

# **glossaries-extra.sty v1.24: documented code**

Nicola L.C. Talbot

Dickimaw Books

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

2017-11-14

## 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	25
1.3	Modifications to Commands Provided by glossaries	33
1.3.1	Existence Checks	37
1.3.2	Document Definitions	45
1.3.3	Existing Glossary Style Modifications	50
1.3.4	Entry Formatting, Hyperlinks and Indexing	54
1.3.5	Entry Counting	89
1.3.6	Acronym Modifications	102
1.3.7	Indexing and Displaying Glossaries	104
1.3.8	Support for bib2gls	132
1.4	Integration with glossaries-accsupp	140
1.5	Categories	151
1.6	Abbreviations	176
1.6.1	Abbreviation Styles Setup	195
1.6.2	Predefined Styles (Default Font)	198
1.6.3	Predefined Styles (Small Capitals)	214
1.6.4	Predefined Styles (Fake Small Capitals)	228
1.6.5	Predefined Styles (Emphasized)	242
1.6.6	Predefined Styles (User Parentheses Hook)	264
1.6.7	Predefined Styles (Hyphen)	273
1.6.8	Predefined Styles (No Short on First Use)	287
1.7	Using Entries in Headings	289
1.8	Multi-Lingual Support	309
<b>2</b>	<b>Style Adjustments (glossaries-extra-stylemods.sty)</b>	<b>310</b>
2.1	Package Initialisation	310
2.2	List-Like Styles	311
2.3	Longtable Styles	314
2.4	Long Ragged Styles	316
2.5	Supertabular Styles	318
2.6	Super Ragged Styles	320
2.7	Inline Style	322
2.8	Tree Styles	322
2.9	Multicolumn Styles	339

<b>3 bookindex style (glossary-bookindex.sty)</b>	<b>345</b>
3.1 Package Initialisation and Options . . . . .	345
<b>Glossary</b>	<b>351</b>
<b>Change History</b>	<b>352</b>
<b>Index</b>	<b>367</b>

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

## 1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/11/14 v1.24 (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{\glxstr@doption}[1]{\setupglossaries{#1}}%
```

```
8 \let\@glxstr@declareoption\@gls@declareoption
```

```
9 }
```

```
10 {%
```

Not already loaded, so pass options to glossaries.

```
11 \newcommand{\glxstr@doption}[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*{\@glxstr@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     \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 }%

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

```

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.

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

`\glstr@recordsee` Does nothing by default.

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

`\ultnumberformat`

```
78 \newcommand*{\@glstr@defaultnumberformat}{\glstrnumberformat}%
```

`\ultNumberFormat`

```

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

```

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.

```

82 \newcommand*{\@glxtr@do@record@wrglossary}[1]{%
83 \beginingroup
84 \ifKV@glslink@noindex
85 \else
86 \edef\@glsl@label{\glstoklabel{#1}}%
87 \let\glslabel\@glsl@label
88 \glswriteentry{#1}%
89 {%
90 \ifdefempty{\@glxtr@thevalue}%
91 {%
92 \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
93 \else
94 \let\theHglentrycounter\@glxtr@theHvalue
95 \fi
96 \glxtr@saveentrycounter
97 \let\@do@wrglossary\@glxtr@dorecord
98 }%
99 {%
100 \let\theHglentrycounter\@glxtr@thevalue
101 \let\theHglentrycounter\@glxtr@theHvalue
102 \let\@do@wrglossary\@glxtr@dorecordnodefer
103 }%
104 \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
105 \glxtr@do@wrglossary{#1}%
106 \else
107 \@glxtrwrglossmark
108 \@do@wrglossary
109 \fi
110 }%
111 \fi
112 \endgroup
113 }

```

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

```

114 \newcommand*{\glxtr@do@alsoindex@wrglossary}[1]{%
115 \glxtr@do@wrglossary{#1}%
116 \glxtr@do@record@wrglossary{#1}%
117 }

```

`@glxtr@record` The `record=only` option sets `\@glxtr@record` to this. This performs the recording if the entry doesn't exist and is done at the start of `\@glsl@field@link` and commands like `\@glsl@` (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 `\glsl`). 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 `\glsladd`).

```

118 \newcommand*{\@glxtr@record}[3]{%
119 \ifglentryexists{#2}{}%
120 {%
121 \@glxtrwrglossmark

```



122     \beginngroup

Save the label in case it's needed.

```
123     \edef\@gls@label{\glsdetoklabel{#2}}%
124     \let\glslabel\@gls@label
125     \let\@glsnumberformat\@glsxtr@defaultnumberformat
126     \def\@glsxtr@thevalue{%
127     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
128     \let\@glsxtr@org@theHvalue\@glsxtr@theHvalue
```

Entry hasn't been defined, so we'll have to assume the page number by default.

```
129     \def\@gls@counter{page}%
```

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

```
130     \@gls@setdefault@glslink@opts
131     \setkeys{#3}{#1}%
132     \ifKV@glslink@noindex
133     \else
134     \glswriteentry{#2}%
135     {%
```

Check if thevalue has been set.

```
136     \ifdefempty{\@glsxtr@thevalue}%
137     {%
```

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

```
138     \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
139     \else
140     \let\theHglentrycounter\@glsxtr@theHvalue
141     \fi
```

Save the entry counter.

```
142     \glsxtr@saveentrycounter
```

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

```
143     \let\@@do@@wrglossary\@glsxtr@dorecord
144     }%
145     {%
```

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.)

```
146     \let\theHglentrycounter\@glsxtr@thevalue
147     \let\theHglentrycounter\@glsxtr@theHvalue
148     \let\@@do@@wrglossary\@glsxtr@dorecordnodefer
149     }%
150     \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
151     \glsxtr@@do@@wrglossary{#2}%
152     \else
```

No need to escape special characters.

```
153     \@@do@@wrglossary
154     \fi
```

```

155     }%
156   \fi
157 \endgroup
158 }%
159 }

```

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

```

160 \newcommand*\@glsxtr@dorecord{%
161   \global\let\@glsrecordlocref\theglsentrycounter
162   \let\@glsxtr@orgprefix\@glo@counterprefix
163   \ifx\theglsentrycounter\theHglentrycounter
164     \def\@glo@counterprefix{}%
165   \else
166     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
167       {\theglsentrycounter}{\theHglentrycounter}}%
168     }%
169     \@do@gls@getcounterprefix
170   \fi
171   \protected@write\@auxout{\let\@glsrecordlocref\relax}{\string\glsxtr@record
172     {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}}%
173     {\@glsrecordlocref}}%
174   \@glsxtr@counterrecordhook
175   \let\@glo@counterprefix\@glsxtr@orgprefix
176 }

```

`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.

```

177 \newcommand*\@glsxtr@dorecordnodefer{%
178   \ifx\theglsentrycounter\theHglentrycounter
179     \protected@write\@auxout{{\string\glsxtr@record
180       {\@gls@label}{\@gls@counter}{\@glsnumberformat}}%
181       {\theglsentrycounter}}%
182   \else
183     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
184       {\theglsentrycounter}{\theHglentrycounter}}%
185     }%
186     \@do@gls@getcounterprefix
187     \protected@write\@auxout{{\string\glsxtr@record
188       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}}%
189       {\theglsentrycounter}}%
190   \fi
191   \@glsxtr@counterrecordhook
192 }

```

`r@recordcounter`

```

193 \newcommand*\@@glsxtr@recordcounter{%
194   \@glsxtr@noop@recordcounter

```

```

195 }

p@recordcounter
196 \newcommand*{\@glxtr@noop@recordcounter}[1]{%
197   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
198     requires record=only or record=alsoindex package option}{}%
199 }

p@recordcounter
200 \newcommand*{\@glxtr@op@recordcounter}[1]{%
201   \eappto\@glxtr@counterrecordhook{\noexpand\@glxtr@docounterrecord{#1}}%
202 }

lsxtr@recordsee Deal with \glssee in record mode.
203 \newcommand*{\@glxtr@recordsee}[2]{%
204   \@@glxtrwrglossmark
205   \def\@gls@xref{#2}%
206   \@onelevel@sanitize\@gls@xref
207   \protected@write\@auxout{}\{\string\glxtr@recordsee{#1}-\@gls@xref}\}%
208 }

srtglossaryunit
209 \newcommand{\printunsrtglossaryunit}{%
210   \print@noop@unsrtglossaryunit
211 }

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

aveentrycounter Only store the entry counter information if the indexing is on.
213 \newcommand*{\glxtr@indexonly@saveentrycounter}{%
214   \ifKV@glslink@noindex
215   \else
216     \glxtr@saveentrycounter
217   \fi
218 }

addloclistfield
219 \newcommand*{\glxtr@addloclistfield}{%
220   \key@ifundefined{glossentry}{loclist}%
221   {%
222     \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
223     \appto\@gls@keymap{,{loclist}{loclist}}%
224     \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
225     \appto\@newglossaryentryposthook{%
226       \gls@assign@field{\@glo@label}{loclist}{\@glo@loclist}%
227     }%
228     \glssetnoexpandfield{loclist}%
229   }%
230 }%

```

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

```

231 \key@ifundefined{glossentry}{location}%
232 {%
233   \define@key{glossentry}{location}{\def\@glo@location{##1}}%
234   \appto\@gls@keymap{,{location}{location}}%
235   \appto\@newglossaryentryprehook{\def\@glo@location{}}%
236   \appto\@newglossaryentryposthook{%
237     \gls@assign@field{\@glo@label}{location}{\@glo@location}%
238   }%
239   \glssetnoexpandfield{location}%
240 }%
241 {}%
```

Add a key to store the group heading.

```

242 \key@ifundefined{glossentry}{group}%
243 {%
244   \define@key{glossentry}{group}{\def\@glo@group{##1}}%
245   \appto\@gls@keymap{,{group}{group}}%
246   \appto\@newglossaryentryprehook{\def\@glo@group{}}%
247   \appto\@newglossaryentryposthook{%
248     \gls@assign@field{\@glo@label}{group}{\@glo@group}%
249   }%
250   \glssetnoexpandfield{group}%
251 }%
252 {}%
253 }
```

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

```

254 \newcommand*{\@glsxtr@record@setting}{off}
```

alsoindex

```

255 \newcommand*{\@glsxtr@record@setting@alsoindex}{alsoindex}
```

only

```

256 \newcommand*{\@glsxtr@record@setting@only}{only}
```

off

```

257 \newcommand*{\@glsxtr@record@setting@off}{off}
```

Now define the record package option.

```

258 \define@choicekey{glossaries-extra.sty}{record}[\val\nr]%
259 {off,only,alsoindex}%
260 [only]%
261 {%
262   \let\@glsxtr@record@setting\val
263   \ifcase\nr\relax
```

Don't record.

```

264   \def\@glsxtr@setup@record{%
```

```

265     \renewcommand*{\@do@seeglossary}{\@glxtr@doseeglossary}%
266     \renewcommand*{\@glxtr@record}[3]{}%
267     \let\@do@wrglossary\glxtr@do@wrglossary
268     \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
269     \let\glxtrundefaction\glxtr@err@undefaction
270     \let\glxtr@warnonexistsordo\@gobble
271     \let\@glxtr@recordcounter\glxtr@noop@recordcounter
272     \def\printunsrtglossaryunit{\print@noop@unsrtglossaryunit}%
273     \undef\glxtrsetaliasnoindex
274 }%
275 \or

```

Only record (don't index).

```

276     \def\glxtr@setup@record{%
277         \@glxtr@autoseeindexfalse
278         \let\@do@seeglossary\glxtr@recordsee
279         \let\@glxtr@record\@glxtr@record
280         \let\@do@wrglossary\glxtr@do@record@wrglossary
281         \let\@glxtr@saveentrycounter\relax
282         \let\glxtrundefaction\glxtr@warn@undefaction
283         \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
284         \glxtr@addloclistfield
285         \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
286         \let\@glxtr@recordcounter\glxtr@op@recordcounter
287         \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%

```

Switch off the index suppression for aliased entries. (bib2gls will deal with them.)

```

288     \def\glxtrsetaliasnoindex{}%

```

\@glxtr@setupsort@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.

```

289     \ifdefined\@glxtr@setupsort@none{\@glxtr@setupsort@none}{}%
290 }%
291 \or

```

Record and index.

```

292     \def\glxtr@setup@record{%
293         \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
294         \let\@glxtr@record\@glxtr@record
295         \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary
296         \let\@glxtr@saveentrycounter\glxtr@indexonly@saveentrycounter
297         \let\glxtrundefaction\glxtr@warn@undefaction
298         \let\glxtr@warnonexistsordo\glxtr@warn@onexistsordo
299         \glxtr@addloclistfield
300         \let\@glxtr@recordcounter\glxtr@op@recordcounter
301         \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
302         \undef\glxtrsetaliasnoindex
303     }%
304 \fi
305 }

```

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.

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

Need to provide conditional commands that are backward compatible:

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

lsxtrdocdeftrue
308 \newcommand*{\@glxtrdocdeftrue}{\@glxtr@docdefval=1 }

sxtrdocdeffalse
309 \newcommand*{\@glxtrdocdeffalse}{\@glxtr@docdefval=0 }
```

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.

```
310 \define@choicekey{glossaries-extra.sty}{docdef}[\val\nr]%
311 {false,true,restricted}[true]%
312 {%
313   \@glxtr@docdefval=\nr\relax
314   \ifnum\@glxtr@docdefval=2\relax
315     \renewcommand*{\@glsdofexistsorwarn}{\glsdofexists}%
316   \fi
317 }
```

```
ocdefrestricted
318 \newcommand*{\if@glxtrdocdefrestricted}{\ifnum\@glxtr@docdefval=2 }
```

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).  
319 \newcommand\*{\@glsdofexistsorwarn}{\glsdofexistsorwarn}

```
indexcrossrefs Automatically index cross references at the end of the document
320 \define@boolkey{glossaries-extra.sty}{@glxtr}{indexcrossrefs}[true]{%
321   \if@glxtrindexcrossrefs
322   \else
323     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
324   \fi
325 }
```

Switch off since this can increase the build time.

```
326 \@glxtrindexcrossrefsfalse
```

But allow see key to switch it on automatically.

oindexcrossrefs

```
327 \newcommand*{\@glxtr@autoindexcrossrefs}{\@glxtr@indexcrossrefstrue}
```

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

```
328 \define@boolkey{glossaries-extra.sty}{@glxtr@}{autoseeindex}[true]{%
329 }
330 \@glxtr@autoseeindextrue
```

iesExtraWarning Allow users to suppress warnings.

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

raWarningNoLine Allow users to suppress warnings.

```
332 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
333 \PackageWarningNoLine{glossaries-extra}{#1}}

334 \@glxtr@declareoption{nowarn}{%
335 \let\GlossariesExtraWarning\@gobble
336 \let\GlossariesExtraWarningNoLine\@gobble
337 \glxtr@doption{nowarn}%
338 }
```

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

```
339 \newcommand*{\@glxtr@defpostpunc}{}
```

postdot Shortcut for nopostdot=false

```
340 \@glxtr@declareoption{postdot}{%
341 \glxtr@doption{nopostdot=false}%
342 \renewcommand*{\@glxtr@defpostpunc}{%
343 \renewcommand*{\glspostdescription}{%
344 \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
345 }%
346 }
```

nopostdot Needs to redefine \@glxtr@defpostpunc

```
347 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
348 \glxtr@doption{nopostdot=#1}%
349 \renewcommand*{\@glxtr@defpostpunc}{%
350 \renewcommand*{\glspostdescription}{%
351 \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
352 }%
353 }
354 %
355 %\begin{option}{postpunc}
356 %Set the post-description punctuation. This also sets
```

```

357%the \cs@ifglsnopostdot} conditional, which now indicates if
358%the post-description punctuation has been suppressed.
359%\changes{1.21}{2017-11-03}{new}
360% \begin{macrocode}
361\define@key{glossaries-extra.sty}{postpunc}{%
362 \glstr@dooption{nopostdot=false}%
363 \ifstrequal{#1}{dot}%
364 {%
365 \renewcommand*\@glstr@defpostpunc}{%
366 \renewcommand*\glspostdescription{.\spacefactor\sfcode'\. }%
367 }%
368 }%
369 {%
370 \ifstrequal{#1}{comma}%
371 {%
372 \renewcommand*\@glstr@defpostpunc}{%
373 \renewcommand*\glspostdescription{,}%
374 }%
375 }%
376 {%
377 \ifstrequal{#1}{none}%
378 {%
379 \glstr@dooption{nopostdot=true}%
380 \renewcommand*\@glstr@defpostpunc}{%
381 \renewcommand*\glspostdescription{}%
382 }%
383 }%
384 {%
385 \renewcommand*\@glstr@defpostpunc}{%
386 \renewcommand*\glspostdescription{#1}%
387 }%
388 }%
389 }%
390 }%
391}

```

`glstrabbrvtype` Glossary type for abbreviations.

```

392\newcommand*\glstrabbrvtype{\glstrdefaulttype}

```

`abbreviationsdef` Set by abbreviations option.

```

393\newcommand*\@glstr@abbreviationsdef{}

```

`abbreviationsdef`

```

394\newcommand*\@glstr@doabbreviationsdef{%
395 \@ifpackageloaded{babel}%
396 {\providecommand{\abbreviationsname}{\acronymname}}%
397 {\providecommand{\abbreviationsname}{Abbreviations}}%
398 \newglossary[glg-abr]{abbreviations}{glg-abr}{glo-abr}{\abbreviationsname}%
399 \renewcommand*\glstrabbrvtype{abbreviations}%

```



```

400 \newcommand*\printabbreviations}[1][\]{%
401   \printglossary[type=\glstrabbrvtype,##1]%
402 }%
403 \disable@keys{glossaries-extra.sty}{abbreviations}%

If the acronym option hasn't been used, change \acronymtype to \glstrabbrvtype.
404 \ifglstracronym
405 \else
406   \renewcommand*\acronymtype{\glstrabbrvtype}%
407 \fi
408 }%

```

**abbreviations** If abbreviations, create a new glossary type for abbreviations.

```

409 \@glstr@declareoption{abbreviations}{%
410   \let\@glstr@abbreviationsdef\@glstr@doabbreviationsdef
411 }

```

**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).

```

412 \newcommand*\GlsXtrDefineAbbreviationShortcuts{%
413   \newcommand*\ab{\cgl}%
414   \newcommand*\abp{\cglsp}%
415   \newcommand*\as{\glstrshort}%
416   \newcommand*\asp{\glstrshortpl}%
417   \newcommand*\al{\glstrlong}%
418   \newcommand*\alp{\glstrlongpl}%
419   \newcommand*\af{\glstrfull}%
420   \newcommand*\afp{\glstrfullpl}%
421   \newcommand*\Ab{\cGls}%
422   \newcommand*\Abp{\cGlspl}%
423   \newcommand*\As{\Glsxtrshort}%
424   \newcommand*\Asp{\Glsxtrshortpl}%
425   \newcommand*\Al{\Glsxtrlong}%
426   \newcommand*\Alp{\Glsxtrlongpl}%
427   \newcommand*\Af{\Glsxtrfull}%
428   \newcommand*\Afp{\Glsxtrfullpl}%
429   \newcommand*\AB{\cGLS}%
430   \newcommand*\ABP{\cGLSpl}%
431   \newcommand*\AS{\GLSxtrshort}%
432   \newcommand*\ASP{\GLSxtrshortpl}%
433   \newcommand*\AL{\GLSxtrlong}%
434   \newcommand*\ALP{\GLSxtrlongpl}%
435   \newcommand*\AF{\GLSxtrfull}%
436   \newcommand*\AFP{\GLSxtrfullpl}%

```

```

437 \providecommand*\newabbr{\newabbreviation}%

```

Disable this command after it's been used.

```

438 \let\GlsXtrDefineAbbreviationShortcuts\relax

```

439 }

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

```
440 \newcommand*{\GlsXtrDefineAcShortcuts}{%
441   \newcommand*{\ac}{\cgl{s}}%
442   \newcommand*{\acp}{\cgl{s}pl}%
443   \newcommand*{\acs}{\gl{s}xtrshort}%
444   \newcommand*{\acsp}{\gl{s}xtrshortpl}%
445   \newcommand*{\acl}{\gl{s}xtrlong}%
446   \newcommand*{\aclp}{\gl{s}xtrlongpl}%
447   \newcommand*{\acf}{\gl{s}xtrfull}%
448   \newcommand*{\acfp}{\gl{s}xtrfullpl}%
449   \newcommand*{\Ac}{\cGls}%
450   \newcommand*{\Acp}{\cGlspl}%
451   \newcommand*{\Acs}{\Glsxtrshort}%
452   \newcommand*{\Acsp}{\Glsxtrshortpl}%
453   \newcommand*{\Acl}{\Glsxtrlong}%
454   \newcommand*{\Aclp}{\Glsxtrlongpl}%
455   \newcommand*{\Acf}{\Glsxtrfull}%
456   \newcommand*{\Acfp}{\Glsxtrfullpl}%
457   \newcommand*{\AC}{\cGLS}%
458   \newcommand*{\ACP}{\cGLSpl}%
459   \newcommand*{\ACS}{\GLSxtrshort}%
460   \newcommand*{\ACSP}{\GLSxtrshortpl}%
461   \newcommand*{\ACL}{\GLSxtrlong}%
462   \newcommand*{\ACLP}{\GLSxtrlongpl}%
463   \newcommand*{\ACF}{\GLSxtrfull}%
464   \newcommand*{\ACFP}{\GLSxtrfullpl}%

465   \providecommand*{\newabbr}{\newabbreviation}%
```

Disable this command after it's been used.

```
466   \let\GlsXtrDefineAcShortcuts\relax
467 }
```

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

```
468 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
469   \newcommand*{\newentry}{\newglossaryentry}%
470   \ifdef\printsymbols
471     {%
472       \newcommand*{\newsym}{\gl{s}xtrnewsymbol}%
473     }{}%
474   \ifdef\printnumbers
475     {%
476       \newcommand*{\newnum}{\gl{s}xtrnewnumber}%
477     }{}%
478   \let\GlsXtrDefineOtherShortcuts\relax
479 }
```

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.

```
480 \newcommand*{\@glxtr@setupshortcuts}{}
```

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

```
481 \newcommand*{\@glxtr@shortcutsval}{\ifglacrshortcuts acro\else none\fi}%
```

