
\@glxstr@s@longnewglossaryentry: new .....	51	\glxstrfieldlistxadd: new .....	37
\@glxstr@entryfmt: new .....	36	\glxstrfieldxifinlist: new .....	38
\@glxstr@indexaliased: new .....	101	\glxstrfmt: new .....	35
\@glxstr@setaliasnoindex: new ....	101	\GlsXtrFmtDefaultOptions: new ....	35
\@newglossaryentryposthook: added check for alias key .....	64	\GlsXtrFmtField: new .....	35
\@no@glxstr@indexaliased: new ....	101	\glxstrifkeydefined: new .....	34
\@printunsrtglossary: new .....	162	\glxstrindexaliased: new .....	101
General: added target key to printgloss family .....	142	\GlsXtrLetField: new .....	44
\apptoglossarypreamble: new .....	49	\GlsXtrLetFieldToField: new .....	44
\csGlsXtrLetField: new .....	44	\GlsXtrLoadResources: removed restriction on only one per document	156
\eGlsXtrSetField: new .....	45	\glxstrlocrangefmt: new .....	149
\gGlsXtrSetField: new .....	44	\glxstrpostlongdescription: new ..	52
\glxnoidxdisplayloc: added redefinition .....	148	\glxstrprovidestoragekey: new ....	34
\glssettocitle: added patch .....	53	\GlsXtrRecordCounter: new .....	159
\glxstr@counterrecord: new .....	159	\glxstrresourcecount: new .....	156
\glxstr@langtag: new .....	157	\glxstrresourcefile: added catcode change for @ .....	156
\glxstr@newabbreviation: new ....	227	\glxstrsetaliasnoindex: new ....	101
\glxstr@org@newignoredglossary: Added check for existence .....	52	\GlsXtrSetField: new .....	44
\glxstr@pluralsuffixes: new .....	157	\glxstrsetfieldifexists: new ....	44
\glxstr@provideignoredglossary: new .....	53	\glxstrunsrtdo: new .....	168
\glxstr@s@newignoredglossary: Added check for existence .....	52	\GlsXtrusefield: new .....	43
\glxstr@s@provideignoredglossary: new .....	54	\glxtrusefield: new .....	43
\glxstr@abbrvpluralsuffix: new ...	232	short-postlong-user: new .....	331
\glxstr@alias: new .....	63	short-postlong-user-desc: new ...	333
\glxstr@copytoglossary: new .....	54	\longnewglossaryentry: added starred version .....	51
\glxstr@deffield: new .....	43	long-postshort-user: new .....	329
\glxstr@displayendloc: new .....	149	long-postshort-user-desc: new ...	331
\glxstr@displayendlochook: new ...	149	postdot: new .....	20
\glxstr@displayingleloc: new ....	148	\pretoglossarypreamble: new .....	49
\glxstr@displaystartloc: new ....	148	\print@noop@unsrtglossaryunit: new .....	168
\glxstr@dohyperlink: added check for alias field .....	105	\print@op@unsrtglossaryunit: new	167
\glxstr@deffield: new .....	43	\printunsrtglossary: added starred form .....	161
\glxstr@entryfmt: new .....	36	\printunsrtglossaryhandler: new .	167
\glxstr@fieldddolistloop: new .....	37	\printunsrtglossaryunit: new ....	13
\glxstr@fieldforlistloop: new .....	37	\printunsrtglossaryunitsetup: new	167
		\provideignoredglossary: new ....	53
		\s@glxstr@provide@storagekey: new	34
		\s@printunsrtglossary: new .....	162
		\xGlsXtrSetField: new .....	45

1.13 (2017-02-07)	
\@glsdisp: removed	
\@glxtr@org@glsdisp .....	77
\glxtrsetaliasnoindex: switched to	
\providecommand .....	101
1.14 (2017-04-18)	
\@gls@link: added redefinition .....	80
\@gls@noidx@getgrouptitle: new ..	145
\@gls@removespaces: new .....	149
\@glxtr@do@automake@err: new ...	159
\@glxtr@org@gloautosee: new .....	30
\@glxtr@record: added third arg .....	7
\@glxtr@recordsee: new .....	13
General: added \glsadd option	
theHvalue .....	83
added \glsadd option thevalue ....	83
\glsdisablehyper: added redefinition	106
\glsenableentrycount: fixed	
assignment of \@cGls@ .....	120
\glsenableentryunitcount: fixed	
assignment of \@cGls@ .....	128
\glsnavigation: new .....	147
\glxtr@org@getgrouptitle: new ..	146
\glxtr@recordsee: new .....	7
\glxtr@writefields: added check for	
automake .....	158
\glxtrdisplayendloc: added check	
for empty format .....	149
\glxtrgetgrouptitle: new .....	146
\glxtrinitwrgloss: new .....	78
\glxtrlocationhyperlink: new ...	150
\glxtrsetgrouptitle: new .....	146
\glxtrsupphypernumber: new .....	150
\ifglxtrwrglossbefore: new .....	78
1.15 (2017-05-10)	
\@glxtr@dorecord: corrected	
premature expansion of \@glslocref	10
short-em-long-em: fixed spelling of	
\glsabbrvfont .....	308
short-long: fixed spelling of	
\glsabbrvfont .....	252
short-long-user: fixed spelling of	
\glsabbrvfont .....	335
short-postfootnote-desc: fixed	
spelling of \glsabbrvfont .....	259
short-postlong-user: fixed spelling of	
\glsabbrvfont .....	332
short-postlong-user-desc: fixed	
spelling of \glsabbrvfont .....	333
long-em-short-em: fixed spelling of	
\glsabbrvfont .....	305
long-postshort-user: fixed spelling of	
\glsabbrvfont .....	329
long-postshort-user-desc: fixed	
spelling of \glsabbrvfont .....	331
long-short: fixed spelling of	
\glsabbrvfont .....	249
long-short-user: fixed spelling of	
\glsabbrvfont .....	328
footnote: fixed spelling of	
\glsabbrvfont .....	255
postfootnote: fixed spelling of	
\glsabbrvfont .....	257
1.16 (2017-06-15)	
\@glo@autosee: added redefinition ....	30
\@gls@noidx@getgrouptitle: fixed	
bug .....	145
\@glxtr@addunusedxrefs: added	
check for seealso field .....	64
\@glxtr@checkgroup: use \csuse	
instead of \csname .....	169
\@glxtr@dorecordnodefer: new ....	11
\@glxtr@record@only@setup: added	
check for \@gls@setupsort@none ..	16
\@print@unsrt@glossary: corrected	
misspelt command .....	162
\@printunsrt@glossary@handler:	
new .....	167
\gls@checkseeallowed: added	
redefinition .....	30
\glxtr@writefields: added	
\providecommand lines .....	157
\glxtrautoindex: new .....	213
\glxtrautoindexassignsort: new ..	213
\glxtrautoindexentry: new .....	213
\glxtrindexseealso: new .....	61
\glxtrseealsolabels: new .....	63
\glxtrseelist: new .....	60
\glxtruseseealso: new .....	59
\glxtruseseealsoformat: new ....	60
\seealso: new .....	61
autoseeindex: new .....	19
1.17 (2017-08-09)	
\@glxtr@mark@wordseps: new .....	226
\@glxtr@markwordseps: new .....	226
\@glxtr@noidx@displaynumberlist:	
replace hard-coded ?? with	
\glxtrundeftag .....	144



\@glxstr@noidx@entrynumberlist: replace hard-coded ?? with \glxstrundeftag .....	145	\Glsxtrsubsequentfmt: new .....	245
\@glxstr@noidx@numberlistloop: replace hard-coded ?? with \glxstrundeftag .....	145	\glxtrsubsequentfmt: new .....	245
\@glxstrifhyphenstart: new .....	337	\Glsxtrsubsequentplfmt: new .....	245
General: removed some inconsistencies in the abbreviation styles .....	249	\glxtrsubsequentplfmt: new .....	245
\glssabbrvhyphenfont: new .....	338	\glxtrword: new .....	226
\glssabbrvonlyfont: new .....	352	\glxtrwordsep: new .....	226
\glssabbrvscfont: new .....	269	short-hyphen-long-hyphen: new ...	347
\glssabbrvsmfont: new .....	285	short-hyphen-long-hyphen-desc: new .....	348
\glssabbrvuserfont: initialised to default font .....	327	short-hyphen-postlong-hyphen: new	349
\glssfirstabbrvhyphenfont: new ...	338	short-hyphen-postlong-hyphen-desc: new .....	351
\glssfirstabbrvonlyfont: new .....	352	short-long-user-desc: corrected first forms .....	336
\glssfirstabbrvscfont: new .....	269	short-nolong-desc-noreg: new ...	264
\glssfirstabbrvsmfont: new .....	285	short-nolong-noreg: new .....	262
\glssfirstlonghyphenfont: new ...	338	long-em-noshort-em-desc-noreg: new .....	321
\glssfirstlongonlyfont: new .....	352	long-em-noshort-em-noreg: new ...	317
\glslonghyphenfont: new .....	338	long-hyphen-noshort-desc-noreg: new .....	340
\glslongonlyfont: new .....	352	long-hyphen-noshort-noreg: new ..	342
\glslonguserfont: initialised to default font .....	327	long-hyphen-postshort-hyphen: new	343
\glxstr@newabbreviation: added \glxstrorgshort and \glxstrorglong .....	227	long-hyphen-postshort-hyphen-desc: new .....	345
\GlsXtrDefineAcShortcuts: new ...	22	long-hyphen-short-hyphen: new ...	338
\glxstrgenabbrvfmt: added check for \ifglxstrinsertinside .....	243	long-hyphen-short-hyphen-desc: new .....	339
\glxstrhyphensuffix: new .....	338	long-noshort-desc-noreg: new ...	267
\glxstrifhyphenstart: new .....	337	long-noshort-noreg: new .....	268
\glxstrlonghyphen: new .....	342	long-only-short-only: new .....	352
\glxstrlonghyphennoshort: new ...	340	long-only-short-only-desc: new ..	354
\glxstrlonghyphenshort: new .....	337	long-short-user-desc: corrected first forms .....	334
\glxstrlongshortdescname: new ...	251	1.18 (2017-08-10) stylemods: changed default value to "default" .....	25
\glxstronlydescname: new .....	354	1.19 (2017-09-09) \@glxstr@defaultnumberformat: new .	7
\glxstronlydescsort: new .....	354	\@glxstr@dorecord: Use \@glxstr@recordloc instead of \@glxstr@recordloc .....	10
\glxstronlysuffix: new .....	352	\@glxstr@dorecordnodefer: Use \theglstrycounter for the location rather than \@glxstr@recordloc ..	11
\glxstrtparen: new .....	230	\@glxstr@record@setting: new .....	14
\glxstrposthyphenlong: new .....	349	\@glxstr@record@setting@alsoindex: new .....	14
\glxstrposthyphenshort: new .....	343	\@glxstrifhasfield: new .....	40
\glxstrposthyphensequent: new	343		
\glxstrshortdescname: new .....	262		
\glxstrshorthyphen: new .....	348		
\glxstrshorthyphenlong: new .....	346		
\glxstrshortlongdescname: new ...	253		
\glxstrshortlongdescsort: new ...	253		

General: added \glslink option		\@rGlspl@: new .....	176
theHvalue .....	79	\@rgls: new .....	175
added \glslink option thevalue ...	79	\@rgls@: new .....	175
\glsxtr@writefields: removed		\@rglspl: new .....	175
double-quotes around \jobname ..	158	\@rglspl@: new .....	175
\glsxtrdoautoindexname: changed		General: adjusted mcolalttree .....	462
format test .....	212	modified index to remove hard coded	
\glsxtrhyperlink: new .....	105	\space .....	439
\glsxtrifhasfield: new .....	40	modified list to remove hard coded	
\GlsXtrSetDefaultNumberFormat:		\space .....	427
new .....	7	moved conditional outside of	
\s@glsxtrifhasfield: new .....	41	\glsgroupskip .....	431–438
1.20 (2017-09-11)		new .....	465
\@glsxtrhypernameprefix: new ....	142	redefined altlistgroup to discourage	
\glsdohypertarget: added redefinition	143	breaks after group headings .....	429
\printunrtglossaryunitsetup:		redefined altlisthypergroup to	
switched from redefining		discourage breaks after group	
\gloinkprefix to		headings .....	430
\@glsxtrhypernameprefix .....	167	redefined alttreegroup to discourage	
1.21 (2017-11-03)		breaks after group headings .....	457
\@glsxtr@record: added check for		redefined alttreehypergroup to	
default options .....	9	discourage breaks after group	
\@glsxtrwrglossmark: new .....	27	headings .....	457
\@glslink: changed \let to \def ....	106	redefined indexgroup to discourage	
\@glsxtr@checkgroup: new .....	168	breaks after group headings .....	440
\@glsxtr@defpostpunc: new .....	19	redefined indexhypergroup to	
\@glsxtr@do@record@wrglossary:		discourage breaks after group	
new .....	8	headings .....	440
\@glsxtr@dosee@alsoindex@glossary:		redefined listgroup to discourage	
new .....	29	breaks after group headings .....	429
\@glsxtr@doseeglossary: new .....	29	redefined listhypergroup to	
\@glsxtr@noidx@do: removed code		discourage breaks after group	
dealing with the group .....	170	headings .....	429
\@glsxtr@record@setting@off: new .	15	redefined mcolalttreegroup to	
\@glsxtr@record@setting@only: new	14	discourage breaks after group	
\@glsxtr@rglstrigger@record: new	174	headings .....	462
\@glsxtrglossentry: new .....	160	redefined mcolalttreehypergroup to	
\@glsxtrnewgls: new .....	171	discourage breaks after group	
\@glsxtrsetaliasnoindex: changed to		headings .....	463
use \glsxtrifhasfield instead of		redefined mcolalttreespannav to	
\ifglshasfield .....	101	discourage breaks after group	
\@glsxtrwrglossmark: new .....	27	headings .....	463
\@rGLS: new .....	177	redefined mcolindexgroup to	
\@rGLS@: new .....	177	discourage breaks after group	
\@rGLSpl: new .....	177	headings .....	458
\@rGLSpl@: new .....	177	redefined mcolindexhypergroup to	
\@rGls: new .....	176	discourage breaks after group	
\@rGls@: new .....	176	headings .....	458
\@rGlspl: new .....	176		

redefined mcolindexspannav to		\glstreechildprelocation: new ...	439
discourage breaks after group		\glstreeprelocation: new .....	439
headings .....	459	\glstriggerrecordformat: new ....	175
redefined mcoltreegroup to		\glseuseabbrvfont: new .....	243
discourage breaks after group		\glseuselongfont: new .....	243
headings .....	459	\glxstr@do@alsoindex@wrglossary:	
redefined mcoltreehypergroup to		new .....	8
discourage breaks after group		\glxstr@org@@do@wrglossary: new ..	31
headings .....	460	\glxstr@org@dohyperlink: new ....	104
redefined mcoltreenonamegroup to		\glxstr@setbookindexmark: new ...	470
discourage breaks after group		\glxstrbookindexatendgroup: new ..	466
headings .....	461	\glxstrbookindexbetween: new ....	466
redefined		\glxstrbookindexbookmark: new ...	466
mcoltreenonamehypergroup to		\glxstrbookindexcols: new .....	465
discourage breaks after group		\glxstrbookindexcolspread: new ..	467
headings .....	461	\glxstrbookindexfirstmark: new ..	471
redefined mcoltreenonamespannav to		\glxstrbookindexfirstmarkfmt: new	471
discourage breaks after group		\glxstrbookindexformatheader: new	466
headings .....	461	\glxstrbookindexgroupskip: new ..	466
redefined mcoltreespannav to		\glxstrbookindexlastmark: new ...	471
discourage breaks after group		\glxstrbookindexlastmarkfmt: new	471
headings .....	460	\glxstrbookindexmarkentry: new ..	470
redefined treegroup to discourage		\glxstrbookindexname: new .....	465
breaks after group headings .....	443	\glxstrbookindexparentchildsep:	
redefined treehypergroup to		new .....	466
discourage breaks after group		\glxstrbookindexparentschildsep:	
headings .....	443	new .....	466
redefined treenonamegroup to		\glxstrbookindexprelocation: new	465
discourage breaks after group		\glxstrbookindexsubatendgroup:	
headings .....	445	new .....	466
redefined treenonamehypergroup to		\glxstrbookindexsubbetween: new ..	466
discourage breaks after group		\glxstrbookindexsubname: new ....	465
headings .....	445	\glxstrbookindexsubprelocation:	
debug: new .....	28	new .....	465
\gglsetwidest: new .....	446	\glxstrbookindexsubsubatendgroup:	
\gl:disablehyper: added check for		new .....	466
existence .....	106	\glxstrbookindexsubsubbetween:	
changed to use \def rather than \let	106	new .....	466
\gl:enablehyper: changed to use \def		\glxstrbookindexthepage: new ....	470
rather than \let .....	106	\glxstrdetoklocation: new .....	173
\Glsfmtname: new .....	370	\glxtrenablerecordcount: new ...	173
\Glsfmtname: new .....	370	\glxstrglossentry: new .....	159
\glshex: new .....	380	\glxstrgroupfield: new .....	168
\glslstchildpostlocation: new ..	427	\Glsxtrheadname: new .....	361
\glslstchildprelocation: new ...	427	\glxtrheadname: new .....	360
\glslstprelocation: new .....	427	\GlsXtrIfFieldEqStr: new .....	45
\gl:navhyperlink: patched .....	104	\glxstriflabelinlist: new .....	167
\gl:seeitemformat: new .....	57	\glxstrifrecordtrigger: new ....	174
\gl:showtarget: new .....	28		

\glxtrindexseealso: added check that the entry exists .....	61	\rGLSpl: new .....	177
\glxtrinithyperoutside: new .....	79	\rGlspl: new .....	176
\GlsXtrLocationRecordCount: new ..	173	\rglspl: new .....	175
\glxtrnewgls: new .....	170, 172	\rGLSplformat: new .....	178
\glxtrnewGLSlike: new .....	172	\rGlsplformat: new .....	178
\glxtrnewglslike: new .....	172	\rglsplformat: new .....	177
\glxtrnewrgls: new .....	172	\s@glxtrifhasfield: switched from \ifdef to \ifundef .....	41
\glxtrnewrGLSlike: new .....	172	1.22 (2017-11-08)	
\glxtrnewrglslike: new .....	172	\@glxtr@nopostpunc: new .....	141
\glxtrprelocation: new .....	426, 465	\@glxtr@orgprintglossary: changed explicit \let for \nopostdesc to \glxtractivatenopost .....	140
\GlsXtrRecordCount: new .....	173	\@glxtrglossentryother: new ....	161
\glxtrrecordtriggervalue: new ..	173	\glossentrynameother: new .....	210
\glxtrresourcefile: now disables record key .....	155	\glseeitemformat: switched check from regular to short .....	57
\glxtrresourceinit: new .....	156	\glxtr@setaccessdisplay: new ...	210
\GlsXtrSetRecordCountAttribute: new .....	173	\glxtr@writefields: provide \glxtr@record in aux file .....	157
\Glsxtrtitlename: new .....	361	\glxtractivatenopost: new .....	141
\glxtrtitlename: new .....	361	\glxtrbookindexprelocation: removed check for no post dot ....	465
\glxtrtitleorpdforheading: new ..	356	\glxtrglossentryother: new .....	160
\GlsXtrTotalRecordCount: new ....	173	\glxtrnopostpunc: new .....	141
\glxtrwrglossmark: new .....	27	1.23 (2017-11-12)	
short-em: new .....	311	\@@glxtrfmt: added check for indexing	36
short-sc: corrected first letter uppercasing .....	274	added grouping .....	35
short-sm: corrected first letter uppercasing .....	290	new .....	35
shortcuts: ac .....	24	\@glxtr@nopostpunc@postdesc: new	141
\ifglxtr@hyperoutside: new .....	79	\@glxtr@restore@postpunc: new ..	141
all: new .....	425	\@glxtrentryfmt: fixed missing label argument .....	36
nolong-short: new .....	264	\@glxtrfmt: new .....	35
nolong-short-em: new .....	313	\eglsupdatewidest: new .....	447
nolong-short-noreg: new .....	265	\gglupdatewidest: new .....	447
nolong-short-sc: new .....	276	\glsupdatewidest: new .....	447
nolong-short-sm: new .....	292	\GlsXtrDefineAbbreviationShortcuts: changed \newabbr definition to use \providecommand .....	22
nopostdot: new .....	20	\GlsXtrDefineAcShortcuts: changed \newabbr definition to use \providecommand .....	22
postpunc: new .....	20	\glxtrfmtdisplay: new .....	36
\printunsrtglossaryentryprocesshook: new .....	166	\glxtrifcustomdiscardperiod: new	221
\printunsrtglossarypredoglossary: new .....	167	\GlsXtrIfFieldUndef: new .....	42
\printunsrtglossaryskipentry: new	166	\glxtrrestorepostpunc: new ....	142
\rGLS: new .....	176	\s@glxtrfmt: new .....	35
\rGls: new .....	176	\s@glxtrfmt: new .....	35
\rgls: new .....	175		
\rGLSformat: new .....	178		
\rGlsformat: new .....	178		
\rglsformat: new .....	177		

\xglupdatewidest:new .....	448	\glxtrcombingdiacriticIrules:	
1.24 (2017-11-14)		new .....	396
\glsadd:added \@gls@setsort .....	84	\glxtrcombingdiacriticIVrules:	
\glxtrforcsvfield:new .....	38	new .....	397
\glxtrlocalsetgrouptitle:new ..	147	\glxtrcombingdiacriticrules:	
1.25 (2017-11-14)		new .....	396
\glxtrbookindexmulticolseenv:new	467	\glxtrcontrolrules:new .....	395
1.25 (2017-11-24)		\glxtrcurrencyrules:new .....	399
\glsextrapostnamehook:new .....	209	\glxtrdigitrules:new .....	422
\glxtrfootnotename:new .....	254	\glxtrfractionrules:new .....	423
\glxtrlongnoshortdescname:new ..	265	\glxtrGeneralLatinIIIrules:new	401
\glxtrlongnoshortname:new .....	267	\glxtrGeneralLatinIIrules:new ..	401
\glxtrlongshortname:new .....	249	\glxtrGeneralLatinIrules:new ..	400
\glxtrlongshortuserdescname:new	330	\glxtrGeneralLatinIVrules:new ..	402
\glxtronlyname:new .....	352	\glxtrGeneralLatinVIIrules:new	405
\glxtrpostlinkAddDescOnFirstUse:		\glxtrGeneralLatinVIrules:new ..	403
changed to use \glxtrparen ....	222	\glxtrGeneralLatinVrules:new ..	403
\glxtrpostlinkAddSymbolOnFirstUse:		\glxtrgeneralpuncIIrules:new ..	400
changed to use \glxtrparen ....	222	\glxtrgeneralpuncIrules:new ...	398
\glxtrshortlongname:new .....	251	\glxtrgeneralpuncrules:new ....	398
\glxtrshortlonguserdescname:new	333	\glxtrhyphenrules:new .....	398
\glxtrshortnolongname:new .....	260	\glxtrLatinA:new .....	406
1.26 (2018-01-05)		\glxtrLatinAA:new .....	408
\@glxtr@do@inc@linkcount:new ..	178	\glxtrLatinAELigature:new ....	408
\glslinkpresetkeys:new .....	79	\glxtrLatinE:new .....	406
\glxtr@inc@linkcount:new .....	79	\glxtrLatinEszettSs:new .....	407
\GlsXtrEnableLinkCounting:new ..	179	\glxtrLatinEszettSz:new .....	407
\GlsXtrIfLinkCounterDef:new ....	179	\glxtrLatinEth:new .....	407
\glxtrinclinkcounter:new .....	179	\glxtrLatinH:new .....	406
\GlsXtrLinkCounterName:new ....	179	\glxtrLatinI:new .....	406
\GlsXtrLinkCounterValue:new ....	179	\glxtrLatinInsularG:new .....	408
\GlsXtrTheLinkCounter:new .....	179	\glxtrLatinK:new .....	406
1.27 (2018-02-26)		\glxtrLatinL:new .....	406
\@glxtrdialecthook:new .....	32	\glxtrLatinLslash:new .....	408
General: added		\glxtrLatinM:new .....	406
glossaries-extra-bib2gls.sty .....	379	\glxtrLatinN:new .....	406
\Alpha:new .....	392	\glxtrLatinO:new .....	406
\Beta:new .....	392	\glxtrLatinOELigature:new ....	408
\Chi:new .....	393	\glxtrLatinOslash:new .....	408
\Digamma:new .....	393	\glxtrLatinP:new .....	407
\Epsilon:new .....	392	\glxtrLatinS:new .....	407
\Eta:new .....	392	\glxtrLatinSchwa:new .....	407
\glxtr@loaddialect:new .....	378	\glxtrLatinT:new .....	407
\glxtrBasicDigitrules:new .....	422	\glxtrLatinThorn:new .....	407
\glxtrcombingdiacriticIIrules:		\glxtrLatinWynn:new .....	408
new .....	397	\glxtrLatinX:new .....	407
\glxtrcombingdiacriticIrules:		\glxtrMathGreekIIrules:new ....	414
new .....	396	\glxtrMathGreekIrules:new ....	413



\glstredfield: changed \csedef to \protected\csedef ..... 43	\glsaddpresetkeys: new ..... 83
\glstrlocalsetgrouptitle: changed \csedef \protected\csedef .... 147	\glsuserdescription: new ..... 327
\glstrsetgrouptitle: changed \csxdef \protected\csxdef .... 146	\glstrabbreviationfont: new ..... 76
1.29 (2018-04-09)	\GlsXtrDualBackLink: new ..... 382
\@gls@removespaces: added expansion 149	\GlsXtrDualField: new ..... 382
\@glstr@dorecord: don't suppress expansion of \@glsrecordlocref if counter isn't page ..... 11	\GlsXtrExpandedFmt: new ..... 80
\@glstr@wrglossary@locationhyperlink: new ..... 27	\GLSxtrlong: added \@glstr@record 239
\glstr@inc@wrglossaryctr: new ... 26	\Glsxtrlong: added \@glstr@record 238
\glstr@wrglossarylocation: new . 381	\glstrlong: added \@glstr@record 237
\GlsXtrBibTeXEntryAliases: new .. 382	\GLSxtrlongpl: added \@glstr@record ..... 242
\glstrfieldforlistloop: corrected argument order in \forlistcsloop 37	\Glsxtrlongpl: added \@glstr@record ..... 242
\GlsXtrIndexCounterLink: new .... 381	\glstrlongpl: added \@glstr@record ..... 241
\GlsXtrInternalLocationHyperlink: new ..... 26	\GLSxtrshort: added \@glstr@record ..... 237
\GlsXtrProvideBibTeXFields: new . 382	\Glsxtrshort: added \@glstr@record ..... 236
indexcounter: new ..... 27	\glstrshort: added \@glstr@record ..... 235
\setentrycounter: new ..... 149	\GLSxtrshortpl: added \@glstr@record ..... 240
1.30 (2018-04-25)	\Glsxtrshortpl: added \@glstr@record ..... 240
\@glstr@record: added check for post-key hook ..... 9	\@glstr@record ..... 239
added check for pre-key hook ..... 9	\GlsXtrStartUnsetBuffering: new . 116
\@GLSxtr@fullpl: added \@glstr@record ..... 235	\GlsXtrStopUnsetBuffering: new .. 116
\@GlsXtrStopUnsetBuffering: new . 116	indexcounter: added check for wrglossary counter ..... 27
\@Glsxtr@fullpl: added \@glstr@record ..... 234	\s@GlsXtrStopUnsetBuffering: new 117
\@glstr@dorecord: don't suppress expansion of \@glsrecordlocref .. 11	1.31 (2018-05-09)
\@glstr@full: added \@glstr@record ..... 232	\@GlsXtrStartUnsetBuffering: new 116
\@glstr@fullpl: added \@glstr@record ..... 234	\@gls@ifaccessattribute@set: new 188
\@glstr@glossadd@postkeys: new .. 10	\@gls@initaccesskeys: new ... 188, 197
\@glstr@glossadd@prekeys: new ... 10	\@gls@setup@default@short@access: new ..... 190
\@glstr@glslink@postkeys: new ... 10	\@glstr@record@noglossarywarning: new ..... 155
\@glstr@glslink@prekeys: new .... 10	\@glstrbuffer@nodup@unset: new . 116
\@glstr@local@textformat: new ... 79	General: added prefix key for glslink . 79
\@glstr@unset: new ..... 115	added prefix key for printgloss .. 142
\@glstrbuffer@unset: new ..... 116	changed \let to \def ..... 142
\glsadd: added \glsaddpostsetkeys . 83	\glsaddeach: new ..... 84
added \glsaddpresetkeys ..... 83	\glsapturedgroup: new ..... 380
\glsaddpostsetkeys: new ..... 83	\glsdefpostdesc: new ..... 220
	\glsdefpostlink: new ..... 221
	\glsdefpostname: new ..... 209

<code>\glsdohypertarget</code> : bug fix: ensure that new version is picked up .....	143	<code>\if@glxtrdocdefrestricted</code> : changed to allow for atom as well ...	18
<code>\glslistdesc</code> : new .....	427	<code>docdef</code> : atom .....	18
<code>\glslocalreseteach</code> : new .....	117	1.35 (2018-08-13)	
<code>\glslocalunseteach</code> : new .....	118	<code>\@gl@link@</code> : initialise post-link hook commands .....	78
<code>\glstreechilddesc</code> : new .....	442	1.36 (2018-08-18)	
<code>\glstreechildsymbol</code> : new .....	442	<code>\glxtrautoindexesc</code> : new .....	213
<code>\glstreedefaultnamefmt</code> : new .....	438	<code>\glxtrdisplaysupplc</code> : new .....	383
<code>\glstreedesc</code> : new .....	441	<code>\glxtrmultisupplcation</code> : new ...	383
<code>\glstreegroupheaderfmt</code> : added redefinition .....	439	1.37 (2018-11-30)	
<code>\glstreenamefmt</code> : added redefinition .	438	<code>\@glxtr@record</code> : added check for auto-add .....	9
<code>\glstreenavigationfmt</code> : added redefinition .....	439	<code>\@dGLS</code> : new .....	391
<code>\glstreenonamechilddesc</code> : new ....	444	<code>\@dGLSpl</code> : new .....	392
<code>\glstreenonamedesc</code> : new .....	443	<code>\@dGls</code> : new .....	391
<code>\glstreenonamesymbol</code> : new .....	444	<code>\@dGlspl</code> : new .....	391
<code>\glstreesymbol</code> : new .....	442	<code>\@dgls</code> : new .....	391
<code>\glxtr@newabbreviation</code> : added <code>\ExtraCustomAbbreviationFields</code> .....	227	<code>\@dglspl</code> : new .....	391
<code>\GlsXtrForUnsetBufferedList</code> : new	117	<code>\@gl@getcounterprefix</code> : new .....	31
<code>\GlsXtrIfFieldCmpNum</code> : new .....	42	<code>\@glslongextrawidestname</code> : new ...	474
<code>\GlsXtrIfFieldEqNum</code> : new .....	41	<code>\@glxtr@bibgls@removespaces</code> : new	385
<code>\GlsXtrIfFieldEqXpStr</code> : new .....	45	<code>\@glxtr@check@bibgls@nameref</code> : new .....	156
<code>\GlsXtrIfFieldNonZero</code> : new .....	41	<code>\@glxtr@do@nameref@record</code> : new ..	12
<code>\GlsXtrIfHasNonZeroChildCount</code> : new .....	380	<code>\@glxtr@get@prefixedlabel</code> : new .	390
<code>\GlsXtrIfXpFieldEqXpStr</code> : new .....	46	<code>\@glxtr@if@record@only</code> : new .....	15
<code>\glxtrpostlinkAddSymbolDescOnFirstUse</code> : new .....	222	<code>\@glxtr@ifnum@mmode</code> : new .....	12
<code>\GlsXtrRecordWarning</code> : new .....	153	<code>\@glxtr@labelprefixes</code> : new .....	389
<code>\glxtrRevertToCmarks</code> : new .....	356	<code>\@glxtr@prefixlabellist</code> : new ...	390
<code>\GlsXtrStandaloneGlossaryType</code> : new .....	160	<code>\@glxtr@providenewgls</code> : new .....	170
<code>\GlsXtrStandaloneSubEntryItem</code> : new .....	160	<code>\@glxtr@record@only@setup</code> : new ..	15
<code>\s@GlsXtrStartUnsetBuffering</code> : new	116	<code>\@glxtr@record@setting@nameref</code> : new .....	15
1.32 (2018-05-24)		<code>\@glxtr@use@equation@counter@or</code> : new .....	80
<code>\GlsXtrForeignText</code> : new .....	46	General: new .....	472
<code>\GlsXtrForeignTextField</code> : new .....	48	page: nameref .....	11
<code>\GlsXtrUnknownDialectWarning</code> : new	48	<code>\dGLS</code> : new .....	391
1.33 (2018-07-26)		<code>\dgls</code> : new .....	391
<code>\ifglsused</code> : added redefinition .....	50	<code>\dglsdisp</code> : new .....	392
1.34 (2018-07-29)		<code>\dglslink</code> : new .....	392
<code>\gls@begindocdefs</code> : atom .....	66	<code>\dGLSpl</code> : new .....	391
<code>\GlsXtrIfUnusedOrUndefined</code> : new ..	32	<code>\glsadd</code> : added grouping .....	83
<code>\glxtrNoGlossaryWarning</code> : added package warning .....	25	ensure that <code>\glsadd</code> performs indexing	84
		<code>\glslongextraDescAlign</code> : new .....	474
		<code>\glslongextraDescFmt</code> : new .....	472
		<code>\glslongextraDescNameHeader</code> : new	479



\glslongextraDescNameTabularFooter:	\glslongextraNameDescSymTabularFooter:
new ..... 479	new ..... 482
\glslongextraDescNameTabularHeader:	\glslongextraNameDescSymTabularHeader:
new ..... 479	new ..... 482
\glslongextraDescSymNameHeader:	\glslongextraNameDescTabularFooter:
new ..... 491	new ..... 474
\glslongextraDescSymNameTabularFooter:	\glslongextraNameDescTabularHeader:
new ..... 491	new ..... 474
\glslongextraDescSymNameTabularHeader:	\glslongextraNameFmt: new ..... 472
new ..... 491	\glslongextraNameSymDescHeader:
\glslongextraGroupHeading: new .. 474	new ..... 485
\glslongextraHeaderFormat: new .. 474	\glslongextraNameSymDescLocationHeader:
\glslongextraLocationAlign: new . 474	new ..... 486
\glslongextraLocationDescNameHeader:	\glslongextraNameSymDescLocationTabularFooter:
new ..... 480	new ..... 487
\glslongextraLocationDescNameTabularFooter:	\glslongextraNameSymDescLocationTabularHeader:
new ..... 480	new ..... 486
\glslongextraLocationDescNameTabularHeader:	\glslongextraNameSymDescTabularFooter:
new ..... 480	new ..... 485
\glslongextraLocationDescSymNameHeader:	\glslongextraNameSymDescTabularHeader:
new ..... 492	new ..... 485
\glslongextraLocationDescSymNameTabularFooter:	\glslongextraSetDescWidth: new .. 475
new ..... 493	\glslongextraSetWidest: new .... 474
\glslongextraLocationDescSymNameTabularHeader:	\glslongextraSubDescFmt: new .... 473
new ..... 493	\glslongextraSubLocationFmt: new 473
\glslongextraLocationFmt: new ... 473	\glslongextraSubNameFmt: new .... 473
\glslongextraLocationSymDescNameHeader:	\glslongextraSubSymbolFmt: new .. 473
new ..... 489	\glslongextraSymbolAlign: new ... 474
\glslongextraLocationSymDescNameTabularFooter:	\glslongextraSymbolFmt: new ..... 472
new ..... 490	\glslongextraSymDescNameHeader:
\glslongextraLocationSymDescNameTabularHeader:	new ..... 488
new ..... 489	\glslongextraSymDescNameTabularFooter:
\glslongextraLocSetDescWidth: new 476	new ..... 488
\glslongextraNameAlign: new ..... 474	\glslongextraSymDescNameTabularHeader:
\glslongextraNameDescHeader: new 474	new ..... 488
\glslongextraNameDescLocationHeader:	\glslongextraSymLocSetDescWidth:
new ..... 477	new ..... 476
\glslongextraNameDescLocationTabularFooter:	\glslongextraSymSetDescWidth: new 475
new ..... 478	\glslongextraTabularVAlign: new . 476
\glslongextraNameDescLocationTabularHeader:	\glslongextraUpdateWidest: new .. 475
new ..... 477	\glslongextraUpdateWidestChild:
\glslongextraNameDescSymHeader:	new ..... 475
new ..... 482	\glsrenewcommand: new ..... 381
\glslongextraNameDescSymLocationHeader:	\glsseeitemformat: removed reference
new ..... 483	to \glslabel ..... 57
\glslongextraNameDescSymLocationTabularFooter:	\glsxtr@dblfloat: new ..... 19
new ..... 483	\glsxtr@do@autoadd: new ..... 80
\glslongextraNameDescSymLocationTabularHeader:	\glsxtr@float: new ..... 19
new ..... 483	\glsxtr@record@nameref: new ..... 159

\glstr@renewcommand: new	381	1.38 (2018-12-01)	
\glstr@writefields: provide			\glslongextraNameFmt: bug fix:
\glstr@record@nameref in aux			removed double param
file	157		all: added glossary-longextra
\glstraddlabelprefix: new	389	1.39 (2019-03-22)	
\GlsXtrAutoAddOnFormat: new	80		\@GlsXtrIfFieldCmpNum: new
\glstrclearlabelprefixes: new	389		\@GlsXtrIfFieldEqNum: new
\glstrdisplaylocnameref: new	383		\@GlsXtrIfFieldEqStr: new
\glstrfmtexternalnameref: new	386		\@GlsXtrIfFieldEqXpStr: new
\glstrfmtinternalnameref: new	386		\@GlsXtrIfFieldNonZero: new
\GLSXRhiername: new	59		\@GlsXtrIfXpFieldEqXpStr: new
\GLSxtrhiername: new	59		\@glstr@removespaces: changed \x to
\GlsXtrhiername: new	58		\@glo@tmp
\Glsxtrhiername: new	58		\@glstr@dorecord: added protection
\glstrhiername: new	58		for fragile commands
\glstrhiernamesep: new	59		General: added label key for
\glstridentifyglsl-like: new	171		printgloss
\glstrifinlabelprefixlist: new	390		\glstrbookindexlocation: new
\GlsXtrLocationField: new	169		\glstrbookindexsublocation: new
\glstrnameloclink: new	385		\glstrentryparentname: new
\glstrnamerefink: new	384		\GlsXtrIfFieldCmpNum: added starred
\glstrprependlabelprefix: new	390		version
\GlsXtrSetAltModifier: write modifier			\GlsXtrIfFieldEqNum: added starred
to aux	104		version
\glstrSetWidest: new	386		\GlsXtrIfFieldEqStr: added starred
\glstrSetWidestFallback: new	388		form
\GlsXtrStandaloneEntryName: new	160		\GlsXtrIfFieldEqXpStr: added starred
\GlsXtrStandaloneEntryOther: new	161		form
\GLSxtrusefield: new	43		\GlsXtrIfFieldNonZero: added starred
\Glsxtrusefield: fixed internal			version
command and added check for			\GlsXtrIfXpFieldEqXpStr: added
\texorpdfstring	43		starred form
\ifGlsLongExtraUseTabular: new	476		\glstrsetglossarylabel: new
floats: new	19		\glstrshortdescname: corrected to
long-desc-name: new	479		show long form as advertised in the
long-desc-sym-name: new	491		manual
long-loc-desc-name: new	481		short-desc: corrected to omit
long-loc-desc-sym-name: new	493		description key as advertised in the
long-loc-sym-desc-name: new	490		manual
long-name-desc: new	476		short-em-desc: bug fix: omit description
long-name-desc-loc: new	478		key as advertised in the manual
long-name-desc-sym: new	482		short-sc-desc: bug fix: omit description
long-name-desc-sym-loc: new	484		key as advertised in the manual
long-name-sym-desc: new	485		short-sm-desc: corrected to omit
long-name-sym-desc-loc: new	487		description key as advertised in the
long-sym-desc-name: new	488		manual
equations: new	19		\s@GlsXtrIfFieldCmpNum: new
			\s@GlsXtrIfFieldEqNum: new
			\s@GlsXtrIfFieldEqStr: new

\s@GlsXtrIfFieldEqXpStr: new . . . . .	46	1.41 (2019-04-09)	General: changed \thisgrptitle to	
\s@GlsXtrIfXpFieldEqXpStr: new . . .	46		\glxtrcurrentgrptitle . . . . .	470
1.4.2 (??)			\glslistgroupskip: new . . . . .	427
\@glossentrysymbol: new . . . . .	217		\glstopicAssignSubIndent: moved	
\glstentrypdfsymbol: new . . . . .	217		\par from \glstopicSubItem . . . .	497
1.40 (2019-03-22)			\glstopicSubItem: added check for	
General: new . . . . .	495		description . . . . .	498
\glstopicAssignSubIndent: new . . .	497		moved \par to	
\glstopicAssignWidest: new . . . . .	498		\glstopicAssignSubIndent . . . .	498
\glstopicCols: new . . . . .	499		\glstopicSubLoc: moved \space to	
\glstopicColsEnv: new . . . . .	499		\glstopicSubPreLocSep . . . . .	499
\glstopicDesc: new . . . . .	497		\glstopicSubPreLocSep: new . . . . .	499
\glstopicGroupHeading: new . . . . .	496		\glstreeChildDescLoc: new . . . . .	442
\glstopicInit: new . . . . .	497		\glstreeDescLoc: new . . . . .	441
\glstopicItem: new . . . . .	496		\glstreegroupskip: new . . . . .	439
\glstopicLoc: new . . . . .	497		\glstreePreHeader: new . . . . .	439
\glstopicMarker: new . . . . .	496		\glxtralttreeSymbolDescLocation:	
\glstopicMidSkip: new . . . . .	498		added check for description . . . . .	446
\glstopicName: new . . . . .	496		topic: added penalty if no description .	495
\glstopicParIndent: new . . . . .	497		topicmcols: added penalty if no	
\glstopicPostSkip: new . . . . .	498		description . . . . .	500
\glstopicPreSkip: new . . . . .	498	1.42 (2020-02-03)		
\glstopicSubIndent: new . . . . .	497		\@@glxtr@record: moved label	
\glstopicSubItem: new . . . . .	498		definition outside of conditional . . . . .	9
\glstopicSubItemBox: new . . . . .	499		\@ACRfull: added redefinition . . . . .	97
\glstopicSubItemSep: new . . . . .	499		\@ACRfullpl: added redefinition . . . . .	97
\glstopicSubLoc: new . . . . .	499		\@Acrfull: added redefinition . . . . .	97
\glstopicSubNameFont: new . . . . .	499		\@Acrfullpl: added redefinition . . . . .	97
\glstopicTitleFont: new . . . . .	497		\@GlsXtrIfFieldValueInCsvList:	
\glstopicwidest: new . . . . .	497		new . . . . .	39
all: added glossary-topic . . . . .	425		\@acrfull: added redefinition . . . . .	97
topic: new . . . . .	495		\@acrfullpl: added redefinition . . . . .	97
topicmcols: new . . . . .	499		\@gl@assign@actual: new . . . . .	189
1.40 (2019-03-31)			\@gl@entry@field: redefined . . . . .	50
\glfirstabbrvdefaultfont: changed			\@gl@setup@default@access: added	
definition from \glabbrvfont to			\gldefaultshortaccess . . . . .	190
\glabbrvdefaultfont for			\@gl@setup@default@short@access:	
consistency . . . . .	231		renamed to	
\GlsXtrDefaultResourceOptions:			\@gl@setup@default@access . .	190
new . . . . .	155		\@gl@link: switched from	
long-hyphen-noshort-noreg:			\gl@dohyperlink to	
corrected formatting commands . . .	342		\glxtr@dohyperlink . . . . .	106
\printunsrtabbreviations: new . . .	380		\@glxtr@abbrlists: new . . . . .	132
\printunsrtacronyms: new . . . . .	379		\@glxtr@acronymlists: new . . . . .	132
\printunsrtindex: new . . . . .	379		\@glxtr@addabbreviationlist: new	132
\printunsrtnumbers: new . . . . .	380		\@glxtr@base@acrcmd: new . . . . .	92
\printunsrtsymbols: new . . . . .	379		\@glxtr@doloadprefix: new . . . . .	24

\@glstr@org@addtoacronymlists: new .....	132	removed \relax and updated \@glshort instead of \glsshorttok .....	228
\@glstr@org@setacronymlists: new	132	replaced explicit \spacefactor with \@ .....	228
\@glstrentryfmt: added \glslabel and scope .....	36	\glstr@writefields: added check for order=letter .....	158
General: added \@afterheading .....	459	\glstrAccSuppAbbrSetFirstLongAttrs: new .....	193, 197
debug: showaccsupp .....	28	\glstrAccSuppAbbrSetNameLongAttrs: new .....	193, 197
\forall abbreviationlists: new ...	132	\glstrAccSuppAbbrSetNameShortAttrs: new .....	193, 197
\forall acronyms: new .....	133	\glstrAccSuppAbbrSetNoLongAttrs: new .....	192, 197
\gldefaultshortaccess: new .....	189	\glstrAccSuppAbbrSetTextShortAttrs: new .....	193, 197
\gldisplaynumberlist: added .....	380	\glstralttreeSymbolDescLocation: switched to using \glstreeDescLoc	446
\glenablehyper: switched from \glsdohyperlink to \glstrdohyperlink .....	106	\glstrassignactualsetup: new ...	189
\glentrynumberlist: added .....	380	\glstrbookindexbookmarkprefix: new .....	467
\GLSfmtfirst: new .....	373	\GlsXtrDiscardUnsetBuffering: new	117
\GLSfmtfirstpl: new .....	374	\glstrdohyperlink: new (was former redefinition of \glsdohyperlink) .	104
\GLSfmtfull: new .....	377	\glstrequationlocfmt: new .....	384
\Glsfmtfull: switched pdf case to use \glspdffmtfull .....	377	\glstrfieldformatcsvlist: new ...	38
\glspdffmtfull: switched pdf case to use \glspdffmtfull .....	376	\glstrfieldformatlist: new .....	37
\GLSfmtfullpl: new .....	378	\glstrfootnotedescname: new ...	256
\Glsfmtfullpl: switched pdf case to use \glspdffmtfullpl .....	377	\glstrfootnotedescsort: new ...	256
\glspdffmtfullpl: switched pdf case to use \glspdffmtfullpl .....	377	\GLSXtrhiername: switched to using \GLSfmtttext and \GLSfmtname ...	59
\GLSfmtlong: new .....	375	\GLSxtrhiername: switched to using \glsfmmttext, \glsfmtname, \GLSfmtttext and \GLSfmtname ...	59
\GLSfmtlongpl: new .....	376	\GlsXtrhiername: switched to using \Glsfmmttext and \Glsfmtname ...	58
\GLSfmtname: new .....	371	\Glsxtrhiername: switched to using \glsfmmttext and \glsfmtname ...	58
\GLSfmtplural: new .....	372	\GlsXtrIfFieldValueInCsvList: new	39
\GLSfmtttext: new .....	372	\glstrpdfentryfmt: new .....	36
\glspdffmtfull: new .....	376	\glstrprovideaccsuppcmd: new ...	192
\glspdffmtfullpl: new .....	376	\glstrscsuffix: added \protect ..	269
\glseeitemformat: switched to using \glsfmmttext and \glsfmtname ...	57	\GlsXtrSetAltModifier: added check	103
\glsshowtarget: added check for \glsshowtargetouter .....	28	\GLSxtrtitlefirst: new .....	364
\glstreeChildDescLoc: added \glstreeNoDescSymbolPreLocation .....	442	\GLSxtrtitlefirstplural: new ...	365
\glstreegroupheaderskip: new ....	439	\GLSxtrtitlefull: new .....	368
\glstreeNoDescSymbolPreLocation: new .....	441		
\glstr@newabbreviation: moved apply abbreviation style to after category key has been obtained ...	227		

\GLSxtrtitlefullpl: new .....	369	moved \protect inside	
\GLSxtrtitlelong: new .....	366	\glxtrscsuffix .....	280
\GLSxtrtitlelongpl: new .....	367	short-sc-footnote-desc: new .....	282
\GLSxtrtitlename: new .....	361	short-sc-long: added missing text key	271
\GLSxtrtitleplural: new .....	363	moved \protect inside	
\GLSxtrtitleshort: new .....	360	\glxtrscsuffix .....	271
\GLSxtrtitleshortpl: new .....	360	short-sc-postfootnote: added	
\GLSxtrtitletext: new .....	362	missing text key .....	282
\glxtrusealias: new .....	60	moved \protect inside	
short-em: removed \protect from		\glxtrscsuffix .....	283
\glxtremsuffix .....	310	short-sc-postfootnote-desc: new .	284
short-em-desc: removed \protect		short-sm: removed \protect from	
from \glxtremsuffix .....	312	\glxtrsmsuffix .....	290
short-em-footnote: added missing text		short-sm-desc: removed \protect	
key .....	321	from \glxtrsmsuffix .....	291
removed \protect from		short-sm-footnote: added missing text	
\glxtremsuffix .....	322	key .....	296
short-em-footnote-desc: new .....	323	removed \protect from	
short-em-long: added missing text key	306	\glxtrsmsuffix .....	297
removed \protect from		short-sm-footnote-desc: new .....	298
\glxtremsuffix .....	307	short-sm-long: added missing text key	287
short-em-long-em: added missing text		removed \protect from	
key .....	308	\glxtrsmsuffix .....	288
removed \protect from		short-sm-postfootnote: added	
\glxtremsuffix .....	309	missing text key .....	299
short-em-postfootnote: added		removed \protect from	
missing text key .....	324	\glxtrsmsuffix .....	300
removed \protect from		short-sm-postfootnote-desc: new .	301
\glxtremsuffix .....	325	\makeglossaries: added	
short-em-postfootnote-desc: new .	326	\@domakeglossaries .....	135
short-footnote-desc: new .....	256	let \@makeglossary to \@gobble	
short-hyphen-long-hyphen: added		instead of \relax .....	136
missing text key .....	347	removed redefinition of	
short-hyphen-postlong-hyphen:		\makeglossary .....	137
added missing text key .....	349	\makenoidxglossaries: added	
short-long: added missing text key ...	252	\@domakeglossaries .....	66
short-long-user: added missing text		long-em-noshort-em: removed	
key .....	335	\protect from \glxtremsuffix .	316
short-postfootnote-desc: added		long-em-noshort-em-desc: removed	
missing text key .....	259	\protect from \glxtremsuffix .	319
new .....	259	long-em-short-em: added missing text	
short-postlong-user: added missing		key .....	304
text key .....	331	removed \protect from	
short-sc: moved \protect inside		\glxtremsuffix .....	305
\glxtrscsuffix .....	273	long-hyphen-noshort-desc-noreg:	
short-sc-desc: moved \protect inside		added missing text key .....	340
\glxtrscsuffix .....	275	long-hyphen-postshort-hyphen:	
short-sc-footnote: added missing text		added missing text key .....	343
key .....	280		

long-hyphen-short-hyphen: added missing text key	338	\@glxtr@noidx@do: replaced	
long-noshort-em: removed \protect from \glxtremsuffix	314	\ifglshasparent with	
long-noshort-em-desc: removed \protect from \glxtremsuffix	318	\@glxtr@ifischild	169
long-noshort-sc: moved \protect inside \glxtrscsuffix	277	\@print@unsrt@innerglossary: new	165
long-noshort-sc-desc: moved \protect inside \glxtrscsuffix	278	General: added groups key	143
long-noshort-sm: removed \protect from \glxtrsmsuffix	293	added leveloffset key	143
long-noshort-sm-desc: removed \protect from \glxtrsmsuffix	295	\doifglossarynoexistsordo: switched to starred form of	
long-only-short-only: added missing text key	352	\ifglossaryexists	56
removed \protect from \glxtronlysuffix	353	\glswriteentry: replaced \ifglused with	
long-postshort-user: added missing text key	329	\GlsXtrIfUnusedOrUndefined	102
long-short: added missing text key	249	\glxtr@printgloss@checkexists: new	140
long-short-em: added missing text key	302	\glxtralttreeSymbolDescLocation: removed duplicate description	446
removed \protect from \glxtremsuffix	303	\ifglossaryexists: added check for starred form	33
long-short-sc: added missing text key	269	\np@glxtr@assign@leveloffset: new	143
moved \protect inside \glxtrscsuffix	269	\p@glxtr@assign@leveloffset: new	143
long-short-sm: added missing text key	286	\pp@glxtr@assign@leveloffset: new	143
removed \protect from \glxtrsmsuffix	286	\printunsrtglossary: added check for \@printgloss@checkexists	161
long-short-user: added missing text key	328	\printunsrtinnerglossary: new	163
footnote: added missing text key	255	printunsrtglossarywrap: new	164
footnote-desc: new	257	1.45 (2020-04-01)	
postfootnote: added missing text key	257	General: removed duplicate description	444
prefix: new	25	\glstreenonameChildDescLoc: new	444
\RestoreAcronyms: added display style	134	\glstreenonameDescLoc: new	444
\s@GlsXtrIfFieldValueInCsvList: new	39	1.46 (2021-09-18)	
\seealso: add check for \also	61	\@glxtrsetaliasnoindex: changed to use starred version of	
1.42 (?)		\glxtrifhasfield	101
postfootnote-desc: new	260	1.46 (2021-09-20)	
1.43 (2020-02-28)		\@@glxtr@record: changed \edef to \protected@edef	9
\@glxtrentryfmt: changed \def to \edef to avoid infinite recursion	36	\@@newglossaryentry@defunitcounters: changed \edef to \protected@edef	124
1.44 (2020-03-23)		\@glossentrysymbol: changed \edef to \protected@edef	217
\@glxtr@assign@leveloffset: new	143	\@gl@increment@currunitcount: changed \edef to \protected@edef	125
\@glxtr@leveloffset: new	143	\@gl@link: changed \edef to \protected@edef	81, 82
		\@gl@link@checkfirsthyper: changed \edef to \protected@edef	100