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).

```
482 \define@choicekey{glossaries-extra.sty}{shortcuts}[\val\nr]%
483 {acronyms,acro,abbreviations,abbr,other,all,true,ac,none,false}[true]{%
484   \let\@glxtr@shortcutsval\val
485   \ifcase\nr\relax % acronyms
486     \renewcommand*{\@glxtr@setupshortcuts}{%
487       \glacrshortcutstrue
488       \DefineAcronymSynonyms
489     }%
490   \or % acro
491     \renewcommand*{\@glxtr@setupshortcuts}{%
492       \glacrshortcutstrue
493       \DefineAcronymSynonyms
494     }%
495   \or % abbreviations
496     \renewcommand*{\@glxtr@setupshortcuts}{%
497       \GlsXtrDefineAbbreviationShortcuts
498     }%
499   \or % abbr
500     \renewcommand*{\@glxtr@setupshortcuts}{%
501       \GlsXtrDefineAbbreviationShortcuts
502     }%
503   \or % other
504     \renewcommand*{\@glxtr@setupshortcuts}{%
505       \GlsXtrDefineOtherShortcuts
506     }%
507   \or % all
508     \renewcommand*{\@glxtr@setupshortcuts}{%
509       \glacrshortcutstrue
510       \GlsXtrDefineAcShortcuts
511       \GlsXtrDefineAbbreviationShortcuts
512       \GlsXtrDefineOtherShortcuts
513     }%
514   \or % true
515     \renewcommand*{\@glxtr@setupshortcuts}{%
```

```

516      \glsacrshortcutstrue

517      \GlsXtrDefineAcShortcuts
518      \GlsXtrDefineAbbreviationShortcuts
519      \GlsXtrDefineOtherShortcuts
520  }%

521  \or % ac
522      \renewcommand*{\@glsxtr@setupshortcuts}{%
523          \glsacrshortcutstrue
524          \GlsXtrDefineAcShortcuts
525      }%

    Leave none and false as last option.

526      \else % none, false
527          \renewcommand*{\@glsxtr@setupshortcuts}{}%
528      \fi
529  }

\lsxtr@doaccsupp
530 \newcommand*{\@glsxtr@doaccsupp}{}

accsupp    If accsupp, load glossaries-accsupp package.
531 \@glsxtr@declareoption{accsupp}{%
532 \renewcommand*{\@glsxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

GlossaryWarning    Warning text displayed in document if the external glossary file given by the argument is miss-
ing.
533 \newcommand{\glsxtrNoGlossaryWarning}[1]{%
534 \@glsxtr@defaultnoglossarywarning{#1}%
535 }

nomissingglsstext    If true, suppress the text produced if the external glossary file is missing.
536 \define@choicekey{glossaries-extra.sty}{nomissingglsstext}[\val\nr]%
537 {true,false}[true]{%
538     \ifcase\nr\relax % true
539         \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
540             \null
541         }%
542     \else % false
543         \renewcommand{\glsxtrNoGlossaryWarning}[1]{%
544             \@glsxtr@defaultnoglossarywarning{#1}%
545         }%
546     \fi
547 }

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

xtr@redefstyles
548 \newcommand*{\@glsxtr@redefstyles}{}

```

stylemods

```
549 \define@key{glossaries-extra.sty}{stylemods}[default]{%
550   \ifstrequal{#1}{default}%
551   {%
552     \renewcommand*{\@glxtr@redefstyles}{%
553       \RequirePackage{glossaries-extra-stylemods}}%
554   }%
555   {%
556     \ifstrequal{#1}{all}%
557     {%
558       \renewcommand*{\@glxtr@redefstyles}{%
559         \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
560         \RequirePackage{glossaries-extra-stylemods}%
561       }%
562     }%
563     {%
564       \renewcommand*{\@glxtr@redefstyles}{}%
565       \@for\@glxtr@tmp:=#1\do{%
566         \IfFileExists{glossary-\@glxtr@tmp.sty}%
567         {%
568           \eappto\@glxtr@redefstyles{%
569             \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
570         }%
571         {%
572           \PackageError{glossaries-extra}%
573             {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
574               doesn’t exist (did you mean to use the ‘style’ key?)}%
575             {The list of values (#1) in the ‘stylemods’ key should
576               match the glossary-xxx.sty files provided with
577               glossaries.sty}%
578         }%
579       }%
580       \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
581     }
582   }%
583 }
```

glxtr@do@style

```
584 \newcommand*{\@glxtr@do@style}{}
```

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

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

Defer actual style change:

```
586 \renewcommand*{\@glxtr@do@style}{%
```

Set this as the default style:

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

Set this style:

```
588 \setglossarystyle{#1}%  
589 }%  
590 }
```

sxtrwrglossmark Marks the place where indexing occurs. Does nothing by default.

```
591 \newcommand*{\@glxtrwrglossmark}{}
```

sxtrwrglossmark Since \glsadd can be used in the preamble, this action needs to be disabled until the start of the document.

```
592 \newcommand*{\@glxtrwrglossmark}{}  
593 \AtBeginDocument{\renewcommand*{\@glxtrwrglossmark}{\@glxtrwrglossmark}}
```

sxtrwrglossmark Does nothing by default.

```
594 \newcommand*{\glxtrwrglossmark}{\ensuremath{\cdot}}
```

debug Provide extra debug options.

```
595 \define@choicekey{glossaries-extra.sty}{debug}[\val\nr]%  
596 {true,false,showtargets,showwrgloss,all}[true]{%  
597 \ifcase\nr\relax % true  
598 \glxtr@doption{debug=true}%  
599 \renewcommand*{\@glxtrwrglossmark}{}%  
600 \or % false  
601 \glxtr@doption{debug=false}%  
602 \renewcommand*{\@glxtrwrglossmark}{}%  
603 \or % showtargets  
604 \glxtr@doption{debug=showtargets}%  
605 \or % showwrgloss  
606 \glxtr@doption{debug=true}%  
607 \renewcommand*{\@glxtrwrglossmark}{\glxtrwrglossmark}%  
608 \or % all  
609 \glxtr@doption{debug=showtargets}%  
610 \renewcommand*{\@glxtrwrglossmark}{\glxtrwrglossmark}%  
611 \fi  
612 }
```

Pass all other options to glossaries.

```
613 \DeclareOptionX*{%  
614 \expandafter\glxtr@doption\expandafter{\CurrentOption}}
```

Process options.

```
615 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
616 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
617 \@glxtr@doaccsupp
```

Redefine \glspostdescription if required.

618 \let\@glstr@defpostpunc

\glsshowtarget This command was introduced to glossaries v4.32 so it may not be defined. Therefore it's defined here using \def.

```
619 \def\glsshowtarget#1{%
620   \glstrtitleorpdforheading
621   {%
622     \ifmmode
623       \texttt{\small [#1]}%
624     \else
625       \ifinner
626         \texttt{\small [#1]}%
627       \else
628         \marginpar{\texttt{\small #1}}%
629       \fi
630     \fi
631   }%
632   {[#1]}%
633   {\texttt{\small [#1]}}%
634 }
```

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

635 \let\@glstr@org@doseeglossary\@do@seeglossary

r@doseeglossary

```
636 \newcommand*{\@glstr@doseeglossary}[2]{%
637   \glsdofexists{#1}%
638   {%
639     \@glstrwrglossmark
640     \@glstr@org@doseeglossary{#1}{#2}%
641   }%
642 }
```

oindex@glossary

```
643 \newcommand*{\@glstr@dosee@alsoindex@glossary}[2]{%
644   \@glstr@recordsee{#1}{#2}%
645   \@glstr@doseeglossary{#1}{#2}%
646 }
```

@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.)

647 \let\@glstr@org@gloautosee\@glo@autosee

Check if user tried autoseeindex=false when it can't be supported.

```
648 \if@glstr@autoseeindex
649 \else
```

```

650 \ifdef\@glxtr@org@gloautosee
651 {}%
652 {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
653 option requires at least v4.30 of glossaries.sty}%
654 {You need to update the glossaries.sty package}%
655 }
656 \fi

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the au-
toseeindex option.
657 \ifdef\@glo@autosee
658 {%
659 \renewcommand*{\@glo@autosee}{%
660 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
661 }%
662 {}

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.
663 \renewcommand*{\gls@checkseeallowed}{%
664 \if@glxtr@autoseeindex\@gls@see@noindex\fi
665 }

Define abbreviations glossaries if required.
666 \@glxtr@abbreviationsdef
667 \let\@glxtr@abbreviationsdef\relax

Setup shortcuts if required.
668 \@glxtr@setupshortcuts

Redefine \@glxtr@redef@for@gl@sentries if required.
669 \@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:
670 \renewcommand{\glxtr@doooption}[1]{\setupglossaries{#1}}%

Now define the user command:
671 \newcommand*{\glossariesextrasetup}[1]{%
672 \let\glxtr@setup@record\relax
673 \let\@glxtr@setupshortcuts\relax
674 \let\@glxtr@redef@for@gl@sentries\relax
675 \setkeys{glossaries-extra.sty}{#1}%
676 \@glxtr@abbreviationsdef
677 \let\@glxtr@abbreviationsdef\relax
678 \@glxtr@setupshortcuts
679 \glxtr@setup@record
680 \@glxtr@redef@for@gl@sentries
681 }

```



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

```
682 \let\glstr@org@@do@wrglossary\@@do@wrglossary
```

`@@do@wrglossary` The new version adds code that can show a marker for debugging.

```
683 \newcommand*{\glstr@@do@wrglossary}[1]{%
684   \@@glstrwrglossmark
685   \glstr@org@@do@wrglossary{#1}%
686 }
```

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

```
687 \let\glstr@saveentrycounter\@gls@saveentrycounter
```

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

```
688 \let\@gls@saveentrycounter\glstr@indexonly@saveentrycounter
```

Set up record option if required.

```
689 \glstr@setup@record
```

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

```
690 \AtBeginDocument{%
691   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
692   \def\@glstrundeftag{\glstrundeftag}%
693 }
```

## 1.2 Extra Utilities

`trifemptyglossary` `\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.)

```
694 \newcommand{\glstrifemptyglossary}[3]{%
695   \ifcsdef{glolist@#1}%
696   {%
697     \ifcsstring{glolist@#1}{,}{#2}{#3}%
698   }%
699   {%
700     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
701     #2%
702   }%
703 }
```

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

```
704 \newcommand*{\glxtrifkeydefined}[3]{%
705   \key@ifundefined{glossentry}{#1}{#3}{#2}%
706 }
```

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

```
707 \newcommand*{\glxtrprovidestoragekey}{%
708   \@ifstar\sglsxtr@provide@storagekey\glxtr@provide@storagekey
709 }
```

vide@storagekey Unstarred version.

```
710 \newcommand*{\@glxtr@provide@storagekey}[3]{%
711   \key@ifundefined{glossentry}{#1}%
712   {%
713     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
714     \appto\@gls@keymap{,{#1}{#1}}%
715     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
716     \appto\@newglossaryentryposthook{%
717       \letcs{\@glo@tmp}{@glo@#1}%
718       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
719     }%

```

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

```
720   \ifblank{#3}
721   {}%
722   {%
723     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
724   }%
725   }%
726   {%

```

Provide the no-link command if not already defined.

```
727   \ifblank{#3}
728   {}%
729   {%
730     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
731   }%
732   }%
733 }
```

vide@storagekey Starred version.

```
734 \newcommand*{\sglsxtr@provide@storagekey}[1]{%
735   \key@ifundefined{glossentry}{#1}%
736   {%
737     \expandafter\newcommand\expandafter*\expandafter
738     {\csname gls@assign@#1@field\endcsname}[2]{%
739       \@gls@expand@field{##1}{#1}{##2}%
740     }%

```

```

741 }%
742 {}%
743 \@glstr@provide@addstoragekey{#1}%
744 }

```

The name of a text-block control sequence can be stored in a field (given by `\GlsXtrFmtField`). This command can then be used with `\glstrfmt[⟨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`

```

745 \newcommand{\GlsXtrFmtField}{useri}

```

`\GlsXtrFmtDefaultOptions`

```

746 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

747 \newrobustcmd*{\glstrfmt}{\@ifstar\s@glstrfmt\@glstrfmt}

```

`\@glstrfmt` Unstarred form.

```

748 \newcommand*{\@glstrfmt}[3][\@glstrfmt{#1}{#2}{#3}]{}

```

`\s@glstrfmt` Starred form.

```

749 \newcommand*{\s@glstrfmt}[3][\@glstrfmt{#1}{#2}{#3}]{}
750 \new@ifnextchar[\s@glstrfmt{#1}{#2}{#3}]{}
751 {\@glstrfmt{#1}{#2}{#3}]{}
752 }

```

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

```

753 \def\s@glstrfmt#1#2#3[#4]{\@glstrfmt{#1}{#2}{#3}{#4}}

```

`\@glstrfmt` Actual inner working.

```

754 \newcommand*{\@glstrfmt}[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).

```

755 \begingroup
756 \def\glslabel{#2}%
757 \glsdoifexistsordo{#2}%
758 {%
759 \ifglshasfield{\GlsXtrFmtField}{#2}%
760 {%
761 \let\do@glsl@link@checkfirsthyper\relax
762 \expandafter\@glsl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
763 {\glstrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
764 }%
765 {\glstrfmtdisplay{@firstofone}{#3}{#4}}%
766 }%
767 {%

```

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

```

768 \begingroup
769 \@gls@setdefault@glslink@opts
770 \setkeys{glslink}{\GlsXtrFmtDefaultOptions,#1}%
771 \ifKV@glslink@noindex\else\glsadd{#2}\fi
772 \endgroup
773 \glsxtrfmtdisplay{@firstofone}{#3}{#4}%
774 }%
775 \endgroup
776 }

```

`\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).

```

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

```

`\glsxtrentryfmt` No link or indexing.

```

778 \ifdef\texorpdfstring
779 {
780 \newcommand*{\glsxtrentryfmt}[2]{%
781 \texorpdfstring{\@glsxtrentryfmt{#1}{#2}}{#2}%
782 }
783 }
784 {
785 \newcommand*{\glsxtrentryfmt}{\@glsxtrentryfmt}
786 }

```

`@glsxtrentryfmt`

```

787 \newrobustcmd*{\@glsxtrentryfmt}[2]{%
788 \glsdoifexistsordo{#1}%
789 {%
790 \ifglshasfield{\GlsXtrFmtField}{#1}%
791 {%
792 \csuse{\glscurrentfieldvalue}{#2}%
793 }%
794 {#2}%
795 }%
796 {#2}%
797 }

```

`\glsxtrfieldlistadd` 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.

```

798 \newcommand*{\glsxtrfieldlistadd}[3]{%
799 \listcsadd{glo@\glsdetoklabel{#1}@\#2}{#3}%
800 }

```

trfieldlistgadd Similarly but uses \listcsgadd.

```

801 \newcommand*\glxtrfieldlistgadd[3]{%
802   \listcsgadd{glo@\glstetoklabel{#1}@\#2}{\#3}%
803 }
```

trfieldlistead Similarly but uses \listcseadd.

```

804 \newcommand*\glxtrfieldlistead[3]{%
805   \listcseadd{glo@\glstetoklabel{#1}@\#2}{\#3}%
806 }
```

trfieldlistxadd Similarly but uses \listcsxadd.

```

807 \newcommand*\glxtrfieldlistxadd[3]{%
808   \listcsxadd{glo@\glstetoklabel{#1}@\#2}{\#3}%
809 }
```

Now provide commands to iterate over these lists.

fielddolistloop

```

810 \newcommand*\glxtrfielddolistloop[2]{%
811   \dolistcsloop{glo@\glstetoklabel{#1}@\#2}%
812 }
```

fieldforlistloop

```

813 \newcommand*\glxtrfieldforlistloop[3]{%
814   \forlistcsloop{glo@\glstetoklabel{#1}@\#2}{\#3}%
815 }
```

List element tests:

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

```

816 \newcommand*\glxtrfieldifinlist[5]{%
817   \ifinlistcs{\#3}{glo@\glstetoklabel{#1}@\#2}{\#4}{\#5}%
818 }
```

trfieldxifinlist Expands item.

```

819 \newcommand*\glxtrfieldxifinlist[5]{%
820   \xifinlistcs{\#3}{glo@\glstetoklabel{#1}@\#2}{\#4}{\#5}%
821 }
```

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

```

822 \newcommand*\glxtrforcsvfield[3]{%
823   \@glxtrifhasfield{\#2}{\#1}%
824   {%
825     \let\glxtrendfor\@endfortrue
```

```

826 \for\@glstr@label:=\glscurrentfieldvalue\do
827 {\expandafter#3\expandafter{\@glstr@label}}}%
828 {}%
829 }

```

`\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.

```

830 \newrobustcmd{\glxtrifhasfield}{%
831 \ifstar{\s@glxtrifhasfield}{\@glxtrifhasfield}%
832 }

```

`\lsxtrifhasfield` Unstarred version adds grouping.

```

833 \newcommand{\@glxtrifhasfield}[4]{%
834 {\s@glxtrifhasfield{#1}{#2}{#3}{#4}}%
835 }

```

`\lsxtrifhasfield` Starred version omits grouping.

```

836 \newcommand{\s@glxtrifhasfield}[4]{%
837 \letcs{\glscurrentfieldvalue}{glo@\glsetoklabel{#2}@#1}%
838 \ifundef\glscurrentfieldvalue
839 {#4}%
840 {%
841 \ifdefempty\glscurrentfieldvalue{#4}{#3}%
842 }%
843 }

```

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

Just uses `\ifcsundef`.

```

844 \newcommand{\GlsXtrIfFieldUndef}[2]{%
845 \ifcsundef{glo@\glsetoklabel{#2}@#1}%
846 }

```

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

```

847 \newcommand*{\glxtrusefield}[2]{%
848 \@gls@entry@field{#1}{#2}%
849 }

```

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

```

850 \newcommand*{\Glsxtrusefield}[2]{%
851 \@gls@entry@field{#1}{#2}%
852 }

```

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

```
853 \newcommand*{\glstrdeffield}[2]{\csdef{glo@\glstoklabel{#1}@#2}}
```

`glstrdeffield` Just use `\csedef` to provide a field value for the given entry.

```
854 \newcommand*{\glstrdeffield}[2]{\csedef{glo@\glstoklabel{#1}@#2}}
```

`etfieldifexists`

```
855 \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.

```
856 \newrobustcmd*{\GlsXtrSetField}[3]{%
857   \glstrsetfieldifexists{#1}{#2}%
858   {\csdef{glo@\glstoklabel{#1}@#2}{#3}}%
859 }
```

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

```
860 \newrobustcmd*{\GlsXtrLetField}[3]{%
861   \glstrsetfieldifexists{#1}{#2}%
862   {\cslet{glo@\glstoklabel{#1}@#2}{#3}}%
863 }
```

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

```
864 \newrobustcmd*{\csGlsXtrLetField}[3]{%
865   \glstrsetfieldifexists{#1}{#2}%
866   {\csletcs{glo@\glstoklabel{#1}@#2}{#3}}%
867 }
```

`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.

```
868 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
869   \glstrsetfieldifexists{#1}{#2}%
870   {\csletcs{glo@\glstoklabel{#1}@#2}{glo@\glstoklabel{#3}@#4}}%
871 }
```

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

```
872 \newrobustcmd*{\gGlsXtrSetField}[3]{%
873   \glstrsetfieldifexists{#1}{#2}%
874   {\csgdef{glo@\glstoklabel{#1}@#2}{#3}}%
875 }
```

`xGlsXtrSetField`

```
876 \newrobustcmd*{\xGlsXtrSetField}[3]{%
877   \glstrsetfieldifexists{#1}{#2}%
878   {\protected@csxdef{glo@\glstoklabel{#1}@#2}{#3}}%
879 }
```

eGlsXtrSetField

```
880 \newrobustcmd*{\eGlsXtrSetField}[3]{%
881   \glstrsetfieldifexists{#1}{#2}%
882   {\protected@csedef{glo@glstdetoklabel{#1}@#2}{#3}}%
883 }
```

XtrIfFieldEqStr

```
884 \newrobustcmd*{\GlsXtrIfFieldEqStr}[5]{%
885   \glstrifhasfield{#1}{#2}%
886   {%
887     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
888   }%
889   {#5}%
890 }
```

\glstrpageref Like \glssrefentry but references the page number instead (if entry counting is on).

```
891 \ifglssentrycounter
892   \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstdetoklabel{#1}}}
893 \else
894   \ifglssubentrycounter
895     \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstdetoklabel{#1}}}
896   \else
897     \newcommand*{\glstrpageref}[1]{\gls{#1}}
898   \fi
899 \fi
```

lossarypreamble

```
900 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
901   \ifcsdef{glolist@#1}%
902   {%
903     \ifcsundef{@glossarypreamble@#1}%
904     {\csdef{@glossarypreamble@#1}{}}%
905   }%
906   \csappto{@glossarypreamble@#1}{#2}%
907   {%
908   {%
909     \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
910   }%
911 }
```

lossarypreamble

```
912 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
913   \ifcsdef{glolist@#1}%
914   {%
915     \ifcsundef{@glossarypreamble@#1}%
916     {\csdef{@glossarypreamble@#1}{}}%
917   }%
918   \cspreto{@glossarypreamble@#1}{#2}%
919   }%
```



```

920  {%
921    \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
922  }%
923 }

```

## 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.

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`

```

924 \renewcommand*{\longnewglossaryentry}{%
925   \@ifstar{\glstr@s@longnewglossaryentry}\glstr@longnewglossaryentry
926 }

```

`ewglossaryentry` Starred version.

```

927 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
928   \glstoifnoexists{#1}%
929   {%
930     \bgroup
931     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
932     \long\def\@newglossaryentryprehook{%
933       \long\def\@glo@desc{#3}%
934       \@org@newglossaryentryprehook
935     }%
936     \renewcommand*{\gls@assign@desc}[1]{%
937       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
938       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
939     }
940     \gls@defglossaryentry{#1}{#2}%
941   \egroup
942   }%
943 }

```

`ewglossaryentry` Unstarred version.

```

944 \newcommand{\@glstr@longnewglossaryentry}[3]{%
945   \glstoifnoexists{#1}%
946   {%
947     \bgroup
948     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
949     \long\def\@newglossaryentryprehook{%
950       \long\def\@glo@desc{#3\glstrpostlongdescription}%
951       \@org@newglossaryentryprehook
952     }%

```

```

953     \renewcommand*{\gls@assign@desc}[1]{%
954     \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
The following is different from the base glossaries.sty:
955     \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
956     }
957     \gls@defglossaryentry{#1}{#2}%
958     \egroup
959 }%
960 }

```

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

```

961 \newcommand*{\glxtrpostlongdescription}{\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.

```

962 \renewcommand{\newignoredglossary}{%
963 \ifstar\glxtr@s@newignoredglossary\glxtr@org@newignoredglossary
964 }

```

`\ignoredglossary` The original definition is patched to check for existence.