<code>\@gls@local@increment@currunitcount:</code>	General: changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 126	<code>\protected@edef</code> ..... 63, 225
<code>\@gls@setup@default@access:</code>	record: added hybrid ..... 16
changed <code>\edef</code> to	<code>\glossentrydesc</code> : changed <code>\edef</code> to
<code>\protected@edef</code> ..... 190, 191	<code>\protected@edef</code> ..... 204, 205
<code>\@glxtr@addabbreviationlist:</code>	<code>\Glossentryname</code> : changed <code>\edef</code> to
changed <code>\eappto</code> to	<code>\protected@edef</code> ..... 208, 209
<code>\protected@eappto</code> ..... 133	<code>\glossentryname</code> : changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 132	<code>\protected@edef</code> ..... 205, 207
<code>\@glxtr@bibgls@removespaces:</code>	<code>\glossentrynameother</code> : changed <code>\edef</code>
changed <code>\x</code> to <code>\@glo@tmp</code> ..... 385	to <code>\protected@edef</code> ..... 210
<code>\@glxtr@do@inc@linkcount</code> : changed	<code>\glsadd</code> : changed <code>\edef</code> to
<code>\x</code> to <code>\@glo@tmp</code> ..... 179	<code>\protected@edef</code> ..... 83
<code>\@glxtr@do@record@wrglossary:</code>	<code>\glsalttreechildpredesc</code> : new .... 445
changed <code>\edef</code> to <code>\protected@edef</code> . 8	<code>\glsalttreepredesc</code> : new ..... 445
<code>\@glxtr@do@redef@forglsentries:</code>	<code>\glsdisablehyper</code> : changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> . 6	<code>\protected@edef</code> ..... 106
<code>\@glxtr@get@prefixedlabel:</code>	<code>\glsdoifexists</code> : changed <code>\edef</code> to
changed <code>\edef</code> to <code>\protected@edef</code> 390	<code>\protected@edef</code> ..... 55
changed <code>\x</code> to <code>\@glo@tmp</code> ..... 390	<code>\glsenableentryunitcount</code> : changed
<code>\@glxtr@mixed@assign@sortkey:</code>	<code>\edef</code> to <code>\protected@edef</code> ..... 128
changed <code>\edef</code> to <code>\protected@edef</code> 144	<code>\glsFindWidestLevelTwo</code> : changed
<code>\@glxtr@op@recordcounter</code> : changed	<code>\edef</code> to <code>\protected@edef</code> ..... 451
<code>\eappto</code> to <code>\protected@eappto</code> ... 13	<code>\glsFindWidestUsedLevelTwo</code> :
<code>\@glxtr@orgprintglossary</code> : changed	changed <code>\edef</code> to <code>\protected@edef</code> 450
<code>\xdef</code> to <code>\protected@xdef</code> ..... 140	<code>\glsnavhyperlink</code> : changed <code>\edef</code> to
<code>\@glxtr@rglstrigger@record:</code>	<code>\protected@edef</code> ..... 104
changed <code>\edef</code> to <code>\protected@edef</code> 174	<code>\glstopicAssignSubIndent</code> : bug 182
<code>\@glxtr@warn@hybrid@noprintgloss:</code>	maintain hangindent for multiple
new ..... 15	paragraphs ..... 497
<code>\@glxtr@entryfmt</code> : changed <code>\edef</code> to	<code>\glstopicsubitemhangindent</code> : new . 497
<code>\protected@edef</code> ..... 36	<code>\glstopicSubItemParIndent</code> : new .. 497
<code>\@glxtr@glossentry</code> : changed <code>\edef</code> to	<code>\glxtr@org@newignoredglossary</code> :
<code>\protected@edef</code> ..... 160	changed <code>\eappto</code> to
<code>\@glxtr@glossentryother</code> : changed	<code>\protected@eappto</code> ..... 52
<code>\edef</code> to <code>\protected@edef</code> ..... 161	changed <code>\edef</code> to <code>\protected@edef</code> . 52
<code>\@glxtr@indexaliased</code> : changed <code>\edef</code>	<code>\glxtr@provideignoredglossary</code> :
to <code>\protected@edef</code> ..... 101	changed <code>\eappto</code> to
<code>\@makeglossaries@warn@noprintglossary:</code>	<code>\protected@eappto</code> ..... 54
new ..... 135	changed <code>\edef</code> to <code>\protected@edef</code> . 54
<code>\@newglossaryentryposthook:</code>	<code>\glxtr@s@newignoredglossary</code> :
changed <code>\edef</code> to <code>\protected@edef</code> 64	changed <code>\edef</code> to <code>\protected@edef</code> 53
<code>\@print@unsrt@glossary</code> : changed	<code>\glxtr@s@provideignoredglossary</code> :
<code>\eappto</code> to <code>\protected@eappto</code> .. 163	changed <code>\edef</code> to <code>\protected@edef</code> 54
<code>\@print@unsrt@innerglossary:</code>	<code>\glxtr@setaccessdisplay</code> : changed
changed <code>\eappto</code> to	<code>\edef</code> to <code>\protected@edef</code> ..... 210
<code>\protected@eappto</code> ..... 166	<code>\glxtralttreeSymbolDescLocation</code> :
<code>\@printunsrt@glossary@handler:</code>	switch to using <code>\glsalttreepredesc</code>
changed <code>\xdef</code> to <code>\protected@xdef</code> 167	and <code>\glsalttreechildpredesc</code> .. 446

\glstrdisplayendloc: changed \edef to \protected@edef .....	149	1.47 (2017-11-14)	\@glstrforcsvfield: new .....	38
\glstrdisplaystartloc: changed \edef to \protected@edef .....	148	1.47 (2021-11-04)	\s@glstrforcsvfield: new .....	38
\glstrdoautoindexname: changed \eappto to \protected@eappto ..	212		\@GlsXtrIfHasNonZeroChildCount: new .....	381
\glstrseelist: changed \edef to \protected@edef .....	60		\@glstrsetaliasnoindex: changed to use \ifcvoid .....	101
\glstrtreechildpredesc: new ....	441		\glsaltlistitem: new .....	428
\glstrtreepredesc: new .....	441		\glslistexpandedname: new .....	428
\makeglossaries: adjust warning on missing glossary for “alsoindex” ...	136		\glslistgroupafterheader: new ...	429
changed \edef to \protected@edef .....	136, 137		\glslistinit: new .....	427
\makenoidxglossaries: changed \edef to \protected@edef .....	67		\glslistitem: new .....	427
topic: added \par (bug 176) .....	496		\glsseefirstitem: new .....	61
grouping added to scope \everypar (bug 182) .....	496		\glsseelastoxfordsep: new .....	61
printunsrtglossarywrap: changed \xdef to \protected@xdef .....	165		\glsseelist: redefined .....	60
\setabbreviationstyle: changed \edef to \protected@edef .....	246		\glunsetcategoryattribute: new .	198
1.47 ()			\glstrapptocsvfield: new .....	44
\@GlsXtrIfValueInFieldCsvList: new .....	39		\glstrcopytoglossary: replaced \cseappto with \protected@cseappto .....	55
\@xGlsXtrIfValueInFieldCsvList: new .....	40		\glstrfieldtitlecasecs: added check for \glscapitalisewords ..	203
\s@GlsXtrIfValueInFieldCsvList: new .....	40		\GlsXtrIfHasNonZeroChildCount: added starred version .....	380
\s@xGlsXtrIfValueInFieldCsvList: new .....	40		\GlsXtrIfValueInFieldCsvList: new	39
			\s@GlsXtrIfFieldNonZero: new .....	41
			\s@GlsXtrIfHasNonZeroChildCount: new .....	381
			\xGlsXtrIfValueInFieldCsvList: new .....	40



# 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	
\\$ .....	380
\, .....	59
\. ....	20, 221, 222, 438
\@ .....	65, 156, 189, 228
\@@cGLS@ .....	120, 128
\@@cGLSpl@ .....	120, 128
\@@cGls@ .....	120, 128
\@@cGlspl@ .....	120, 128
\@@cgls@ .....	120, 128
\@@cglspl@ .....	120, 128
\@@do@@wrglossary .....	8, 10, 137
\@@do@wrglossary ....	13, 15–17, 31, 84, 101
\@@glo@assign@sortkey .....	144
\@@glo@list .....	6
\@@glo@type .....	162
\@@glossarysec .....	466
\@@glossaryseclabel .....	142
\@@gls@expand@field .....	34
\@@glslocalreset .....	117
\@@glslocalunset .....	117
\@@glsreset .....	117
\@@glsunset .....	115
\@@glsxtr@autoindex@escspch ...	214–216
\@@glsxtr@base@acrcmd@warn ....	92, 133
\@@glsxtr@checkspch .....	213, 214, 216
\@@glsxtr@disabledflycommand .....	71
\@@glsxtr@org@postdescription ....	141
\@@glsxtr@record .....	15, 17
\@@glsxtr@recordcounter .....	15–17, 159
\@@glsxtrfmt .....	35
\@@glsxtrp .....	110, 111
\@@glsxtrpostloctag .....	74
\@@glsxtrpreloctag .....	74
\@@glsxtrwrglossmark .....	8, 9, 13, 29, 31, 61, 66, 137
\@@newglossaryentry@defcounters ...	119
\@@newglossaryentry@defunitcounters	126
\@@par .....	446
\@ACRlong .....	107
\@ACRlongpl .....	107
\@ACRshort .....	107
\@ACRshortpl .....	107
\@Acrlong .....	107
\@Acrlongpl .....	107
\@Acrshort .....	107
\@Acrshortpl .....	107
\@GLS@ .....	107, 122, 123, 177, 391
\@GLSdesc@ .....	89
\@GLSpl@ .....	107, 123, 177, 392
\@GLSplural@ .....	108
\@GLSsymbol@ .....	90
\@GLSxtr@full .....	233
\@GLSxtr@fullpl .....	235
\@GLSxtr@p@acrlong@ .....	107
\@GLSxtr@p@acrlongpl@ .....	107
\@GLSxtr@p@acrshort@ .....	107
\@GLSxtr@p@acrshortpl@ .....	107
\@GLSxtr@p@long@ .....	107
\@GLSxtr@p@longpl@ .....	107
\@GLSxtr@p@plural@ .....	107
\@GLSxtr@p@short@ .....	107
\@GLSxtr@p@shortpl@ .....	107
\@GLSxtr@p@text@ .....	107
\@GLSxtrlong .....	107, 238
\@GLSxtrlongpl .....	107, 242
\@GLSxtrp .....	114, 115
\@GLSxtrshort .....	107, 237
\@GLSxtrshortpl .....	107, 240
\@Gls@ .....	107, 122, 123, 176, 391
\@Gls@acrentryname .....	131
\@Gls@entry@field ....	43, 98, 113, 114, 211
\@Gls@entryname .....	131

\@GlsXtrEnableOnTheFly .....	68	\@cGLSpl .....	123
\@GlsXtrIfFieldCmpNum .....	41, 42	\@cGLSpl@ .....	120, 123, 128
\@GlsXtrIfFieldEqNum .....	41	\@cGls@ .....	120, 128
\@GlsXtrIfFieldEqStr .....	45	\@cGlspl@ .....	120, 128
\@GlsXtrIfFieldEqXpStr .....	45	\@cgl@ .....	120, 128
\@GlsXtrIfFieldNonZero .....	41, 381	\@cglspl@ .....	120, 128
\@GlsXtrIfFieldValueInCsvList .....	39	\@currentlabelname .....	142
\@GlsXtrIfHasNonZeroChildCount .....	380	\@dGLS .....	391
\@GlsXtrIfValueInFieldCsvList .....	39	\@dGLSpl .....	391
\@GlsXtrIfXpFieldEqXpStr .....	46	\@dGls .....	391
\@GlsXtrStartUnsetBuffering .....	116	\@dGlspl .....	391
\@GlsXtrStopUnsetBuffering .....	116	\@dblfloat .....	19
\@Glspl@ .....	107, 122, 123, 176, 391	\@dgls .....	391
\@Glsplural@ .....	108	\@dglspl .....	391
\@Gls@ .....	107	\@disable@onlypremakeg .....	137
\@Glsxtr .....	69, 71	\@do@auxoutstuff .....	150, 151
\@Glsxtr@full .....	233	\@do@gls@getcounterprefix .....	11, 12
\@Glsxtr@fullpl .....	234	\@do@glssee .....	63, 64
\@Glsxtr@p@acrlong@ .....	107	\@do@newglossaryentry .....	131, 229
\@Glsxtr@p@acrlongpl@ .....	107	\@do@seeglossary .....	15–17, 29, 66, 137
\@Glsxtr@p@acrshort@ .....	107	\@do@wrglossary .....	82, 83, 175
\@Glsxtr@p@acrshortpl@ .....	107	\@domakeglossaries .....	66, 135
\@Glsxtr@p@long@ .....	107	\@dtl@formatlist@handler .....	37
\@Glsxtr@p@longpl@ .....	107	\@dtl@formatlist@itemsep .....	37
\@Glsxtr@p@plural@ .....	107	\@dtl@formatlist@lastitem .....	37
\@Glsxtr@p@short@ .....	107	\@dtl@formatlist@prelastitem .....	37
\@Glsxtr@p@shortpl@ .....	107	\@dtl@formatlist@prelastitemsep .....	37
\@Glsxtr@p@text@ .....	107	\@dtlformatlist .....	38
\@Glsxtrlong .....	107, 238	\@empty .....	32, 84, 92– 96, 133, 141, 142, 149, 213, 214, 232–242
\@Glsxtrlongpl .....	107, 242	\@end@glsxtr@addunused .....	65
\@Glsxtrp .....	113, 114	\@end@glsxtr@gettype .....	139, 143
\@Glsxtrpl .....	70, 71	\@end@glsxtr@usesee .....	57
\@Glsxtrshort .....	107, 236	\@end@glsxtrifhyphenstart .....	337
\@Glsxtrshortpl .....	107, 240	\@endfortrue .....	38, 210, 246, 390
\@acrlong .....	107	\@firstofone .....	84, 163, 166, 189, 204, 205, 212, 217, 218, 384, 385, 427
\@acrlongpl .....	107	\@firstofthree .....	78, 84, 92–94, 96, 103, 232, 234, 236, 237, 239, 241
\@acrshort .....	107	\@firstoftwo .....	85–90, 93, 94, 96, 100, 103, 134, 169, 210, 223, 224, 233–235, 239–242, 356, 357
\@acrshortpl .....	107	\@float .....	19
\@addtoacronymlists .....	131–134	\@for .....	6, 25, 38, 60, 65, 84, 118, 119, 130, 132–134, 136, 139, 147, 163, 165, 173, 180, 203, 210, 218, 390
\@addtoreset .....	179	\@glo@alias .....	62, 63
\@afterheading .....	428, 429, 440, 443, 445, 458–462, 470, 496	\@glo@assign@sortkey .....	139
\@alt@gls@hyp@opt .....	103	\@glo@autosee .....	30
\@auxout .....	11–13, 67, 75, 104, 120, 129, 136, 137, 150, 151, 155, 157–159, 170, 171, 390, 470		
\@bibgls@restoreat .....	156		
\@cGLS .....	123		
\@cGLS@ .....	120, 123, 128		

\@glo@autoseehook .....	63	\@gls@checkmkidxchars .....	62, 213
\@glo@category .....	124	\@gls@codepage .....	151
\@glo@check@sortallowed .....	139	\@gls@counter .....	
\@glo@counterprefix . 11, 12, 31, 32, 149, 385		..... 9, 11, 12, 27, 32, 80, 81, 83, 101, 174	
\@glo@countunit .....	124	\@gls@currentlettergroup 147, 162, 165, 169	
\@glo@default@sorttype .....	139	\@gls@declareoption .....	5
\@glo@desc .....	51	\@gls@default@longpl .....	227–229
\@glo@descplural .....	51	\@gls@deffile .....	66
\@glo@group .....	14	\@gls@doautomake .....	139, 158
\@glo@label .....		\@gls@doautomake@err .....	158, 159
... 14, 34, 56, 57, 62–64, 98, 106, 448–455		\@gls@dolast .....	60, 61
\@glo@location .....	14	\@gls@donext .....	60, 61
\@glo@loclist .....	14	\@gls@enablesavenonumberlist ....	65, 66
\@glo@name .....	212, 213	\@gls@encapchar .....	214
\@glo@no@assign@sortkey .....	144	\@gls@entry@count .....	120
\@glo@parent .....	450, 451	\@gls@entry@field .....	
\@glo@see .....	56, 57, 60, 63–65	..... 34, 43, 63, 98, 112–115, 119, 161	
\@glo@seealso .....	62, 63	\@gls@entry@unitcount .....	128, 129
\@glo@sort .....	213	\@gls@field@font .....	84–92
\@glo@sorttype .....	139, 147	\@gls@field@link .....	85–92, 98, 99
\@glo@text .....	78	\@gls@firstaccess .....	187, 188, 192
\@glo@thislettergrp .....	169	\@gls@firstpluralaccess .....	188, 192
\@glo@thisvalue .....	327	\@gls@get@counterprefix .....	31, 32
\@glo@tmp . 31, 32, 34, 60, 98, 149, 179, 385, 390		\@gls@getcounterprefix .....	11
\@glo@type .....		\@gls@getgrouptitle .....	146, 162, 165
.. 64, 104, 131–133, 136, 140, 141, 143,		\@gls@grptitle .....	104, 147
144, 147, 148, 150, 151, 154, 155, 162–165		\@gls@hyp@opt .....	98,
\@glo@types .....	15, 135, 201, 448–455	99, 103, 123, 171, 175–177, 232–242, 391	
\@glossary@default@style .....		\@gls@hyp@opt@cs .....	103
..... 71, 72, 140, 164, 464		\@gls@ifaccessattribute@set ...	190, 191
\@glossarystyle .....	140, 164, 165	\@gls@ifinlist .....	167
\@glossentrysymbol .....	217	\@gls@increment@currcount .....	119
\@gls@ .....	107, 121, 123, 175, 391	\@gls@increment@currunitcount ....	127
\@gls@@automake@immediate .....	136	\@gls@initaccesskeys .....	227
\@gls@@link .....	78	\@gls@keymap .....	14, 34, 57, 62, 98, 157, 210
\@gls@ReturnAfterFi .....	149, 386	\@gls@label .....	
\@gls@actualchar .....	213	... 8, 9, 11, 12, 67, 102, 103, 137, 159, 246	
\@gls@actuallong .....	189, 190	\@gls@levelchar .....	214
\@gls@actuallongpl .....	189–191	\@gls@link .....	35, 76, 78, 92–96, 232–243
\@gls@actualshort .....	189–191	\@gls@link@checkfirsthyper .....	77, 134
\@gls@actualshortpl .....	189–191	\@gls@link@label .....	81, 174
\@gls@adjustmode .....	83	\@gls@link@nocheckfirsthyper .....	
\@gls@alt@hyp@opt .....	103	..... 76, 92–96, 232–242	
\@gls@alt@hyp@opt@char .....	103, 104	\@gls@link@opts .....	81
\@gls@alt@hyp@opt@keys .....	103, 104	\@gls@list .....	147
\@gls@assign@actual .....	190	\@gls@local@increment@currcount ...	119
\@gls@automake .....	139	\@gls@local@increment@currunitcount	127
\@gls@between .....	147	\@gls@location .....	169, 170
\@gls@checkedmkidx .....	213, 214, 216	\@gls@loclist .....	144, 145, 169, 170

\@gls@long	228	\@gls@xref	13, 61, 62
\@gls@longaccess	188, 190	\@glsabbrv@current@abbreviation	227, 243
\@gls@longaccesspl	188, 191	\@glsacronymlists	131–134
\@gls@longpl	189, 190, 225, 227–229	\@glsdoifexistsorwarn	18, 205, 207–210
\@gls@map	210	\@glsentry	66, 120, 129
\@gls@nameaccess	187, 188, 191	\@glslink	82, 104, 106
\@gls@nohyperlist	52, 54	\@glslocalreset	118
\@gls@noidx@do	147	\@glslocalunset	117, 118
\@gls@noidx@getgrouptitle	162, 165	\@glslongextra@begintab	476–490, 492–494
\@gls@noidx@nosanitizesort	138	\@glslongextrawidestname	474, 475
\@gls@noidx@sanitizesort	138	\@glsnextpages	140, 165
\@gls@noidx@loclist@finalsep	144	\@glsnonextpages	140, 165
\@gls@noidx@loclist@prev	144	\@glsnumberformat	9, 11, 12, 80, 81, 83, 101, 174, 209, 212, 385
\@gls@noidx@loclist@sep	144	\@glsorder	136
\@gls@noref@warn	137, 148	\@glspl@	107, 122, 123, 176, 391
\@gls@org@glsnoidxdisplayloc	145	\@glsplural@	107
\@gls@org@glsseeformat	145	\@glsplunc@token	224
\@gls@org@longpl	229	\@glsrecordlocref	11
\@gls@org@shortpl	229	\@glsseeitem	60, 61
\@gls@pluralaccess	187, 188, 191	\@glsseelastsep	60, 61
\@gls@preglossaryhook	141, 165, 218	\@glsshowtarget	105
\@gls@prevlevel	456–458, 462, 463	\@glsstyle@altlist	428
\@gls@quotechar	213	\@glsstyle@altlistgroup	429
\@gls@reference	67, 136, 137	\@glsstyle@altlisthypergroup	430
\@gls@restoreat	65, 66	\@glsstyle@alttree	445
\@gls@saveentrycounter	15–17, 31, 81, 83, 174	\@glsstyle@alttreegroup	457
\@gls@see@noindex	30, 155, 156	\@glsstyle@alttreehypergroup	457
\@gls@setdefault@glslink@opts	9, 36, 81, 102	\@glsstyle@index	439
\@gls@setsort	82, 84	\@glsstyle@indexgroup	440
\@gls@setup@default@access	229	\@glsstyle@indexhypergroup	440
\@gls@setupsort@none	16	\@glsstyle@inline	438
\@gls@short	228	\@glsstyle@list	427
\@gls@shortaccess	188, 190–192	\@glsstyle@listdotted	426
\@gls@shortaccesspl	188, 190–192	\@glsstyle@listgroup	429
\@gls@shortpl	189, 190, 225, 228, 229	\@glsstyle@listhypergroup	429
\@gls@sort	168	\@glsstyle@mcolalttree	462
\@gls@textaccess	187, 188, 191	\@glsstyle@mcolalttreegroup	462
\@gls@thisHloc	31	\@glsstyle@mcolalttreehypergroup	463
\@gls@thislabel	60, 61, 84, 118, 390–392	\@glsstyle@mcolalttreespannav	463
\@gls@thisloc	31	\@glsstyle@mcolindexgroup	458
\@gls@thisval	210	\@glsstyle@mcolindexhypergroup	458
\@gls@tmp	40, 45, 46, 147, 189, 190	\@glsstyle@mcolindexspannav	459
\@gls@tmpb	216	\@glsstyle@mcoltreegroup	459
\@gls@type	133, 134, 137, 139, 246, 448–455	\@glsstyle@mcoltreehypergroup	460
\@gls@write@entrycounts	120	\@glsstyle@mcoltreenamegroup	461
\@gls@write@entryunitcounts	128	\@glsstyle@mcoltreenamehypergroup	461
\@gls@write@entryunitcounts@do	129	\@glsstyle@mcoltreenamepannav	461
\@gls@writedef	66	\@glsstyle@mcoltreespannav	460

<code>\@glsstyle@tree</code> .....	441	<code>\@glsxtr@bookindex@subsep</code> .....	467–469
<code>\@glsstyle@treegroup</code> .....	443	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@treehypergroup</code> .....	443	.....	468, 470
<code>\@glsstyle@treenoname</code> .....	443	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@treenonamegroup</code> .....	445	.....	467–469
<code>\@glsstyle@treenonamehypergroup</code> ...	445	<code>\@glsxtr@bookindexgroupskip</code> ...	468, 470
<code>\@gls@target</code> .....	106, 142, 143	<code>\@glsxtr@cat</code> .....	119, 130, 173, 174, 218
<code>\@gls@text@</code> .....	107	<code>\@glsxtr@check@bibgls@nameref</code> .....	156
<code>\@gls@unset</code> .....	116, 117	<code>\@glsxtr@checkgroup</code> .....	163, 166
<code>\@gls@widestname</code> .....	448, 455	<code>\@glsxtr@counterrecordhook</code> .....	11–13
<code>\@glsxtr</code> .....	69, 71	<code>\@glsxtr@csname</code> .....	125, 126, 128
<code>\@glsxtr@@do@@wrglossary</code> .....	137	<code>\@glsxtr@current@style</code> .....	72, 464
<code>\@glsxtr@abbreviationsdef</code> .....	21, 30, 31	<code>\@glsxtr@currentunitcount</code> ..	125, 126, 128
<code>\@glsxtr@abbrlists</code> .....	132, 133	<code>\@glsxtr@currunitcount</code> .....	127, 129
<code>\@glsxtr@accessdisplay</code> .....	210, 211	<code>\@glsxtr@debugnr</code> .....	28
<code>\@glsxtr@acronymlists</code> .....	133, 134	<code>\@glsxtr@debugval</code> .....	28
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@declareoption</code> 5, 19–21, 24, 25, 27	
.....	218, 219	<code>\@glsxtr@defaultnoglossarywarning</code> ..	25
<code>\@glsxtr@addabbreviationlist</code> .....	230	<code>\@glsxtr@defaultnumberformat</code> .....	
<code>\@glsxtr@addunitcounter</code> .....	124	.....	7, 9, 81, 83, 101, 209, 212
<code>\@glsxtr@addunused</code> .....	65	<code>\@glsxtr@defpostpunc</code> .....	20, 28
<code>\@glsxtr@addunusedxrefs</code> .....	64, 65	<code>\@glsxtr@deprecated@abbrstyle</code> .....	
<code>\@glsxtr@altmodifier</code> .....	104	.....	278, 280, 281,
<code>\@glsxtr@assign@leveloffset</code> .....	143	284, 295, 296, 298, 301, 315, 319, 323, 325	
<code>\@glsxtr@attrval</code> .....	82, 204–212, 217	<code>\@glsxtr@dialect</code> .....	47, 48
<code>\@glsxtr@autoindex@at</code> .....	213, 214	<code>\@glsxtr@disabledflycommand</code> .....	71
<code>\@glsxtr@autoindex@doextra@esc</code> ...	213	<code>\@glsxtr@display@loc</code> .....	148
<code>\@glsxtr@autoindex@encap</code> ...	212, 214, 215	<code>\@glsxtr@do@@wrintex</code> .....	102
<code>\@glsxtr@autoindex@esc</code> .....	213, 215, 216	<code>\@glsxtr@do@autoadd</code> .....	80
<code>\@glsxtr@autoindex@escat</code> .....	214	<code>\@glsxtr@do@glstdisablehyperinlist</code> .	100
<code>\@glsxtr@autoindex@escencap</code> ...	214, 215	<code>\@glsxtr@do@inc@linkcount</code> .....	180
<code>\@glsxtr@autoindex@esclevel</code> ...	214, 215	<code>\@glsxtr@do@nameref@record</code> .....	11, 12
<code>\@glsxtr@autoindex@escquote</code> ...	213, 215	<code>\@glsxtr@do@record@wrglossary</code> ....	8, 15
<code>\@glsxtr@autoindex@level</code> .....	214, 215	<code>\@glsxtr@do@redef@forglsentries</code> .....	7
<code>\@glsxtr@autoindex@setname</code> .....	212	<code>\@glsxtr@do@style</code> .....	26, 379
<code>\@glsxtr@autoindex@crossrefs</code> 15, 18, 57, 62		<code>\@glsxtr@do@titlecaps@warn</code> .....	
<code>\@glsxtr@autoindex@false</code> .....	15	.....	204–207, 211, 219
<code>\@glsxtr@autoindex@true</code> .....	19	<code>\@glsxtr@doabbreviationsdef</code> .....	21
<code>\@glsxtr@base@acrcmd</code> .....	92–97, 133, 134	<code>\@glsxtr@doaccsupp</code> .....	24, 28
<code>\@glsxtr@bibgls@removespaces</code> .....	385	<code>\@glsxtr@docdefsetting</code> .....	18, 66
<code>\@glsxtr@bookindex@atendgroup</code> .....		<code>\@glsxtr@docdefval</code> .....	18, 65–67
.....	467, 468, 470	<code>\@glsxtr@doccounterrecord</code> .....	13
<code>\@glsxtr@bookindex@atsubendgroup</code> ..	469	<code>\@glsxtr@doglossary</code> .....	163, 165, 166
<code>\@glsxtr@bookindex@atsubsubendgroup</code>	469	<code>\@glsxtr@doiflabelinlist</code> .....	167
<code>\@glsxtr@bookindex@between</code> .	467, 468, 470	<code>\@glsxtr@doloadprefix</code> .....	25, 28, 31
<code>\@glsxtr@bookindex@sep</code> .....	467–469	<code>\@glsxtr@doloctag</code> .....	74
<code>\@glsxtr@bookindex@subatendgroup</code> ..		<code>\@glsxtr@dorecord</code> .....	8, 10
.....	467, 468, 470	<code>\@glsxtr@dorecordnodefer</code> .....	8, 10
<code>\@glsxtr@bookindex@subbetween</code> .	467–469	<code>\@glsxtr@dosee@alsoindex@glossary</code> ..	17

\@glxtr@doseeglossary .....	16, 30	\@glxtr@mark@wordseps .....	226
\@glxtr@dostylewarn .....	246	\@glxtr@mark@wordseps@next .....	226
\@glxtr@enabletagging .....	218	\@glxtr@markwordseps .....	228, 229
\@glxtr@end@ .....	68	\@glxtr@mixed@assign@sortkey .....	139
\@glxtr@endescspch .....	213–216	\@glxtr@newglslike .....	170, 171
\@glxtr@entrycount@org@localreset	120	\@glxtr@noidx@displaynumberlist ..	138
\@glxtr@entrycount@org@localunset	119	\@glxtr@noidx@do .....	168
\@glxtr@entrycount@org@reset .	119, 120	\@glxtr@noidx@entrynumberlist .....	138
\@glxtr@entrycount@org@unset .....	119	\@glxtr@noidx@numberlistloop .....	138
\@glxtr@entryunitcount@org@localreset		\@glxtr@nomissingglstextnr .....	25
.....	128	\@glxtr@nomissingglstextval .....	25
\@glxtr@entryunitcount@org@localunset		\@glxtr@noop@recordcounter .....	13, 16
.....	127	\@glxtr@nopostpunc .....	141
\@glxtr@entryunitcount@org@reset .	127	\@glxtr@nopostpunc@postdesc .....	141
\@glxtr@entryunitcount@org@unset .	127	\@glxtr@notfoundinlist .....	224
\@glxtr@equationsfalse .....	19	\@glxtr@op@recordcounter .....	15, 17
\@glxtr@err@undefaction .....	7, 16	\@glxtr@optlist .....	70, 71
\@glxtr@field@linkdefs .....	76, 78	\@glxtr@org@@starttoc .....	356
\@glxtr@floatsfalse .....	19	\@glxtr@org@GLS@ .....	77
\@glxtr@format@overridefalse .....	211	\@glxtr@org@GLSpl@ .....	77
\@glxtr@format@override>true .....	212	\@glxtr@org@Gls@ .....	77
\@glxtr@foundinlist .....	224	\@glxtr@org@Glspl@ .....	77
\@glxtr@full .....	232	\@glxtr@org@Glsxtrtitlefirst .	357, 358
\@glxtr@fullpl .....	234	\@glxtr@org@Glsxtrtitlefirstplural	
\@glxtr@get@prefixedlabel .....	391, 392	.....	357, 358
\@glxtr@gettype .....	139	\@glxtr@org@Glsxtrtitlefull ..	357, 358
\@glxtr@glossdescfont .....	204, 205	\@glxtr@org@Glsxtrtitlefullpl	357, 358
\@glxtr@glossnamefont .....	206–209, 211	\@glxtr@org@Glsxtrtitlelong ..	357, 358
\@glxtr@glosssymbolfont .....	217	\@glxtr@org@Glsxtrtitlelongpl	357, 358
\@glxtr@gobbleto@endescspch .....	216	\@glxtr@org@Glsxtrtitlename ..	357, 358
\@glxtr@groupheading ..	163, 166, 168, 169	\@glxtr@org@Glsxtrtitleplural	357, 358
\@glxtr@idx@displaynumberlist .....	138	\@glxtr@org@Glsxtrtitleshort .	357, 358
\@glxtr@idx@entrynumberlist .....	138	\@glxtr@org@Glsxtrtitleshortpl	357, 358
\@glxtr@if@record@only .....	135, 158	\@glxtr@org@Glsxtrtitletext ..	357, 358
\@glxtr@ifcsstart .....	68	\@glxtr@org@MakeUppercase .....	357, 358
\@glxtr@ifischild .....	169	\@glxtr@org@addtoacronymlists	131, 134
\@glxtr@ifnum@mmode .....	80	\@glxtr@org@checkfirsthyper ...	99, 134
\@glxtr@ifpunctoken .....	224	\@glxtr@org@currentfieldvalue .....	47
\@glxtr@ifunitcounter .....	124	\@glxtr@org@delimN .....	74, 75
\@glxtr@insert@dots .....	226	\@glxtr@org@delimR .....	74, 75
\@glxtr@insert@dots@next .....	226	\@glxtr@org@doseeglossary .....	29, 137
\@glxtr@insertdots .....	190, 228	\@glxtr@org@gloautosee .....	30
\@glxtr@label .....	38, 65, 180, 203	\@glxtr@org@glolinkprefix .....	81, 83
\@glxtr@labelprefixes .....	389, 390	\@glxtr@org@gls@ .....	77
\@glxtr@leveloffset .....	143, 169	\@glxtr@org@glsdohypertarget .....	143
\@glxtr@loadstyles .....	425, 426	\@glxtr@org@glsignore .....	74, 75
\@glxtr@local@textformat .....	81, 82	\@glxtr@org@glspl@ .....	77
\@glxtr@locale .....	47, 48	\@glxtr@org@Glsxtrtitlefirst .	357, 358
\@glxtr@longnewglossaryentry .....	51		



\@glxtr@org@glxtrtitlefirstplural .....	357, 358	\@glxtr@record .....	15–17, 76–78, 83, 232, 234–242
\@glxtr@org@glxtrtitlefull ..	357, 358	\@glxtr@record@noglossarywarning ..	16
\@glxtr@org@glxtrtitlefullpl ..	357, 358	\@glxtr@record@only@setup .....	16, 17
\@glxtr@org@glxtrtitlelong ..	357, 358	\@glxtr@record@setting ..	8, 10–12, 15, 16,
\@glxtr@org@glxtrtitlelongpl ..	357, 358		32, 61, 66, 67, 104, 136, 153, 155–158, 171
\@glxtr@org@glxtrtitlename ..	357, 358	\@glxtr@record@setting@alsoindex ..	
\@glxtr@org@glxtrtitleorpdforheading .....	357, 358		8, 10, 16, 17, 61, 136
\@glxtr@org@glxtrtitleplural ..	357, 358	\@glxtr@record@setting@nameref ..	
\@glxtr@org@glxtrtitleshort ..	357, 358		11, 12, 15, 32, 156, 157
\@glxtr@org@glxtrtitleshortpl ..	357, 358	\@glxtr@record@setting@off ..	66, 104, 171
\@glxtr@org@glxtrtitletext ..	357, 358	\@glxtr@record@setting@only .....	15, 32
\@glxtr@org@makeglossaries .....	135, 136	\@glxtr@recordsee .....	15, 29, 61
\@glxtr@org@markboth .....	355, 356	\@glxtr@redef@forglsentries ..	7, 30, 31
\@glxtr@org@markright .....	355, 356	\@glxtr@redefstyles .....	25, 26, 379
\@glxtr@org@newacronymstyle ..	132, 134	\@glxtr@reg@glosslist .....	136–139, 144
\@glxtr@org@postdescription ..	141, 220	\@glxtr@restore@postpunc .....	141
\@glxtr@org@see@noindex .....	155, 156	\@glxtr@rglstrigger@record ..	175–177
\@glxtr@org@setacronymlists .....	134	\@glxtr@save@longnewglossaryentry .....	51
\@glxtr@org@setacronymstyle ..	132, 134	\@glxtr@save@preloctag .....	74, 75
\@glxtr@org@theHvalue .....	8–10	\@glxtr@setentrycountunsetattr ..	118
\@glxtr@org@unset@buffer .....	116, 117	\@glxtr@setentryunitcountunsetattr ..	130
\@glxtr@org@prefix .....	11	\@glxtr@setup@shortcuts .....	23, 24, 30, 31
\@glxtr@org@printglossary .....	71, 142	\@glxtr@shortcutsnr .....	23
\@glxtr@org@warndep .....	227	\@glxtr@shortcutsval .....	23, 158
\@glxtr@p@acrlong@ .....	107	\@glxtr@swaptwo .....	225
\@glxtr@p@acrlongpl@ .....	107	\@glxtr@tag .....	218
\@glxtr@p@acrshort@ .....	107	\@glxtr@taggingcs .....	218
\@glxtr@p@acrshortpl@ .....	107	\@glxtr@textformat .....	82
\@glxtr@p@long@ .....	107	\@glxtr@theHentrycounter .....	11
\@glxtr@p@longpl@ .....	107	\@glxtr@theHvalue .....	8–10, 79, 81–84, 174
\@glxtr@p@plural@ .....	107	\@glxtr@theentrycounter .....	11
\@glxtr@p@short@ .....	107	\@glxtr@thevalue .....	8–10, 79, 81–84, 174
\@glxtr@p@shortpl@ .....	107	\@glxtr@thisloctag .....	74, 75
\@glxtr@p@text@ .....	107	\@glxtr@titlelabel .....	146, 147, 168
\@glxtr@pagestag .....	74	\@glxtr@tmp .....	25, 26, 80, 149
\@glxtr@pagetag .....	74	\@glxtr@type .....	203
\@glxtr@prefix .....	390	\@glxtr@unitcountlist .....	125
\@glxtr@prevunitcount .....	127	\@glxtr@unset .....	115–117
\@glxtr@printglossnr .....	142	\@glxtr@unset@buffer .....	116, 117
\@glxtr@printglossopts ..	71, 139, 142, 164	\@glxtr@unsrt@getgroupptitle ..	162, 165
\@glxtr@printglossval .....	142	\@glxtr@use@equation@counter ..	9, 80, 81
\@glxtr@printunsrtglossaryskipentry .....	163, 166	\@glxtr@use@see .....	57
\@glxtr@provide@addstoragekey .....	35	\@glxtr@warn@hybrid@noprintgloss ..	136
\@glxtr@provide@storagekey .....	34	\@glxtr@warn@onexistsordo .....	7, 15, 17
\@glxtr@providenewgls .....	171	\@glxtr@warn@undefaction .....	7, 15, 17
		\@glxtr@wrglossary@locationhyperlink .....	27
		\@glxtr@wrglossnr .....	79

<code>\@glxtr@wrglossval</code> .....	79	<code>\@makeglossaries@warn@noprintglossary</code>	137
<code>\@glxtrbuffer@nodup@unset</code> .....	116	<code>\@makeglossary</code> .....	136
<code>\@glxtrbuffer@unset</code> .....	116	<code>\@mfu@domakefirstuc</code> .....	218, 219
<code>\@glxtrdialecthook</code> .....	379	<code>\@mfu@nocaplist</code> .....	219
<code>\@glxtrdocdeffalse</code> .....	67	<code>\@one</code> .....	120, 129, 171
<code>\@glxtrentryfmt</code> .....	36	<code>\@newglossaryentry@defcounters</code> .....	119, 126
<code>\@glxtrfmt</code> .....	35	<code>\@newglossaryentryposthook</code> .....	14, 34, 62, 98
<code>\@glxtrforcsvfield</code> .....	38	<code>\@newglossaryentryprehook</code> .....	14, 34, 51, 62, 98
<code>\@glxtrglossentry</code> .....	159	<code>\@nil</code> .....	149, 168, 385, 386
<code>\@glxtrglossentryother</code> .....	161	<code>\@nnil</code> .....	60, 213, 214, 216, 224, 226
<code>\@glxtrhypernameprefix</code> .....	142, 143, 167, 168	<code>\@no@glxtrindexaliased</code> .....	101
<code>\@glxtrifhasfield</code> .....	38–40, 45, 46	<code>\@no@makeglossaries</code> .....	154
<code>\@glxtrifhyphenstart</code> .....	337	<code>\@nocounterr</code> .....	180
<code>\@glxtrindexaliased</code> .....	101	<code>\@nopostdesc</code> .....	141, 165
<code>\@glxtrindexcrossrefsfalse</code> .....	18	<code>\@onelevel@sanitize</code> .....	11, 13, 62, 70, 146, 147, 168
<code>\@glxtrindexcrossrefstrue</code> .....	18	<code>\@onlypreamble</code> .....	71, 74, 129, 156, 159, 180, 212, 214, 215, 218
<code>\@glxtrinmark</code> .....	355, 356	<code>\@org@glossaryentrynumbers</code> .....	140, 141, 164, 165
<code>\@glxtrlong</code> .....	107, 237	<code>\@org@newglossaryentryprehook</code> .....	51
<code>\@glxtrlongpl</code> .....	107, 241	<code>\@print@unsrt@glossary</code> .....	162
<code>\@glxtrnewgls</code> .....	172	<code>\@print@unsrt@innerglossary</code> .....	164
<code>\@glxtrnewgls@inner</code> .....	170, 171	<code>\@printgloss@checkexists</code> .....	140, 162
<code>\@glxtrnewgls@innercsname</code> .....	171	<code>\@printgloss@checkexists@allowignored</code> .....	162
<code>\@glxtrnotinmark</code> .....	355, 356	<code>\@printgloss@setsort</code> .....	139, 140, 164
<code>\@glxtrp</code> .....	112, 113	<code>\@printglossary</code> .....	71, 162
<code>\@glxtrp@opt</code> .....	110	<code>\@printunsrt@glossary@handler</code> .....	163, 166
<code>\@glxtrpl</code> .....	70, 71	<code>\@printunsrtglossary</code> .....	162
<code>\@glxtrpostloctag</code> .....	73–75	<code>\@rGLS</code> .....	176
<code>\@glxtrpreloctag</code> .....	73–75	<code>\@rGLS@</code> .....	177
<code>\@glxtrsetaliasnoindex</code> .....	100, 102	<code>\@rGLSpl</code> .....	177
<code>\@glxtrshort</code> .....	107, 235	<code>\@rGLSpl@</code> .....	177
<code>\@glxtrshortpl</code> .....	107, 239	<code>\@rGls</code> .....	176
<code>\@glxtrundeftag</code> .....	6, 32	<code>\@rGls@</code> .....	176
<code>\@glxtrwrglossmark</code> .....	27, 28	<code>\@rGlspl</code> .....	176
<code>\@gobble</code> .....	7, 16, 19, 85, 133, 136, 166–168, 226, 427, 467–470	<code>\@rGlspl@</code> .....	176
<code>\@gobbletwo</code> .....	134, 227	<code>\@rGLs</code> .....	176
<code>\@gtempa</code> .....	381	<code>\@rGLs@</code> .....	176
<code>\@ifdefinable</code> .....	381	<code>\@rGLspl</code> .....	176
<code>\@ifglossaryexists</code> .....	33	<code>\@rGLspl@</code> .....	176
<code>\@ifnextchar</code> .....	103, 143	<code>\@rc@ifdefinable</code> .....	381
<code>\@ifpackageloaded</code> .....	5, 21, 158, 180, 204, 205, 208, 210, 212, 379, 393, 424	<code>\@rgls</code> .....	175
<code>\@ifstar</code> .....	33–35, 38–42, 45, 46, 51–53, 68, 103, 116, 162, 218, 380	<code>\@rgls@</code> .....	175
<code>\@ifundefined</code> .....	378, 381	<code>\@rglspl</code> .....	175
<code>\@ignored@glossaries</code> .....	52–54	<code>\@rglspl@</code> .....	175
<code>\@input</code> .....	156	<code>\@sGlsXtrEnableOnTheFly</code> .....	68
<code>\@input@</code> .....	150	<code>\@secondofthree</code> .....	85–87, 93–96, 98, 233, 234, 236, 238, 240, 242
<code>\@istfilename</code> .....	136	<code>\@secondoftwo</code> .....	78, 84, 88–96, 100, 106, 134, 142, 169,



210, 224, 232, 233, 236–242, 258, 260, 283, 285, 300, 302, 324, 326, 356, 358, 427	\abbreviationsname .....	21
\@sglsxtr@provide@storagekey .....	\abbrvpluralsuffix .....	158, 228, 250, 252, 255, 258, 261, 263, 266, 269, 271, 273, 275, 277, 278, 280, 283, 286, 288, 290, 291, 294, 295, 297, 300, 303, 305, 307, 309, 310, 312, 314, 316, 318, 319, 322, 325, 328, 330, 332, 335, 339, 341, 344, 347, 350, 353
\@star@or@long .....	\ABP .....	22
\@starttoc .....	\Abp .....	21
\@thirdofthree .....	\abp .....	21
85–87, 93–96, 99, 233, 235, 237, 239, 241, 242, 357	\AC .....	22
\@thirdoftwo .....	\Ac .....	22
88–92	\ac .....	22
\@this@key .....	\ACF .....	22
210	\Acf .....	22
\@tracklang@lang .....	\acf .....	22
47, 48	\ACFP .....	22
\@warn@nomakeglossaries .....	\Acfp .....	22
151	\acfp .....	22
\@xGlsXtrIfValueInFieldCsvList .....	\ACL .....	22
40	\Acl .....	22
\@xdy@main@language .....	\acl .....	22
151	\ACLP .....	22
\@xdycrossrefhook .....	\Aclp .....	22
61	\aclp .....	22
\@xdy@language .....	\ACP .....	22
151	\Acp .....	22
\@xdy@locationclassorder .....	\acp .....	22
61	\ACRfull .....	97
\@xfor@nextelement .....	\Acrfull .....	97
60	\acrfull .....	97
[ package .....	\ACRfullfmt .....	97, 131
383	\Acrfullfmt .....	97, 131
\\ .....	\acrfullfmt .....	97, 131
149, 385	\ACRfullpl .....	97
	\Acrfullpl .....	97
	\acrfullpl .....	97
	\ACRfullplfmt .....	97, 132
	\Acrfullplfmt .....	97, 132
	\acrfullplfmt .....	97, 132
	\ACRlong .....	95
	\Acrlong .....	95
	\acrlong .....	94
	\ACRlongpl .....	96
	\Acrlongpl .....	96
	\acrlongpl .....	95
	\acronymentry .....	131
	\acronymfont .....	92–96, 109, 133, 134
<b>A</b>		
\AA .....		408
\aa .....		408
\AB .....		22
\Ab .....		21
\ab .....		21
abbreviation styles:		
footnote .....		257
long-hyphen-postshort-hyphen ...		343, 345
long-hyphen-short-hyphen .....		339, 343
long-postshort-user .....		331
long-short .....		189
long-short-user .....		329
nolong-short .....		265
short .....		262
short-em-footnote .....		323
short-em-postfootnote .....		326
short-footnote .....		256, 298
short-hyphen-long-hyphen .....		348, 349
short-hyphen-postlong-hyphen		348, 349, 351
short-long-user .....		331
short-nolong .....		262, 264
short-nolong-desc .....		264
short-postfootnote .....		259
short-postlong-user .....		333
short-sc-footnote .....		282, 284
short-sm-postfootnote .....		301





<code>\DTLifint</code> .....	146	<code>\equal</code> .....	154, 221
<b>E</b>			
<code>\eappto</code> .....	26, 190–192, 425	<code>equation (counter)</code> .....	19, 80
<code>\edef</code> .....	11, 47, 65, 146, 149–151, 156, 179, 213, 214, 216, 385, 390, 424, 467–469, 476–490, 492, 493, 500	<code>\escapechar</code> .....	381
<code>\eglssetwidest</code> .....	448–455	<code>etoolbox package</code> .....	5
<code>\egroup</code> .....	51, 141	<code>\everypar</code> .....	497
<code>\else</code> .....	8– 13, 15, 18–21, 23–25, 29–32, 36, 42, 49, 60, 66–68, 73, 75, 78, 82, 83, 101, 102, 121, 133, 134, 136, 138, 140–142, 146, 148, 149, 152, 154, 156, 164, 169, 174, 175, 212–214, 216, 224, 226, 229, 236– 243, 245, 250, 252, 253, 255, 256, 258, 259, 261–267, 270, 272–281, 283, 284, 286–298, 300, 301, 303, 305, 307, 309, 311–323, 325, 328–330, 332, 333, 336– 338, 340, 343, 345, 346, 349, 351, 353, 354, 385–389, 428, 431–438, 440, 443, 444, 456, 457, 464, 466–469, 477–494, 499	<code>\expandafter</code> .....	28, 34, 35, 38– 40, 47, 57, 60, 61, 64, 65, 68–70, 80, 98, 99, 103, 110, 116, 123, 124, 133, 137– 139, 144, 147–149, 163–166, 168–171, 178, 190, 203, 206, 208, 210, 211, 213, 215, 216, 224, 228, 229, 337, 381, 385, 428
<code>\emph</code> .....	189, 302	<code>\expandonce</code> .....	131, 169, 189–191, 213, 214, 250, 340, 428, 476–490, 492–494
<code>\empty</code> .....	148, 149, 189, 385, 389, 390	<code>\ExtraCustomAbbreviationFields</code> ....	190–192, 227, 229
<code>\encapchar</code> .....	216	<b>F</b>	
<code>\end</code> .....	17, 135, 147, 152, 153, 163, 165, 428, 430–437, 459, 460, 462, 463, 468, 477–484, 486, 487, 489–494, 500	<code>\fi</code> .....	7–13, 15, 17–21, 23–25, 28–30, 32, 36, 42, 49, 57, 60–62, 64, 66– 68, 72, 73, 75, 78, 79, 81–83, 101, 102, 121, 129, 130, 133, 134, 136, 138–143, 146, 148–152, 154–158, 163, 164, 166, 169, 174, 175, 212–214, 216, 224, 226, 229, 236–243, 245, 250, 252, 253, 255, 256, 258, 259, 261–267, 270, 272–281, 283, 284, 286–298, 300, 301, 303, 305, 307, 309–323, 325, 328–330, 332, 333, 336–338, 340, 343, 345, 346, 349, 351, 353, 354, 379, 385–389, 428, 431–438, 440, 442–444, 447–457, 464, 466–469, 475, 477–489, 491, 492, 494, 495, 499, 500
<code>\end@getprefix</code> .....	31, 32	<code>first use</code> .....	501
<code>\end@glstr@display@loc</code> .....	148	<code>flag</code> .....	501
<code>\endcsname</code> .....	6, 34, 53, 62, 67, 72, 75, 78, 81, 83, 92– 96, 98, 99, 101, 110, 126, 137, 147, 150, 151, 154, 155, 163–165, 171, 173, 174, 179, 203, 227, 232–243, 248, 249, 428, 456	<code>text</code> .....	501
<code>\endfoot</code> .....	474, 477, 479, 480, 482, 483, 485, 486, 488, 489, 491, 493	<code>\firstacronymfont</code> .....	133, 134
<code>\endgroup</code> .....	8, 10, 36, 37, 80, 84, 101, 160–162, 164, 179, 189, 190, 217, 381, 385, 390, 496	<code>fontspec package</code> .....	158
<code>\endhead</code> .....	474, 477, 479, 480, 482, 483, 485, 486, 488, 489, 491, 493	<code>\footnote</code> .....	254
<code>\ensuremath</code> .....	27	<code>\forall abbreviations</code> .....	133
entry categories:		<code>\forall glossaries</code> .....	64, 162, 201, 203, 448–455
abbreviation .....	227, 243	<code>\forall glsentries</code> .....	66, 120, 129
general .....	197, 199	<code>\ForEachTrackedDialect</code> .....	379, 424
index .....	201	<code>\foreignlanguage</code> .....	47, 48
nameshortaccess .....	312	<code>\for glsentries</code> .....	6, 64, 201, 203, 448–455
<code>\entryname</code> .....	474, 475, 477, 479, 480, 482, 483, 485, 486, 488, 490, 491, 493	<code>\forlistcsloop</code> .....	37, 129, 147
<code>\epreto</code> .....	213	<code>\forlistloop</code> .....	116, 117, 144, 145, 219
		<code>\futurelet</code> .....	224
<b>G</b>			
<code>\gdef</code> .....	74, 214, 215		
<code>\Genacrformat</code> .....	131, 132		