```

965 \newcommand*{\glxtr@org@newignoredglossary}[1]{%
966 \ifcsdef{glolist@#1}
967 {%
968 \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
969 }%
970 {%
971 \ifdefempty\@ignored@glossaries
972 {%
973 \edef\@ignored@glossaries{#1}%
974 }%
975 {%
976 \eappto\@ignored@glossaries{,#1}%
977 }%
978 \csgdef{glolist@#1}{,}%
979 \ifcsundef{gls@#1@entryfmt}%
980 {%
981 \defglsentryfmt[#1]{\glsentryfmt}%
982 }%
983 }%
984 \ifdefempty\@gls@nohyperlist
985 {%
986 \renewcommand*{\@gls@nohyperlist}{#1}%
987 }%
988 {%
989 \eappto\@gls@nohyperlist{,#1}%
990 }%

```

```

991 }%
992 }

```

ignoredglossary Starred form.

```

993 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
994   \ifcsdef{glolist@#1}
995   {%
996     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
997   }%
998   {%
999     \ifdefempty\@ignored@glossaries
1000     {%
1001       \edef\@ignored@glossaries{#1}%
1002     }%
1003     {%
1004       \eappto\@ignored@glossaries{, #1}%
1005     }%
1006     \csgdef{glolist@#1}{,}%
1007     \ifcsundef{gls@#1@entryfmt}%
1008     {%
1009       \defglsentryfmt[#1]{\glsentryfmt}%
1010     }%
1011     {}%
1012   }%
1013 }

```

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

```

1014 \glsifusetranslator
1015 {%
1016   \renewcommand*{\glsettoctitle}[1]{%
1017     \ifcsdef{gls@tr@set@#1@toctitle}%
1018     {%
1019       \csuse{gls@tr@set@#1@toctitle}%
1020     }%
1021     {%
1022       \ifcsdef{@glotype@#1@title}%
1023       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1024       {\def\glossarytoctitle{\glossarytitle}}%
1025     }%
1026   }%
1027 }
1028 {
1029   \renewcommand*{\glsettoctitle}[1]{%
1030     \ifcsdef{@glotype@#1@title}%
1031     {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1032     {\def\glossarytoctitle{\glossarytitle}}%
1033   }
1034 }

```

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

```
1035 \newcommand{\provideignoredglossary}{%
1036 \ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1037 }
```

ignoredglossary Unstarred version.

```
1038 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1039 \ifcsdef{glolist@#1}
1040 {}%
1041 {%
1042 \ifdefempty\@ignored@glossaries
1043 {%
1044 \edef\@ignored@glossaries{#1}%
1045 }%
1046 {%
1047 \eappto\@ignored@glossaries{,#1}%
1048 }%
1049 \csgdef{glolist@#1}{,}%
1050 \ifcsundef{gls@#1@entryfmt}%
1051 {%
1052 \defglsentryfmt[#1]{\glsentryfmt}%
1053 }%
1054 {}%
1055 \ifdefempty\@gls@nohyperlist
1056 {%
1057 \renewcommand*{\@gls@nohyperlist}{#1}%
1058 }%
1059 {%
1060 \eappto\@gls@nohyperlist{,#1}%
1061 }%
1062 }%
1063 }
```

ignoredglossary Starred form.

```
1064 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1065 \ifcsdef{glolist@#1}
1066 {}%
1067 {%
1068 \ifdefempty\@ignored@glossaries
1069 {%
1070 \edef\@ignored@glossaries{#1}%
1071 }%
1072 {%
1073 \eappto\@ignored@glossaries{,#1}%
1074 }%
1075 \csgdef{glolist@#1}{,}%
1076 \ifcsundef{gls@#1@entryfmt}%
1077 {%
1078 \defglsentryfmt[#1]{\glsentryfmt}%
1079 }
```

```

1079     }%
1080     {}%
1081   }%
1082 }

```

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

```

1083 \newcommand*{\glxtrcopytoglossary}[2]{%
1084   \glsdoidexists{#1}%
1085   {%
1086     \ifcsdef{glolist@#2}
1087     {%
1088       \cseappto{glolist@#2}{#1,}%
1089     }%
1090     {%
1091       \glxtrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1092     }%
1093   }%
1094 }

```

### 1.3.1 Existence Checks

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

```

1095 \renewcommand{\glsdoidexists}[2]{%
1096   \ifglstryexists{#1}{#2}%
1097   {%

```

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

```

1098     \edef\glslabel{\glsdetoklabel{#1}}%
1099     \glxtrundefaction{Glossary entry ‘\glslabel’
1100       has not been defined}{You need to define a glossary entry before
1101       you can reference it.}%
1102   }%
1103 }

```

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

```

1104 \renewcommand{\glsdoidnoexists}[2]{%
1105   \ifglstryexists{#1}{%
1106     \glxtrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1107       has already been defined}{}}{#2}%
1108 }

```

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

```

1109 \ifdef\glsdoidexistsordo
1110 {%
1111   \renewcommand{\glsdoidexistsordo}[3]{%

```

```

1112 \ifglentryexists{#1}{#2}%
1113 {%
1114 \glstrundefaction{Glossary entry '\glsetoklabel{#1}'
1115 has not been defined}{You need to define a glossary entry
1116 before you can use it.}%
1117 #3%
1118 }%
1119 }%
1120 }
1121 {%
1122 \glstr@warnonexistsordo\glsoifexistsordo
1123 \newcommand{\glsoifexistsordo}[3]{%
1124 \ifglentryexists{#1}{#2}%
1125 {%
1126 \glstrundefaction{Glossary entry '\glsetoklabel{#1}'
1127 has not been defined}{You need to define a glossary entry
1128 before you can use it.}%
1129 #3%
1130 }%
1131 }%
1132 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1133 \ifdef\doifglossarynoexistsordo
1134 {%
1135 \renewcommand{\doifglossarynoexistsordo}[3]{%
1136 \ifglossaryexists{#1}%
1137 {%
1138 \glstrundefaction{Glossary type '#1' already exists}{}%
1139 #3%
1140 }%
1141 {#2}%
1142 }%
1143 }
1144 {%
1145 \glstr@warnonexistsordo\doifglossarynoexistsordo
1146 \newcommand{\doifglossarynoexistsordo}[3]{%
1147 \ifglossaryexists{#1}%
1148 {%
1149 \glstrundefaction{Glossary type '#1' already exists}{}%
1150 #3%
1151 }%
1152 {#2}%
1153 }%
1154 }
1155

```

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.

```
1156 \appto\@newglossaryentryposthook{%
1157   \ifdefvoid\@glo@see
1158     {\csxdef{glo@\@glo@label @see}{}}%
1159     {%
1160       \csxdef{glo@\@glo@label @see}{\@glo@see}%
1161       \if@glxtr@autoindex
1162         \@glxtr@autoindexcrossrefs
1163       \fi
1164     }%
1165 }
1166 \appto\@gls@keymap{,{see}{see}}
```

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

```
1167 \newcommand*{\glxtrusesee}[1]{%
1168   \glsdoifexists{#1}%
1169   {%
1170     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
1171     \ifdefempty\@glo@see
1172       {}%
1173       {%
1174         \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1175       }%
1176     }%
1177 }
```

`\glxtr@usesee`

```
1178 \newcommand*{\glxtr@usesee}[1][\@seenname]{%
1179   \@glxtr@usesee{#1}%
1180 }
```

`\@glxtr@usesee`

```
1181 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1182   \glxtruseseeformat{#1}{#2}%
1183 }
```

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

```
1184 \newcommand*{\glxtruseseeformat}[2]{%
1185   \glsseeformat{#1}{#2}%
1186 }
```

`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 restores the original definition as it

makes more sense to use the name in the cross-reference list. This still uses `\glsaccesstext` for abbreviations.

```
1187 \renewcommand*{\glsseeitemformat}[1]{%
1188   \ifglshasshort{\glslabel}{\glsaccesstext{#1}}{\glsaccessname{#1}}%
1189 }
```

`\glstruseealso` Apply `\glsseeformat` to the `seealso` key if not empty. There's no optional tag to worry about here.

```
1190 \newcommand*{\glstruseealso}[1]{%
1191   \glsdoidexists{#1}%
1192   {%
1193     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@seealso}%
1194     \ifdefempty\@glo@see
1195     {}%
1196     {%
1197       \expandafter\glstruseealsoformat\expandafter{\@glo@see}%
1198     }%
1199   }%
1200 }
```

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

```
1201 \newcommand*{\glsseealsoformat}[1]{%
1202   \glsseeformat[\seealso]{#1}%
1203 }
```

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

```
1204 \newrobustcmd{\glstrseelist}[1]{%
1205   \edef\@glo@tmp{\noexpand\glsseelist{#1}}\@glo@tmp
1206 }
```

`\seealso` In case this command hasn't been defined. (Should be provided by language packages.)

```
1207 \providecommand{\seealso}{see also}
```

`\glstrindexseealso` 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.

```
1208 \ifdef\@xdycrossrefhook
1209 {
1210   Add the cross-reference class definition to the hook.
1211   \appto\@xdycrossrefhook{%
1212     \write\glswrite{(define-crossref-class \string"seealso\string"
1213       :unverified )}%
1214     \write\glswrite{(markup-crossref-list
1215       :class \string"seealso\string"^^J\space\space\space
1216       :open \string"\string\glstruseealsoformat\glsopenbrace\string"
```



```

1216         :close \string"glsclosebrace\string"))}%
1217   }

```

Append to class list.

```

1218   \appto\@xdylocationclassorder{\space\string"seealso\string"}
    This essentially works like \@do@seeglossary but uses the seealso class.
1219   \newrobustcmd*{\glxtrindexseealso}[2]{%
1220     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
1221       \@glxtr@recordsee{#1}{#2}%
1222     \fi
1223     \glsdofexists{#1}%
1224     {%
1225       \@@glxtrwrglossmark
1226       \def\@glx@xref{#2}%
1227       \@onelevel@sanitize\@glx@xref
1228       \@gl@checkmkidxchars\@glx@xref
1229       \gl@glossary{\csname glo@#1@type\endcsname}{%
1230         (indexentry
1231           :tkey (\csname glo@#1@index\endcsname)
1232           :xref (\string"\@glx@xref\string")
1233           :attr \string"seealso\string"
1234         )
1235       }%
1236     }%
1237   }
1238 }
1239 {

```

xindy not in use or glossaries version too old to support this.

```

1240   \newrobustcmd*{\glxtrindexseealso}{\glsee[\seealso]}
1241 }

```

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

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

```

1242 \ifdef\gl@set@xr@key
1243 {

```

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

```

1244   \define@key{glossentry}{alias}{%
1245     \gl@set@xr@key{alias}{\@glo@alias}{#1}%
1246   }
1247   \define@key{glossentry}{seealso}{%
1248     \gl@set@xr@key{seealso}{\@glo@seealso}{#1}%
1249   }

```

Add to the key mappings.

```
1250 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}
```

Set the default value.

```
1251 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%
```

Assign the field values.

```
1252 \appto\@newglossaryentryposthook{%
1253   \ifdefvoid\@glo@seealso
1254     {\csxdef{glo@\@glo@label @seealso}{}}%
1255     {%
1256       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1257       \if@glstr@autoseeindex
1258         \@glstr@autoindexcrossrefs
1259       \fi
1260     }%
}
```

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

```
1261 \ifdefvoid\@glo@alias
1262   {\csxdef{glo@\@glo@label @alias}{}}%
1263   {%
1264     \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1265   }%
1266 }
```

Provide user-level commands to access the values.

\glstralias

```
1267 \newcommand*\glstralias[1]{\@gls@entry@field{#1}{alias}}
```

trseealsolabels

```
1268 \newcommand*\glstrseealsolabels[1]{\@gls@entry@field{#1}{seealso}}
```

Add to the \@glo@autosee hook.

```
1269 \appto\@glo@autoseehook{%
1270   \ifdefvoid\@glo@alias
1271     {%
1272       \ifdefvoid\@glo@seealso
1273         {}%
1274       {%
1275         \edef\@do@glsee{\noexpand\glstrindexseealso
1276           {\@glo@label}{\@glo@seealso}}%
1277         \@do@glsee
1278       }%
1279     }%
1280   {%
}
```

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

```
1281 \ifdefvoid\@glo@see
1282   {%
}
```

```

1283     \edef\@do@glsssee{\noexpand\glsssee{\@glo@label}{\@glo@alias}}}%
1284     \@do@glsssee
1285     }%
1286     {}%
1287     }%
1288     }%
1289 }
1290 {

```

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

`\glstralias`

```

1291 \glsaddstoragekey*{alias}{}{\glstralias}

```

`trseealsolabels`

```

1292 \glsaddstoragekey*{seealso}{}{\glstrseealsolabels}

```

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.

```

1293 \appto\@newglossaryentryposthook{%
1294   \ifcsvoid{glo@\@glo@label @alias}%
1295   {%
1296     \ifcsvoid{glo@\@glo@label @seealso}%
1297     {}%
1298     {%
1299       \edef\@do@glsssee{\noexpand\glstrindexseealso
1300         {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1301       \@do@glsssee
1302     }%
1303   }%
1304   {%

```

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

```

1305   \ifdefvoid\@glo@see
1306   {%
1307     \edef\@do@glsssee{\noexpand\glsssee
1308       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1309     \@do@glsssee
1310   }%
1311   {}%
1312   }%
1313 }
1314 }

```

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

```

1315 \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.

```

1316 \newcommand*{\glxtraddallcrossrefs}{%
1317   \forallglossaries{\@glo@type}%
1318   {%
1319     \forallglsentries[\@glo@type]{\@glo@label}%
1320     {%
1321       \ifglsused{\@glo@label}%
1322       {\expandafter\glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
1323     }%
1324   }%
1325 }

```

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

```

1326 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
1327   \letcs{\@glo@see}{glo\glsdetoklabel{#1}@see}%
1328   \ifdefvoid\@glo@see
1329   {%
1330     {%
1331       \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1332     }%
1333     \letcs{\@glo@see}{glo\glsdetoklabel{#1}@seealso}%
1334     \ifdefvoid\@glo@see
1335     {%
1336       {%
1337         \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1338       }%
1339     }%

```

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

```

1340 \newcommand*{\glxtr@addunused}[1][]{%
1341   \@glxtr@addunused
1342 }

```

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

```

1343 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1344   \@for\@glxtr@label:=#1\do
1345   {%
1346     \ifglsused{\@glxtr@label}{}%
1347     {%
1348       \glsadd[format=glxtrunusedformat]{\@glxtr@label}%
1349       \glsunset{\@glxtr@label}%
1350       \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1351     }%
1352   }%
1353 }

```

xtrunusedformat

```
1354 \newcommand*{\glxtrunusedformat}[1]{\unskip}
```

### 1.3.2 Document Definitions

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

```
1355 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
1356 \renewcommand{\makenoidxglossaries}{%
1357   \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off
1358   {%
1359     \glxtr@orgmakenoidxglossaries
```

Add marker to \@do@seeglossary

```
1360   \renewcommand{\@do@seeglossary}[2]{%
1361     \@glxtrwrglossmark
1362     \edef\@gls@label{\glsdetoklabel{##1}}%
1363     \protected@write\@auxout{}{%
1364       \string\@gls@reference
1365       {\curname glo@\@gls@label @type\endcurname}%
1366       {\@gls@label}%
1367       {%
1368         \string\glsseeformat##2}%
1369       }%
1370   }%
1371 }
```

Check for docdefs=restricted:

```
1372 \if@glxtrdocdefrestricted
```

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

```
1373   \renewcommand*{\@gls@reference}[3]{%
1374     \ifcsundef{\@glsref@##1}{\csgdef{\@glsref@##1}{}}{}%
1375     \ifinlistcs{##2}{\@glsref@##1}%
1376     {}%
1377     {\listcsgadd{\@glsref@##1}{##2}}%
1378     \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1379     {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}{}%
1380     {}%
1381     \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1382   }%
1383   \else
```

Disable document definitions.

```
1384     \@glxtrdocdeffalse
1385   \fi
1386   \disable@keys{glossaries-extra}{docdef}%
1387 }%
1388 {}%
```

```

1389 \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1390 not permitted\MessageBreak
1391 with record=\@glxtr@record@setting\space package option}%
1392 {You may only use \string\makenoidxglossaries\ space with the
1393 record=off option}%
1394 }%
1395 }

```

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

```

1396 \renewcommand*{\gls@defdocnewglossaryentry}{%
1397 \ifcase\@glxtr@docdefval
1398 docdef=false:
1399 \renewcommand*{\newglossaryentry}[2]{%
1400 \PackageError{glossaries-extra}{Glossary entries must
1401 be \MessageBreak defined in the preamble with \MessageBreak
1402 package option ‘docdef=false’\MessageBreak(consider using
1403 ‘docdef=restricted’)}{Move your glossary definitions to
1404 the preamble. You can also put them in a \MessageBreak separate file
1405 and load them with \string\loadglsentries.}%
1406 }%
1407 \or
1408 docdef=true Since the see value is now saved in a field, it can be used by entries that have
1409 been defined in the document.
1410 \let\gls@checkseeallowed\relax
1411 \let\newglossaryentry\newglossaryentry
1412 \or
1413 Restricted mode just needs to allow the see value.
1414 \let\gls@checkseeallowed\relax
1415 \fi
1416 }%

```

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`

```

1413 \newcommand*{\GlsXtrEnableOnTheFly}{%
1414 \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1415 }

```

`\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.)

```

1416 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1417 \renewcommand*{\glsdetoklabel}[1]{%

```

```

1418 \expandafter\@glxtr@ifcsstart\string##1 \@glxtr@end@
1419 {%
1420 \expandafter\detokenize\expandafter{##1}%
1421 }%
1422 {\detokenize{##1}}%
1423 }%
1424 \@GlsXtrEnableOnTheFly
1425 }
1426 \def\@glxtr@ifcsstart#1#2\@glxtr@end@#3#4{%
1427 \expandafter\if\glslackspace#1%
1428 #3%
1429 \else
1430 #4%
1431 \fi
1432 }

```

sxtrstarflywarn

```

1433 \newcommand*\@glxtrstarflywarn{%
1434 \GlossariesExtraWarning{Experimental starred version of
1435 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1436 read the warnings in the glossaries-extra user manual)}}%
1437 }

```

rEnableOnTheFly

```

1438 \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

```

1439 \newcommand*\glxtrcat{general}

```

\glxtr

```

1440 \newcommand*\glxtr[1][]{%
1441 \def\glxtr@keylist{##1}%
1442 \@glxtr
1443 }

```

\@glxtr

```

1444 \newcommand*\@glxtr[2][]{%
1445 \ifglentryexists{##2}%
1446 {%
1447 \ifblank{##1}{-}\@GlsXtrWarning{##1}{##2}}%
1448 }%
1449 {%
1450 \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1451 description={\nopostdesc},##1}%
1452 }%

```

```

1453 \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
1454 }

\Glsxtr
1455 \newcommand*{\Glsxtr}[1] [] {%
1456 \def\glsxtr@keylist{##1}%
1457 \@Glsxtr
1458 }

\@Glsxtr
1459 \newcommand*{\@Glsxtr}[2] [] {%
1460 \ifglsentryexists{##2}%
1461 {%
1462 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1463 }%
1464 {%
1465 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1466 description={\nopostdesc},##1}%
1467 }%
1468 \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
1469 }

\glsxtrpl
1470 \newcommand*{\glsxtrpl}[1] [] {%
1471 \def\glsxtr@keylist{##1}%
1472 \@glsxtrpl
1473 }

\@glsxtrpl
1474 \newcommand*{\@glsxtrpl}[2] [] {%
1475 \ifglsentryexists{##2}%
1476 {%
1477 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1478 }%
1479 {%
1480 \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1481 description={\nopostdesc},##1}%
1482 }%
1483 \expandafter\glspl\expandafter[\glsxtr@keylist]{##2}%
1484 }

\Glsxtrpl
1485 \newcommand*{\Glsxtrpl}[1] [] {%
1486 \def\glsxtr@keylist{##1}%
1487 \@Glsxtrpl
1488 }

\@Glsxtrpl

```



```

1489 \newcommand*{\@GlsXtrPl}[2][\%
1490 \ifglentryexists{##2}
1491 {%
1492   \ifblank{##1}{\@GlsXtrWarning{##1}{##2}}%
1493 }%
1494 {%
1495   \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1496     description={\nopostdesc},##1}%
1497 }%
1498 \expandafter\GlsPl\expandafter[\glstr@keylist]{##2}%
1499 }

```

\GlsXtrWarning

```

1500 \newcommand*{\GlsXtrWarning}[2]{%
1501   \def\@glstr@optlist{##1}%
1502   \@onelevel@sanitize\@glstr@optlist
1503   \GlossariesExtraWarning{The options '\@glstr@optlist' have
1504     been ignored for entry '##2' as it has already been defined}%
1505 }

```

Disable commands after the glossary:

```

1506 \renewcommand\@printglossary[2]{%
1507   \def\@glstr@printglossopts{##1}%
1508   \@glstr@orgprintglossary{##1}{##2}%
1509   \def\@glstr{\@glstr@disabledflycommand\glstr}%
1510   \def\@glstrpl{\@glstr@disabledflycommand\glstrpl}%
1511   \def\@GlsXtr{\@glstr@disabledflycommand\GlsXtr}%
1512   \def\@GlsXtrpl{\@glstr@disabledflycommand\GlsXtrpl}%
1513 }

```

abledflycommand

```

1514 \newcommand*{\@glstr@disabledflycommand}[1]{%
1515   \PackageError{glossaries-extra}%
1516     {\string##1\space can't be used after any of the \MessageBreak
1517     glossaries have been displayed}%
1518     {The on-the-fly commands enabled by
1519     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1520     before the glossaries. If you want to use any entries \MessageBreak
1521     after any of the glossaries, you must use the standard \MessageBreak
1522     method of first defining the entry and then using the \MessageBreak
1523     entry with commands like \string\glstr}%
1524   \@glstr@disabledflycommand
1525 }%
1526 \newcommand*{\@glstr@disabledflycommand}[2][\%

```

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

```

1527 \let\GlsXtrEnableOnTheFly\relax
1528 }
1529 \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.

```
1530 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

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

`etglossarystyle`

```
1531 \renewcommand*{\setglossarystyle}[1]{%
1532   \ifcsundef{\@glstyle@#1}%
1533   {%
1534     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1535   }%
1536   {%
1537     \csname \@glstyle@#1\endcsname
```

Only set the current style if it exists.