<code>\genacrfullformat</code> .....	131, 132	<code>longragged-booktabs</code> .....	476
<code>\GenericAcronymFields</code> .....	131	<code>mcolalttree</code> .....	462
<code>\Genplacrfullformat</code> .....	132	<code>mcolalttreegroup</code> .....	462
<code>\genplacrfullformat</code> .....	132	<code>mcolalttreehypergroup</code> .....	463
<code>\GetTitleStringDisableCommands</code> ....	427	<code>mcolalttreespannav</code> .....	463
<code>\GetTitleStringSetup</code> .....	427	<code>mcolindexgroup</code> .....	458
<code>\GetTrackedDialectFromLanguageTag</code> ..	47	<code>mcolindexhypergroup</code> .....	458
<code>\GetTrackedDialectToMapping</code> .....	47	<code>mcolindexspannav</code> .....	459
<code>\glo@grabfirst</code> .....	168	<code>mcoltreegroup</code> .....	459
<code>\glo@name</code> .....	206–208, 211	<code>mcoltreehypergroup</code> .....	460
<code>\gloaliaslabel</code> .....	105	<code>mcoltreenonamegroup</code> .....	461
<code>\global</code> .....	11, 51, 66, 141, 165, 169	<code>mcoltreenonamehypergroup</code> .....	461
<code>\glolinkprefix</code> ..	79, 81–83, 105, 106, 142, 158	<code>mcoltreenonamespannav</code> .....	461
<code>glossaries</code> package .....	12, 16, 28, 30, 31, 49, 50, 56, 61–63, 135, 136, 139, 161, 189, 203, 426, 428	<code>mcoltreespannav</code> .....	460
<code>glossaries-accsupp</code> package .....	24, 28, 180, 189, 229	<code>name-desc</code> .....	479
<code>glossaries-extra</code> package .....	2, 50, 424	<code>sublistdotted</code> .....	427
<code>glossaries-extra-bib2gls</code> package .....	16, 32, 379, 424, 475	<code>tree</code> .....	441
<code>glossaries-extra-stylemods</code> package .....	25, 220, 379, 387	<code>treegroup</code> .....	443
<code>glossaries-prefix</code> package .....	25, 28, 391	<code>treehypergroup</code> .....	443
<code>glossaries-stylemods</code> package .....	465	<code>treenoname</code> .....	443
<code>glossaries.sty</code> package .....	51	<code>treenonamegroup</code> .....	445
<code>\GlossariesExtraWarning</code> .....	6, 17, 19, 25, 32, 47–50, 69, 71, 82, 92, 133, 137, 149, 152, 155, 156, 162, 165, 204–209, 211, 217–219, 248, 381, 389	<code>treenonamehypergroup</code> .....	445
<code>\GlossariesExtraWarningNoLine</code> .....	15, 19, 121, 129, 158	<code>glossary-bookindex</code> package .....	426
<code>\GlossariesWarning</code> .....	74, 131, 138, 140, 144, 145, 164, 246	<code>glossary-hypernav</code> package .....	104
<code>\GlossariesWarningNoLine</code> .....	135, 151	<code>glossary-list</code> package .....	427, 428
<code>glossary</code> styles:		<code>glossary-long</code> package .....	432
<code>altlist</code> .....	428	<code>glossary-longbooktabs</code> package .....	432
<code>altlistgroup</code> .....	429	<code>glossary-longextra</code> package .....	386–388
<code>altlisthypergroup</code> .....	430	<code>glossary-tree</code> package .....	387, 388, 439, 498
<code>alttree</code> ....	386–388, 441, 445, 446, 456, 474	<code>\glossaryentrynumbers</code> .....	75, 140, 141, 164, 165, 169, 170
<code>alttreegroup</code> .....	457	<code>\glossaryheader</code> .....	147, 163, 165, 428–437, 439, 440, 442–445, 456, 458–461, 463, 468, 477, 478, 480–484, 486, 487, 489–495, 499
<code>alttreehypergroup</code> .....	457	<code>\glossaryname</code> .....	140
<code>index</code> .....	439	<code>\glossarypostamble</code> .....	147, 163, 165, 168
<code>indexgroup</code> .....	440	<code>\glossarypreamble</code> .....	147, 162, 165
<code>indexhypergroup</code> .....	440	<code>\glossarysection</code> .....	147, 148, 154, 155, 162, 165, 168
<code>inline</code> .....	438	<code>\glossarytitle</code> .....	53, 140, 147, 148, 154, 155, 162, 164, 165
<code>list</code> .....	428	<code>\glossarytoctitle</code> .....	53, 140, 142, 147, 148, 154, 155, 162, 164, 165
<code>listdotted</code> .....	426	<code>\glossentry</code> .....	140, 165, 170, 426, 428, 430–437, 439, 442, 444, 456, 468, 477, 478, 480, 481, 483, 484, 486, 487, 489, 491, 492, 494, 495, 499
<code>listdottedstyle</code> .....	427	<code>\Glossentrydesc</code> .....	497
<code>listgroup</code> .....	429		
<code>listhypergroup</code> .....	429		

<code>\glossentrydesc</code> .....	<code>\glsabbrvsmfont</code> .....
.... 426, 427, 430–438, 441–444, 472, 498	..... 285–291, 293, 295, 297, 299–301
<code>\Glossentryname</code> .....	<code>\glsabbrvuserfont</code> .....
496	327–333, 335
<code>\glossentryname</code> . 160, 426, 427, 430–437,	<code>\GLSaccessdesc</code> .....
439, 440, 442, 444, 456, 457, 465, 472, 498	88
<code>\glossentrynameother</code> .....	<code>\Glsaccessdesc</code> .....
161	88, 204, 216
<code>\glossentrysymbol</code> .....	<code>\glsaccessdesc</code> .....
.... 431–438, 442, 444, 446, 472, 496, 498	88, 204, 222
<code>\glossxtrsetopts</code> .....	<code>\GLSaccessdescplural</code> .....
220	89
<code>\GLS</code> .....	<code>\Glsaccessdescplural</code> .....
118, 130, 173	89
<code>\Gls</code> .....	<code>\glsaccessdescplural</code> .....
70, 118, 130, 173	88
<code>\gls</code> .....	<code>\GLSaccessfirst</code> .....
49, 69, 71, 118, 130, 138, 152, 173	86
<code>\gls@assign@desc</code> .....	<code>\Glsaccessfirst</code> .....
51	86
<code>\gls@assign@field</code> .....	<code>\glsaccessfirst</code> .....
14, 34, 98	85
<code>\gls@checkseeallowed</code> .....	<code>\GLSaccessfirstplural</code> .....
66, 68, 137	87
<code>\gls@codepage</code> .....	<code>\Glsaccessfirstplural</code> .....
151	87
<code>\gls@defdocnewglossaryentry</code> . 66, 119, 127	<code>\Glsaccesslong</code> .....
<code>\gls@defglossaryentry</code> .....	95, 230,
51, 66, 69, 70	238, 250, 261, 266, 270, 276, 277, 279,
<code>\gls@dotocitle</code> .....	287, 293–296, 303, 305, 313–318, 320,
140, 164, 165	329, 330, 339, 341, 342, 345, 347, 353, 354
<code>\gls@glossary</code> .....	<code>\glsaccesslong</code> .....
62	95, 230,
<code>\gls@grplabel</code> .....	238, 239, 250, 252, 253, 255, 256, 259,
104	261, 263, 264, 266, 267, 270, 272–279,
<code>\gls@ifnotmeasuring</code> .....	281, 284, 286, 288–290, 292–298, 300,
12, 117, 118	301, 303, 305, 307, 309, 311–323, 325,
<code>\gls@level</code> .....	328, 330, 333, 336, 339, 341, 345, 347, 353
169, 170	<code>\Glsaccesslongpl</code> .....
<code>\gls@noidxglossary</code> .....	. 96, 230, 242, 250, 261, 266, 270, 276–
137	279, 287, 293–296, 303, 305, 313–320,
<code>\gls@org@glossaryentryfield</code> ... 140, 165	329, 330, 339, 341, 342, 345, 348, 353, 354
<code>\gls@org@glossarysubentryfield</code> ....	<code>\glsaccesslongpl</code> .....
140, 141, 165	..... 96, 230, 241, 242, 250, 252, 253,
<code>\gls@orgTrackLangRequireDialectPrefix</code>	255, 256, 259, 261, 263–267, 270, 272–
..... 424	281, 284, 286, 288–290, 292–298, 301,
<code>\gls@save@numberlist</code> .....	303, 305, 307, 309, 311–323, 325, 328,
73, 75	330, 333, 336, 339, 341, 342, 345, 347, 353
<code>\gls@set@xr@key</code> .....	<code>\GLSaccessname</code> .....
62	88
<code>\gls@tmplen</code> .....	<code>\Glsaccessname</code> .....
448–457, 475, 476	88
<code>\gls@type</code> .....	<code>\glsaccessname</code> .....
137	87
<code>\glsabbrvdefaultfont</code> ... 231, 250, 252,	<code>\GLSaccessplural</code> .....
255, 258, 261, 263, 266, 327, 338, 341, 352	86
<code>\glsabbrvmfont</code> .....	<code>\Glsaccessplural</code> .....
302–310, 312, 314, 316, 318, 319, 321–326	86
<code>\glsabbrvfont</code> .....	<code>\glsaccessplural</code> .....
108, 109, 133, 236,	86
237, 239–241, 243, 245, 249–263, 266,	<code>\Glsaccessshort</code> .....
267, 269, 271, 273, 275, 277, 278, 280,	93, 236,
283, 286, 288, 290, 291, 293, 295, 297,	245, 252, 255, 256, 258, 259, 263, 265,
300, 303, 305, 307, 309, 310, 312, 314,	272, 274, 275, 281, 283, 284, 288, 290,
316, 318, 319, 322, 325, 328, 330, 332,	292, 298, 300, 301, 307, 309, 311, 312,
334–336, 339, 341, 343–345, 347, 350, 353	322, 323, 325, 332, 333, 336, 344, 350, 351
<code>\glsabbrvhyphenfont</code> 338, 339, 344, 346–351	<code>\glsaccessshort</code> .....
<code>\glsabbrvonlyfont</code> .....	92, 93,
352–354	230, 236, 237, 245, 250, 252, 255, 256,
<code>\glsabbrvscfont</code> .....	258, 261, 263, 264, 266, 270, 272–277,
..... 269–275, 277, 278, 280, 282–284	

279, 281, 283, 284, 286–288, 290–294,  
 296–298, 300, 303, 305, 307, 309–313,  
 315–318, 320, 322, 325, 328–330, 332,  
 333, 336, 339, 341, 344, 347, 350, 351, 353  
 \Glsaccesssshortpl ..... 94, 240,  
 245, 253, 255, 256, 258, 259, 263, 265,  
 272, 274, 275, 281, 284, 289, 290, 292,  
 298, 300, 301, 307, 309, 311, 312, 322,  
 323, 325, 332, 333, 336, 345, 350, 351, 354  
 \glsaccesssshortpl . 93, 94, 230, 239, 241,  
 245, 250, 252, 255, 256, 258, 259, 261,  
 263, 264, 266, 270, 272–279, 281, 283,  
 284, 286–288, 290–294, 296–298, 300,  
 303, 305, 307, 309, 311–313, 315–320,  
 322, 323, 325, 328–330, 332, 333, 336,  
 339, 341, 344, 347, 348, 350, 351, 353, 354  
 \GLSaccessssymbol ..... 89  
 \Glsaccessssymbol ..... 89, 217  
 \glsaccessssymbol ..... 89, 217, 222  
 \GLSaccessssymbolplural ..... 90  
 \Glsaccessssymbolplural ..... 90  
 \glsaccessssymbolplural ..... 89  
 \GLSaccessstext ..... 85  
 \Glsaccessstext ..... 85  
 \glsaccessstext ..... 85  
 \glsacrshortcutstrue ..... 23, 24  
 \glsacspacemax ..... 134  
 \glsadd ..... 36, 65, 80, 84, 152  
 \glsadd options  
   theHvalue ..... 10  
   thevalue ..... 9, 10  
 \glsaddpostsetkeys ..... 10, 83  
 \glsaddpresetkeys ..... 10, 83  
 \glsaddstoragekey ..... 63, 197, 382, 383  
 \glsaltlistitem ..... 428  
 \glsalttreechildpredesc ..... 446  
 \glsalttreepredesc ..... 445, 446  
 \glsbackslash ..... 68  
 \glscapitalisewords ..... 203  
 \glscapscase ..... 78, 84–96, 98, 99, 232–244  
 \glscategory ..... 76, 84, 100, 108,  
 109, 198–200, 204, 205, 207–210, 216,  
 217, 220, 221, 232–234, 236, 237, 239–241  
 \glscategorylabel ..... 100, 188,  
 191, 192, 225, 227–229, 246, 249, 251–  
 254, 256, 257, 259, 260, 262, 268–274,  
 280, 282–284, 286–289, 291, 293, 297–  
 299, 301, 302, 304–310, 312, 314, 315,  
 317, 321, 323, 324, 326, 328, 329, 331–  
 336, 338, 339, 342, 344–349, 351, 352, 354  
 \glsclosebrace ..... 61, 153, 154  
 \glscounter ..... 9, 19  
 \glscurrententrylabel .... 73–75, 140,  
 141, 150, 160, 161, 163, 165–167, 219, 220  
 \glscurrentfieldvalue .....  
 35, 36, 38–42, 45–47, 58, 59, 327, 381, 382  
 \glscustomtext .. 76, 78, 92–96, 232–243, 245  
 \glsdefaultshortaccess ..... 190, 191  
 \glsdefaulttype .....  
 .... 6, 21, 49, 139, 140, 152, 162, 164, 167  
 \glsdescriptionaccessdisplay 184, 185, 204  
 \glsdescriptionpluralaccessdisplay 185  
 \glsdescwidth ..... 430–437, 474–476  
 \glsdetoklabel ..... 8, 9, 33, 37, 38, 41–  
 45, 49–51, 55–57, 60, 65, 67, 68, 81, 83,  
 101, 105, 119, 120, 125–129, 137, 140,  
 141, 144, 145, 160, 161, 165, 168, 169,  
 173, 174, 203, 206–208, 211, 428, 450, 451  
 \glsdisp ..... 392  
 \glsdisplaynumberlist ..... 138, 144  
 \glsdohyperlink ..... 104, 386  
 \glsdohypertarget ..... 106, 142  
 \glsdoifexists ..... 18, 29, 43,  
 44, 50, 55, 57–61, 76, 77, 83, 92–96, 106,  
 118, 137, 144, 145, 160, 161, 232–242, 380  
 \glsdoifexistsordo ..... 35, 36, 78  
 \glsdoifexistsorwarn 18, 204, 205, 216, 217  
 \glsdoifnoexists ..... 51  
 \glsdonohyperlink ..... 82, 106  
 \glsdosanitizesort ..... 138  
 \glsenableentrycount ..... 118, 121, 129  
 \glsenableentryunitcount ... 120, 129, 130  
 \glsentrycounter ..... 27, 149, 385  
 \GlsEntryCounterLabelPrefix ..... 49  
 \glsentrycurrcount ..... 119, 120, 127  
 \Glsentrydesc ..... 184, 195, 205  
 \glsentrydesc ..... 184, 185, 195, 205  
 \Glsentrydescplural ..... 185, 196  
 \glsentrydescplural ..... 185, 195, 196  
 \Glsentryfirst ..... 124, 178, 182, 194  
 \glsentryfirst ..... 124, 177, 182, 194, 373  
 \Glsentryfirstplural ... 124, 178, 183, 195  
 \glsentryfirstplural .....  
 ..... 124, 177, 182, 183, 194, 195, 374  
 \glsentryfmt ..... 52–54, 133  
 \Glsentryfull ..... 132  
 \glsentryfull ..... 132



<code>\Glsentryfullpl</code> .....	132	<code>\glsfieldfetch</code> .....	105
<code>\glsentryfullpl</code> .....	132	<code>\glsfieldxdef</code> .....	203
<code>\glsentryitem</code> ...	160, 161, 426, 427, 430– 437, 439, 442, 444, 456, 468, 472, 496, 498	<code>\glsFindWidestLevelTwo</code> .....	389
<code>\Glsentrylong</code> ...	109, 110, 124, 178, 187, 196	<code>\glsFindWidestTopLevelName</code> .....	389
<code>\glsentrylong</code> .....	109, 110, 124, 177, 186, 187, 196, 258, 259, 283, 285, 299, 301, 324, 326, 332, 334, 349, 374–376	<code>\glsfindwidesttoplevelname</code> .....	448
<code>\Glsentrylongpl</code> ....	109, 110, 124, 187, 196	<code>\GLSfirst</code> .....	363, 364
<code>\glsentrylongpl</code> .....	109, 110, 124, 187, 196, 197, 375, 376	<code>\Glsfirst</code> .....	364
<code>\Glsentrylongplural</code> .....	178	<code>\glsfirst</code> .....	363
<code>\glsentrylongplural</code> .....	177	<code>\glsfirstabbrvdefaultfont</code> .....	231, 250, 252, 255, 258, 261, 263, 266, 341
<code>\Glsentryname</code> .....	180, 194, 206–209	<code>\glsfirstabbrvmfont</code> .....	302–326
<code>\glsentryname</code> .....	159, 160, 180, 181, 194, 213, 370, 371, 448–455, 471	<code>\glsfirstabbrvfont</code> .....	133, 230, 249–266, 269, 271, 273, 275, 277, 278, 280, 283, 286, 288, 290, 291, 294, 295, 297, 300, 303, 305, 307, 309, 310, 312, 314, 316, 318, 319, 322, 325, 328, 330, 332, 335, 339, 341, 344, 347, 350, 353
<code>\glsentrynumberlist</code> ....	138, 145, 453–455	<code>\glsfirstabbrvhyphenfont</code> .....	337–339, 343, 344, 346–351
<code>\glsentrypdfsymbol</code> .....	217	<code>\glsfirstabbrvonlyfont</code> .....	353, 354
<code>\Glsentryplural</code> .....	181, 194	<code>\glsfirstabbrvscfont</code> .....	269–284
<code>\glsentryplural</code> .....	181, 182, 194, 372	<code>\glsfirstabbrvsmfont</code> .....	286–301
<code>\glsentryprevcount</code> .....	119, 121, 127	<code>\glsfirstabbrvuserfont</code> .....	328–336
<code>\glsentryprevmaxcount</code> .....	127	<code>\glsfirstaccessdisplay</code> .....	182
<code>\glsentryprevtotalcount</code> .....	127	<code>\glsfirstlongdefaultfont</code> .....	250, 252, 261, 263, 266, 269–280, 286– 296, 302–304, 306–308, 310–315, 318, 319
<code>\Glsentryshort</code> .....	108, 109, 186, 196	<code>\glsfirstlongemfont</code> .....	304–306, 308–310, 316, 317, 319–321
<code>\glsentryshort</code> .....	108, 109, 134, 185, 186, 196, 329, 331, 343, 369, 370, 376	<code>\glsfirstlongfont</code> .....	230, 249–253, 255, 258, 261, 263–268, 270, 271, 273, 275, 277, 278, 280, 283, 286, 288, 290, 291, 294, 295, 297, 300, 303, 305, 307, 309, 310, 312, 314, 316, 318, 320, 322, 325, 328, 330, 332, 335, 339, 341, 344, 347, 350, 353
<code>\Glsentryshortpl</code> .....	108, 109, 186, 196	<code>\glsfirstlongfootnotefont</code> .....	254–259, 280–285, 297–301, 321–326
<code>\glsentryshortpl</code> .....	108, 109, 186, 196, 369, 370, 376	<code>\glsfirstlonghyphenfont</code> .....	337–350
<code>\Glsentrysymbol</code> .....	183, 195	<code>\glsfirstlongonlyfont</code> .....	352–354
<code>\glsentrysymbol</code> ....	183, 195, 217, 452–454	<code>\glsfirstlonguserfont</code> .....	328–336
<code>\Glsentrysymbolplural</code> .....	184, 195	<code>\GLSfirstplural</code> .....	364, 365
<code>\glsentrysymbolplural</code> .....	184, 195	<code>\Glsfirstplural</code> .....	365
<code>\Glsentrytext</code> .....	181, 194	<code>\glsfirstplural</code> .....	364
<code>\glsentrytext</code> .....	106, 181, 194, 371, 372	<code>\glsfirstpluralaccessdisplay</code> ..	182, 183
<code>\glsentrytype</code> .....	160, 230	<code>\GLSfmtname</code> .....	59
<code>\Glsentryuseri</code> .....	90	<code>\Glsfmtname</code> .....	58, 59
<code>\glsentryuseri</code> .....	90	<code>\glsfmtname</code> .....	57–59
<code>\Glsentryuserii</code> .....	90	<code>\GLSfmttext</code> .....	59
<code>\glsentryuserii</code> .....	90	<code>\Glsfmttext</code> .....	58, 59
<code>\Glsentryuseriii</code> .....	91		
<code>\glsentryuseriii</code> .....	91		
<code>\Glsentryuseriv</code> .....	91		
<code>\glsentryuseriv</code> .....	91		
<code>\Glsentryuserv</code> .....	91		
<code>\glsentryuserv</code> .....	91		
<code>\Glsentryuservi</code> .....	92		
<code>\glsentryuservi</code> .....	92		
<code>\glsextrapostnamehook</code> .....	209		



<code>\glsfmttext</code> .....	57–59	106, 134, 174, 178, 179, 221, 222, 243–
<code>\glsforeachincategory</code> .....	246	245, 258, 259, 283, 285, 299, 301, 324,
<code>\glsgenentryfmt</code> .....	76	326, 329, 331, 332, 334, 344, 346, 349–351
<code>\glsgetattribute</code> .....	82, 105, 121,	<code>\glslabeltok</code> .....
125–127, 150, 174, 179, 204–210, 212, 217		.... 131, 227, 229, 230, 249–258, 260,
<code>\glsgetcategoryattribute</code> .....	199	262–273, 275, 277, 280, 282, 283, 285–
<code>\glsgetgrouptitle</code> .....	427, 429, 430	291, 293, 297, 299–310, 312, 314, 316,
<code>\glsgetwidestname</code> .....	446	317, 319, 321–324, 326, 328, 329, 331–
<code>\glsgroupheading</code> .....		336, 338–340, 342, 344, 346–348, 350–354
.... 169, 428–437, 439, 440, 442–445,		<code>\glsletentryfield</code> .....
456–463, 469, 477, 478, 480, 481, 483,		213
484, 486, 487, 489, 491, 492, 494, 495, 499		<code>\glslink</code> .....
<code>\glsgroupskip</code> .....	169, 428, 430–438,	131, 132, 392
440, 443, 444, 457, 469, 477, 479–483,		<code>\glslink options</code>
485, 486, 488, 489, 491, 492, 494, 496, 500		counter .....
<code>\glshasattribute</code> .....		format .....
.... 82, 104, 105, 120, 121, 125,		hyper .....
126, 128, 129, 150, 174, 179, 204, 205,		hyperoutside .....
207–210, 212, 217, 250–253, 255, 257,		noindex .....
258, 260, 262, 264, 265, 267–269, 271,		textformat .....
272, 280, 282, 283, 285–289, 297, 299–		theHvalue .....
301, 303–306, 308, 310, 317, 321–324,		thevalue .....
326, 328, 329, 331, 332, 334–336, 338–		wrgloss .....
340, 342, 344, 346–348, 350, 351, 353, 354		8, 78
<code>\glshascategoryattribute</code> .....	199	<code>\glslinkcheckfirsthyperhook</code> .....
<code>\glshex</code> ..	395–400, 403–408, 411–413, 415–424	100
<code>\glshyperlink</code> .....	106, 382	<code>\glslinkpostsetkeys</code> .....
<code>\glshypernavsep</code> .....	147	10, 81, 174
<code>\glshypernumber</code> .....	150, 212, 383–385	<code>\glslinkpresetkeys</code> .....
<code>\glsifattribute</code> .....		10, 81, 174
.... 78, 79, 85, 100, 102, 112, 178,		<code>\glslinkvar</code> .....
201, 204–207, 211, 219, 222, 223, 359–368		103
<code>\glsifcategory</code> .....	201	<code>\glslistchildpostlocation</code> .....
<code>\glsifcategoryattribute</code> .....		428
.... 100, 188, 191, 192, 199, 200, 227–229		<code>\glslistchildprelocation</code> .....
<code>\glsifnotregular</code> .....	84	428, 429
<code>\glsifnotregularcategory</code> .....	200	<code>\glslistdesc</code> .....
<code>\glsifplural</code> 78, 84, 86–90, 92–96, 223, 232–244		428, 429
<code>\glsifregular</code> .....	76, 84, 124, 177, 178	<code>\glslistdottedwidth</code> .....
<code>\glsifregularcategory</code> .....	200	426
<code>\glsifusetranslator</code> .....	53	<code>\glslistexpandedname</code> .....
<code>\glsignore</code> .....	74, 75	427
<code>\glsinlinedescformat</code> .....	438	<code>\glslistgroupafterheader</code> .....
<code>\glsinlinesubdescformat</code> .....	438	429, 430
<code>\glsinsert</code> .....	78,	<code>\glslistgroupheaderfmt</code> .....
84, 92–96, 232–245, 337, 344, 346, 349–351		427, 429, 430
<code>\glskeylisttok</code> .....	131, 227, 229	<code>\glslistgroupheaderitem</code> .....
<code>\glslabel</code> .....	8,	429, 430
9, 35, 36, 55, 76, 78–82, 99–101, 104–		<code>\glslistgroupskip</code> .....
		428
		<code>\glslistinit</code> .....
		428
		<code>\glslistitem</code> .....
		428
		<code>\glslistnavigationitem</code> .....
		429, 430
		<code>\glslistprelocation</code> .....
		427, 428
		<code>\glslocalunset</code> .....
		78, 175
		<code>\glslongaccessdisplay</code> .....
		186, 187
		<code>\glslongdefaultfont</code> .
		231, 232, 250, 252,
		254, 261, 263, 266, 270, 271, 273, 275,
		277–279, 286, 288, 290, 291, 293–295,
		303, 307, 310, 312, 314, 318, 327, 338, 352
		<code>\glslongemfont</code> 302, 305, 308, 309, 316, 319, 320
		<code>\glslongextraDescAlign</code> .....
		.... 476–487, 489, 490, 492–494
		<code>\glslongextraDescFmt</code> 473, 477, 478, 480,
		481, 483, 484, 486, 488, 489, 491, 492, 494

<code>\glslongextraDescNameHeader</code> .....	480	<code>\glslongextraNameDescSymLocationTabularHeader</code>	.....	483, 484
<code>\glslongextraDescNameTabularFooter</code>	479	<code>\glslongextraNameDescSymTabularFooter</code>	.....	482
<code>\glslongextraDescNameTabularHeader</code>	.....	<code>\glslongextraNameDescSymTabularHeader</code>	.....	482
.....	479, 480	<code>\glslongextraNameDescTabularFooter</code>	.....	474, 477
<code>\glslongextraDescSymNameHeader</code> ....	492	<code>\glslongextraNameDescTabularHeader</code>	.....	474, 477
<code>\glslongextraDescSymNameTabularFooter</code>	.....	<code>\glslongextraNameFmt</code> ...	477, 478, 480,	
.....	491, 492	.....	481, 483, 484, 486, 488, 489, 491, 492, 494	
<code>\glslongextraDescSymNameTabularHeader</code>	.....	<code>\glslongextraNameSymDescHeader</code> ....	486	
.....	491, 492	<code>\glslongextraNameSymDescLocationHeader</code>	.....	487
<code>\glslongextraGroupHeading</code> 477, 478, 480,		<code>\glslongextraNameSymDescLocationTabularFooter</code>	.....	486, 487
481, 483, 484, 486, 487, 489, 491, 492, 494		<code>\glslongextraNameSymDescLocationTabularHeader</code>	.....	486, 487
<code>\glslongextraHeaderFmt</code> .	474, 475, 477,	<code>\glslongextraNameSymDescTabularFooter</code>	.....	485
479, 480, 482, 483, 485–488, 490, 491, 493		<code>\glslongextraNameSymDescTabularHeader</code>	.....	485, 486
<code>\glslongextraLocationAlign</code> .....		<code>\glslongextraSetDescWidth</code> .....		
.....	478, 481, 484, 487, 490, 493, 494	.....	475–477, 479, 480	
<code>\glslongextraLocationDescNameHeader</code>	481	<code>\glslongextraSubDescFmt</code> ....	477, 479–	
<code>\glslongextraLocationDescNameTabularFooter</code>	.....	.....	481, 483, 485, 486, 488, 489, 491, 492, 494	
.....	480, 481	<code>\glslongextraSubLocationFmt</code> .....		
<code>\glslongextraLocationDescNameTabularHeader</code>	.....	.....	479, 481, 485, 488, 491, 494	
.....	480, 481	<code>\glslongextraSubNameFmt</code> ....	477, 479–	
<code>\glslongextraLocationDescSymNameHeader</code>	.....	.....	481, 483, 484, 486, 488, 489, 491, 492, 494	
.....	494	<code>\glslongextraSubSymbolFmt</code> .....		
<code>\glslongextraLocationDescSymNameTabularFooter</code>	.....	.....	483, 485, 486, 488, 489, 491, 492, 494	
.....	493	<code>\glslongextraSymbolAlign</code>	482–490, 492–494	
<code>\glslongextraLocationDescSymNameTabularHeader</code>	.....	<code>\glslongextraSymbolFmt</code> .....		
.....	493	.....	473, 483, 484, 486, 488, 489, 491, 492, 494	
<code>\glslongextraLocationFmt</code> .....		<code>\glslongextraSymDescNameHeader</code> ....	489	
.....	478, 481, 484, 488, 491, 494	<code>\glslongextraSymDescNameTabularFooter</code>	.....	488, 489
<code>\glslongextraLocationSymDescNameHeader</code>	.....	<code>\glslongextraSymDescNameTabularHeader</code>	.....	488, 489
.....	491	<code>\glslongextraSymLocSetDescWidth</code> ...		
<code>\glslongextraLocationSymDescNameTabularFooter</code>	.....	.....	484, 487, 490, 493	
.....	489, 490	<code>\glslongextraSymSetDescWidth</code> .....		
<code>\glslongextraLocationSymDescNameTabularHeader</code>	.....	.....	476, 482, 485, 486, 488, 489, 492	
.....	489, 490	<code>\glslongextraTabularVAlign</code> .....		
<code>\glslongextraLocSetDescWidth</code> ..	478, 481	.....	476, 478, 479,	
<code>\glslongextraNameAlign</code> .....		<code>\glslongextraNameDescSymLocationHeader</code>	.....	484
.....	476–487, 489, 490, 492–494	<code>\glslongextraNameDescSymLocationTabularFooter</code>	.....	483, 484
<code>\glslongextraNameDescHeader</code> .....	477	.....	483, 484	
<code>\glslongextraNameDescLocationHeader</code>	478			
<code>\glslongextraNameDescLocationTabularFooter</code>	.....			
.....	477, 478			
<code>\glslongextraNameDescLocationTabularHeader</code>	.....			
.....	477, 478			
<code>\glslongextraNameDescSymHeader</code> ....	483			
<code>\glslongextraNameDescSymLocationHeader</code>	.....			
.....	484			
<code>\glslongextraNameDescSymLocationTabularFooter</code>	.....			
.....	483, 484			

<code>\glslongextraUpdateWidestChild</code>	387–389	<code>\glsnumberlistloop</code>	138
<code>\GlsLongExtraUseTabularfalse</code>	476	<code>\glsnumlistlastsep</code>	144, 380
<code>\glslongfont</code>	109, 232, 238, 239, 241–243, 250–253, 255, 256, 258, 261–263, 265, 266, 268, 270, 271, 273, 275, 277, 278, 280, 283, 286, 288, 290, 291, 294, 295, 297, 300, 303, 305, 307, 309, 310, 312, 314, 316, 318, 320, 322, 325, 328, 330, 332, 335, 339, 341, 344, 347, 350, 353, 354	<code>\glsnumlistsep</code>	144, 380
<code>\glslongfootnotefont</code>	254, 255, 258, 280, 283, 297, 300, 322, 325	<code>\glsopenbrace</code>	61, 153, 154
<code>\glslonghyphenfont</code>	338–342, 344, 347, 349, 350	<code>\glsorder</code>	136, 158
<code>\glslongonlyfont</code>	352, 353	<code>\glspagelistwidth</code>	431, 433, 435, 437, 474, 476
<code>\glslongpltok</code>	229, 249, 251–254, 257, 265, 268–272, 277, 280, 282, 286–289, 293, 297, 299, 302, 304, 306, 308, 310, 314, 316, 319, 321, 323, 328, 329, 331– 336, 338–340, 342, 344, 346–348, 352, 354	<code>\glspar</code>	168, 496
<code>\glslongpluralaccessdisplay</code>	187	<code>\glspatchLToutput</code>	477, 478, 480–482, 484, 486, 487, 489, 490, 492, 493
<code>\glslongtok</code>	131, 227–229, 249, 251–254, 256, 257, 260, 262, 265, 268–273, 277, 280, 282, 286–290, 293, 297, 299, 302–306, 308– 310, 314, 316, 319, 321, 323, 324, 328– 336, 338–340, 342, 344, 346–349, 352–354	<code>\glspdffmtfull</code>	376, 377
<code>\glslonguserfont</code>	327, 328, 330, 332, 333, 335	<code>\glspdffmtfullpl</code>	377, 378
<code>\glsmcols</code>	459–463	<code>\glspenaltygroupskip</code>	477, 479, 480, 482, 483, 485, 486, 488, 489, 491, 492, 494
<code>\GLSname</code>	361	<code>\GLSpl</code>	118, 130, 173
<code>\Glsname</code>	361	<code>\Glspl</code>	70, 118, 130, 173
<code>\glsname</code>	361	<code>\glspl</code>	70, 118, 130, 173
<code>\glsnameaccessdisplay</code>	180, 206, 208	<code>\GLSplural</code>	362, 363
<code>\glsnamefont</code>	206–209, 211, 475	<code>\Glsplural</code>	363
<code>\glsnavhyperlink</code>	147	<code>\glsplural</code>	362, 363
<code>\glsnavhyperlinkname</code>	104	<code>\glspluralaccessdisplay</code>	181, 182
<code>\glsnavhypertarget</code>	427, 429, 430, 440, 443, 445, 458–463	<code>\glspluralsuffix</code>	158, 227, 228, 232
<code>\glsnavigation</code>	427, 429, 430, 440, 443, 445, 458–461, 463	<code>\glspostdescription</code>	20, 141, 220, 426, 427, 430–438, 441–444, 472, 497, 498
<code>\glsnextpages</code>	140, 165	<code>\glspostinline</code>	438
<code>\glsnoidxdisplayloc</code>	145, 384	<code>\glspostlinkhook</code>	76, 78, 92–96, 110, 232–243
<code>\glsnoidxdisplaylocclisthandler</code>	144	<code>\glsprestandardsort</code>	138
<code>\glsnoidxloclist</code>	145, 169, 170	<code>\glsresetentrylist</code>	147, 163, 165
<code>\glsnoidxnumberlistloophandler</code>	145	<code>\glssee</code>	62–64
<code>\glsnonnextpages</code>	140, 165	<code>\glsseefirstitem</code>	60
<code>\glsnonnumberlistfalse</code>	73	<code>\glsseeformat</code>	57, 60, 67, 137, 145
<code>\glsnonnumberlisttrue</code>	73	<code>\glsseeitem</code>	61
<code>\glsnopostdotfalse</code>	141	<code>\glsseelastoxfordsep</code>	61
<code>\glsnopostdottrue</code>	141	<code>\glsseelastsep</code>	60, 61
		<code>\glsseelist</code>	60
		<code>\glsseesep</code>	61
		<code>\glssetabbrvfmt</code>	76, 84, 108, 109, 204, 205, 207–210, 216, 217, 232–234, 236, 237, 239–241, 243
		<code>\glssetattribute</code>	250–253, 255, 257, 258, 260, 262–269, 271, 273, 275, 277, 280, 282, 283, 285–291, 293, 297, 299–301, 303– 308, 310, 312, 314, 316, 317, 319, 321– 324, 326, 328, 329, 331, 332, 334–336, 338–340, 342, 344, 346–348, 350, 352–354
		<code>\glssetcategoryattribute</code>	119, 130, 134, 174, 180, 192, 193, 198–200, 202, 218
		<code>\glssetnoexpandfield</code>	14
		<code>\glssettoctitle</code>	140, 164

<code>\glsssetwidest</code> .....	387, 388	<code>\glstopicsubitemhangindent</code> .....	497
<code>\glssshortaccessdisplay</code> .....	185, 186	<code>\glstopicSubItemParIndent</code> .....	497
<code>\glssshortaccsupp</code> .....	192	<code>\glstopicSubItemSep</code> .....	498
<code>\glssshortpltok</code> 229, 249, 251–257, 259, 260, 262, 263, 269–275, 280, 282, 284, 286– 291, 297, 299, 301, 303–306, 308, 310, 312, 321, 323, 324, 326, 328, 329, 331– 336, 338, 339, 344, 346–349, 351, 353, 354		<code>\glstopicSubLoc</code> .....	498
<code>\glssshortpluralaccessdisplay</code> .....	186	<code>\glstopicSubNameFont</code> .....	498
<code>\glssshorttok</code> .....	131, 227–229, 249, 251–257, 259, 260, 262, 263, 267– 274, 276, 280, 282, 284, 286–291, 293, 297–299, 301–306, 308–310, 312, 314, 316, 321, 323, 324, 326, 328, 329, 331– 336, 338, 339, 342, 344, 346–349, 351–354	<code>\glstopicSubPreLocSep</code> .....	498
<code>\glssshowtargetfont</code> .....	29	<code>\glstopicTitle</code> .....	496
<code>\glssshowtargetouter</code> .....	28, 29	<code>\glstopicTitleFont</code> .....	496
<code>\glssubentryitem</code> .....	160, 426, 428–437, 440, 442, 444, 456, 469, 473	<code>\glstopicwidest</code> .....	497–499
<code>\glssymbolaccessdisplay</code> .....	183	<code>\glstreechilddesc</code> .....	442
<code>\glssymbolpluralaccessdisplay</code> .....	184	<code>\glstreeChildDescLoc</code> .....	440, 443
<code>\glstarget</code> .....	160, 161, 426–440, 442, 444, 456, 457, 468, 469, 472, 473, 496, 498	<code>\glstreechildpredesc</code> .....	441
<code>\GLStext</code> .....	361, 362	<code>\glstreechildprelocation</code> .....	442, 444
<code>\Glstext</code> .....	362	<code>\glstreechildsymbol</code> .....	440, 442
<code>\glstext</code> .....	362	<code>\glstreedefaultnamefmt</code> .....	438, 439
<code>\glstextaccessdisplay</code> .....	181	<code>\glstreedefsc</code> .....	441
<code>\glstextformat</code> .....	78, 82	<code>\glstreeDescLoc</code> .....	439, 442, 446
<code>\glstextup</code> .....	269	<code>\glstreegroupheaderfmt</code> .....	440, 443, 445, 457–463, 466
<code>\glstopic@postchildren</code> .....	499, 500	<code>\glstreegroupheaderskip</code> .....	440, 443, 445, 457–463
<code>\glstopic@prechildren</code> .....	495, 499, 500	<code>\glstreegroupskip</code> ..	439, 440, 443, 444, 457
<code>\glstopic@prevlevel</code> .....	495, 499, 500	<code>\glstreeindent</code> .....	442, 444, 455–457
<code>\glstopicAssignSubIndent</code> .....	496, 500	<code>\glstreeitem</code> .....	439, 459, 468, 469
<code>\glstopicAssignWidest</code> .....	497	<code>\glstreenamebox</code> .....	456, 457
<code>\glstopicCols</code> .....	500	<code>\glstreenamefmt</code> .....	438–440, 442, 444, 446, 448–457
<code>\glstopicColsEnv</code> .....	500	<code>\glstreenavigationfmt</code> .....	440, 443, 445, 458–461, 463
<code>\glstopicDesc</code> .....	496	<code>\glstreeNoDescSymbolPreLocation</code>	441, 442
<code>\glstopicGroupHeading</code> .....	495, 499	<code>\glstreenonamechilddesc</code> .....	444
<code>\glstopicInit</code> .....	495, 499	<code>\glstreenonameChildDescLoc</code> .....	444
<code>\glstopicItem</code> .....	495, 500	<code>\glstreenonamedesc</code> .....	444
<code>\glstopicLoc</code> .....	496	<code>\glstreenonameDescLoc</code> .....	444
<code>\glstopicMarker</code> .....	496	<code>\glstreenonamesymbol</code> .....	444
<code>\glstopicMidSkip</code> .....	496	<code>\glstreepredesc</code> .....	441, 443
<code>\glstopicParIndent</code> .....	495, 500	<code>\glstreePreHeader</code> ..	440, 443, 445, 457–463
<code>\glstopicPostSkip</code> .....	496	<code>\glstreeprelocation</code> .....	439, 441, 444
<code>\glstopicPreSkip</code> .....	496	<code>\glstreesubitem</code> .....	439, 469
<code>\glstopicSubIndent</code> .....	497	<code>\glstreesubsubitem</code> .....	439, 469
<code>\glstopicSubItem</code> .....	496, 500	<code>\glstreesymbol</code> .....	439, 442
<code>\glstopicSubItemBox</code> .....	498	<code>\GlsrLetField</code> .....	44
		<code>\glstype</code> .....	78, 81, 92–96, 174, 232–243
		<code>\glset</code> .....	65, 78, 121, 122, 175
		<code>\glupdatewidest</code> .....	386, 387
		<code>\gluserdescription</code> ....	328, 329, 332, 335
		<code>\glswrite</code> .....	61, 136
		<code>\glswriteentry</code> .....	8, 9
		<code>\Glsxtr</code> .....	71

<code>\glsxtr</code> .....	71	<code>\glsxtr@renewcommand</code> .....	381
<code>\glsxtr@do@wrglossary</code> .....	8, 10, 13, 16	<code>\glsxtr@resource</code> .....	155, 157
<code>\glsxtr@addloclistfield</code> .....	15, 17	<code>\glsxtr@s@newignoredglossary</code> .....	52
<code>\glsxtr@addunused</code> .....	65	<code>\glsxtr@s@provideignoredglossary</code> ...	53
<code>\glsxtr@applyabbrvfmt</code> .....	243	<code>\glsxtr@saveentrycounter</code> ...	8, 10, 13, 101
<code>\glsxtr@applyabbrvstyle</code> .....	227, 246	<code>\glsxtr@setaccessdisplay</code> .....	210
<code>\glsxtr@beginbookindex</code> .....	467	<code>\glsxtr@setbookindexmark</code> .....	470
<code>\glsxtr@counterrecord</code> .....	159	<code>\glsxtr@setup@record</code> .....	15–17, 31, 32
<code>\glsxtr@dblfloat</code> .....	19	<code>\glsxtr@shortcutsval</code> .....	157, 158
<code>\glsxtr@do@alsoindex@wrglossary</code> ....	17	<code>\glsxtr@texencoding</code> .....	158
<code>\glsxtr@do@autoadd</code> .....	9, 80, 81	<code>\glsxtr@undefaction@nr</code> .....	7
<code>\glsxtr@doooption</code> .....	5, 19, 20, 27, 28, 30	<code>\glsxtr@undefaction@val</code> .....	7
<code>\glsxtr@endbookindex</code> .....	468	<code>\glsxtr@usesee</code> .....	57
<code>\glsxtr@fields</code> .....	157	<code>\glsxtr@warnonexistsordo</code> ....	7, 15–17, 56
<code>\glsxtr@float</code> .....	19	<code>\glsxtr@writefields</code> .....	155
<code>\glsxtr@grptitle</code> ...	440, 443, 445, 457–463	<code>\glsxtr@abbreviationfont</code> .....	76
<code>\glsxtr@headentry@p</code> .....	112, 113	<code>\glsxtr@abbrvfootnote</code> 254–256, 258, 259,	
<code>\glsxtr@hyperoutsidefalse</code> .....	79	280–283, 285, 297–299, 301, 321–324, 326	
<code>\glsxtr@hyperoutsidettrue</code> .....	79	<code>\glsxtr@abbrvpluralsuffix</code> ...	158, 190,
<code>\glsxtr@ifnextpunc</code> .....	224	232, 250, 252, 255, 258, 261, 263, 266,	
<code>\glsxtr@ifpunctoken</code> .....	224	269, 285, 302, 327, 338, 341, 344, 350, 352	
<code>\glsxtr@inc@linkcount</code> .....	81, 180	<code>\glsxtr@abbrvtype</code> .....	21, 229
<code>\glsxtr@inc@wrglossaryctr</code> ....	8, 9, 27, 31	<code>\glsxtr@AccSuppAbbrSetFirstLongAttrs</code>	
<code>\glsxtr@indexonly@saveentrycounter</code>		.....	249, 252,
.....	16, 17, 31	269, 271, 286, 288, 302, 304, 306, 308,	
<code>\glsxtr@keylist</code> .....	69, 70	328, 329, 331, 335, 338, 344, 347, 349, 352	
<code>\glsxtr@label</code> .....	471	<code>\glsxtr@AccSuppAbbrSetNameLongAttrs</code>	
<code>\glsxtr@langtag</code> .....	157	.....	256,
<code>\glsxtr@linkprefix</code> .....	157, 158	259, 274, 282, 284, 289, 298, 301, 323, 326	
<code>\glsxtr@loaddialect</code> .....	379, 424	<code>\glsxtr@AccSuppAbbrSetNameShortAttrs</code>	
<code>\glsxtr@locationhypertext</code> .....	385	.....	268, 293, 314, 315, 317, 342
<code>\glsxtr@makeglossaries</code> .....	136	<code>\glsxtr@AccSuppAbbrSetNoLongAttrs</code> ..	
<code>\glsxtr@newabbreviation</code> .....	133, 227	.....	254, 257, 260, 262, 273,
<code>\glsxtr@next</code> .....	224	280, 282, 291, 297, 299, 310, 312, 321, 324	
<code>\glsxtr@org@@do@wrglossary</code> .....	31	<code>\glsxtr@AccSuppAbbrSetTextShortAttrs</code>	
<code>\glsxtr@org@dohyperlink</code> .....	104	.....	251, 253,
<code>\glsxtr@org@getgrouptitle</code> .....	146	270, 272, 287, 289, 304, 305, 307, 309,	
<code>\glsxtr@org@newignoredglossary</code> ....	52	331, 333, 334, 336, 339, 345, 348, 351, 354	
<code>\glsxtr@org@makenoidxglossaries</code> ....	66	<code>\glsxtr@activatenopost</code> .....	140
<code>\glsxtr@pluralsuffixes</code> .....	157, 158	<code>\glsxtr@addallcrossrefs</code> .....	64
<code>\glsxtr@printgloss@checkexists</code> ....	141	<code>\glsxtr@alias</code> .....	101
<code>\glsxtr@printgloss@groupstrue</code> ....	143	<code>\glsxtr@AltTreeIndent</code> .....	446
<code>\glsxtr@process</code> .....	163, 166	<code>\glsxtr@altrtreeInit</code> .....	456, 462, 463
<code>\glsxtr@provideignoredglossary</code> ....	53	<code>\glsxtr@AltTreePar</code> .....	446
<code>\glsxtr@punctlist</code> .....	223, 224	<code>\glsxtr@AltTreeSetHangIndent</code> ...	446, 456
<code>\glsxtr@record</code> .....	11, 12, 157	<code>\glsxtr@AltTreeSetSubHangIndent</code> ....	457
<code>\glsxtr@record@nameref</code> .....	12, 157	<code>\glsxtr@altrtreeSubSymbolDescLocation</code>	457
<code>\glsxtr@record@nr</code> .....	16	<code>\glsxtr@altrtreeSymbolDescLocation</code> ..	
<code>\glsxtr@recordsee</code> .....	13	.....	446, 456