```
1538     \protected@edef\@glxtr@current@style{#1}%
1539   }%
1540   \ifx\@glossary@default@style\relax
1541     \protected@edef\@glossary@default@style{#1}%
1542   \fi
1543 }
```

In case we have an old version of `glossaries`:

```
1544 \ifdef\@glossary@default@style
1545 {}
1546 {%
1547   \let\@glossary@default@style\relax
1548 }
```

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

```
1549 \ifdef\glslistdottedwidth
1550 {%
1551   \ifdim\glslistdottedwidth=.5\hsize
1552     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1553     \AtBeginDocument{%
1554       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1555         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1556       \fi
1557     }%
1558   \fi
1559 }
1560 {}%
```

Similarly for `\glstdescwidth`:

\glsdescwidth

```
1561 \ifdef\glsdescwidth
1562 {%
1563   \ifdim\glsdescwidth=.6\hsize
1564     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1565     \AtBeginDocument{%
1566       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1567         \setlength{\glsdescwidth}{.6\columnwidth}%
1568       \fi
1569     }%
1570   \fi
1571 }
1572 {}%
```

and for \glspagelistwidth:

\glspagelistwidth

```
1573 \ifdef\glspagelistwidth
1574 {%
1575   \ifdim\glspagelistwidth=.1\hsize
1576     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1577     \AtBeginDocument{%
1578       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1579         \setlength{\glspagelistwidth}{.1\columnwidth}%
1580       \fi
1581     }%
1582   \fi
1583 }
1584 {}%
```

aryentrynumbers Has the nonumberlist option been used?

```
1585 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1586 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1587   \glsnonumberlistfalse
1588   \renewcommand*{\glossaryentrynumbers}[1]{%
1589     \ifglsentryexists{\glscurrententrylabel}%
1590     {%
1591       \@glsxtrpreloctag
1592       \GlsXtrFormatLocationList{#1}%
1593       \@glsxtrpostloctag
1594       \gls@save@numberlist{#1}%
1595     }{}%
1596   }%
1597 \else
1598   \glsnonumberlisttrue
1599   \renewcommand*{\glossaryentrynumbers}[1]{%
1600     \ifglsentryexists{\glscurrententrylabel}%
1601     {%
1602       \gls@save@numberlist{#1}%

```

```

1603     }{}%
1604   }%
1605 \fi

```

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

```

1606 \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`

```

1607 \newcommand*\GlsXtrEnablePreLocationTag}[2]{%
1608   \let\@glxtrpreloctag\@glxtrpreloctag
1609   \let\@glxtrpostloctag\@glxtrpostloctag
1610   \renewcommand*\@glxtr@pagetag{#1}%
1611   \renewcommand*\@glxtr@pagetag{#2}%
1612   \renewcommand*\@glxtr@savepreloctag}[2]{%
1613     \csgdef{\@glxtr@preloctag@##1}{##2}%
1614   }%
1615   \renewcommand*\@glxtr@doloctag}{%
1616     \ifcsundef{\@glxtr@preloctag\@glscurrententrylabel}%
1617     {%
1618       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
1619         Rerun required}%
1620     }%
1621     {%
1622       \csuse{\@glxtr@preloctag\@glscurrententrylabel}%
1623     }%
1624   }%
1625 }
1626 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`\glxtrpreloctag`

```

1627 \newcommand*\@glxtrpreloctag{%
1628   \let\@glxtr@org@delimN\delimN
1629   \let\@glxtr@org@delimR\delimR
1630   \let\@glxtr@org@glsgignore\glsgignore

```

`\gdef` is required as the delimiters may occur inside a scope.

```

1631   \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
1632   \renewcommand*\@delimN{%
1633     \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%
1634     \@glxtr@org@delimN}%
1635   \renewcommand*\@delimR{%
1636     \gdef\@glxtr@thisloctag{\@glxtr@pagetag}%

```

```

1637 \glxtr@org@delimR}%
1638 \renewcommand*{\glsglignore}[1]{%
1639 \gdef\glxtr@thisloctag{\relax}%
1640 \glxtr@org@glsglignore{##1}}%
1641 \glxtr@doloctag
1642 }

```

glxtrpreloctag

```

1643 \newcommand*{\glxtrpreloctag}{}

```

@glxtr@pagetag

```

1644 \newcommand*{\glxtr@pagetag}{}%

```

glxtr@pagetag

```

1645 \newcommand*{\glxtr@pagetag}{}%

```

lsxtrpostloctag

```

1646 \newcommand*{\glxtrpostloctag}{%
1647 \let\delimN\glxtr@org@delimN
1648 \let\delimR\glxtr@org@delimR
1649 \let\glsglignore\glxtr@org@glsglignore
1650 \protected@write\@auxout{}%
1651 {\string\glxtr@savepreloctag{\glscurrententrylabel}\glxtr@thisloctag}}%
1652 }

```

lsxtrpostloctag

```

1653 \newcommand*{\glxtrpostloctag}{}

```

lsxtr@preloctag

```

1654 \newcommand*{\glxtr@savepreloctag}[2]{%
1655 \protected@write\@auxout{}%
1656 \string\providecommand\string\glxtr@savepreloctag[2]}

```

glxtr@doloctag

```

1657 \newcommand*{\glxtr@doloctag}{}

```

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

```

1658 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1659 \XKV@plfalse
1660 \XKV@sttrue
1661 \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1662 {%
1663 \csname glsnonumberlist\XKV@resa\endcsname
1664 \ifglsnonumberlist
1665 \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1666 \else
1667 \def\glossaryentrynumbers##1{%

```

```

1668      \@glsxtrpreloctag
1669      \GlsXtrFormatLocationList{##1}%
1670      \@glsxtrpostloctag
1671      \gls@save@numberlist{##1}}%
1672  \fi
1673 }%
1674 }

```

### 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.

```

1675 \renewcommand*{\glsentryfmt}{%
1676   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}{}}%
1677   \glsifregular{\glslabel}%
1678   {\glsxtrregularfont{\glsgenentryfmt}}%
1679   {%
1680     \ifglshasshort{\glslabel}%
1681     {\glsxtrgenabbrvfmt}%
1682     {\glsxtrregularfont{\glsgenentryfmt}}%
1683   }%
1684 }

```

`sxtrregularfont` Font used for regular entries.

```

1685 \newcommand*{\glsxtrregularfont}[1]{#1}

```

Commands like `\glsifplural` 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, `\glsfirst` or `\glsplural`. This can provide better consistency with the formatting of the `\gls`-like commands, even though they don't use `\glsentryfmt`.

`@gls@field@link` Redefine `\@gls@field@link` so that commands like `\glsfirst` can setup `\glsxtrifwasfirstuse` etc to allow the postlink hook to work better. This now has an optional argument that sets up the defaults.

```

1686 \renewcommand{\@gls@field@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).

```

1687   \@glsxtr@record{#2}{#3}{\glslink}%
1688   \glsdoifexists{#3}%
1689   {%

```

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

```

1690 \let\glxstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1691 \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
1692 \def\glscustomtext{#4}%
1693 \@glxstr@field@linkdefs
1694 #1%
1695 \@glsl@link[#2]{#3}{#4}%
1696 \let\ifKV@glslink@hyper\glxstrorg@ifKV@glslink@hyper
1697 }%
1698 \glspostlinkhook
1699 }

```

The commands `\glsl`, `\Gls` etc don't use `\@glsl@field@link`, so they need modifying as well to use `\@glxstr@record`.

`\@glsl@` Save the original definition and redefine.

```

1700 \let\@glxstr@org@glsl@\@glsl@
1701 \def\@glsl@#1#2{%
1702   \@glxstr@record{#1}{#2}{glslink}%
1703   \@glxstr@org@glsl@{#1}{#2}%
1704 }%

```

`\@glspl@` Save the original definition and redefine.

```

1705 \let\@glxstr@org@glspl@\@glspl@
1706 \def\@glspl@#1#2{%
1707   \@glxstr@record{#1}{#2}{glslink}%
1708   \@glxstr@org@glspl@{#1}{#2}%
1709 }%

```

`\@Gls@` Save the original definition and redefine.

```

1710 \let\@glxstr@org@Gls@\@Gls@
1711 \def\@Gls@#1#2{%
1712   \@glxstr@record{#1}{#2}{glslink}%
1713   \@glxstr@org@Gls@{#1}{#2}%
1714 }%

```

`\@Glspl@` Save the original definition and redefine.

```

1715 \let\@glxstr@org@Glspl@\@Glspl@
1716 \def\@Glspl@#1#2{%
1717   \@glxstr@record{#1}{#2}{glslink}%
1718   \@glxstr@org@Glspl@{#1}{#2}%
1719 }%

```

`\@GLS@` Save the original definition and redefine.

```

1720 \let\@glxstr@org@GLS@\@GLS@
1721 \def\@GLS@#1#2{%
1722   \@glxstr@record{#1}{#2}{glslink}%
1723   \@glxstr@org@GLS@{#1}{#2}%
1724 }%

```

`\@GLSp1@` Save the original definition and redefine.

```
1725 \let\@glxtr@org@GLSp1@\@GLSp1@
1726 \def\@GLSp1@#1#2{%
1727   \@glxtr@record{#1}{#2}{glslink}%
1728   \@glxtr@org@GLSp1@{#1}{#2}%
1729 }%
```

`\@glsdisp` Save the original definition and redefine. Can't save and restore `\@glsdisp` since it has an optional argument.

```
1730 \renewcommand*{\@glsdisp}[3][{}]{%
1731   \@glxtr@record{#1}{#2}{glslink}%
1732   \glsdoifexists{#2}{%
1733     \let\do@gl@link@checkfirsthyper\@gl@link@checkfirsthyper
1734     \let\glsifplural\@secondoftwo
1735     \let\glscapscase\@firstofthree
1736     \def\glscustomtext{#3}%
1737     \def\glsinsert{}%
1738     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1739     \@gl@link[#1]{#2}{\@glo@text}%
1740     \ifKV@glslink@local
1741       \glslocalunset{#2}%
1742     \else
1743       \glsunset{#2}%
1744     \fi
1745   }%
1746   \glspostlinkhook
1747 }
```

`\@gl@link@` Redefine to include `\@glxtr@record`

```
1748 \renewcommand*{\@gl@link}[3][{}]{%
1749   \@glxtr@record{#1}{#2}{glslink}%
1750   \glsdoifexistsordo{#2}%
1751   {%
1752     \let\do@gl@link@checkfirsthyper\relax
1753     \@gl@link[#1]{#2}{#3}%
1754   }%
1755   {%
1756     \glstextformat{#3}%
1757   }%
1758   \glspostlinkhook
1759 }
```

`sxtrinitwrgloss` Set the default if the `wrgloss` is omitted.

```
1760 \newcommand*{\glxtrinitwrgloss}{%
1761   \glsifattribute{\glslabel}{wrgloss}{after}%
1762   {%
1763     \glxtrinitwrglossbeforefalse
1764   }%
1765   {%
```



```

1766 \glstrinitwrglossbeforetrue
1767 }%
1768 }

```

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

```

1769 \newif\ifglstrinitwrglossbefore
1770 \glstrinitwrglossbeforetrue

```

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

```

1771 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1772 {%
1773   \ifcase\nr\relax
1774     \glstrinitwrglossbeforetrue
1775   \or
1776     \glstrinitwrglossbeforefalse
1777   \fi
1778 }

```

```

1779 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}

```

```

1780 \define@key{glslink}{theHvalue}{\def\@glstr@theHvalue{#1}}

```

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

```

1781 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{}
1782 \glstr@hyperoutsidetrue

```

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

```

1783 \newcommand*{\glstrnithyperoutside}{%
1784   \glsifattribute{glslabel}{hyperoutside}{false}%
1785   {%
1786     \glstr@hyperoutsidetrue
1787   }%
1788   {%
1789     \glstr@hyperoutsidefalse
1790   }%
1791 }

```

`\@gls@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`.

```

1792 \def\@gls@link[#1]#2#3{%
1793   \leavevmode
1794   \edef\glslabel{\glsdetoklabel{#2}}%
1795   \def\@gls@link@opts{#1}%
1796   \let\@gls@link@label\glslabel
1797   \let\@glsnumberformat\@glstr@defaultnumberformat
1798   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1799   \edef\glstype{\csname glo@\glslabel @type\endcsname}%
1800   \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

```

Initialise thevalue and theHvalue (v1.19).

```
1801 \def\@glxstr@thevalue{%
1802 \def\@glxstr@theHvalue{\@glxstr@thevalue}%
```

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

```
1803 \glxstrinitwrgloss
```

Initialise whether \hyperlink should be outside \glstextformat (new to v1.21).

```
1804 \glxstrinithyperoutside
```

As the original definition. Note that the default link options may override \glxstrinitwrgloss.

```
1805 \@gls@setdefault@glslink@opts
1806 \do@gl:disablehyperinlist
1807 \do@gl:link@checkfirsthyper
1808 \setkeys{glslink}{#1}%
1809 \glslinkpostsetkeys
```

Check thevalue and theHvalue before saving (v1.19).

```
1810 \ifdefempty{\@glxstr@thevalue}%
1811 {%
1812 \@gls@saveentrycounter
1813 }%
1814 {%
1815 \let\theglentrycounter\@glxstr@thevalue
1816 \def\theHglentrycounter{\@glxstr@theHvalue}%
1817 }%
1818 \@gls@setsort{\glslabel}%
```

Check textformat attribute (new to v1.21).

```
1819 \glshasattribute{\glslabel}{textformat}%
1820 {%
1821 \edef\@glxstr@attrval{\glsggetattribute{\glslabel}{textformat}}%
1822 \ifcsdef{\@glxstr@attrval}%
1823 {%
1824 \letcs{\@glxstr@textformat}{\@glxstr@attrval}%
1825 }%
1826 {%
1827 \GlossariesExtraWarning{Unknown control sequence name
1828 '\@glxstr@attrval' supplied in textformat attribute
1829 for entry '\glslabel'. Reverting to default \string\glstextformat}%
1830 \let\@glxstr@textformat\glstextformat
1831 }%
1832 }%
1833 {%
1834 \let\@glxstr@textformat\glstextformat
1835 }%
```

Do write if it should occur before the link text:

```
1836 \ifglxstrinitwrglossbefore
1837 \do@wrglossary{#2}%
1838 \fi
```

Do the link text:

```

1839 \ifKV@glslink@hyper
1840 \ifglxtr@hyperoutside
1841 \@glslink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1842 \else
1843 \@glxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
1844 \fi
1845 \else
1846 \ifglxtr@hyperoutside
1847 \glndonohyperlink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1848 \else
1849 \@glxtr@textformat{\glndonohyperlink{\glolinkprefix\glslabel}{#3}}%
1850 \fi
1851 \fi

```

Do write if it should occur after the link text:

```

1852 \ifglxtr@nritwrglossbefore
1853 \else
1854 \do@wrglossary{#2}%
1855 \fi

```

As the original definition:

```

1856 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
1857 }

```

```

1858 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

```

```

1859 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

```

`\glsadd` Redefine to include `\@glxtr@record` and suppress in headings

```

1860 \renewrobustcmd*{\glsadd}[2][\%
1861 \glxtr@finmark
1862 {}%
1863 {%
1864 \@gls@adjustmode
1865 \@glxtr@record{#1}{#2}{glossadd}%
1866 \glsoifexists{#2}%
1867 {%
1868 \let\@glsnumberformat\@glxtr@defaultnumberformat
1869 \edef\@gls@counter{\csname glo@\glxtr@detoklabel{#2}@counter\endcsname}%
1870 \def\@glxtr@thevalue{}%
1871 \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1872 \setkeys{glossadd}{#1}%
1873 \ifdefempty{\@glxtr@thevalue}%
1874 {%
1875 \@gls@saveentrycounter
1876 }%
1877 {%
1878 \let\thegl@entrycounter\@glxtr@thevalue
1879 \def\theHgl@entrycounter{\@glxtr@theHvalue}%

```

```

1880      }%
      Define sort key if necessary (in case of sort=use):
1881      \@gls@setsort{#2}%
1882      \@do@wrglossary{#2}%
1883      }%
1884  }%
1885 }

```

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

```

1886 \newcommand*{\@glsxtr@field@linkdefs}{%
1887   \let\glsxtrifwasfirstuse\@secondoftwo
1888   \let\glsifplural\@secondoftwo
1889   \let\glscapscase\@firstofthree
1890   \let\glsinsert\@empty
1891 }

```

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

`assignfieldfont`

```

1892 \newcommand*{\glsxtrassignfieldfont}[1]{%
1893   \ifglstryexists{#1}%
1894   {%
1895     \ifglshasshort{#1}%
1896     {%
1897       \glssetabbrvfmt{\glscategory{#1}}%
1898       \glsifregular{#1}%
1899       {\let\@gls@field@font\glsxtrregularfont}%
1900       {\let\@gls@field@font\@firstofone}%
1901     }%
1902     {%
1903       \glsifnotregular{#1}%
1904       {\let\@gls@field@font\@firstofone}%
1905       {\let\@gls@field@font\glsxtrregularfont}%
1906     }%
1907   }%
1908   {%
1909     \let\@gls@field@font\@gobble
1910   }%
1911 }

```

`\@glstext@` The abbreviation format may also need setting.

```

1912 \def\@glstext@#1#2[#3]{%
1913   \glsxtrassignfieldfont{#2}%
1914   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1915 }

```

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

```

1916 \def\@GLStext@#1#2[#3]{%
1917   \glstrassignfieldfont{#2}%
1918   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1919   {\@gls@field@font{\Glsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1920 }

```

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

```

1921 \def\@Glstext@#1#2[#3]{%
1922   \glstrassignfieldfont{#2}%
1923   \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1924   {\@gls@field@font{\Glsaccesstext{#2}#3}}%
1925 }

```

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.

ecknohyperfirst

```

1926 \newcommand*{\glstrchecknohyperfirst}[1]{%
1927   \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1928 }

```

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

```

1929 \def\@glsfirst@#1#2[#3]{%
1930   \glstrassignfieldfont{#2}%
1931   \@gls@field@link
1932   [\let\glstrifwasfirstuse\@firstoftwo
1933   \glstrchecknohyperfirst{#2}%
1934   ]{#1}{#2}%
1935   {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1936 }

```

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

```

1937 \def\@Glsfirst@#1#2[#3]{%
1938   \glstrassignfieldfont{#2}%
1939   \@gls@field@link
1940   [\let\glstrifwasfirstuse\@firstoftwo
1941   \let\glscapscase\@secondofthree
1942   \glstrchecknohyperfirst{#2}%
1943   ]%
1944   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1945 }

```

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

```

1946 \def\@GLSfirst@#1#2[#3]{%
1947   \glstrassignfieldfont{#2}%

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

```
1948 \@gls@field@link
1949 [\let\glstrifwasfirstuse\@firstoftwo
1950 \let\glscapscase\@thirdofthree
1951 \glstrchecknohyperfirst{#2}%
1952 ]%
1953 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1954 }
```

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

```
1955 \def\@glsplural@#1#2[#3]{%
1956 \glstrassignfieldfont{#2}%
1957 \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1958 {\@gls@field@font{\glsaccessplural{#2}#3}}%
1959 }
```

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

```
1960 \def\@Glsplural@#1#2[#3]{%
1961 \glstrassignfieldfont{#2}%
1962 \@gls@field@link
1963 [\let\glsifplural\@firstoftwo
1964 \let\glscapscase\@secondofthree
1965 ]%
1966 {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1967 }
```

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

```
1968 \def\@GLSplural@#1#2[#3]{%
1969 \glstrassignfieldfont{#2}%
1970 \@gls@field@link
1971 [\let\glsifplural\@firstoftwo
1972 \let\glscapscase\@thirdofthree
1973 ]%
1974 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1975 }
```

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

```
1976 \def\glsfirstplural@#1#2[#3]{%
1977 \glstrassignfieldfont{#2}%
1978 \@gls@field@link
1979 [\let\glstrifwasfirstuse\@firstoftwo
1980 \let\glsifplural\@firstoftwo
1981 \glstrchecknohyperfirst{#2}%
1982 ]%
1983 {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1984 }
```

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

```
1985 \def\@Glsfirstplural@#1#2[#3]{%
1986   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1987   \@gls@field@link
1988   [\let\glstrifwasfirstuse\@firstoftwo
1989   \let\glstifplural\@firstoftwo
1990   \let\glscapscase\@secondofthree
1991   \glstrchecknohyperfirst{#2}%
1992   ]%
1993   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1994 }
```

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

```
1995 \def\@GLSfirstplural@#1#2[#3]{%
1996   \glstrassignfieldfont{#2}%
    Ensure that \glfirstplural honours the nohyperfirst attribute.
1997   \@gls@field@link
1998   [\let\glstrifwasfirstuse\@firstoftwo
1999   \let\glstifplural\@firstoftwo
2000   \let\glscapscase\@thirdofthree
2001   \glstrchecknohyperfirst{#2}%
2002   ]%
2003   {#1}{#2}%
2004   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2005 }
```

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

```
2006 \def\@glsname@#1#2[#3]{%
2007   \glstrassignfieldfont{#2}%
2008   \@gls@field@link{#1}{#2}{\@gls@field@font{\Glsaccessname{#2}#3}}%
2009 }
```

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

```
2010 \def\@Glsname@#1#2[#3]{%
2011   \glstrassignfieldfont{#2}%
2012   \@gls@field@link
2013   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2014   {\@gls@field@font{\Glsaccessname{#2}#3}}%
2015 }
```

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

```
2016 \def\@GLSname@#1#2[#3]{%
2017   \glstrassignfieldfont{#2}%
2018   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2019   {#1}{#2}%
2020   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2021 }
```

```

\@glsdesc@
2022 \def\@glsdesc@#1#2[#3]{%
2023   \glstrassignfieldfont{#2}%
2024   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2025 }

\@Glsdesc@   First letter uppercase version.
2026 \def\@Glsdesc@#1#2[#3]{%
2027   \glstrassignfieldfont{#2}%
2028   \@gls@field@link
2029   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2030   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2031 }

\@GLSdesc@   All uppercase version.
2032 \def\@GLSdesc@#1#2[#3]{%
2033   \glstrassignfieldfont{#2}%
2034   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2035   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2036 }

@glsdescplural@   No case-changing version.
2037 \def\@glsdescplural@#1#2[#3]{%
2038   \glstrassignfieldfont{#2}%
2039   \@gls@field@link
2040   [\let\glscapscase\@secondoftwo
2041    \let\glsifplural\@firstoftwo
2042   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2043 }

@Glsdescplural@   First letter uppercase version.
2044 \def\@Glsdescplural@#1#2[#3]{%
2045   \glstrassignfieldfont{#2}%
2046   \@gls@field@link
2047   [\let\glscapscase\@secondoftwo
2048    \let\glsifplural\@firstoftwo
2049   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2050 }

@GLSdescplural@   All uppercase version.
2051 \def\@GLSdesc@#1#2[#3]{%
2052   \glstrassignfieldfont{#2}%
2053   \@gls@field@link
2054   [\let\glscapscase\@thirdoftwo
2055    \let\glsifplural\@firstoftwo
2056   ]%
2057   {#1}{#2}%
2058   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2059 }

```



\@glssymbol@

```
2060 \def\@glssymbol@#1#2[#3]{%
2061   \glstrassignfieldfont{#2}%
2062   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2063 }
```

\@Glsymbol@ First letter uppercase version.