<code>\glstrassignactualsetup</code> .....	189, 190	<code>\glstrbookindexthepage</code> .....	470, 471
<code>\glstrassignfieldfont</code> .....	85–92	<code>\glstrcat</code> .....	69, 70
<code>\glstrautoindex</code> .....	213	<code>\glstrchecknohyperfirst</code> .....	85–87
<code>\glstrautoindexassignsort</code> .....	213	<code>\glstrcombiningdiacriticIIrules</code> ..	396
<code>\glstrautoindexentry</code> .....	213	<code>\glstrcombiningdiacriticIrules</code> ...	396
<code>\glstrautoindexesc</code> .....	213	<code>\glstrcombiningdiacriticIVrules</code> ..	396
<code>\glstrbibaddress</code> .....	382	<code>\glstrComputeTreeIndent</code> .....	456, 457
<code>\glstrbibauthor</code> .....	383	<code>\glstrComputeTreeSubIndent</code> ...	456, 457
<code>\glstrbibbooktitle</code> .....	383	<code>\glstrcounterprefix</code> .....	149
<code>\glstrbibchapter</code> .....	383	<code>\glstrcurrencyrules</code> .....	398
<code>\glstrbibedition</code> .....	383	<code>\glstrcurrentgrptitle</code> .....	470
<code>\glstrbibhowpublished</code> .....	383	<code>\GlsXtrDefaultResourceOptions</code> ....	155
<code>\glstrbibinstitution</code> .....	383	<code>\GlsXtrdefaultsubsequentfmt</code> ...	245, 247
<code>\glstrbibjournal</code> .....	383	<code>\glstrdefaultsubsequentfmt</code> ...	245, 247
<code>\glstrbibmonth</code> .....	383	<code>\GlsXtrdefaultsubsequentplfmt</code> .	245, 247
<code>\glstrbibnote</code> .....	383	<code>\glstrdefaultsubsequentplfmt</code> .	245, 247
<code>\glstrbibnumber</code> .....	383	<code>\GlsXtrDefineAbbreviationShortcuts</code> .....	23, 24
<code>\glstrbiborganization</code> .....	383	<code>\GlsXtrDefineAcShortcuts</code> .....	24
<code>\glstrbibpages</code> .....	383	<code>\GlsXtrDefineOtherShortcuts</code> .....	24
<code>\glstrbibpublisher</code> .....	383	<code>\glstrdetoklocation</code> .....	173
<code>\glstrbibschool</code> .....	383	<code>\glstrdiscardperiod</code> .....	221
<code>\glstrbibseries</code> .....	383	<code>\glstrdisplayendloc</code> .....	148
<code>\glstrbibtitle</code> .....	383	<code>\glstrdisplayendlochook</code> .....	149
<code>\glstrbibtype</code> .....	383	<code>\glstrdisplaysingleloc</code> .....	148, 149
<code>\glstrbibvolume</code> .....	383	<code>\glstrdisplaystartloc</code> .....	148
<code>\glstrbookindexatendgroup</code> .....	468	<code>\glstrdoautoindexname</code> ....	102, 103, 209
<code>\glstrbookindexatsubendgroup</code> ....	469	<code>\glstrdohyperlink</code> .....	104, 106
<code>\glstrbookindexatsubsubendgroup</code> ..	469	<code>\glstrdopostpunc</code> .....	258, 259, 283, 285, 299, 301, 324, 326
<code>\glstrbookindexbetween</code> .....	468	<code>\glstrdowrglossaryhook</code> .....	102
<code>\glstrbookindexbookmark</code> .....	470	<code>\GlsXtrDualField</code> .....	382
<code>\glstrbookindexbookmarkprefix</code> ....	470	<code>\glstremssuffix</code> .....	303, 305, 307, 309, 310, 312, 314, 316, 318, 319, 322, 325
<code>\glstrbookindexcols</code> .....	467, 468	<code>\GlsXtrEnableEntryCounting</code> .....	130
<code>\glstrbookindexcolspread</code> .....	467	<code>\GlsXtrEnableEntryUnitCounting</code> ....	118
<code>\glstrbookindexfirstmarkfmt</code> .....	471	<code>\GlsXtrEnableOnTheFly</code> .....	69, 71
<code>\glstrbookindexformatheader</code> .....	470	<code>\glstrendfor</code> .....	38
<code>\glstrbookindexgroupskip</code> .....	470	<code>\glstrfieldlistgadd</code> .....	159
<code>\glstrbookindexlastmarkfmt</code> .....	471	<code>\glstrfieldtitlecase</code> .....	204–207, 211
<code>\glstrbookindexlocation</code> .....	466, 468	<code>\glstrfieldtitlecasecs</code> .....	203
<code>\glstrbookindexmulticolseenv</code> ..	467, 468	<code>\glstrfieldxifinlist</code> .....	167
<code>\glstrbookindexname</code> .....	465, 468	<code>\glstrfirstscfont</code> .....	269
<code>\glstrbookindexparentchildsep</code> 466–468		<code>\glstrfirstsmfont</code> .....	285
<code>\glstrbookindexparentsubchildsep</code> . .....	467–469	<code>\GlsXtrFmtDefaultOptions</code> .....	35, 36
<code>\glstrbookindexprelocation</code> ...	465, 468	<code>\glstrfmtdisplay</code> .....	35, 36
<code>\glstrbookindexsubbetween</code> .....	469	<code>\glstrfmtexternalnameref</code> .....	385
<code>\glstrbookindexsublocation</code> .....	469	<code>\GlsXtrFmtField</code> .....	35, 36
<code>\glstrbookindexsubname</code> .....	469		
<code>\glstrbookindexsubprelocation</code> ....	469		
<code>\glstrbookindexsubsubbetween</code> ....	469		



<code>\glxtrfmtinternalnameref</code> .....	385	<code>\glxtrgeneralpuncIIrules</code> .....	398
<code>\glxtrfootnotedescname</code> .....		<code>\glxtrgeneralpuncIrules</code> .....	398
..... 256, 259, 282, 284, 298, 301, 323, 326		<code>\glxtrgetgrouptitle</code> .....	
<code>\glxtrfootnotedescsort</code> .....		..... 147, 440, 443, 445, 457–463, 470	
..... 256, 259, 282, 284, 298, 301, 323, 326		<code>\glxtrgroupfield</code> .....	169
<code>\glxtrfootnotename</code> .....		<code>\Glsxtrheadfirst</code> .....	357
..... 254, 257, 280, 282, 297, 299, 321, 324		<code>\glxtrheadfirst</code> .....	357
<code>\GlsXtrForeignTextField</code> .....	47	<code>\Glsxtrheadfirstplural</code> .....	358
<code>\GlsXtrFormatLocationList</code> 73, 75, 453–455		<code>\glxtrheadfirstplural</code> .....	358
<code>\GLSxtrfull</code> .....	22, 97, 367, 368	<code>\Glsxtrheadfull</code> .....	358
<code>\Glsxtrfull</code> .....	22, 97, 368	<code>\glxtrheadfull</code> .....	358
<code>\glxtrfull</code> .....	21, 22, 97, 367	<code>\Glsxtrheadfullpl</code> .....	358
<code>\Glsxtrfullformat</code> .....		<code>\glxtrheadfullpl</code> .....	358
..... 231, 245, 247, 250, 252, 255,		<code>\Glsxtrheadlong</code> .....	358
258, 261, 263, 267, 270, 272, 274, 275,		<code>\glxtrheadlong</code> .....	358
278, 279, 281, 283, 286, 288, 291, 292,		<code>\Glsxtrheadlongpl</code> .....	358
295–297, 300, 303, 305, 307, 309, 311,		<code>\glxtrheadlongpl</code> .....	358
313, 315, 317, 319, 321, 322, 325, 329,		<code>\Glsxtrheadname</code> .....	357
330, 332, 336, 339, 342, 345, 347, 350, 353		<code>\glxtrheadname</code> .....	160, 357
<code>\glxtrfullformat</code> .....		<code>\Glsxtrheadplural</code> .....	357
..... 231, 245, 247, 250, 252, 255,		<code>\glxtrheadplural</code> .....	357
258, 261, 263, 267, 270, 272, 274, 275,		<code>\Glsxtrheadshort</code> .....	357
278–280, 283, 286, 288, 290, 292, 294,		<code>\glxtrheadshort</code> .....	357
296, 297, 300, 303, 305, 307, 309, 311,		<code>\Glsxtrheadshortpl</code> .....	357
312, 315, 317, 319, 320, 322, 325, 328,		<code>\glxtrheadshortpl</code> .....	357
330, 332, 335, 339, 341, 345, 347, 350, 353		<code>\Glsxtrheadtext</code> .....	357
<code>\GLSxtrfullpl</code> .....	22, 97, 367–369	<code>\glxtrheadtext</code> .....	357
<code>\Glsxtrfullpl</code> .....	22, 97, 369	<code>\glxtrhiernamesep</code> .....	58, 59
<code>\glxtrfullpl</code> .....	21, 22, 97, 368	<code>\glxtrhyperlink</code> .....	26, 27, 105
<code>\Glsxtrfullplformat</code> .....		<code>\glxtrhyphensuffix</code> .....	339, 347
..... 231, 244, 247, 250, 253, 255, 258,		<code>\glxtridentifyglslike</code> .....	171
261, 263, 267, 270, 272, 274, 275, 278,		<code>\glxtrifcounttrigger</code> .....	121, 122
279, 281, 283, 287, 289, 291, 292, 295,		<code>\glxtrifcustomdiscardperiod</code> .....	221
296, 298, 300, 303, 305, 307, 309, 311,		<code>\glxtrifemptyglossary</code> .	148, 154, 162, 165
313, 315, 317, 319, 321, 322, 325, 329,		<code>\GlsXtrIfFieldEqNum</code> .....	160
330, 332, 336, 339, 342, 345, 347, 350, 353		<code>\glxtrifhasfield</code> ..	47, 58, 59, 381, 382, 465
<code>\glxtrfullplformat</code> .....		<code>\glxtrifhyphenstart</code> 337, 340, 343, 346, 349	
..... 244, 247, 250, 252, 255,		<code>\glxtrifindexing</code> .....	102
258, 261, 263, 267, 270, 272, 274, 275,		<code>\glxtrifinmark</code> .....	83, 112–115, 356–358
278, 279, 281, 283, 286, 288, 291, 292,		<code>\glxtrifnextpunc</code> .....	224, 225
295–297, 300, 303, 305, 307, 309, 311,		<code>\glxtrifperiod</code> .....	221, 223
313, 315, 317, 319, 320, 322, 325, 328,		<code>\glxtrifrecordtrigger</code> .....	175–177
330, 332, 336, 339, 342, 345, 347, 350, 353		<code>\GlsXtrIfUnusedOrUndefined</code> .....	102
<code>\glxtrfullsep</code> .....		<code>\glxtrifwasfirstuse</code> .....	
..... 230, 249–253, 256, 258, 259, 261–		84–87, 92–96, 100, 134, 222, 233, 236–	
266, 269–279, 281, 284, 286–294, 296,		242, 257–260, 283–285, 299–302, 324,	
298, 300–313, 315–320, 322, 323, 325,		326, 329, 331, 332, 334, 344, 346, 349, 351	
327, 337–339, 341, 343, 346–349, 353, 354		<code>\glxtrinclinkcounter</code> .....	179
<code>\glxtrgenabbrvfmt</code> .....	76	<code>\glxtrindexaliased</code> .....	101

<code>\glxstrindexseealso</code> .....	63, 64	<code>\glxstrlocrangefmt</code> .....	148, 149
<code>\glxstrinithyperoutside</code> .....	81	<code>\GLSxtrlong</code> .....	22, 95, 365, 366
<code>\glxstrnitrwgloss</code> .....	81, 174	<code>\Glsxtrlong</code> .....	22, 95, 366
<code>\glxstrnitrwglossbeforefalse</code> ...	78, 79	<code>\glxstrlong</code> .....	21, 22, 94, 365
<code>\glxstrnitrwglossbeforetrue</code> ....	78, 79	<code>\glxstrlonghyphen</code> .....	345
<code>\Glsxtrinlinefullformat</code> .....	231, 233, 247, 256, 259, 261, 263, 264, 266, 274–277, 279, 281, 284, 290, 292–294, 296, 298, 301, 311–313, 315, 317, 318, 320, 323, 325, 330, 333, 341, 345, 351, 354	<code>\glxstrlonghyphennoshort</code> .....	341, 342
<code>\glxstrinlinefullformat</code> 231–233, 247, 255, 258, 261, 263, 264, 266, 273, 275– 277, 279, 281, 284, 290, 291, 293, 294, 296, 298, 300, 310, 312–314, 316, 318, 320, 322, 325, 330, 333, 341, 345, 350, 353		<code>\glxstrlonghyphenshort</code> .....	339
<code>\Glsxtrinlinefullplformat</code> .. 231, 235, 247, 256, 259, 261, 263, 265, 266, 274– 276, 278, 279, 281, 284, 290, 292–294, 296, 298, 301, 311–313, 315, 317, 319, 320, 323, 325, 330, 333, 341, 345, 351, 354		<code>\glxstrlongnoshortdescname</code> .	265, 319, 340
<code>\glxstrinlinefullplformat</code> 231, 234, 235, 247, 256, 259, 261, 263, 264, 266, 273, 275–277, 279, 281, 284, 290, 292–294, 296, 298, 300, 311–313, 315, 316, 318, 320, 323, 325, 330, 333, 341, 345, 351, 353		<code>\glxstrlongnoshortname</code> .....	268, 276, 293, 314, 316, 342
<code>\glxstrinsertinsidefalse</code> .....	249	<code>\GLSxtrlongpl</code> .....	22, 96, 366, 367
<code>\GlsXtrInternalLocationHyperlink</code> 27, 150		<code>\Glsxtrlongpl</code> .....	22, 96, 367
<code>\glxstrLatinA</code> .....	400–405	<code>\glxstrlongpl</code> .....	21, 22, 95, 366
<code>\glxstrLatinAELigature</code> .....	402, 404, 405	<code>\glxstrlongshortdescname</code> .....	251, 270, 287, 304, 306, 339, 345
<code>\glxstrLatinE</code> .....	400–405	<code>\glxstrlongshortdescsort</code> .....	251, 270, 287, 304, 306, 334, 339, 345
<code>\glxstrLatinEszettSs</code> .....	401–403, 405	<code>\glxstrlongshortname</code> .....	249, 269, 286, 302, 304, 328, 329, 338, 344
<code>\glxstrLatinEszettSz</code> .....	402, 404	<code>\glxstrlongshortuserdescname</code> ..	331, 334
<code>\glxstrLatinEth</code> .....	401–404	<code>\glxstrmarkhook</code> .....	355, 356
<code>\glxstrLatinH</code> .....	400–405	<code>\glxstrMathItalicAlpha</code> .....	410, 413, 414
<code>\glxstrLatinI</code> .....	400–405	<code>\glxstrMathItalicBeta</code> .....	410, 413, 414
<code>\glxstrLatinInsularG</code> .....	404	<code>\glxstrMathItalicChi</code> ...	410, 411, 414, 415
<code>\glxstrLatinK</code> .....	400–405	<code>\glxstrMathItalicDelta</code> .....	410, 413, 415
<code>\glxstrLatinL</code> .....	400–405	<code>\glxstrMathItalicEpsilon</code> ...	410, 413, 415
<code>\glxstrLatinM</code> .....	400–405	<code>\glxstrMathItalicEta</code> .....	410, 414, 415
<code>\glxstrLatinN</code> .....	400–405	<code>\glxstrMathItalicGamma</code> .....	410, 413, 414
<code>\glxstrLatinO</code> .....	400–405	<code>\glxstrMathItalicIota</code> .....	410, 414, 415
<code>\glxstrLatinOELigature</code> .....	402, 405	<code>\glxstrMathItalicKappa</code> .....	410, 414, 415
<code>\glxstrLatinP</code> .....	400–405	<code>\glxstrMathItalicLambda</code> ....	410, 414, 415
<code>\glxstrLatinS</code> .....	400–405	<code>\glxstrMathItalicMu</code> .....	410, 414, 415
<code>\glxstrLatinT</code> .....	400–405	<code>\glxstrMathItalicNu</code> .....	410, 414, 415
<code>\glxstrLatinThorn</code> .....	405	<code>\glxstrMathItalicOmega</code> .	410, 411, 414, 415
<code>\glxstrLatinX</code> .....	401–405	<code>\glxstrMathItalicOmicron</code> ...	410, 414, 415
<code>\GlsXtrLocationField</code> .....	169	<code>\glxstrMathItalicPhi</code> ...	410, 411, 414, 415
<code>\glxstrlocationhyperlink</code> .....	149	<code>\glxstrMathItalicPi</code> .....	410, 414, 415
		<code>\glxstrMathItalicPsi</code> ...	410, 411, 414, 415
		<code>\glxstrMathItalicRho</code> .....	410, 414, 415
		<code>\glxstrMathItalicSigma</code> .....	410, 414, 415
		<code>\glxstrMathItalicTau</code> .....	410, 414, 415
		<code>\glxstrMathItalicTheta</code> ....	410, 414, 415
		<code>\glxstrMathItalicUpsilon</code> ...	410, 414, 415
		<code>\glxstrMathItalicXi</code> .....	410, 414, 415
		<code>\glxstrMathItalicZeta</code> .....	410, 414, 415
		<code>\glxstrmultisupplcation</code> .....	383
		<code>\glxstrnamereflink</code> .....	384
		<code>\glxstrnewabbrevpresetkeyhook</code> .....	228
		<code>\glxstrnewnumber</code> .....	23



\glxtrnewsymbol .....	23	308, 310, 312, 314, 316, 317, 319, 321–	
\glxstrNoGlossaryWarning .....	16, 25, 150	324, 326, 328, 329, 331, 332, 334–336,	
\GlsXtrNoGlsWarningAutoMake .....	154	338–340, 342, 344, 346–349, 351, 353, 354	
\GlsXtrNoGlsWarningBuildInfo .....	154	\glxstrpostreset .....	117, 119, 127, 128
\GlsXtrNoGlsWarningCheckFile .....	154	\glxstrpostunset .....	115, 119, 127
\GlsXtrNoGlsWarningEmptyMain .....	154	\glxstrprelocation .....	
\GlsXtrNoGlsWarningEmptyNotMain ...	154	.....	427, 430, 432, 434, 436, 439, 465
\GlsXtrNoGlsWarningEmptyStart .....	154	\glxstrprotectlinks .....	105, 106
\GlsXtrNoGlsWarningHead .....	154	\glxstrprovideaccsuppcmd .....	193
\GlsXtrNoGlsWarningMisMatch .....	154	\GlsXtrRecordCounter .....	13
\GlsXtrNoGlsWarningNoOut .....	155	\glxstrrecordtriggervalue .....	174
\GlsXtrNoGlsWarningTail .....	155	\GlsXtrRecordWarning .....	155
\glxstrnopostpunc .....	141	\glxstrregularfont .....	76, 84
\glxstronlydescname .....	354	\glxstrresourcecount .....	156
\glxstronlydescsort .....	354	\glxstrresourcefile .....	156
\glxstronlyname .....	352	\glxstrresourceinit .....	155
\glxstronlysuffix .....	353	\glxstrrestoremarkhook .....	355, 356
\glxstrorg@ifKV@glslink@hyper .....	76	\glxstrrestorepostpunc .....	141, 142
\glxstrorglong .....	189, 190, 227, 250, 340	\glxstrscfont .....	269
\glxstrorgshort .....	189, 190, 227, 250	\glxstrscsuffix .....	
\GLSxtrp .....	111	....	269, 271, 273, 275, 277, 278, 280, 283
\Glsxtrp .....	111	\GlsXtrSetActualChar .....	215
\glxtrp .....	111, 113	\glxtrsetaliasnoindex .....	16, 17, 101
\glxtrparen .....	222,	\GlsXtrSetEncapChar .....	216
230, 249–253, 256, 259, 261–266, 269–		\GlsXtrSetEscChar .....	215
279, 281, 284, 286–290, 292–294, 296,		\glxtrsetfieldifexists .....	44, 45
298, 300–313, 315–320, 322, 323, 325,		\glxtrsetglossarylabel .....	142
327, 337–339, 341, 343, 346–349, 353, 354		\GlsXtrSetLevelChar .....	216
\glxstrpdfentryfmt .....	36	\glxtrsetpopts .....	110
\Glsxtrpl .....	71	\glxtrsetupfulldefs .....	232–
\glxtrpl .....	71	235, 258, 260, 283, 285, 300, 302, 324, 326	
\glxstrpostdescription .	141, 202, 220, 438	\GLSxtrshort .....	22, 93, 114, 115, 359, 360
\glxstrpostthyphenlong .....	349, 351	\Glsxtrshort .....	21, 22, 92, 360
\glxstrpostthyphenshort .....	344, 346	\glxtrshort .....	21, 22, 92, 359
\glxstrpostthyphensubsequent .....		\glxtrshortdescname ...	262, 274, 291, 312
.....	344, 346, 350, 351	\glxtrshorthyphen .....	350
\glxstrpostlink .....	221	\glxtrshorthyphenlong .....	347, 348
\glxstrpostlinkendsentence .....	221	\glxtrshortlongdescname .....	
\glxstrpostlinkhook .....	221	.....	253, 272, 289, 307, 309, 348, 351
\glxstrpostlocalreset .....	117, 120, 128	\glxtrshortlongdescsort .....	
\glxstrpostlocalunset .....	117, 119, 127	....	253, 272, 289, 307, 309, 336, 348, 351
\glxstrpostlongdescription .....	51	\glxtrshortlongname .....	
\glxstrpostnamehook .....	207–209, 211	252, 271, 288, 306, 308, 332, 335, 347, 349	
\GlsXtrPostNewAbbreviation .....		\glxtrshortlonguserdescname ..	333, 336
....	230, 247, 250–253, 255, 257, 259,	\glxtrshortnolongname .	260, 273, 290, 310
260, 262–273, 275, 277, 280, 282–284,		\GLSxtrshortpl .....	22, 94, 359, 360
286–291, 293, 297, 299, 301, 303–306,		\Glsxtrshortpl .....	21, 22, 94, 360
		\glxtrshortpl .....	21, 22, 93, 359
		\glxtrsmfont .....	285

<code>\glxtrrmsuffix</code> .....	<code>\glxtrrtitleshortpl</code> .....
..... 286, 288, 290, 291, 294, 295, 297, 300	357, 358, 369
<code>\GlsXtrStandaloneEntryName</code> .....	<code>\GLSxtrrtitletext</code> .....
160	372
<code>\GlsXtrStandaloneEntryOther</code> .....	<code>\Glsxtrrtitletext</code> .....
161	357, 358, 371
<code>\GlsXtrStandaloneGlossaryType</code> .	<code>\Glsxtrrtitletext</code> .....
160, 161	357, 358, 371
<code>\GlsXtrStandaloneSubEntryItem</code> .	<code>\GlsXtrTotalRecordCount</code> .....
160, 161	173
<code>\Glsxtrsubsequentfmt</code> .....	<code>\glxtrtreechildpredesc</code> .....
..... 244, 247, 266, 277, 279,	442, 446
294, 295, 314, 316, 318, 320, 341, 344, 350	<code>\glxtrtreepredesc</code> .....
<code>\glxtrsubsequentfmt</code> .....	441, 446
..... 244, 247, 266, 277, 278,	<code>\glxtrtreetopindent</code> .....
294, 295, 314, 316, 318, 320, 341, 344, 350	446, 455
<code>\Glsxtrsubsequentplfmt</code> .....	<code>\glxtrundefaction</code> ..
..... 243, 247, 266, 277, 279,	7, 15–17, 33, 52, 55, 56
294, 295, 314, 316, 318, 320, 341, 344, 350	<code>\glxtrundeftag</code> .....
<code>\glxtrsubsequentplfmt</code> .....	32, 144, 145
..... 243, 247, 266, 277, 279,	<code>\GlsXtrUnknownDialectWarning</code> .....
294, 295, 314, 316, 318, 320, 341, 344, 350	48
<code>\glxtrsupplocationurl</code> .....	<code>\glxtrunsrtdo</code> .....
150, 383, 385	167
<code>\glxtrtagfont</code> .....	<code>\glxtrUpAlpha</code> .....
219	408, 409, 413, 414
<code>\GLSxtrrtitlefirst</code> .....	<code>\glxtrUpBeta</code> .....
373	408, 409, 413, 414
<code>\Glsxtrrtitlefirst</code> .....	<code>\glxtrUpChi</code> .....
357, 358, 373	409, 414, 415
<code>\glxtrrtitlefirst</code> .....	<code>\glxtrUpDelta</code> .....
357, 358, 373	408, 409, 413, 415
<code>\GLSxtrrtitlefirstplural</code> .....	<code>\glxtrUpDigamma</code> .....
374	408, 410, 414
<code>\Glsxtrrtitlefirstplural</code> ....	<code>\glxtrUpEpsilon</code> .....
357, 358, 374	408, 409, 414, 415
<code>\glxtrrtitlefirstplural</code> ....	<code>\glxtrUpEta</code> .....
357, 358, 374	408, 409, 414, 415
<code>\GLSxtrrtitlefull</code> .....	<code>\glxtrUpGamma</code> .....
377	408, 409, 413, 415
<code>\Glsxtrrtitlefull</code> .....	<code>\glxtrUpIota</code> .....
357, 358, 377	409, 414, 415
<code>\glxtrrtitlefull</code> .....	<code>\glxtrUpKappa</code> .....
357, 358, 376	409, 414, 415
<code>\GLSxtrrtitlefullpl</code> .....	<code>\glxtrUpLambda</code> .....
378	409, 414, 415
<code>\Glsxtrrtitlefullpl</code> .....	<code>\glxtrUpMu</code> .....
357, 358, 377, 378	409, 414, 415
<code>\glxtrrtitlefullpl</code> .....	<code>\glxtrUpNu</code> .....
357, 358, 377	409, 414, 415
<code>\GLSxtrrtitlelong</code> .....	<code>\glxtrUpOmega</code> .....
375	409, 414, 415
<code>\Glsxtrrtitlelong</code> .....	<code>\glxtrUpOmicron</code> .....
357, 358, 375	409, 414, 415
<code>\glxtrrtitlelong</code> .....	<code>\glxtrUpPhi</code> .....
357, 358, 374, 375	409, 414, 415
<code>\GLSxtrrtitlelongpl</code> .....	<code>\glxtrUpPi</code> .....
376	409, 414, 415
<code>\Glsxtrrtitlelongpl</code> .....	<code>\glxtrUpPsi</code> .....
357, 358, 376	409, 414, 415
<code>\glxtrrtitlelongpl</code> .....	<code>\glxtrUpRho</code> .....
357, 358, 375	409, 414, 415
<code>\GLSxtrrtitlename</code> .....	<code>\glxtrUpSigma</code> .....
371	409, 414, 415
<code>\Glsxtrrtitlename</code> .....	<code>\glxtrUpTau</code> .....
357, 358, 370, 371	409, 414, 415
<code>\glxtrrtitlename</code> .....	<code>\glxtrUpTheta</code> .....
357, 358, 370	408, 409, 414, 415
<code>\glxtrrtitleorpdforheading</code> .....	<code>\glxtrUpUpsilon</code> .....
..... 28, 29, 159–161, 357, 358	409, 414, 415
<code>\GLSxtrrtitleplural</code> .....	<code>\glxtrUpXi</code> .....
372, 373	409, 414, 415
<code>\Glsxtrrtitleplural</code> .....	<code>\glxtrUpZeta</code> .....
357, 358, 372	408, 409, 414, 415
<code>\glxtrrtitleplural</code> .....	<code>\GlsXtrUseAbbrStyleFmts</code> .....
357, 358, 372	..... 251, 254, 257, 260, 262,
<code>\Glsxtrrtitleshort</code> .....	264, 265, 267, 268, 271, 273, 276, 282,
357, 358, 370	285, 287, 289, 292, 299, 302, 304, 306,
<code>\glxtrrtitleshort</code> .....	308, 310, 313, 318, 321, 324, 326, 331,
357, 358, 369	334, 335, 337, 340–342, 346, 348, 352, 355
<code>\Glsxtrrtitleshortpl</code> .....	<code>\GlsXtrUseAbbrStyleSetup</code> .....
357, 358, 370	..... 262, 264, 265, 267,
<code>\glxtrtrtitleshortpl</code> .....	268, 276, 278, 292, 295, 313, 317, 318, 321
357, 358, 370	<code>\glxtrusefield</code> .....
<code>\glxtrtrtitleshort</code> .....	380
357, 358, 369	<code>\glxtruserfield</code> .....
<code>\Glsxtrtrtitleshortpl</code> .....	327
357, 358, 370	<code>\glxtruserparen</code> .....
	328–336