```
2064 \def\@Glsymbol@#1#2[#3]{%
2065   \glstrassignfieldfont{#2}%
2066   \@gls@field@link
2067   [\let\glscapscase\@secondoftwo]%
2068   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2069 }
```

\@GLSsymbol@ All uppercase version.

```
2070 \def\@GLSsymbol@#1#2[#3]{%
2071   \glstrassignfieldfont{#2}%
2072   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2073   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2074 }
```

lssymbolplural@ No case-changing version.

```
2075 \def\@lssymbolplural@#1#2[#3]{%
2076   \glstrassignfieldfont{#2}%
2077   \@gls@field@link
2078   [\let\glscapscase\@secondoftwo
2079   \let\glsifplural\@firstoftwo
2080   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2081 }
```

lssymbolplural@ First letter uppercase version.

```
2082 \def\@lssymbolplural@#1#2[#3]{%
2083   \glstrassignfieldfont{#2}%
2084   \@gls@field@link
2085   [\let\glscapscase\@secondoftwo
2086   \let\glsifplural\@firstoftwo
2087   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2088 }
```

LSsymbolplural@ All uppercase version.

```
2089 \def\@LSsymbol@#1#2[#3]{%
2090   \glstrassignfieldfont{#2}%
2091   \@gls@field@link
2092   [\let\glscapscase\@thirdoftwo
2093   \let\glsifplural\@firstoftwo
2094   ]%
2095   {#1}{#2}%
2096   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2097 }
```

\@Glsuseri@ First letter uppercase version.

```
2098 \def\@Glsuseri@#1#2[#3]{%
2099   \glstrassignfieldfont{#2}%
2100   \@gls@field@link
2101   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2102   {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2103 }
```

\@GLSuseri@ All uppercase version.

```
2104 \def\@GLSuseri@#1#2[#3]{%
2105   \glstrassignfieldfont{#2}%
2106   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2107   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
2108 }
```

\@Glsuserii@ First letter uppercase version.

```
2109 \def\@Glsuserii@#1#2[#3]{%
2110   \glstrassignfieldfont{#2}%
2111   \@gls@field@link
2112   [\let\glscapscase\@secondoftwo]%
2113   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2114 }
```

\@GLSuserii@ All uppercase version.

```
2115 \def\@GLSuserii@#1#2[#3]{%
2116   \glstrassignfieldfont{#2}%
2117   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2118   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2119 }
```

\@Glsuseriii@ First letter uppercase version.

```
2120 \def\@Glsuseriii@#1#2[#3]{%
2121   \glstrassignfieldfont{#2}%
2122   \@gls@field@link
2123   [\let\glscapscase\@secondoftwo]%
2124   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2125 }
```

\@GLSuseriii@ All uppercase version.

```
2126 \def\@GLSuseriii@#1#2[#3]{%
2127   \glstrassignfieldfont{#2}%
2128   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2129   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2130 }
```

\@Glsuseriv@ First letter uppercase version.

```
2131 \def\@Glsuseriv@#1#2[#3]{%
2132   \glstrassignfieldfont{#2}%
2133 }
```

```

2133 \@gls@field@link
2134 [\let\glscapscase\@secondoftwo]%
2135 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2136 }

```

\@GLSuseriv@ All uppercase version.

```

2137 \def\@GLSuseriv@#1#2[#3]{%
2138 \glstrassignfieldfont{#2}%
2139 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2140 {#1}{#2}%
2141 {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2142 }

```

\@Glsuserv@ First letter uppercase version.

```

2143 \def\@Glsuserv@#1#2[#3]{%
2144 \glstrassignfieldfont{#2}%
2145 \@gls@field@link
2146 [\let\glscapscase\@secondoftwo]%
2147 {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
2148 }

```

\@GLSuserv@ All uppercase version.

```

2149 \def\@GLSuserv@#1#2[#3]{%
2150 \glstrassignfieldfont{#2}%
2151 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2152 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2153 }

```

\@Glsuservi@ First letter uppercase version.

```

2154 \def\@Glsuservi@#1#2[#3]{%
2155 \glstrassignfieldfont{#2}%
2156 \@gls@field@link
2157 [\let\glscapscase\@secondoftwo]%
2158 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}%
2159 }

```

\@GLSuservi@ All uppercase version.

```

2160 \def\@GLSuservi@#1#2[#3]{%
2161 \glstrassignfieldfont{#2}%
2162 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2163 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2164 }

```

Commands like \acrshort already set \glsifplural, but they don't set \glxtrifwasfirstuse so they need adjusting.

\@acrshort No case change.

```

2165 \def\@acrshort#1#2[#3]{%

```

```

2166 \glsdoifexists{#2}%
2167 {%
2168   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2169   \let\glxtrifwasfirstuse\@secondoftwo
2170   \let\gl@ifplural\@secondoftwo
2171   \let\glscapscase\@firstofthree
2172   \let\glinsert\@empty
2173   \def\glscustomtext{%
2174     \acronymfont{\glaccessshort{#2}}#3%
2175   }%
2176   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2177 }%
2178 \glspostlinkhook
2179 }

```

\@Acrshort First letter uppercase.

```

2180 \def\@Acrshort#1#2[#3]{%
2181   \glsdoifexists{#2}%
2182   {%
2183     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2184     \let\glxtrifwasfirstuse\@secondoftwo
2185     \let\gl@ifplural\@secondoftwo
2186     \let\glscapscase\@secondofthree
2187     \let\glinsert\@empty
2188     \def\glscustomtext{%
2189       \acronymfont{\Glsaccessshort{#2}}#3%
2190     }%
2191     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2192   }%
2193   \glspostlinkhook
2194 }

```

\@ACRshort All uppercase.

```

2195 \def\@ACRshort#1#2[#3]{%
2196   \glsdoifexists{#2}%
2197   {%
2198     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2199     \let\glxtrifwasfirstuse\@secondoftwo
2200     \let\gl@ifplural\@secondoftwo
2201     \let\glscapscase\@thirdofthree
2202     \let\glinsert\@empty
2203     \def\glscustomtext{%
2204       \mfirstucMakeUppercase{\acronymfont{\glaccessshort{#2}}#3}%
2205     }%
2206     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2207   }%
2208   \glspostlinkhook
2209 }

```

\@acrshortpl No case change.

```
2210 \def\@acrshortpl#1#2[#3]{%
2211   \glsdoifexists{#2}%
2212   {%
2213     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2214     \let\glxtrifwasfirstuse\@secondoftwo
2215     \let\gl@sifplural\@firstoftwo
2216     \let\glscapscase\@firstofthree
2217     \let\gl$insert\@empty
2218     \def\glscustomtext{%
2219       \acronymfont{\gl@saccesssshortpl{#2}}#3%
2220     }%
2221     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2222   }%
2223   \glspostlinkhook
2224 }
```

\@Acrshortpl First letter uppercase.

```
2225 \def\@Acrshortpl#1#2[#3]{%
2226   \glsdoifexists{#2}%
2227   {%
2228     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2229     \let\glxtrifwasfirstuse\@secondoftwo
2230     \let\gl@sifplural\@firstoftwo
2231     \let\glscapscase\@secondofthree
2232     \let\gl$insert\@empty
2233     \def\glscustomtext{%
2234       \acronymfont{\Glsaccesssshortpl{#2}}#3%
2235     }%
2236     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2237   }%
2238   \glspostlinkhook
2239 }
```

\@ACRshortpl All uppercase.

```
2240 \def\@ACRshortpl#1#2[#3]{%
2241   \glsdoifexists{#2}%
2242   {%
2243     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2244     \let\glxtrifwasfirstuse\@secondoftwo
2245     \let\gl@sifplural\@firstoftwo
2246     \let\glscapscase\@thirdofthree
2247     \let\gl$insert\@empty
2248     \def\glscustomtext{%
2249       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshortpl{#2}}#3}%
2250     }%
2251     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2252   }%
2253   \glspostlinkhook
```

2254 }

\@acrlong No case change.

```
2255 \def\@acrlong#1#2[#3]{%
2256   \glsdoifexists{#2}%
2257   {%
2258     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2259     \let\glxtrifwasfirstuse\@secondoftwo
2260     \let\glsifplural\@secondoftwo
2261     \let\glscapscase\@firstofthree
2262     \let\glsinsert\@empty
2263     \def\glscustomtext{%
2264       \acronymfont{\glsaccesslong{#2}}#3%
2265     }%
2266     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2267   }%
2268   \glspostlinkhook
2269 }
```

\@Acrlong First letter uppercase.

```
2270 \def\@Acrlong#1#2[#3]{%
2271   \glsdoifexists{#2}%
2272   {%
2273     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2274     \let\glxtrifwasfirstuse\@secondoftwo
2275     \let\glsifplural\@secondoftwo
2276     \let\glscapscase\@secondofthree
2277     \let\glsinsert\@empty
2278     \def\glscustomtext{%
2279       \acronymfont{\Glsaccesslong{#2}}#3%
2280     }%
2281     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2282   }%
2283   \glspostlinkhook
2284 }
```

\@ACRlong All uppercase.

```
2285 \def\@ACRlong#1#2[#3]{%
2286   \glsdoifexists{#2}%
2287   {%
2288     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2289     \let\glxtrifwasfirstuse\@secondoftwo
2290     \let\glsifplural\@secondoftwo
2291     \let\glscapscase\@thirdofthree
2292     \let\glsinsert\@empty
2293     \def\glscustomtext{%
2294       \mfirstucMakeUppercase{\acronymfont{\glsaccesslong{#2}}#3}%
2295     }%
2296     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2297   }
```

```

2297 }%
2298 \glspostlinkhook
2299 }

```

\@acrlongpl No case change.

```

2300 \def\@acrlongpl#1#2[#3]{%
2301   \glsdoidexists{#2}%
2302   {%
2303     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2304     \let\glxtrifwasfirstuse\@secondoftwo
2305     \let\gl@sifplural\@firstoftwo
2306     \let\glscapscase\@firstofthree
2307     \let\gl$insert\@empty
2308     \def\glscustomtext{%
2309       \acronymfont{\glaccesslongpl{#2}}#3%
2310     }%
2311     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2312   }%
2313   \glspostlinkhook
2314 }

```

\@Acrlongpl First letter uppercase.

```

2315 \def\@Acrlongpl#1#2[#3]{%
2316   \glsdoidexists{#2}%
2317   {%
2318     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2319     \let\glxtrifwasfirstuse\@secondoftwo
2320     \let\gl@sifplural\@firstoftwo
2321     \let\glscapscase\@secondofthree
2322     \let\gl$insert\@empty
2323     \def\glscustomtext{%
2324       \acronymfont{\Glsaccesslongpl{#2}}#3%
2325     }%
2326     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2327   }%
2328   \glspostlinkhook
2329 }

```

\@ACRlongpl All uppercase.

```

2330 \def\@ACRlongpl#1#2[#3]{%
2331   \glsdoidexists{#2}%
2332   {%
2333     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2334     \let\glxtrifwasfirstuse\@secondoftwo
2335     \let\gl@sifplural\@firstoftwo
2336     \let\glscapscase\@thirdofthree
2337     \let\gl$insert\@empty
2338     \def\glscustomtext{%
2339       \mfirstucMakeUppercase{\acronymfont{\glaccesslongpl{#2}}#3}%

```

```

2340 }%
2341 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2342 }%
2343 \glspostlinkhook
2344 }

```

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

\@glsaddkey

```

2345 \renewcommand*{\@glsaddkey}[7]{%
2346 \key@ifundefined{glossentry}{#1}%
2347 {%
2348 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2349 \appto\@gls@keymap{,{#1}{#1}}%
2350 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2351 \appto\@newglossaryentryposthook{%
2352 \letcs{\@glo@tmp}{@glo@#1}%
2353 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2354 }%
2355 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2356 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2357 \ifcsdef{@gls@user@#1@}%
2358 {%
2359 \PackageError{glossaries}%
2360 {Can't define '\string#5' as helper command
2361 '\expandafter\string\csname @gls@user@#1@\endcsname' already
2362 exists}%
2363 }%
2364 }%
2365 {%
2366 \expandafter\newcommand\expandafter*\expandafter
2367 {\csname @gls@user@#1\endcsname}[2][ ]{%
2368 \new@ifnextchar[%
2369 {\csuse{@gls@user@#1@}{##1}{##2}}%
2370 {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2371 \csdef{@gls@user@#1@}##1##2[##3]{%
2372 \@gls@field@link{##1}{##2}{#3{##2}##3}%
2373 }%
2374 \newrobustcmd*{#5}{%
2375 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
2376 }%

```

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

```

2377 \ifcsdef{@Gls@user@#1@}%
2378 {%
2379 \PackageError{glossaries}%
2380 {Can't define '\string#6' as helper command

```



```

2381         '\expandafter\string\csname @Gls@user@#1@\endcsname' already
2382         exists}%
2383     {}%
2384 }%
2385 {%
2386     \expandafter\newcommand\expandafter*\expandafter
2387     {\csname @Gls@user@#1@\endcsname}[2] []{%
2388         \new@ifnextchar[%
2389             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2390             {\csuse{@Gls@user@#1@}{##1}{##2} []}}}%
2391     \csdef{@Gls@user@#1@}##1##2[##3]{%
2392         \@gls@field@link[\let\gls@caps@case\@secondofthree]%
2393         {##1}{##2}{#4{##2}##3}%
2394     }%
2395     \newrobustcmd*{#6}{%
2396         \expandafter\@gls@hyp@opt\csname @Gls@user@#1@\endcsname}%
2397     }%

```

Finally the all caps version (modified):

```

2398     \ifcsdef{@GLS@user@#1@}%
2399     {%
2400         \PackageError{glossaries}%
2401         {Can't define '\string#7' as helper command
2402         '\expandafter\string\csname @GLS@user@#1@\endcsname' already
2403         exists}%
2404     {}%
2405 }%
2406 {%
2407     \expandafter\newcommand\expandafter*\expandafter
2408     {\csname @GLS@user@#1@\endcsname}[2] []{%
2409         \new@ifnextchar[%
2410             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2411             {\csuse{@GLS@user@#1@}{##1}{##2} []}}}%
2412     \csdef{@GLS@user@#1@}##1##2[##3]{%
2413         \@gls@field@link[\let\gls@caps@case\@thirdofthree]%
2414         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
2415     }%
2416     \newrobustcmd*{#7}{%
2417         \expandafter\@gls@hyp@opt\csname @GLS@user@#1@\endcsname}%
2418     }%
2419 }%
2420 {%
2421     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2422 }%
2423 }

```

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

```

2424 \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.

```
2425 \let\@glstr@org@checkfirsthyper\@gls@link@checkfirsthyper
2426 \renewcommand*{\@gls@link@checkfirsthyper}{%
```

\ifglsused 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.

```
2427 \ifglsused{\glslabel}%
2428 {\let\glstr@ifwasfirstuse\@secondoftwo}
2429 {\let\glstr@ifwasfirstuse\@firstoftwo}%
```

Store the category label for convenience.

```
2430 \edef\glscategorylabel{\glscategory{\glslabel}}%
2431 \ifglsused{\glslabel}%
2432 {%
2433   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2434   {\KV@glslink@hyperfalse}{}%
2435 }%
2436 {%
2437   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2438   {\KV@glslink@hyperfalse}{}%
2439 }%
2440 \glslinkcheckfirsthyperhook
2441 }
```

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.

```
2442 \ifdef\do@glstdisablehyperinlist
2443 {%
2444   \let\@glstr@do@glstdisablehyperinlist\do@glstdisablehyperinlist
2445   \renewcommand*{\do@glstdisablehyperinlist}{%
2446     \@glstr@do@glstdisablehyperinlist
2447     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
2448   }
2449 }
2450 {}
```

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

```
2451 \define@boolkey{glslink}{noindex}[true]{}
2452 \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

```
2453 \ifdef\@gls@setdefault@glslink@opts
2454 {
2455   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2456     \KV@glslink@noindexfalse
```

```

2457 \glxtrsetaliasnoindex
2458 }
2459 }
2460 {
    Not defined so prepend it to \do@glstdisablehyperinlist to achieve the same effect.
2461 \newcommand*{\@glstdisablehyperinlist@glslink@opts}{%
2462 \KV@glslink@noindexfalse
2463 \glxtrsetaliasnoindex
2464 }
2465 \pretoto\do@glstdisablehyperinlist{\@glstdisablehyperinlist@glslink@opts}
2466 }

```

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.)

```

2467 \providecommand*\glxtrsetaliasnoindex{%
2468 \KV@glslink@noindextrue
2469 }

```

setaliasindex

```

2470 \newcommand*{\@glxtrsetaliasindex}{%
2471 \glxtrifhasfield{alias}{\glslabel}%
2472 {%
2473 \let\glxtrindexaliased\@glxtrindexaliased
2474 \glxtrsetaliasnoindex
2475 \let\glxtrindexaliased\@no@glxtrindexaliased
2476 }%
2477 }%
2478 }

```

xtrindexaliased

```

2479 \newcommand{\@glxtrindexaliased}{%
2480 \ifKV@glslink@noindex
2481 \else
2482 \begingroup
2483 \let\@glslnumberformat\@glxtr@defaultnumberformat
2484 \edef\@glscounter{\csname glo@glstdetoklabel{\glslabel}@counter\endcsname}%
2485 \glxtr@saveentrycounter
2486 \do@wrglossary{\glxtralias{\glslabel}}%
2487 \endgroup
2488 \fi
2489 }

```

xtrindexaliased

```

2490 \newcommand{\@no@glxtrindexaliased}{%
2491 \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
2492 not permitted outside definition of \string\glxtrsetaliasnoindex}%
2493 {}%
2494 }

```

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

```
2495 \let\glxtrindexaliased\@no\glxtrindexaliased
```

tDefaultGlsOpts Set the default options for \glslink etc.

```
2496 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2497   \renewcommand*{\@glslsetdefault@glslink@opts}{%
2498     \setkeys{glslink}{#1}%
2499     \@glxtrsetaliasnoindex
2500   }%
2501 }
```

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

```
2502 \newcommand*{\glxtrifindexing}[2]{%
2503   \ifKV@glslink@noindex #2\else #1\fi
2504 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2505 \renewcommand*{\glswriteentry}[2]{%
2506   \glxtrifindexing
2507   {%
2508     \ifglindexonlyfirst
2509       \ifglused{#1}
2510       {\glxtrdoautoindexname{#1}{dualindex}}%
2511       {#2}%
2512     \else
2513       \gl@ifattribute{#1}{indexonlyfirst}{true}%
2514       {\ifglused{#1}
2515        {\glxtrdoautoindexname{#1}{dualindex}}%
2516        {#2}}%
2517       {#2}%
2518     \fi
2519   }%
2520   {}%
2521 }
```

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

```
2522 \appto\@do@wrglossary{\@glxtr@do@wrindex
2523   \glxtrdowrglossaryhook{\@glsl@label}%
2524 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2525 \appto\glsl@noidxglossary{\@glxtr@do@wrindex
2526   \glxtrdowrglossaryhook{\@glsl@label}%
2527 }
```

xtr@do@@wrindex

```
2528 \newcommand*{\@glsxtr@do@@wrindex}{%
2529   \glsxtrdoautoindexname{\@gls@label}{dualindex}%
2530 }
```

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.)

```
2531 \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.

```
2532 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2533   \let\glslinkvar\@firstofthree
2534   \let\@gls@hyp@opt@cs#1\relax
2535   \@ifstar{\s@gls@hyp@opt}%
2536   {\@ifnextchar+%
2537     {\@firstoftwo{\p@gls@hyp@opt}}%
2538     {%
2539       \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2540       {\@firstoftwo{\@alt@gls@hyp@opt}}%
2541       {#1}%
2542     }%
2543   }%
2544 }
```

alt@gls@hyp@opt User version

```
2545 \newcommand*{\@alt@gls@hyp@opt}[1] [] {%
2546   \let\glslinkvar\@firstofthree
2547   \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}
```

lt@hyp@opt@char Contains the character used as the command modifier.

```
2548 \newcommand*{\@gls@alt@hyp@opt@char}{{}}
```

lt@hyp@opt@keys Contains the option list used as the command modifier.

```
2549 \newcommand*{\@gls@alt@hyp@opt@keys}{{}}
```

rSetAltModifier

```
2550 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2551   \let\@gls@hyp@opt\@gls@alt@hyp@opt
2552   \def\@gls@alt@hyp@opt@char{#1}%
2553   \def\@gls@alt@hyp@opt@keys{#2}%
2554 }
```

org@dohyperlink

```
2555 \let\glsxtr@org@dohyperlink\glsdohyperlink
```

`\glsnavhyperlink` Now that `\glsdohyperlink` (used by `\@glslink`) references `\glslabel` it's necessary to patch `\glsnavhyperlink` to avoid using it (since `\glslabel` won't be defined). This means temporarily redefining `\glsdohyperlink` to its original definition.

This command is provided by `glossary-hypernav` so it may not exist.

```
2556 \ifdef\glsnavhyperlink
2557 {
2558   \renewcommand*\glsnavhyperlink}[3][\@glo@type]{%
2559     \edef\gls@grplabel{#2}\protected@edef\gls@grptitle{#3}%
```

Scope:

```
2560   {%
2561     \let\glsdohyperlink\glsxtr@org@dohyperlink
2562     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2563   }%
2564 }%
2565 }
2566 {}
```

`\glsdohyperlink` 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.

```
2567 \renewcommand*\glsdohyperlink}[2]{%
2568   \glsattribute{\glslabel}{targeturl}%
2569   {%
2570     \glsattribute{\glslabel}{targetname}%
2571     {%
2572       \glsattribute{\glslabel}{targetcategory}%
2573       {%
2574         \hyperref{\glsattribute{\glslabel}{targeturl}}%
2575           {\glsattribute{\glslabel}{targetcategory}}%
2576           {\glsattribute{\glslabel}{targetname}}%
2577           {\glsxtrprotectlinks#2}}%
2578       }%
2579     }%
2580     \hyperref{\glsattribute{\glslabel}{targeturl}}%
2581       {%
2582         {\glsattribute{\glslabel}{targetname}}%
2583         {\glsxtrprotectlinks#2}}%
2584     }%
2585   }%
2586   {%
2587     \href{\glsattribute{\glslabel}{targeturl}}%
2588       {\glsxtrprotectlinks#2}}%
2589   }%
```

```

2590 }%
2591 {%
    Check for alias.
2592 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2593 \ifvoid\gloaliaslabel
2594 {%
2595     \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2596 }%
2597 {%

```

Redirect link to the alias target.

```

2598 \glsxtrhyperlink
2599 {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2600 {\glsxtrprotectlinks#2}}%
2601 }%
2602 }%
2603 }

```

`\glsxtrhyperlink` Allows integration with the base glossaries package's `debug=showtargets` option.

```

2604 \ifdef\@glsshowtarget
2605 {
2606     \newcommand{\glsxtrhyperlink}[2]{%
2607         \@glsshowtarget{#1}%
2608         \hyperlink{#1}{#2}%
2609     }%
2610 }
2611 {
2612     \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2613 }

```

`\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.

```

2614 \renewrobustcmd*{\gls hyperlink}[2][\glsentrytext{\@glo@label}]{%
2615     \glsdoifexists{#2}%
2616     {%
2617         \def\@glo@label{#2}%
2618         {\edef\glslabel{#2}%
2619         \@glslink{\glolinkprefix\glslabel}{#1}}%
2620     }%
2621 }

```

`\glsdisablehyper` Redefine in case we have an old version of glossaries. This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdohyperlink`.

```

2622 \renewcommand{\glsdisablehyper}{%
2623     \KV@glslink@hyperfalse
2624     \def\@glslink{\glsdohyperlink}%

```

```

2625 \let\@glstarget\@secondoftwo
2626 }

```

`\glsenablehyper` This now uses `\def` rather than `\let` to allow for redefinitions of `\glsdohypertarget` and `\glsdohyperlink`.

```

2627 \renewcommand{\glsenablehyper}{%
2628 \KV@glslink@hypertrue
2629 \def\@glslink{\glsdohyperlink}%
2630 \def\@glstarget{\glsdohypertarget}%
2631 }

```

`\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.

```

2632 \def\glsdonohyperlink#1#2{{\glsxtrprotectlinks #2}}

```

`\@glslink` Reset `\@glslink` with patched versions:

```

2633 \ifcsundef{hyperlink}%
2634 {%
2635 \def\@glslink{\glsdonohyperlink}
2636 }%
2637 {%
2638 \def\@glslink{\glsdohyperlink}
2639 }

```

`\glsxtrprotectlinks` Make `\gls` (and variants) behave like the corresponding `\gls{text}` (and variants) with hyperlinking and indexing off.

```

2640 \newcommand*{\glsxtrprotectlinks}{%
2641 \KV@glslink@hyperfalse
2642 \KV@glslink@noindextrue
2643 \let\@gls\@glsxtr@p@text@
2644 \let\@Gls\@Glsxtr@p@text@
2645 \let\@GLS\@GLSxtr@p@text@
2646 \let\@glspl\@glsxtr@p@plural@
2647 \let\@Glspl\@Glsxtr@p@plural@
2648 \let\@GLSpl\@GLSxtr@p@plural@
2649 \let\@glsxtrshort\@glsxtr@p@short@
2650 \let\@Glsxtrshort\@Glsxtr@p@short@
2651 \let\@GLSxtrshort\@GLSxtr@p@short@
2652 \let\@glsxtrlong\@glsxtr@p@long@
2653 \let\@Glsxtrlong\@Glsxtr@p@long@
2654 \let\@GLSxtrlong\@GLSxtr@p@long@
2655 \let\@glsxtrshortpl\@glsxtr@p@shortpl@
2656 \let\@Glsxtrshortpl\@Glsxtr@p@shortpl@
2657 \let\@GLSxtrshortpl\@GLSxtr@p@shortpl@
2658 \let\@glsxtrlongpl\@glsxtr@p@longpl@
2659 \let\@Glsxtrlongpl\@Glsxtr@p@longpl@
2660 \let\@GLSxtrlongpl\@GLSxtr@p@longpl@

```



```

2661 \let\@acrshort\@glsxtrp@acrshort@
2662 \let\@Acrshort\@Glsxtrp@acrshort@
2663 \let\@ACRshort\@GLSxtrp@acrshort@
2664 \let\@acrshortpl\@glsxtrp@acrshortpl@
2665 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
2666 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
2667 \let\@acrlong\@glsxtrp@acrlong@
2668 \let\@Acrlong\@Glsxtrp@acrlong@
2669 \let\@ACRlong\@GLSxtrp@acrlong@
2670 \let\@acrlongpl\@glsxtrp@acrlongpl@
2671 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
2672 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
2673 }

```

These protected versions need grouping to prevent the label from getting confused.

@glsxtrp@text@

```
2674 \def\@glsxtrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2675 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2676 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2677 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2678 \def\@lsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
2679 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtrp@short@

```

2680 \def\@glsxtrp@short@#1#2[#3]{%
2681 {%
2682   \glssetabbrvfmt{\glscategory{#2}}%
2683   \glsabbrvfont{\glstryshort{#2}}#3%
2684 }%
2685 }

```

Glsxtrp@short@

```

2686 \def\@Glsxtrp@short@#1#2[#3]{%
2687 {%
2688   \glssetabbrvfmt{\glscategory{#2}}%
2689   \glsabbrvfont{\Glsentryshort{#2}}#3%
2690 }%
2691 }

```

GLSxtr@p@short@

```
2692 \def\@GLSxtr@p@short@#1#2[#3]{%
2693   {%
2694     \glsetabbrvfmt{\glscategory{#2}}%
2695     \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
2696   }%
2697 }
```

sxtr@p@shortpl@

```
2698 \def\@glxtr@p@shortpl@#1#2[#3]{%
2699   {%
2700     \glsetabbrvfmt{\glscategory{#2}}%
2701     \glabbrvfont{\glentryshortpl{#2}}#3%
2702   }%
2703 }
```

sxtr@p@shortpl@

```
2704 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2705   {%
2706     \glsetabbrvfmt{\glscategory{#2}}%
2707     \glabbrvfont{\Glsentryshortpl{#2}}#3%
2708   }%
2709 }
```

Sxtr@p@shortpl@

```
2710 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2711   {%
2712     \glsetabbrvfmt{\glscategory{#2}}%
2713     \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
2714   }%
2715 }
```

@glxtr@p@long@

```
2716 \def\@glxtr@p@long@#1#2[#3]{\{\glentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2717 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2718 \def\@GLSxtr@p@long@#1#2[#3]{%
2719   {\mfirstucMakeUppercase{\glslongfont{\glentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2720 \def\@glxtr@p@longpl@#1#2[#3]{\{\glentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2721 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2722 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2723   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2724 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2725 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2726 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2727   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2728 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2729 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2730 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2731   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2732 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
2733 \def\@GLSxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
2734 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2735   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
2736 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2737 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2738 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2739   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2740 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

`\glsxtrsetpopts` Used in glossary to switch hyperlinks on for the `\glsxtrp` type of commands.

```
2741 \newcommand*{\glsxtrsetpopts}[1]{%
2742   \renewcommand*{\@glsxtrp@opt}{#1}%
2743 }
```

`\glossxtrsetpopts` Used in glossary to switch hyperlinks on for the `\glsxtrp` type of commands.

```
2744 \newcommand*{\glossxtrsetpopts}{%
2745   \glsxtrsetpopts{noindex}%
2746 }
```

`\@@glsxtrp`

```
2747 \newrobustcmd*{\@@glsxtrp}[2]{%
```

Add scope.

```
2748   {%
2749     \let\glspostlinkhook\relax
2750     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2751   }%
2752 }
```

`\@glsxtrp`

```
2753 \newrobustcmd*{\@glsxtrp}[2]{%
2754   \ifcsdef{gls#1}%
2755     {%
2756       \@glsxtrp{gls#1}{#2}%
2757     }%
2758     {%
2759       \ifcsdef{glsxtr#1}%
2760         {%
2761           \@glsxtrp{glsxtr#1}{#2}%
2762         }%
2763         {%
2764           \PackageError{glossaries-extra}{‘#1’ not recognised by
2765             \string\glsxtrp}{}%
2766         }%
2767       }%
2768 }
```

`\@Glsxtrp`

```
2769 \newrobustcmd*{\@Glsxtrp}[2]{%
2770   \ifcsdef{Gls#1}%
2771     {%
2772       \@glsxtrp{Gls#1}{#2}%
2773     }%
2774     {%
2775       \ifcsdef{Glsxtr#1}%
2776         {%
2777           \@glsxtrp{Glsxtr#1}{#2}%
2778         }%
2779     }
```

```

2779   {%
2780     \PackageError{glossaries-extra}{‘#1’ not recognised by
2781       \string\Glsxtrp}{}%
2782   }%
2783 }%
2784 }

```

\@GLSxtrp

```

2785 \newrobustcmd*{\@GLSxtrp}[2]{%
2786   \ifcsdef{GLS#1}%
2787   {%
2788     \@@glxtrp{GLS#1}{#2}%
2789   }%
2790   {%
2791     \ifcsdef{GLSxtr#1}%
2792     {%
2793       \@@glxtrp{GLSxtr#1}{#2}%
2794     }%
2795     {%
2796       \PackageError{glossaries-extra}{‘#1’ not recognised by
2797         \string\GLSxtrp}{}%
2798     }%
2799   }%
2800 }

```

\glxtr@entry@p

```

2801 \newrobustcmd*{\glxtr@headentry@p}[2]{%
2802   \gl@ifattribute{#1}{headuc}{true}%
2803   {%
2804     \mfirstucMakeUppercase{\@gl@entry@field{#1}{#2}}%
2805   }%
2806   {%
2807     \@gl@entry@field{#1}{#2}%
2808   }%
2809 }

```

\glxtrp Not robust as it needs to expand somewhat.

```

2810 \ifdef\texorpdfstring
2811 {
2812   \newcommand{\glxtrp}[2]{%
2813     \protect\NoCaseChange
2814     {%
2815       \protect\texorpdfstring
2816       {%
2817         \protect\glxtrifinmark
2818         {%
2819           \ifcsdef{glxtrhead#1}%
2820           {%
2821             {\protect\csuse{glxtrhead#1}{#2}}%

```

```

2822         }%
2823         {%
2824         \glxtr@headentry@p{#2}{#1}%
2825         }%
2826     }%
2827     {%
2828     \@glxtrp{#1}{#2}%
2829     }%
2830 }%
2831 {%
2832 \protect\@gls@entry@field{#2}{#1}%
2833 }%
2834 }%
2835 }
2836 }
2837 {
2838 \newcommand{\glxtrp}[2]{%
2839 \protect\NoCaseChange
2840 {%
2841 \protect\glxtrifinmark
2842 {%
2843 \ifcsdef{glxtrhead#1}%
2844 {%
2845 {\protect\csuse{glxtrhead#1}}%
2846 }%
2847 {%
2848 \glxtr@headentry@p{#2}{#1}%
2849 }%
2850 }%
2851 {%
2852 \@glxtrp{#1}{#2}%
2853 }%
2854 }%
2855 }
2856 }

```

Provide short synonyms for the most common option.

`\glsps`

```
2857 \newcommand*{\glsps}{\glxtrp{short}}
```

`\glspt`

```
2858 \newcommand*{\glspt}{\glxtrp{text}}
```

`\Glsxtrp` As above but use first letter upper case (but not for the bookmarks, which can't process `\uppercase`).

```

2859 \ifdef\texorpdfstring
2860 {
2861 \newcommand{\Glsxtrp}[2]{%

```

```

2862 \protect\NoCaseChange
2863 {%
2864 \protect\texorpdfstring
2865 {%
2866 \protect\glxtrifinmark
2867 {%
2868 \ifcsdef{Glsxtrhead#1}%
2869 {%
2870 {\protect\csuse{Glsxtrhead#1}{#2}}%
2871 }%
2872 {%
2873 \protect\@Gls@entry@field{#2}{#1}%
2874 }%
2875 }%
2876 {%
2877 \@Glsxtrp{#1}{#2}%
2878 }%
2879 }%
2880 {%
2881 \protect\@gls@entry@field{#2}{#1}%
2882 }%
2883 }%
2884 }
2885 }
2886 {
2887 \newcommand{\Glsxtrp}[2]{%
2888 \protect\NoCaseChange
2889 {%
2890 \protect\glxtrifinmark
2891 {%
2892 \ifcsdef{Glsxtrhead#1}%
2893 {%
2894 {\protect\csuse{Glsxtrhead#1}}%
2895 }%
2896 {%
2897 \protect\@Gls@entry@field{#2}{#1}%
2898 }%
2899 }%
2900 {%
2901 \@Glsxtrp{#1}{#2}%
2902 }%
2903 }%
2904 }
2905 }

```

`\GLSxtrp` As above but all upper case (but not for the bookmarks, which can't process `\uppercase`).

```

2906 \ifdef\texorpdfstring
2907 {
2908 \newcommand{\GLSxtrp}[2]{%

```

```

2909 \protect\NoCaseChange
2910 {%
2911 \protect\texorpdfstring
2912 {%
2913 \protect\glsxtrifinmark
2914 {%
2915 \ifcsdef{GLSxtr#1}%
2916 {%
2917 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2918 }%
2919 {%
2920 \protect\mfirstucMakeUppercase
2921 {%
2922 \protect\@gls@entry@field{#2}{#1}%
2923 }%
2924 }%
2925 }%
2926 {%
2927 \@GLSxtrp{#1}{#2}%
2928 }%
2929 }%
2930 {%
2931 \protect\@gls@entry@field{#2}{#1}%
2932 }%
2933 }%
2934 }
2935 }
2936 {
2937 \newcommand{\GLSxtrp}[2]{%
2938 \protect\NoCaseChange
2939 {%
2940 \protect\glsxtrifinmark
2941 {%
2942 \ifcsdef{GLSxtr#1}%
2943 {%
2944 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2945 }%
2946 {%
2947 \protect\mfirstucMakeUppercase
2948 {%
2949 \protect\@gls@entry@field{#2}{#1}%
2950 }%
2951 }%
2952 }%
2953 {%
2954 \@GLSxtrp{#1}{#2}%
2955 }%
2956 }%
2957 }

```



2958 }

### 1.3.5 Entry Counting

The 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's instead.

First adjust definitions of the unset and reset commands to provide a hook.

\@glsunset Global unset.

```
2959 \renewcommand*{\@glsunset}[1]{%
2960   \@glsunset{#1}%
2961   \glsxtrpostunset{#1}%
2962 }%
```

glsxtrpostunset

```
2963 \newcommand*{\glsxtrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2964 \renewcommand*{\@glslocalunset}[1]{%
2965   \@glslocalunset{#1}%
2966   \glsxtrpostlocalunset{#1}%
2967 }%
```

rpostlocalunset

```
2968 \newcommand*{\glsxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2969 \renewcommand*{\@glsreset}[1]{%
2970   \@glsreset{#1}%
2971   \glsxtrpostreset{#1}%
2972 }%
```

glsxtrpostreset

```
2973 \newcommand*{\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2974 \renewcommand*{\@glslocalreset}[1]{%
2975   \@glslocalreset{#1}%
2976   \glsxtrpostlocalreset{#1}%
2977 }%
```

rpostlocalreset

```
2978 \newcommand*{\glsxtrpostlocalreset}[1]{}
```

\leEntryCounting The first argument is the list of categories and the second argument is the value of the entrycount attribute.

```
2979 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2980 \glsenableentrycount
```

Redefine \gls etc:

```
2981 \renewcommand*{\gls}{\cglsl}%
2982 \renewcommand*{\Gls}{\cGls}%
2983 \renewcommand*{\glspl}{\cglspl}%
2984 \renewcommand*{\Glspl}{\cGlspl}%
2985 \renewcommand*{\GLS}{\cGLS}%
2986 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2987 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2988 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2989 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2990 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2991 can't be used with \string\GlsXtrEnableEntryCounting}%
2992 {Use one or other but not both commands}}%
2993 }
```

ycountunsetattr

```
2994 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2995 \@for\@glsxtr@cat:=#1\do
2996 {%
2997 \ifdefempty{\@glsxtr@cat}{}%
2998 {%
2999 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3000 }%
3001 }%
3002 }
```

Redefine the entry counting commands to take into account the entrycount attribute.

nableentrycount

```
3003 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
3004 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

Just in case the user has switched on the docdef option.

```
3005 \renewcommand*{\gls@defdocnewglossaryentry}{%
3006 \renewcommand*\newglossaryentry[2]{%
3007 \PackageError{glossaries}{\string\newglossaryentry\space
3008 may only be used in the preamble when entry counting has
3009 been activated}{If you use \string\glsenableentrycount\space
3010 you must place all entry definitions in the preamble not in
3011 the document environment}%
3012 }%
3013 }%
```

New commands to access new fields:

```

3014 \newcommand*{\glsentrycurrcount}[1]{%
3015   \ifcsundef{glo@\glsdetoklabel{##1}@currcount}%
3016   {0}{\@gls@entry@field{##1}{currcount}}%
3017 }%
3018 \newcommand*{\glsentryprevcount}[1]{%
3019   \ifcsundef{glo@\glsdetoklabel{##1}@prevcount}%
3020   {0}{\@gls@entry@field{##1}{prevcount}}%
3021 }%

```

Adjust post unset and reset:

```

3022 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
3023 \renewcommand*{\glsxtrpostunset}[1]{%
3024   \@glsxtr@entrycount@org@unset{##1}%
3025   \@gls@increment@currcount{##1}%
3026 }%
3027 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
3028 \renewcommand*{\glsxtrpostlocalunset}[1]{%
3029   \@glsxtr@entrycount@org@localunset{##1}%
3030   \@gls@local@increment@currcount{##1}%
3031 }%
3032 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3033 \renewcommand*{\glsxtrpostreset}[1]{%
3034   \@glsxtr@entrycount@org@reset{##1}%
3035   \csgdef{glo@\glsdetoklabel{##1}@currcount}{0}%
3036 }%
3037 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3038 \renewcommand*{\glsxtrpostlocalreset}[1]{%
3039   \@glsxtr@entrycount@org@localreset{##1}%
3040   \csdef{glo@\glsdetoklabel{##1}@currcount}{0}%
3041 }%

```

Modifications to take into account the attributes that govern whether the entry should be unset.

```

3042 \let\@cgl@s\@cgl@s
3043 \let\@cgl spl\@cgl spl

3044 \let\@cGls\@cGls
3045 \let\@cGlspl\@cGlspl
3046 \let\@cGLS\@cGLS
3047 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

3048 \AtEndDocument{\@gls@write@entrycounts}%
3049 \renewcommand*{\@gls@entry@count}[2]{%
3050   \csgdef{glo@\glsdetoklabel{##1}@prevcount}{##2}%
3051 }%
3052 \let\glsenableentrycount\relax
3053 \renewcommand*{\glsenableentryunitcount}{%
3054   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space

```

```

3055     can't be used with \string\glsenableentrycount}%
3056     {Use one or other but not both commands}%
3057 }%
3058 }

```

`writeentrycounts` 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.

```

3059 \newcommand*{\@gls@write@entrycounts}{%
3060   \immediate\write\@auxout
3061   {\string\providecommand*\string\@gls@entry@count}[2]{}%
3062   \count@=0\relax
3063   \forallglsentries{\@glsentry}{%
3064     \glshasattribute{\@glsentry}{entrycount}%
3065     {%
3066       \ifglsused{\@glsentry}%
3067       {%
3068         \immediate\write\@auxout
3069         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3070       }%
3071     }%
3072     \advance\count@ by \@ne
3073   }%
3074 }%
3075 }%
3076 \ifnum\count@=0
3077   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3078   \MessageBreak with \string\glsenableentrycount\space but the
3079   \MessageBreak attribute 'entrycount' hasn't
3080   \MessageBreak been assigned to any of the defined
3081   \MessageBreak entries}%
3082 \fi
3083 }

```

`trifcounttrigger` `\glstrifcounttrigger{<label>}{<trigger format>}{<normal>}`

```

3084 \newcommand*{\glstrifcounttrigger}[3]{%
3085   \glshasattribute{#1}{entrycount}%
3086   {%
3087     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
3088     #3%
3089   \else
3090     #2%
3091   \fi
3092 }%
3093 {#3}%
3094 }

```

Actual internal definitions of \cgl's used when entry counting is enabled.

\@@cgl's@

```

3095 \def\@@cgl's@#1#2[#3]{%
3096   \gl'sxtrifcounttrigger{#2}%
3097   {%
3098     \cgl'sformat{#2}{#3}%
3099     \gl'sunset{#2}%
3100   }%
3101   {%
3102     \@gl's@{#1}{#2}[#3]%
3103   }%
3104 }%
```

\@@cgl'spl@

```

3105 \def\@@cgl'spl@#1#2[#3]{%
3106   \gl'sxtrifcounttrigger{#2}%
3107   {%
3108     \cgl'splformat{#2}{#3}%
3109     \gl'sunset{#2}%
3110   }%
3111   {%
3112     \@gl'spl@{#1}{#2}[#3]%
3113   }%
3114 }%
```

\@@cGl's@

```

3115 \def\@@cGl's@#1#2[#3]{%
3116   \gl'sxtrifcounttrigger{#2}%
3117   {%
3118     \cGl'sformat{#2}{#3}%
3119     \gl'sunset{#2}%
3120   }%
3121   {%
3122     \@Gl's@{#1}{#2}[#3]%
3123   }%
3124 }%
```

\@@cGl'spl@

```

3125 \def\@@cGl'spl@#1#2[#3]{%
3126   \gl'sxtrifcounttrigger{#2}%
3127   {%
3128     \cGl'splformat{#2}{#3}%
3129     \gl'sunset{#2}%
3130   }%
3131   {%
3132     \@Gl'spl@{#1}{#2}[#3]%
3133   }%
3134 }%
```

\@@cGLS@

```
3135 \def\@@cGLS@#1#2[#3]{%
3136   \glxtrifcounttrigger{#2}%
3137   {%
3138     \cGLSformat{#2}{#3}%
3139     \glset{#2}%
3140   }%
3141   {%
3142     \@GLS@{#1}{#2}[#3]%
3143   }%
3144 }%
```

\@@cGLSpl@

```
3145 \def\@@cGLSpl@#1#2[#3]{%
3146   \glxtrifcounttrigger{#2}%
3147   {%
3148     \cGLSplformat{#2}{#3}%
3149     \glset{#2}%
3150   }%
3151   {%
3152     \@GLSpl@{#1}{#2}[#3]%
3153   }%
3154 }%
```

Remove default warnings from \cgl's etc so that it can be used interchangeable with \gl's etc.

\@cgl's@