<code>\glstrusersuffix</code> .....	328, 330, 332, 335	
<code>\glstruseeseealsoformat</code> .....	60, 61	
<code>\glstruseeseeformat</code> .....	57, 60	
<code>\GlsXtrWarnDeprecatedAbbrStyle</code>	227, 248	
<code>\GlsXtrWarning</code> .....	69, 70	
<code>\glstrword</code> .....	226	
<code>\glstrwordsep</code> ..	226, 337, 340, 343, 346, 349	
<code>\glstrwrglossmark</code> .....	28	

**H**

<code>\hangindent</code> .....	442, 444, 446, 456–458, 462, 463, 495, 497, 500	
<code>\hbox</code> .....	426	
<code>\hfill</code> .....	426	
<code>\href</code> .....	105	
<code>\hsize</code> .....	72, 73	
<code>\hss</code> .....	426	
<code>\hyperlink</code> .....	17, 105, 384	
<code>\hyperpage</code> .....	212	
<code>\hyperref</code> .....	105, 150, 381, 386	
<code>hyperref package</code> .....	106, 212, 355, 369, 384	

**I**

<code>\if</code> .....	68	
<code>\if@display</code> .....	12	
<code>\if@glstr@autoseeindex</code> .....	30, 57, 62	
<code>\if@glstr@equations</code> .....	9, 81	
<code>\if@glstr@floats</code> .....	19	
<code>\if@glstr@format@override</code> .....	212	
<code>\if@glstr@docdefrestricted</code> .....	67	
<code>\if@glstr@indexcrossrefs</code> .....	18, 64	
<code>\ifblank</code> .....	34, 69, 70, 136	
<code>\ifbool</code> .....	33, 50	
<code>\ifcase</code> ..	7, 16, 23, 25, 28, 67, 79, 142, 440, 468	
<code>\ifcsdef</code> .....	33, 43, 44, 49, 52–55, 79, 82, 98, 99, 110–115, 125, 140, 146, 147, 160, 168, 171, 173, 178, 204–210, 217, 221, 227, 243, 247, 384, 430–437, 498	
<code>\ifcsname</code> .....	428	
<code>\ifcsstring</code> .....	33, 199, 246	
<code>\ifcsundef</code> .....	33, 42, 47–50, 52–54, 67, 71, 74, 106, 119, 125– 128, 146, 147, 150, 167, 179, 180, 192, 199, 246–248, 426, 447, 448, 455, 470, 498	
<code>\ifcsvoid</code> .....	64, 101, 198	
<code>\ifdef</code> .....	16, 23, 28, 30, 33, 36, 43, 47, 49, 50, 55, 56, 61, 62, 65, 72, 73, 100, 104, 105, 112–114, 136, 139, 140, 144, 145, 156–158, 162, 171, 189, 201–203, 215–217, 219, 327, 356, 369–378, 381, 386–389, 424, 426–430, 438–441, 443, 445, 457–463, 466, 470, 471	
<code>\ifdefempty</code> .....	7–9, 41, 42, 47, 48, 52–54, 57, 60, 81, 83, 119, 130– 132, 136, 139, 149, 155, 163, 165, 169, 174, 190–192, 218, 226, 243, 389, 390, 467	
<code>\ifdefequal</code> .....	45, 46, 66, 104, 131, 133, 154, 169, 171, 210	
<code>\ifdefstring</code> .....	..... 6, 15, 27, 45, 135, 158, 212, 218, 466	
<code>\ifdefvoid</code> .....	56, 62–65, 105, 124, 146, 150, 169, 170, 385	
<code>\ifdim</code> .....	72, 73, 134, 447–455, 475, 499	
<code>\IfFileExists</code>	25, 150, 154, 155, 158, 424, 425	
<code>\ifglossaryexists</code> .....	56, 164, 379, 380	
<code>\ifglssacronym</code> .....	21, 154, 379	
<code>\ifglssacrshortcuts</code> .....	23	
<code>\ifglssautomake</code> .....	139, 154, 158	
<code>\ifglssentrycounter</code> .....	49	
<code>\ifglssentryexists</code> .....	9, 33, 55, 56, 69, 70, 73, 84, 169, 199, 220, 221, 390	
<code>\ifglssfieldeq</code> .....	197	
<code>\ifglsshasdesc</code> ...	441, 442, 495, 496, 498, 500	
<code>\ifglsshasfield</code> .....	35, 36, 327	
<code>\ifglsshaslong</code> .....	124, 177, 178	
<code>\ifglsshasparent</code> .....	..... 160, 161, 163, 166, 448, 450, 451	
<code>\ifglsshashshort</code> .....	57–59, 76, 84	
<code>\ifglsshashsymbol</code> .....	..... 222, 441, 442, 444, 446, 496, 498	
<code>\ifglssindexonlyfirst</code> .....	102	
<code>\ifGlsLongExtraUseTabular</code>	476, 478, 479, 481, 482, 484, 485, 487, 488, 490, 491, 493	
<code>\ifglssnogroupskip</code> .....	428, 430– 438, 440, 443, 444, 457, 466, 477, 479– 481, 483, 485, 486, 488, 489, 491, 492, 494	
<code>\ifglssnonumberlist</code> .....	75	
<code>\ifglssnopostdot</code> .....	20, 141	
<code>\ifglssanitizesort</code> .....	138	
<code>\ifglssubentrycounter</code> .....	49	
<code>\ifglssused</code> .....	64, 65, 99, 100, 120, 129, 134, 243, 448–450, 452–454	
<code>\ifglssxindy</code> .....	150, 152	
<code>\ifglstr@hyperoutside</code> .....	82	
<code>\ifglstr@printgloss@groups</code> ...	163, 166	
<code>\ifglstr@trinitwrglossbefore</code>	78, 82, 83, 175	
<code>\ifglstr@insertinside</code> .....	. 236–243, 245, 250, 252, 253, 255, 256, 258, 259, 261–267, 270, 272–281, 283,	

284, 286–298, 300, 301, 303, 305, 307, 309–323, 325, 328–330, 332, 333, 336– 338, 340, 343, 345, 346, 349, 351, 353, 354	\LaTeX ..... 152, 153
\ifHy@hyperindex ..... 212	\leaders ..... 426
\ifinlist ..... 116	\leavevmode ..... 52, 80
\ifinlistcs ..... 38, 67	\let ..... 5, 7–11, 13, 15–17, 19, 21–23, 29–31, 35, 38, 46–48, 51, 60, 61, 66, 68, 71, 72, 74–78, 81–96, 98–101, 103, 104, 106, 107, 110, 115– 120, 127–137, 139–147, 155, 156, 158, 162–166, 169, 170, 174, 180, 189, 191, 204–214, 217–220, 224, 226–229, 232– 242, 245, 247, 258, 260, 283, 285, 300, 302, 324, 326, 355–358, 380, 381, 384, 385, 424, 427, 439, 446, 448, 459, 467–470
\ifinner ..... 29	\letabbreviationstyle ..... ..... 256, 257, 259, 260, 262, 264, 267, 268, 274, 276, 291, 292, 311, 313
\ifKV@glslink@hyper ..... 76, 81–83	\letcs ..... 34, 41, 42, 57, 60, 65, 79, 82, 98, 144–146, 168, 169, 204–211, 217, 471
\ifKV@glslink@local ..... 78, 175	\levelchar ..... 216
\ifKV@glslink@noindex 8, 9, 13, 36, 101, 102	\linewidth ..... 475
\ifmmode ..... 12, 28, 29	\listadd ..... 125
\ifnum ..... 18, 42, 65, 66, 121, 128, 129, 146, 156, 169, 174, 386–389, 442, 444, 456, 457, 467, 468, 495, 499, 500	\listbreak ..... 218
\ifst@rred ..... 12	\listcsadd ..... 37
\ifstrempty ..... 160, 171, 180, 384–390	\listcseadd ..... 37, 126
\ifstrequal ..... 20, 25, 80, 103, 384	\listcsgadd ..... 37, 67
\ifthenelse ..... 154, 221	\listcsxadd ..... 37, 125
\IfTrackedDialectHasMapping ..... 47	\listxadd ..... 116
\IfTrackedLanguageFileExists ..... 378	\loadglsentries ..... 68, 152
\ifundef ..... 17, 27, 41, 42, 47, 136, 218–220, 384, 424, 475	\long ..... 51
\ifx ..... 8, 10–12, 15, 31, 32, 60, 61, 72, 73, 82, 133, 136, 140, 143, 148, 149, 156–158, 164, 212–214, 216, 224, 226, 229, 337, 385, 464	
\immediate ..... 66, 120, 129, 150, 151, 158	
\index ..... 213	
\indexspace ..... 427, 439, 466, 470	
\input ..... 378	
\inputencodingname ..... 158	
\InputIfFileExists ..... 66	
\istfilename ..... 136	
\item 152, 153, 426, 427, 429, 439, 440, 458, 459	
<b>J</b>	
\jobname ..... 66, 150, 152–156, 158	
<b>K</b>	
\key@ifundefined 13, 14, 34, 97, 162, 165, 168	
\KV@glslink@hyperfalse .. 85, 100, 106, 107	
\KV@glslink@hypertrue ..... 106	
\KV@glslink@noindexfalse ..... 84, 100	
\KV@glslink@noindextrue ..... 101, 107	
<b>L</b>	
\L ..... 405, 408	
\l ..... 408	
\label ..... 27, 142	
\large ..... 497	
	<b>M</b>
	\m@ne ..... 381
	\MakeAcronymsAbbreviations ..... 134
	\makeatletter ..... 66, 150, 156, 214
	\makeatother ..... 214
	\makebox ..... 426, 456, 457, 499
	\makefirstuc ..... 219
	makeglossaries ..... 143
	\makeglossaries .... 135, 151, 153, 154, 159
	makeindex ..... 501
	makeindex ..... 16, 135, 501
	\makenoidxglossaries ..... 153
	\MakeTextUppercase ..... 357
	\MakeUppercase ..... 357, 358
	\marginpar ..... 29
	\markboth ..... 356
	\markright ..... 356
	\mathit ..... 393
	\mathrm ..... 392–394

<code>\maxdimen</code> .....	72, 73	<code>\newentry</code> .....	23
<code>\mbox</code> .....	428, 429, 456	<code>\newenvironment</code> .....	164
<code>\medskip</code> .....	154, 168, 498	<code>\newglossary</code> .....	21, 136
<code>\MessageBreak</code> .....	67, 68, 71, 121, 129, 130, 136, 139, 140, 155, 164, 246	<code>\newglossaryentry</code> .....	23, 66–68, 119, 127, 131, 202, 229
<code>mfirstuc package</code> .....	218, 219	<code>\newglossaryentry options</code>	
<code>\mfirstucMakeUppercase</code> .....	43, 85–96, 99, 108–110, 112, 114, 115, 123, 124, 132, 178, 181–187, 194–197, 206, 207, 211, 233, 235, 237, 239, 241–245	access .....	191
<code>\mfu@checkword@arg</code> .....	218, 219	alias .....	19, 56, 60, 62–64
<code>\mfu@checkword@do</code> .....	219	desc .....	184, 195
<code>\midrule</code> .....	474, 477, 479, 480, 482, 483, 485, 487, 488, 490, 491, 493	descplural .....	185, 195, 196
<b>N</b>			
<code>\NeedsTeXFormat</code> ...	5, 379, 425, 465, 472, 495	description .....	522
<code>\new@atom@glossaryentry</code> .....	66	first ...	104, 182, 194, 249, 363–365, 373, 501
<code>\new@command</code> .....	381	firstaccess .....	192
<code>\new@glossaryentry</code> .....	68, 139	firstplural	182, 183, 194, 195, 249, 364, 374, 501
<code>\new@ifnextchar</code> .....	35, 98, 99, 123, 171, 175–177, 223, 232–242, 391, 392	group .....	168, 169
<code>\newabbr</code> .....	22	loclist .....	37
<code>\newabbreviation</code> .....	22	long .....	187, 196, 374
<code>\newabbreviationhook</code> .....	229	longplural .....	187, 197, 375
<code>\newabbreviationstyle</code> ..	249, 251, 253, 254, 256, 257, 259, 260, 262, 264, 265, 267–274, 276, 278, 280, 282, 284, 286, 287, 289, 291–293, 295, 296, 298, 299, 301–311, 313–315, 317–319, 321, 323, 324, 326, 328, 329, 331, 333–336, 338– 340, 342, 343, 345, 347–349, 351, 352, 354	name .....	57, 180, 194, 212, 360, 361, 370
<code>\newacronym</code> .....	131, 133	plural .....	181, 182, 194, 249, 362, 363, 372
<code>\newacronymhook</code> .....	131	see .....	18, 19, 30, 56, 57, 62, 64, 68, 137
<code>\newacronymstyle</code> .....	132–134	seealso .....	19, 56, 59, 62, 64, 512
<code>\newcommand</code> .....	5–15, 17–19, 21–27, 29, 31–49, 51–61, 63–65, 68–71, 73–76, 78–80, 83–85, 92, 98–105, 107, 110, 112–119, 121, 123– 127, 129, 130, 132–135, 140–157, 159– 162, 164–171, 173–190, 192–203, 209, 210, 212–227, 230–243, 245–251, 253, 254, 256, 260, 262, 265, 267–269, 285, 302, 327, 330, 333, 337, 338, 340, 343, 346, 348, 349, 352, 354, 356–378, 380– 386, 389–392, 395–423, 425, 427–429, 438, 439, 441–448, 455, 456, 465–467, 470–480, 482, 483, 485–491, 493, 496–499	short .....	57, 186, 196, 225
<code>\newcount</code> .....	143, 156, 170	shortaccess .....	190
<code>\newcounter</code> .....	27, 178	shortaccessplural .....	190
		shortplural .....	186, 196, 225
		symbol .....	183, 195
		symbolplural .....	183, 184, 195
		text	57, 104, 181, 194, 249, 251, 361, 362, 371
		textaccess .....	191
		user2 .....	48
		<code>\newglossarystyle</code> .....	467, 476, 478, 479, 481, 482, 484, 485, 487, 488, 490, 491, 493, 495, 499
		<code>\newif</code> .....	78, 211, 249, 476
		<code>\newlength</code> .....	446, 497
		<code>\newnum</code> .....	23
		<code>\newrobustcmd</code> .....	35–38, 40, 44–46, 60–62, 66, 80, 84, 98, 99, 110– 112, 123, 141, 146, 160, 161, 167, 168, 171, 172, 175–177, 210, 217–219, 232– 243, 337, 356, 359–369, 391, 392, 448–455
		<code>\newsym</code> .....	23
		<code>\newterm</code> .....	201
		<code>\newtoks</code> .....	225
		<code>\newwrite</code> .....	66, 136
		<code>\nfss@text</code> .....	29
		<code>\nobreak</code> .....	427–429, 439, 496
		<code>\NoCaseChange</code> .....	112–115, 161, 359–368

`\noexpand` ..... 11, 13, 26, 60, 63–  
     65, 80, 131, 150, 151, 156, 163, 166, 167,  
     169, 179, 189, 190, 213, 214, 229, 381,  
     390, 425, 467–469, 476–490, 492, 493, 500  
`\nofiles` ..... 154  
`\noindent` ..... 154, 443, 445, 459–463, 496  
`\nopagebreak` ..... 439, 466, 470, 495, 500  
`\nopostdesc` ..... 52, 69, 70, 141, 165, 202  
`\np@glstr@assign@leveloffset` ..... 143  
`\ns@GLSxtrfull` ..... 233  
`\ns@GLSxtrfull` ..... 233  
`\ns@glstrfull` ..... 232  
`\ns@GLSxtrfullpl` ..... 235  
`\ns@GLSxtrfullpl` ..... 234  
`\ns@glstrfullpl` ..... 234  
`\ns@GLSxtrlong` ..... 238  
`\ns@GLSxtrlong` ..... 238  
`\ns@glstrlong` ..... 237  
`\ns@GLSxtrlongpl` ..... 242  
`\ns@GLSxtrlongpl` ..... 241, 242  
`\ns@glstrlongpl` ..... 241  
`\ns@GLSxtrshort` ..... 236, 237  
`\ns@GLSxtrshort` ..... 236  
`\ns@glstrshort` ..... 235  
`\ns@GLSxtrshortpl` ..... 240  
`\ns@GLSxtrshortpl` ..... 240  
`\ns@glstrshortpl` ..... 239  
`\null` ..... 25  
`\number` ..... 65, 126–128, 156, 169–171  
`\numexpr` ..... 126, 128, 169

## O

`\O` ..... 405, 408  
`\o` ..... 408  
`\openout` ..... 66  
`\or` ..... 7, 16, 17, 23, 24, 28, 68, 79, 440, 469  
`\org@glossaryentrynumbers` ... 73, 140, 164  
`\org@glossarytitle` ..... 140, 164  
`\org@ifKV@glslink@hyper` ..... 81, 83

## P

`\p@glshyp@opt` ..... 103  
`\p@glstr@assign@leveloffset` ..... 143  
package options:  
    abbreviations ..... 21  
    accsupp ..... 24, 180  
    acronym ..... 21  
    automake ..... 139, 152, 158  
        immediate ..... 158  
        true ..... 158

autoseeindex ..... 30  
    false ..... 30  
counter  
    wrglossary ..... 27  
debug  
    showtargets ..... 105  
docdef ..... 17, 66, 67, 119, 127  
    atom ..... 66  
    false ..... 66, 67  
    restricted ..... 18  
    true ..... 65, 68  
docdefs  
    restricted ..... 67  
equations ..... 9, 81  
indexcounter ..... 381  
nonumberlist ..... 73  
nopostdot ..... 19  
    false ..... 20  
numbers ..... 23  
order  
    letter ..... 158  
postdot ..... 19  
prefix ..... 25  
record ..... 7,  
    14, 16, 66, 76, 135, 155, 232, 234–242, 510  
    alsoindex ..... 8, 10, 14, 16, 158  
    hybrid ..... 10, 14, 16  
    nameref ..... 12, 31, 32, 156, 159  
    only ..... 8, 153  
shortcuts ..... 23  
    ac ..... 23  
    all ..... 23  
    false ..... 23  
    none ..... 23  
    true ..... 23  
sort  
    use ..... 84  
style ..... 26  
stylemods ..... 26  
symbols ..... 23, 202  
undefaction ..... 55  
    error ..... 6  
    warn ..... 6  
    xindy ..... 61, 62  
`\PackageError` ..... 6, 13, 26, 30, 66,  
    67, 71, 79, 98, 99, 101, 103, 111, 118–  
    120, 127, 129, 130, 133, 136, 138, 139,  
    147, 159, 166, 168, 171, 221, 246–248, 426  
`\PackageWarning` ..... 19





<code>\rGls</code> .....	173	<code>\settowidth</code> .....	134, 446–456, 475, 498
<code>\rgls</code> .....	173	<code>\setupglossaries</code> .....	5, 30
<code>\rGLSformat</code> .....	177	<code>\sfcode</code> .....	20, 221, 222, 438
<code>\rGlsformat</code> .....	176	<code>\small</code> .....	29, 59
<code>\rglsformat</code> .....	175, 178	<code>\smallskip</code> .....	498
<code>\rGLSpl</code> .....	173	soul package .....	116
<code>\rGlspl</code> .....	173	<code>\space</code> 6, 13, 15, 61, 67, 69, 71, 92, 101, 118–	
<code>\rglspl</code> .....	173	121, 127, 129–131, 133–136, 138, 140,	
<code>\rGLSplformat</code> .....	177	151, 154, 155, 159, 164, 166, 168, 171,	
<code>\rGlsplformat</code> .....	176	222, 226, 230, 250, 426, 428, 438, 441,	
<code>\rglsplformat</code> .....	176, 178	442, 444, 446, 455, 456, 465, 496, 498, 499	
<code>\robustify</code> .....	380	<code>\spacefactor</code> .....	20, 221, 222, 438
<code>\romannumeral</code> .....	446–448, 455, 456, 498	<code>\stepcounter</code> .....	179
<b>S</b>			
<code>\s@glsxtrfmt</code> .....	35	<code>\string</code> 6, 11–13, 15, 32, 61, 62, 67–69, 71, 75,	
<code>\s@gls@hyp@opt</code> .....	103	82, 92, 98, 99, 101, 104, 111, 118–121,	
<code>\s@glsxtr@enabletagging</code> .....	218	127, 129–131, 133–140, 151–155, 157–	
<code>\s@glsxtrfmt</code> .....	35	159, 164, 166, 168, 170, 171, 206–209,	
<code>\s@glsxtrforcsvfield</code> .....	38	211, 213, 221, 380, 381, 390, 395–424, 470	
<code>\s@GlsXtrIfFieldCmpNum</code> .....	41, 42	<code>\strut</code> .....	426, 428–438, 444, 473
<code>\s@GlsXtrIfFieldEqNum</code> .....	41	<code>\sty</code> .....	135
<code>\s@GlsXtrIfFieldEqStr</code> .....	45	<code>\subglossentry</code> .....	140,
<code>\s@GlsXtrIfFieldEqXpStr</code> .....	45	165, 169, 170, 426, 428, 430–437, 440,	
<code>\s@GlsXtrIfFieldNonZero</code> .....	41, 381	442, 444, 456, 468, 477, 479–481, 483,	
<code>\s@GlsXtrIfFieldValueInCsvList</code> .....	39	484, 486, 488, 489, 491, 492, 494, 495, 500	
<code>\s@glsxtrifhasfield</code> .....	38–41, 45, 46	<code>\subitem</code> .....	439, 440
<code>\s@GlsXtrIfHasNonZeroChildCount</code> ...	380	<code>\subsubitem</code> .....	439, 440
<code>\s@GlsXtrIfValueInFieldCsvList</code> .....	39	<code>\symbolname</code> .....	
<code>\s@GlsXtrIfXpFieldEqXpStr</code> .....	46	475, 482, 483, 485, 486, 488, 490, 491, 493	
<code>\s@GlsXtrStartUnsetBuffering</code> .....	116	<b>T</b>	
<code>\s@GlsXtrStopUnsetBuffering</code> .....	116	<code>\tabcolsep</code> .....	475, 476
<code>\s@ifglossaryexists</code> .....	33	<code>\tablehead</code> .....	434–437
<code>\s@printunsrtglossary</code> .....	162, 167	<code>\tabletail</code> .....	434–437
<code>\s@xGlsXtrIfValueInFieldCsvList</code> ....	40	<code>\tabularnewline</code> ....	430–438, 474, 477–494
<code>\seealsoname</code> .....	60, 62	<code>\TeX</code> .....	152
<code>\seename</code> .....	57, 60	<code>\texorpdfstring</code> .....	
<code>\setabbreviationstyle</code> .....	133, 250, 262	.....	36, 43, 112–114, 217, 356, 369–378
<code>\SetAcronymLists</code> .....	132–134	<code>\textbf</code> .....	189, 438, 474, 497, 499
<code>\setacronymstyle</code> .....	132–134	textcase package .....	355
<code>\SetDefaultAcronymDisplayStyle</code> ....	134	<code>\textit</code> .....	189
<code>\setentrycounter</code> .....	148, 383, 385	<code>\textmd</code> .....	189
<code>\SetGenericNewAcronym</code> .....	134	<code>\textrm</code> .....	189
<code>\setglossarystyle</code> .....	26, 140, 164,	<code>\textsc</code> .....	189, 268
	426, 428–430, 440, 443, 445, 457–464, 467	<code>\textsf</code> .....	189
<code>\setkeys</code> .....	9, 26, 31,	<code>\textsl</code> .....	189
	36, 81, 83, 102, 131, 140, 164, 174, 227, 229	<code>\textsmaller</code> .....	285
<code>\setlength</code> .....	72, 73, 439, 442, 444,	<code>\texttt</code> .....	29, 151–154, 189
	446, 457, 459–461, 467, 475, 476, 497, 498	<code>\the</code> .....	131,
			149, 156, 216, 229, 230, 249–260, 262–
			277, 280, 282–291, 293, 297–310, 312,



314, 316, 317, 319, 321–324, 326, 328–336, 338–340, 342, 344, 346–354, 385, 498	\unskip ..... 52, 65, 426
\theglsentrycounter .. 8, 10–12, 82, 84, 174	upgreek package ..... 393
\theH ..... 32	\usepackage ..... 153, 154
\theHglentrycounter . 8, 10–13, 82, 84, 174	
\theindex ..... 212	<b>W</b>
\thewrglossary ..... 27	\warn@nomakeglossaries ..... 137
\this@dialect ..... 378, 379, 424	\warn@noprintglossary .. 136, 137, 141, 165
\toks@ ..... 149, 216, 385	wrglossary (counter) ..... 26, 27
\toprule ..... 474, 477, 479, 480, 482, 483, 485, 486, 488, 490, 491, 493	\write ..... 61, 120, 129, 136, 150, 151, 158
\TrackedDialectClosestSubMatch ..... 47	
tracklang package ..... 47, 157, 394	<b>X</b>
\TrackLangGetDefaultScript ..... 424	\xcapitalisewords ..... 203
\TrackLangIfHasDefaultScript ..... 424	\xdef ..... 381
\TrackLangRequireDialectPrefix .... 424	\xifinlist ..... 125
\triangleright ..... 59	\xifinlistcs ..... 38
	xindy ..... 501
<b>U</b>	xindy ..... 16, 135, 501
\u ..... 380	xkeyval package ..... 5
\undef ..... 16, 17, 66, 218	\XKV@checkchoice ..... 75
\underline ..... 219	\XKV@plfalse ..... 75
	\XKV@resa ..... 75
	\XKV@sttrue ..... 75