```
3155 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
3156 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
3157 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```

\@cGL'spl@

```
3158 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
3159 \newrobustcmd*{\cGLS}{\@gl'shyp@opt\@cGLS}
```

\@cGLS Defined the un-starred form. Need to determine if there is a final optional argument

```
3160 \newcommand*{\@cGLS}[2][ ]{%
3161   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}]%
3162 }
```

```

\@cGLS@
3163 \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.
3164 \newcommand*\cGLSformat[2]{%
3165   \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
3166 }

\cGLSpl
3167 \newrobustcmd*\cGLSpl{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
3168 \newcommand*\@cGLSpl[2][{}]{%
3169   \new@ifnextchar[\@cGLSpl@{#1}{#2}}{\@cGLSpl@{#1}{#2}[{}]}%
3170 }

\@cGLSpl@
3171 \def\@cGLSpl@#1#2[#3]{\@GLSpl@{#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.
3172 \newcommand*\cGLSplformat[2]{%
3173   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
3174 }

      Modify the trigger formats to check for the regular attribute.

\cglformat
3175 \renewcommand*\cglformat[2]{%
3176   \glsifregular{#1}
3177   {\glsentryfirst{#1}}%
3178   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
3179 }

\cGlsformat
3180 \renewcommand*\cGlsformat[2]{%
3181   \glsifregular{#1}
3182   {\Glsentryfirst{#1}}%
3183   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
3184 }

\cglsplformat
3185 \renewcommand*\cglsplformat[2]{%
3186   \glsifregular{#1}
3187   {\glsentryfirstplural{#1}}%
3188   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
3189 }

```

\cGlsplformat

```
3190 \renewcommand*{\cGlsplformat}[2]{%
3191   \glsifregular{#1}
3192   {\Glsentryfirstplural{#1}}%
3193   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
3194 }
```

New code similar to above for unit counting.

defunitcounters

```
3195 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3196   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}%
3197   \ifdefined\@glo@countunit
3198     {}%
3199     {%
3200       \@glxtr@ifunitcounter{\@glo@countunit}%
3201       {}%
3202       {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
3203     }%
3204 }
```

r@unitcountlist List to keep track of which counters are being used by the entry unit count facility.

```
3205 \newcommand*{\@glxtr@unitcountlist}{}%
```

@addunitcounter

```
3206 \newcommand*{\@glxtr@addunitcounter}[1]{%
3207   \listadd{\@glxtr@unitcountlist}{#1}%
3208   \ifcsundef{glxtr@theunit@#1}
3209     {%
3210       \ifcsdef{theH#1}%
3211       {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
3212       {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
3213     }%
3214   {}%
3215 }
```

r@ifunitcounter

```
3216 \newcommand*{\@glxtr@ifunitcounter}[3]{%
3217   \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
3218 }
```

urrentunitcount

```
3219 \newcommand*{\@glxtr@currentunitcount}[1]{%
3220   glo@\glsetoklabel{#1}@currunit@\glsetattribute{#1}{unitcount}.%
3221   \csuse{glxtr@theunit@\glsetattribute{#1}{unitcount}}%
3222 }
```



previousunitcount

```
3223 \newcommand*\@glsxtr@previousunitcount[1]{%
3224   glo@glstdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
3225   \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}%
3226 }
```

t@currunitcount

```
3227 \newcommand*\@gls@increment@currunitcount[1]{%
3228   \glshasattribute{#1}{unitcount}%
3229   {%
3230     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3231     \ifcsundef{\@glsxtr@csname}%
3232     {%
3233       \csgdef{\@glsxtr@csname}{1}%
3234       \listcsxadd
3235         {glo@glstdetoklabel{#1}@unitlist}%
3236         {\glsggetattribute{#1}{unitcount}.%
3237          \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}}%
3238       }%
3239     }%
3240     {%
3241       \csxdef{\@glsxtr@csname}%
3242         {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3243       }%
3244     }%
3245     {}%
3246 }
```

t@currunitcount

```
3247 \newcommand*\@gls@local@increment@currunitcount[1]{%
3248   \glshasattribute{#1}{unitcount}%
3249   {%
3250     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3251     \ifcsundef{\@glsxtr@csname}%
3252     {%
3253       \csdef{\@glsxtr@csname}{1}%
3254       \listcseadd
3255         {glo@glstdetoklabel{#1}@unitlist}%
3256         {\glsggetattribute{#1}{unitcount}.%
3257          \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}}%
3258       }%
3259     }%
3260     {%
3261       \csedef{\@glsxtr@csname}%
3262         {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3263       }%
3264     }%
3265     {}%
3266 }
```

r@currunitcount

```
3267 \newcommand*{\@glxstr@currunitcount}[2]{%
3268   \ifcsundef
3269   {glo@\glstetoklabel{#1}@currunit@#2}%
3270   {0}%
3271   {\csuse{glo@\glstetoklabel{#1}@currunit@#2}}%
3272 }%
```

r@prevunitcount

```
3273 \newcommand*{\@glxstr@prevunitcount}[2]{%
3274   \ifcsundef
3275   {glo@\glstetoklabel{#1}@prevunit@#2}%
3276   {0}%
3277   {\csuse{glo@\glstetoklabel{#1}@prevunit@#2}}%
3278 }%
```

entryunitcount

```
3279 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3280   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3281   \renewcommand*{\glstetoknewglossaryentry}{%
3282     \renewcommand*\newglossaryentry[2]{%
3283       \PackageError{glossaries}{\string\newglossaryentry\space
3284         may only be used in the preamble when entry counting has
3285         been activated}{If you use \string\glsenableentryunitcount\space
3286         you must place all entry definitions in the preamble not in
3287         the document environment}%
3288     }%
3289   }%
```

New commands to access new fields:

```
3290 \newcommand*{\glstentrycurrcount}[1]{%
3291   \@glxstr@currunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3292   \csuse{glstetok@theunit@\glstgetattribute{##1}{unitcount}}}%
3293 }%
3294 \newcommand*{\glstentryprevcount}[1]{%
3295   \@glxstr@prevunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3296   \csuse{glstetok@theunit@\glstgetattribute{##1}{unitcount}}}%
3297 }%
```

Access total count:

```
3298 \newcommand*{\glstentryprevtotalcount}[1]{%
3299   \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
3300   {0}%
3301   {%
3302     \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
3303   }%
3304 }%
```

Access max value:

```

3305 \newcommand*{\glstentryprevmaxcount}[1]{%
3306   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
3307     {0}%
3308     {%
3309       \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
3310     }%
3311   }%

```

Adjust post unset and reset:

```

3312 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
3313 \renewcommand*{\glstxtrpostunset}[1]{%
3314   \@glstxtr@entryunitcount@org@unset{##1}%
3315   \@glst@increment@currunitcount{##1}%
3316 }%
3317 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
3318 \renewcommand*{\glstxtrpostlocalunset}[1]{%
3319   \@glstxtr@entryunitcount@org@localunset{##1}%
3320   \@glst@local@increment@currunitcount{##1}%
3321 }%
3322 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
3323 \renewcommand*{\glstxtrpostreset}[1]{%
3324   \glshasattribute{##1}{unitcount}%
3325   {%
3326     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3327     \ifcsundef{\@glstxtr@csname}%
3328       {}%
3329       {\csgdef{\@glstxtr@csname}{0}}%
3330   }%
3331   {}%
3332 }%
3333 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
3334 \renewcommand*{\glstxtrpostlocalreset}[1]{%
3335   \@glstxtr@entryunitcount@org@localreset{##1}%
3336   \glshasattribute{##1}{unitcount}%
3337   {%
3338     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3339     \ifcsundef{\@glstxtr@csname}%
3340       {}%
3341       {\csdef{\@glstxtr@csname}{0}}%
3342   }%
3343   {}%
3344 }%

```

Modifications to take into account the attributes that govern whether the entry should be unset.

```

3345 \let\@cglst@\@cglst@
3346 \let\@cglstpl@\@cglstpl@
3347 \let\@cGls@\@cGls@

```

```

3348 \let\@cGlspl@\@cGlspl@
3349 \let\@cGLS@\@cGLS@
3350 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3351 \AtEndDocument{\@gls@write@entryunitcounts}%
3352 \renewcommand*{\@gls@entry@unitcount}[3]{%
3353   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
3354   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
3355   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
3356   {%
3357     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
3358       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
3359     }%
3360     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
3361     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
3362     {%
3363       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
3364       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
3365       \fi
3366     }%
3367   }%
3368 \let\glsenableentryunitcount\relax
3369 \renewcommand*{\glsenableentrycount}{%
3370   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3371     can't be used with \string\glsenableentryunitcount}%
3372   {Use one or other but not both commands}%
3373 }%
3374 }
3375 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3376 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3377 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3378   \immediate\write\@auxout
3379   {\string\@gls@entry@unitcount
3380     {\@glsentry}%
3381     {\@glsxtr@currunitcount{\@glsentry}{#1}%
3382     }%
3383     {#1}}%
3384 }

```

entryunitcounts

```

3385 \newcommand*{\@gls@write@entryunitcounts}{%
3386   \immediate\write\@auxout
3387   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3388   \count@=0\relax

```

```

3389 \forallglentries{\@glentry}{%
3390   \glshasattribute{\@glentry}{unitcount}%
3391   {%
3392     \ifglused{\@glentry}%
3393     {%
3394       \forlistcsloop
3395         {\@glswrite@entryunitcounts@do}%
3396         {glo@\glsetoklabel{\@glentry}@unitlist}%
3397     }%
3398   }%
3399   \advance\count@ by \@ne
3400 }%
3401 {}%
3402 }%
3403 \ifnum\count@=0
3404   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3405     \MessageBreak with \string\glsenableentryunitcount\space but the
3406     \MessageBreak attribute ‘unitcount’ hasn’t
3407     \MessageBreak been assigned to any of the defined
3408     \MessageBreak entries}%
3409 \fi
3410 }

```

`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.

```

3411 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3412   \glsenableentryunitcount

```

Redefine `\gls` etc:

```

3413   \renewcommand*{\gls}{\cgl}%
3414   \renewcommand*{\Gls}{\cGls}%
3415   \renewcommand*{\glspl}{\cglspl}%
3416   \renewcommand*{\Glspl}{\cGlspl}%
3417   \renewcommand*{\GLS}{\cGLS}%
3418   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3419   \@glxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3420   \let\GlsXtrEnableEntryUnitCounting\@glxtr@setentryunitcountunsetattr
3421   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3422     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3423       can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3424     {Use one or other but not both commands}}%
3425 }

```

`tcountunsetattr`

```

3426 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3427   \@for\@glsxtr@cat:=#1\do
3428   {%
3429     \ifdefempty{\@glsxtr@cat}{}%
3430     {%
3431       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3432       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3433     }%
3434   }%
3435 }

```

### 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.

`\SetGenericNewAcronym`

```

3436 \renewcommand*{\SetGenericNewAcronym}{%
3437   \let\@Gls@entryname\@Gls@acrenryname
3438   \renewcommand{\newacronym}[4][]{%
3439     \ifdefempty{\@glsacronymlists}%
3440     {%
3441       \def\@glo@type{\acronymtype}%
3442       \setkeys{glossentry}{##1}%
3443       \DeclareAcronymList{\@glo@type}%
3444     }%
3445   }%
3446   \glskeylisttok{##1}%
3447   \glslabeltok{##2}%
3448   \glsshorttok{##3}%
3449   \glslongtok{##4}%
3450   \newacronymhook
3451   \protected@edef\@do@newglossaryentry{%
3452     \noexpand\newglossaryentry{\the\glslabeltok}%
3453     {%
3454       type=\acronymtype,%
3455       name={\expandonce{\acronymentry{##2}}},%
3456       sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%
3457       text={\the\glsshorttok},%
3458       short={\the\glsshorttok},%
3459       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3460       long={\the\glslongtok},%
3461       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3462       category=acronym,

```

```

3463     \GenericAcronymFields,%
3464     \the\glskeylisttok
3465   }%
3466 }%
3467 \do@newglossaryentry
3468 }%
3469 \renewcommand*{\acrfullfmt}[3]{%
3470   \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}}%
3471 \renewcommand*{\Acrfullfmt}[3]{%
3472   \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}}%
3473 \renewcommand*{\ACRfullfmt}[3]{%
3474   \glslink{##1}{##2}{%
3475     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3476 \renewcommand*{\acrfullplfmt}[3]{%
3477   \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
3478 \renewcommand*{\Acrfullplfmt}[3]{%
3479   \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
3480 \renewcommand*{\ACRfullplfmt}[3]{%
3481   \glslink{##1}{##2}{%
3482     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3483 \renewcommand*{\glstryfull}[1]{\genacrfullformat{##1}{}}}%
3484 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3485 \renewcommand*{\glstryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3486 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3487 }

```

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:

```

3488 \let\@glstr@org@setacronymstyle\setacronymstyle
3489 \let\@glstr@org@newacronymstyle\newacronymstyle

```

**msAbbreviations** Make acronyms use the same interface as abbreviations. Note that `\newacronymstyle` has a different implementation to `\newabbreviationstyle` so disable `\newacronymstyle` and `\setacronymstyle`.

```

3490 \newcommand*{\MakeAcronymsAbbreviations}{%
3491   \renewcommand*{\newacronym}[4][]{%
3492     \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3493   }%
3494   \renewcommand*{\firstacronymfont}[1]{\glstr@firstabbrvfont{##1}}%
3495   \renewcommand*{\acronymfont}[1]{\glstr@abbrvfont{##1}}%
3496   \renewcommand*{\setacronymstyle}[1]{%
3497     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3498     unavailable.
3499     Use \string\setabbreviationstyle\space instead.
3500     The original acronym interface can be restored with
3501     \string\RestoreAcronyms}{}%
3502   }%
3503   \renewcommand*{\newacronymstyle}[1]{%

```

```

3504     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3505     available unless you restore the original acronym interface with
3506     \string\RestoreAcronyms}%
3507     \@glxtr@org@newacronymstyle{##1}%
3508 }%
3509 }

```

Switch acronyms to abbreviations:

```
3510 \MakeAcronymsAbbreviations
```

**RestoreAcronyms** Restore acronyms to glossaries interface.

```

3511 \newcommand*{\RestoreAcronyms}{%
3512   \SetGenericNewAcronym
3513   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3514   \renewcommand{\acronymfont}[1]{##1}%
3515   \let\setacronymstyle\@glxtr@org@setacronymstyle
3516   \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.

```

3517   \renewcommand*\@gls@link@checkfirsthyper{%
3518     \ifglused{\glslabel}%
3519     {\let\glxtrifwasfirstuse\@secondoftwo}
3520     {\let\glxtrifwasfirstuse\@firstoftwo}%
3521     \@glxtr@org@checkfirsthyper
3522   }
3523   \glssetcategoryattribute{acronym}{regular}{false}%
3524   \setacronymstyle{long-short}%
3525 }

```

**\glsacspace** Allow the user to customise the maximum value.

```

3526 \renewcommand*{\glsacspace}[1]{%
3527   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3528   \ifdim\dimen@<\glsacspacemax~\else\space\fi
3529 }

```

**\glsacspacemax** Value used in the above.

```
3530 \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**

```
3531 \newcommand*{\@glxtr@reg@glosslist}{}
```



Save the original definition of `\makeglossaries`:

```
3532 \let\@glxtr@org@makeglossaries\makeglossaries
```

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`

```
3533 \renewcommand*{\makeglossaries}[1][]{%
3534   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3535     \PackageError{glossaries-extra}{\string\makeglossaries\space
3536       not permitted\MessageBreak with record=only package option}%
3537     {You may only use \string\makeglossaries\space with
3538       record=off or record=alsoindex options}%
3539   \else
3540     \ifblank{#1}%
3541     {\@glxtr@org@makeglossaries}%
3542     {%
3543       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3544         \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3545           not permitted\MessageBreak with record=alsoindex package option}%
3546         {You may only use the hybrid \string\makeglossaries[...]\space with
3547           record=off option}%
3548       \else
3549         \edef\@glxtr@reg@glosslist{#1}%
3550         \ifundef{\glswrite}{\newwrite\glswrite}{}%
3551         \protected@write\@auxout{}{\string\providecommand
3552           \string\@glxtr@order[1]{}}
3553         \protected@write\@auxout{}{\string\providecommand
3554           \string\@istfilename[1]{}}
3555         \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3556         \protected@write\@auxout{}{\string\@glxtr@order{\glxtr@order}}
3557         \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}
3558         \write\@auxout{\string\providecommand\string\@glxtr@reference[3]{}}%
3559       }%
3560     }%
3561   }
```

Iterate through each supplied glossary type and activate it.

```
3559   \@for\@glo@type:=#1\do{%
3560     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3561   }
```

New glossaries must be created before `\makeglossaries`:

```
3562   \renewcommand*\newglossary[4][]{%
3563     \PackageError{glossaries}{New glossaries
3564       must be created before \string\makeglossaries}{You need
3565       to move \string\makeglossaries\space after all your
3566       \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
3567   \let\@makeglossary\relax
3568   \let\makeglossary\relax
```

```

3569      \renewcommand\makeglossaries[1][{}]{%
Disable all commands that have no effect after \makeglossaries
3570      \@disable@onlypremakeg
Allow see key:
3571      \let\gls@checkseeallowed\relax
Adjust \@do@seeglossary. This needs to check for the entries existence.
3572      \renewcommand*{\@do@seeglossary}[2]{%
3573      \glsdoifexists{##1}%
3574      {%
3575      \edef\gls@label{\glsdetoklabel{##1}}%
3576      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3577      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3578      {\glsxtr@org@doseeglossary{##1}{##2}}%
3579      {%
3580      @@glsxtrwrglossmark
3581      \protected@write\@auxout{}{%
3582      \string\gls@reference
3583      {\gls@type}{\gls@label}{\string\glsseeformat##2{}}%
3584      }%
3585      }%
3586      }%
3587      }%
Adjust \@do@@wrglossary
3588      \let\glsxtr@@do@@wrglossary\@do@@wrglossary
3589      \def\@do@@wrglossary{%
3590      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3591      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3592      {\glsxtr@@do@@wrglossary}%
3593      {\gls@noidxglossary}%
3594      }%
Suppress warning about no \makeglossaries
3595      \let\warn@nomakeglossaries\relax
3596      \def\warn@noprintglossary{%
3597      \GlossariesWarningNoLine{No \string\printglossary\space
3598      or \string\printglossaries\space
3599      found.^^J(Remove \string\makeglossaries\space if you don't want
3600      any glossaries.)^^JThis document will not have a glossary}%
3601      }%
Only warn for glossaries not listed.
3602      \renewcommand{\@gls@noref@warn}[1]{%
3603      \edef\gls@type{##1}%
3604      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3605      {%
3606      \GlossariesExtraWarning{Can't use
3607      \string\printnoidxglossary[type={\gls@type}]
3608      when '\gls@type' is listed in the optional argument of

```

```

3609         \string\makeglossaries}%
3610     }%
3611     {%
3612         \GlossariesWarning{Empty glossary for
3613         \string\printnoidxglossary[type={##1}].
3614         Rerun may be required (or you may have forgotten to use
3615         commands like \string\gls)}%
3616     }%
3617 }%

```

Adjust display number list to check for type:

```

3618     \renewcommand*{\glsdisplaynumberlist}[1]{%
3619     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3620     {\@glxtr@idx@displaynumberlist{##1}}%
3621     {\@glxtr@noidx@displaynumberlist{##1}}%
3622 }%

```

Adjust entry list:

```

3623     \renewcommand*{\glsentrynumberlist}[1]{%
3624     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3625     {\@glxtr@idx@entrynumberlist{##1}}%
3626     {\@glxtr@noidx@entrynumberlist{##1}}%
3627 }%

```

Adjust number list loop

```

3628     \renewcommand*{\glsnumberlistloop}[2]{%
3629     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3630     {%
3631         \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3632         not available for glossary ‘##1’}{}%
3633     }%
3634     {\@glxtr@noidx@numberlistloop{##1}{##2}}%
3635 }%

```

Only sanitize sort for normal indexing glossaries.

```

3636     \renewcommand*{\glsprestandardsort}[3]{%
3637     \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
3638     {%
3639         \glsdosanitizesort
3640     }%
3641     {%
3642         \ifglssanitizesort
3643         \@gls@noidx@sanitizesort
3644         \else
3645         \@gls@noidx@nosanitizesort
3646         \fi
3647     }%
3648 }%

```

Unlike `\makenoidxglossaries` we can't automatically set `sanitizesort=false`. All entries must be defined in the preamble.

```

3649 \renewcommand*\new@glossaryentry[2]{%
3650 \PackageError{glossaries-extra}{Glossary entries must be defined
3651 in the preamble\MessageBreak when you use the optional argument
3652 of \string\makeglossaries}{Either move your definitions to the
3653 preamble or don't use the optional argument of
3654 \string\makeglossaries}%
3655 }%

```

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).

```

3656 \let\@glo@assign@sortkey\@glstr@mixed@assign@sortkey
3657 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3658 \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
3659 type=\glsdefaulttype,\@end@glstr@gettype
3660 \def\@glo@sorttype{\@glo@default@sorttype}%
3661 }%

```

Check automake setting:

```

3662 \ifglsautomake
3663 \renewcommand*\@gls@doautomake{%
3664 \@for\@gls@type:=\@glstr@reg@glosslist\do{%
3665 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3666 }%
3667 }%
3668 \fi

```

Check the sort setting (glossaries v4.30 onwards):

```

3669 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3670 \fi
3671 }%
3672 \fi
3673 }

```

The optional argument version of `\makeglossaries` needs an adjustment to `\@printglossary` to allow `\@glo@assign@sortkey` to pick up the glossary type.

`\@printglossary` This no longer simply saves `\@printglossary` with `\let` but is actually defined to check for the existence of the title, since ignored glossaries don't have a title assigned. (bib2gls writes `\provideignoredglossary` to the `glstex` file for some settings, so the glossary might not have been defined.) (This command is also used for on-the-fly setting.)

```

3674 \newcommand{\@glstr@orgprintglossary}[2]{%
3675 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3676 \def\glossarytitle{%
3677 \ifcsdef{\@glo@type@\@glo@type @title}%
3678 {\csuse{\@glo@type@\@glo@type @title}}%
3679 {\glossaryname}}%
3680 \def\glossarytoctitle{\glossarytitle}%

```

```

3681 \let\org@glossarytitle\glossarytitle
3682 \def\@glossarystyle{%
3683   \ifx\@glossary@default@style\relax
3684     \GlossariesWarning{No default glossary style provided \MessageBreak
3685       for the glossary '\@glo@type'. \MessageBreak
3686       Using deprecated fallback. \MessageBreak
3687       To fix this set the style with \MessageBreak
3688       \string\setglossarystyle\space or use the \MessageBreak
3689       style key=value option}%
3690   \fi
3691 }%
3692 \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
3693 \let\org@glossaryentrynumbers\glossaryentrynumbers
3694 \bgroup
3695   \@printgloss@setsort
3696   \setkeys{printgloss}{#1}%
3697   \ifx\glossarytitle\org@glossarytitle
3698   \else
3699     \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3700   \fi
3701   \let\currentglossary\@glo@type
3702   \let\org@glossaryentrynumbers\glossaryentrynumbers
3703   \let\glsnonextpages\@glsnonextpages
3704   \let\glsnextpages\@glsnextpages

3705   \glsxtractivatenopost
3706   \gls@dotoc@title
3707   \@glossarystyle
3708   \let\gls@org@glossaryentryfield\glossentry
3709   \let\gls@org@glossarysubentryfield\subglossentry
3710   \renewcommand{\glossentry}[1]{%
3711     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3712     \gls@org@glossaryentryfield{##1}%
3713   }%
3714   \renewcommand{\subglossentry}[2]{%
3715     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3716     \gls@org@glossarysubentryfield{##1}{##2}%
3717   }%
3718   \@gls@preglossaryhook
3719   #2%
3720 \egroup
3721 \global\let\glossaryentrynumbers\org@glossaryentrynumbers
3722 \global\let\warn@noprntglossary\relax
3723 }

```

ractivatenopost Change \nopostdesc and \glsxtrnopostpunc to behave as they do in the glossary.

```

3724 \newcommand*\glsxtractivatenopost{%
3725   \let\nopostdesc\@nopostdesc
3726   \let\glsxtrnopostpunc\@glsxtrnopostpunc
3727 }

```

lsxtrnopostpunc

```
3728 \newrobustcmd*{\glxtrnopostpunc}{}
```

lsxtr@nopostpunc Provide a command that works like \nopostdesc but only switches of the punctuation without suppressing the post-description hook.

```
3729 \newcommand{\@glxtr@nopostpunc}{%
3730 \let\@glxtr@org@postdescription\glspostdescription
3731 \ifglsnopostdot
3732 \renewcommand{\glspostdescription}{%
3733 \glsnopostdottrue
3734 \let\glspostdescription\@glxtr@org@postdescription
3735 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3736 \glxtrpostdescription
3737 \@glxtr@nopostpunc@postdesc}%
3738 \else
3739 \renewcommand{\glspostdescription}{%
3740 \let\glspostdescription\@glxtr@org@postdescription
3741 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3742 \glxtrpostdescription
3743 \@glxtr@nopostpunc@postdesc}%
3744 \fi
3745 \glsnopostdotfalse
3746 }
```

stpunc@postdesc

```
3747 \newcommand*{\@glxtr@nopostpunc@postdesc}{}
```

restore@postpunc

```
3748 \newcommand*{\@glxtr@restore@postpunc}{%
3749 \def\@glxtr@nopostpunc@postdesc{%
3750 \@glxtr@org@postdescription
3751 \let\@glxtr@nopostpunc@postdesc\@empty
3752 \let\glxtrrestorepostpunc\@empty
3753 }%
3754 }
```

restorepostpunc Does nothing outside of glossary.

```
3755 \newcommand*{\glxtrrestorepostpunc}{}
```

\@printglossary Redefine.

```
3756 \renewcommand{\@printglossary}[2]{%
3757 \def\@glxtr@printglossopts{#1}%
3758 \@glxtr@orgprintglossary{#1}{#2}%
3759 }
```

Add a key that switches off the entry targets:

```
3760 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3761 \ifcase\nr
```

```

3762 \let\@glstarget\glsdohypertarget
3763 \else
3764 \let\@glstarget\@secondoftwo
3765 \fi
3766 }

```

hypernameprefix

```

3767 \newcommand{\@glxtrhypernameprefix}{}

```

New to v1.20:

```

3768 \define@key{printgloss}{targetnameprefix}{%
3769 \renewcommand{\@glxtrhypernameprefix}{#1}%
3770 }

```

glsdohypertarget Redefine to insert \@glxtrhypernameprefix before the target name.

```

3771 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3772 \renewcommand{\glsdohypertarget}[2]{%
3773 \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3774 }

```

@makeglossaries For the benefit of makeglossaries

```

3775 \newcommand*{\@glxtr@makeglossaries}[1]{}

```

@glxtr@gettype Get just the type.

```

3776 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3777 \def\@glo@type{#2}%
3778 }

```

@assign@sortkey Assign the sort key.

```

3779 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3780 \edef\@glo@type{\@glo@type}%
3781 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3782 {%
3783 \@glo@no@assign@sortkey{#1}%
3784 }%
3785 {%
3786 \@glo@assign@sortkey{#1}%
3787 }%
3788 }%

```

Display number list for the regular version:

displaynumberlist

```

3789 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3790 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
3791   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3792   \ifdef\@gls@loclist
3793   {%
3794     \def\@gls@noidxloclist@sep{%
3795       \def\@gls@noidxloclist@sep{%
3796         \def\@gls@noidxloclist@sep{%
3797           \glsnumlistsep
3798         }%
3799         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3800       }%
3801     }%
3802     \def\@gls@noidxloclist@finalsep{}%
3803     \def\@gls@noidxloclist@prev{}%
3804     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3805     \@gls@noidxloclist@finalsep
3806     \@gls@noidxloclist@prev
3807   }%
3808   {%

3809   \glsxtrundeftag
3810   \glsdoifexists{#1}%
3811   {%
3812     \GlossariesWarning{Missing location list for ‘#1’. Either
3813       a rerun is required or you haven’t referenced the entry.}%
3814   }%
3815 }%
3816 }%
3817
```

And for the number list loop:

@numberlistloop

```

3818 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3819   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3820   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3821   \let\@gls@org@glsseeformat\glsseeformat
3822   \let\glsnoidxdisplayloc#2\relax
3823   \let\glsseeformat#3\relax
3824   \ifdef\@gls@loclist
3825   {%
3826     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3827   }%
3828   {%

3829   \glsxtrundeftag
3830   \glsdoifexists{#1}%
3831   {%
3832     \GlossariesWarning{Missing location list for ‘##1’. Either
```



```

3833         a rerun is required or you haven't referenced the entry.}%
3834     }%
3835 }%
3836 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3837 \let\glsseeformat\@gls@org@glsseeformat
3838 }%

```

Same for entry number list.

entrynumberlist

```

3839 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3840   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3841   \ifdef\@gls@loclist
3842   {%
3843     \glsnoidxloclist{\@gls@loclist}%
3844   }%
3845   {%
3846     \glsxtrundeftag
3847     \glsdoifexists{#1}%
3848     {%
3849       \GlossariesWarning{Missing location list for '#1'. Either
3850         a rerun is required or you haven't referenced the entry.}%
3851     }%
3852   }%
3853 }%

```

entrynumberlist

```

3854 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3855 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3856   \protected@edef\@glsxtr@titlelabel{#1}%
3857   \ifdefvoid\@glsxtr@titlelabel
3858   {%
3859   {%
3860     \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle{#1}}}%
3861   }%
3862   \ifdefvoid{\@glsxtr@titlelabel}%
3863   {%
3864     \DTLifint{#1}%
3865     {%
3866       \ifnum#1<256\relax
3867       \edef#2{\char#1\relax}%
3868     \else
3869     \edef#2{#1}%
3870     \fi
3871   }%
3872   {%
3873     \ifcsundef{#1groupname}%

```

```

3874      {\def#2{#1}}%
3875      {\letcs#2{#1groupname}}}%
3876  }%
3877 }%
3878 {%
3879   \let#2\@glsxtr@titlelabel
3880 }%
3881 }

```

**g@getgrouptitle** Save original definition of \@gls@getgrouptitle

```

3882 \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.

```

3883 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3884   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3885   \@onelevel@sanitize\@glsxtr@titlelabel
3886   \ifcsdef{\@glsxtr@titlelabel}
3887     {\letcs{#2}{\@glsxtr@titlelabel}}%
3888     {\glsxtr@org@getgrouptitle{#1}{#2}}%
3889 }
3890 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

**trsetgrouptitle** Sets the title for the given group label.

```

3891 \newcommand{\glsxtrsetgrouptitle}[2]{%
3892   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3893   \@onelevel@sanitize\@glsxtr@titlelabel
3894   \csxdef{\@glsxtr@titlelabel}{#2}%
3895 }

```

**alsetgrouptitle** As above put only locally defines the title.

```

3896 \newcommand{\glsxtrlocalsetgrouptitle}[2]{%
3897   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3898   \@onelevel@sanitize\@glsxtr@titlelabel
3899   \csedef{\@glsxtr@titlelabel}{#2}%
3900 }

```

**\glsnavigation** Redefine to use new user-level command.

```

3901 \renewcommand*{\glsnavigation}{%
3902   \def\@gls@between{}%
3903   \ifcsundef{\@gls@hypergroupelist@\@glo@type}%
3904   {%
3905     \def\@gls@list{}%
3906   }%
3907   {%
3908     \expandafter\let\expandafter\@gls@list
3909       \csname \@gls@hypergroupelist@\@glo@type\endcsname
3910   }%

```

```

3911 \@for\@gls@tmp:=\@gls@list\do{%
3912   \@gls@between
3913   \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3914   \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3915   \let\@gls@between\glshypernavsep
3916 }%
3917 }

```

@noidx@glossary

```

3918 \renewcommand*{\@print@noidx@glossary}{%
3919   \ifcsdef{\glsref@{\@glo@type}%
3920   {%
3921     \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
3922     {%
3923       \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
3924     }%
3925     {%
3926       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3927     }%
3928     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3929     \glossarypreamble

```

Moved this command definition outside of environment in case of scoping issues (e.g. in tabular-like styles).

```

3930   \def\@gls@currentlettergroup{}%
3931   \begin{theglossary}%
3932   \glossaryheader
3933   \glsresetentrylist
3934   \forlistcsloop{\@gls@noidx@do}{\@glsref@{\@glo@type}%
3935   \end{theglossary}%
3936   \glossarypostamble
3937 }%
3938 {%

```

Add section header if there are actually entries defined in this glossary as the document is likely pending a re-run.

```

3939   \glsxtrifemptyglossary{\@glo@type}%
3940   }%
3941   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3942   \@gls@noref@warn{\@glo@type}%
3943 }%
3944 }

```

noidxdisplayloc Patch to check for range formations.

```

3945 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3946   \setentrycounter[#1]{#2}%
3947   \@glsxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
3948 }

```

tr@display@loc Patch to check for range formations.

```
3949 \def\@glxtr@display@loc#1#2\end@glxtr@display@loc#3{%
3950   \ifx#1\relax
3951     \glxtrdisplaystartloc{#2}{#3}%
3952   \else
3953     \ifx#1)\relax
3954       \glxtrdisplayendloc{#2}{#3}%
3955     \else
3956       \glxtrdisplaysingleloc{#1#2}{#3}%
3957     \fi
3958   \fi
3959 }
```

isplaysingleloc Single location.

```
3960 \newcommand*\glxtrdisplaysingleloc}[2]{%
3961   \csuse{#1}{#2}%
3962 }
```

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 `\glxtrlocrangefmt`.

displaystartloc Start of a location range.

```
3963 \newcommand*\glxtrdisplaystartloc}[2]{%
3964   \edef\glxtrlocrangefmt{#1}%
3965   \ifx\glxtrlocrangefmt\empty
3966     \def\glxtrlocrangefmt{\glsnnumberformat}%
3967   \fi
3968   \expandafter\glxtrdisplaysingleloc
3969   \expandafter{\glxtrlocrangefmt}{#2}%
3970 }
```

trdisplayendloc End of a location range.

```
3971 \newcommand*\glxtrdisplayendloc}[2]{%
3972   \edef\@glxtr@tmp{#1}%
3973   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{\glsnnumberformat}}{}}%
3974   \ifx\glxtrlocrangefmt\@glxtr@tmp
3975   \else
3976     \GlossariesExtraWarning{Mismatched end location range
3977       (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3978   \fi
3979   \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3980   \expandafter\glxtrdisplaysingleloc
3981   \expandafter{\glxtrlocrangefmt}{#2}%
3982   \def\glxtrlocrangefmt{}%
3983 }
```

isplayendloohook Allow the user to hook into the end of range command.

```
3984 \newcommand*\glxtrdisplayendloohook}[2]{}
```

lsxtrlocrangefmt Current range format. Empty if not in a range.

```
3985 \newcommand*{\glsxtrlocrangefmt}{}%
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
3986 \def\@gls@removespaces#1 #2\@nil{%
3987 \toks@=\expandafter{\the\toks@#1}%
3988 \ifx\#2\%
3989 \edef\x{\the\toks@}%
3990 \ifx\x\empty
3991 \else
3992 \glsxtrlocationhyperlink{\glstrycounter}{\@glo@counterprefix}{\the\toks@}%
3993 \fi
3994 \else
3995 \@gls@ReturnAfterFi{%
3996 \@gls@removespaces#2\@nil
3997 }%
3998 \fi
3999 }
```

locationhyperlink

```
4000 \newcommand*{\glsxtrlocationhyperlink}[3]{%
4001 \ifdefined\glxtrsupplocationurl
4002 {%
4003 \glxtrhyperlink{#1#2#3}{#3}%
4004 }%
4005 {%
4006 \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
4007 }%
4008 }
```

supphypernumber

```
4009 \newcommand*{\glxtrsupphypernumber}[1]{%
4010 {%
4011 \glshasattribute{\glscurrententrylabel}{externallocation}%
4012 {%
4013 \def\glxtrsupplocationurl{%
4014 \glsggetattribute{\glscurrententrylabel}{externallocation}}%
4015 }%
4016 {%
4017 \def\glxtrsupplocationurl{}%
4018 }%
4019 \glshypernumber{#1}%
4020 }%
4021 }
```

Give a bit of assistance to new users who are confused and don't know how to read transcript messages.

@print@glossary

```

4022 \renewcommand{\@print@glossary}{%
4023   \makeatletter
4024   \@input@{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4025   \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4026   {}%
4027   {\glstrNoGlossaryWarning{\@glo@type}}%
4028   \ifglxindy
4029     \ifcsundef{@xdy@\@glo@type @language}%
4030     {%
4031       \edef\@do@auxoutstuff{%
4032         \noexpand\AtEndDocument{%
4033           \noexpand\immediate\noexpand\write\@auxout{%
4034             \string\providecommand\string\@xdylanguage[2]{}}%
4035           \noexpand\immediate\noexpand\write\@auxout{%
4036             \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4037         }%
4038       }%
4039     }%
4040     {%
4041       \edef\@do@auxoutstuff{%
4042         \noexpand\AtEndDocument{%
4043           \noexpand\immediate\noexpand\write\@auxout{%
4044             \string\providecommand\string\@xdylanguage[2]{}}%
4045           \noexpand\immediate\noexpand\write\@auxout{%
4046             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4047               @language\endcsname}}%
4048         }%
4049       }%
4050     }%
4051     \@do@auxoutstuff
4052     \edef\@do@auxoutstuff{%
4053       \noexpand\AtEndDocument{%
4054         \noexpand\immediate\noexpand\write\@auxout{%
4055           \string\providecommand\string\@gls@codepage[2]{}}%
4056         \noexpand\immediate\noexpand\write\@auxout{%
4057           \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4058       }%
4059     }%
4060     \@do@auxoutstuff
4061   \fi
4062   \renewcommand*{\@warn@nomakeglossaries}{%
4063     \GlossariesWarningNoLine{\string\makeglossaries\space
4064       hasn't been used,^^Jthe glossaries will not be updated}%
4065   }%
4066 }

```

Setup the warning text to display if the external file for the given glossary is missing.

`\oGlsWarningHead` Header message.

```

4067 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%

```

```

4068 This document is incomplete. The external file associated with
4069 the glossary ‘#1’ (which should be called \texttt{#2})
4070 hasn’t been created.%
4071 }

arningEmptyStart No entries have been added to the glossary.
4072 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4073 This has probably happened because there are no entries defined
4074 in this glossary.%
4075 }

arningEmptyMain The default “main” glossary is empty.
4076 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4077 If you don’t want this glossary,
4078 add \texttt{nomain} to your package option list when you load
4079 \texttt{glossaries-extra.sty}. For example:%
4080 }

ingEmptyNotMain A glossary that isn’t the default “main” glossary is empty.
4081 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4082 Did you forget to use \texttt{type=#1} when you defined your
4083 entries? If you tried to load entries into this glossary with
4084 \texttt{\string\loadglsentries} did you remember to use
4085 \texttt{[#1]} as the optional argument? If you did, check that
4086 the definitions in the file you loaded all had the type set
4087 to \texttt{\string\glsdefaulttype}.%
4088 }

arningCheckFile Advisory message to check the file contents.
4089 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4090 Check the contents of the file \texttt{#1}. If
4091 it’s empty, that means you haven’t indexed any of your entries in this
4092 glossary (using commands like \texttt{\string\gls} or
4093 \texttt{\string\glsadd}) so this list can’t be generated.
4094 If the file isn’t empty, the document build process hasn’t been
4095 completed.%
4096 }

WarningAutoMake Message when automake option has been used.
4097 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4098 You may need to rerun \LaTeX. If you already have, it may be that
4099 \TeX’s shell escape doesn’t allow you to run
4100 \ifglxindy xindy\else makeindex\fi. Check the
4101 transcript file \texttt{\jobname.log}. If the shell escape is
4102 disabled, try one of the following:
4103
4104 \begin{itemize}
4105 \item Run the external (Lua) application:
4106

```

```

4107      \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4108
4109      \item Run the external (Perl) application:
4110
4111      \texttt{makeglossaries \string"\jobname\string"}
4112 \end{itemize}
4113
4114 Then rerun \LaTeX\ on this document.
4115 \GlossariesExtraWarning{Rerun required to build the
4116 glossary ‘#1’ or check TeX’s shell escape allows
4117 you to run \ifglxindy xindy\else makeindex\fi}%
4118 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4119 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
4120 You need to either replace \texttt{\string\makenoidxglossaries}
4121 with \texttt{\string\makeglossaries} or replace
4122 \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4123 \texttt{\string\printnoidxglossary}
4124 (or \texttt{\string\printnoidxglossaries}) and then rebuild
4125 this document.%
4126 }

```

WarningBuildInfo Build advice.

```

4127 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4128 Try one of the following:
4129 \begin{itemize}
4130 \item Add \texttt{automake} to your package option list when you load
4131 \texttt{glossaries-extra.sty}. For example:
4132
4133 \texttt{\string\usepackage[automake]%
4134 \glsopenbrace glossaries-extra\glsclosebrace}
4135
4136 \item Run the external (Lua) application:
4137
4138 \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4139
4140 \item Run the external (Perl) application:
4141
4142 \texttt{makeglossaries \string"\jobname\string"}
4143 \end{itemize}
4144
4145 Then rerun \LaTeX\ on this document.%
4146 }

```

oGlsWarningTail Final paragraph.

```

4147 \newcommand{\GlsXtrNoGlsWarningTail}{%
4148 This message will be removed once the problem has been fixed.%
4149 }

```



GlsWarningNoOut No out file created. Build advice.

```
4150 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4151   The file \texttt{#1} doesn't exist. This most likely means you haven't used
4152   \texttt{\string\makeglossaries} or you have used
4153   \texttt{\string\nofiles}. If this is just a draft version of the
4154   document, you can suppress this message using the
4155   \texttt{\nomissingglstext} package option.%
4156 }
```

glossarywarning

```
4157 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
4158   \glossarysection[\glossarytoctitle]{\glossarytitle}
4159   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
4160   \par
4161   \glstrifemptyglossary{#1}%
4162   {%
4163     \GlsXtrNoGlsWarningEmptyStart\space
4164     \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4165     \medskip
4166     \noindent\texttt{\string\usepackage[nomain\ifglacronym ,acronym\fi]%
4167       \glsoopenbrace glossaries-extra\glsclosebrace}
4168     \medskip
4169     }%
4170     {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4171   }%
4172   {%
4173     \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
4174     {%
4175       \GlsXtrNoGlsWarningCheckFile
4176       {\jobname.\csname @glotype@\@glo@type @out\endcsname}
4177
4178       \ifglautomake
4179
4180       \GlsXtrNoGlsWarningAutoMake{#1}
4181
4182       \else
4183
4184       \ifthenelse{\equal{#1}{main}}{%
4185         {%
4186           \GlsXtrNoGlsWarningEmptyMain\par
4187           \medskip
4188           \noindent\texttt{\string\usepackage[nomain]%
4189             \glsoopenbrace glossaries-extra\glsclosebrace}
4190           \medskip
4191         }%
4192       }%
4193
4194       \ifdequal\makeglossaries\@no@makeglossaries
4195       {%
```

```

4196         \GlsXtrNoGlsWarningMisMatch
4197     }%
4198     {%
4199         \GlsXtrNoGlsWarningBuildInfo
4200     }%
4201     \fi
4202 }%
4203 {%
4204     \GlsXtrNoGlsWarningNoOut
4205     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4206 }%
4207 }%
4208 \par
4209 \GlsXtrNoGlsWarningTail
4210 }

```

Provide some commands to accompany the record option for use with **bib2gls**.

**glstrresourcefile** 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.

```
4211 \newcommand*{\glstrresourcefile}[2] [] {%
```

The record option can't be set after this command.

```

4212     \disable@keys{glossaries-extra.sty}{record}%
4213     \glstr@writefields
4214     \protected@write\@auxout{\glstrresourceinit}{\string\glstr@resource{#1}{#2}}%
4215     \let\@glstr@org@see@noindex\@glstr@see@noindex
4216     \let\@glstr@see@noindex\relax
4217     \IfFileExists{#2.glstex}%
4218     {%

```

Can't scope \@input so save and restore the category code of @ to allow for internal commands in the location list.

```

4219     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4220     \makeatletter
4221     \@input{#2.glstex}%
4222     \@bibgls@restreat
4223 }%
4224 {%
4225     \GlossariesExtraWarning{No file '#2.glstex'}%
4226 }%
4227     \let\@glstr@see@noindex\@glstr@org@see@noindex
4228 }
4229 \@onlypreamble\glstrresourcefile

```

**glstrresourcecount**

```
4230 \newcount\glstrresourcecount
```

**glstrLoadResources** Short cut that uses \glstrresourcefile with \jobname as the mandatory argument.

```
4231 \newcommand*{\GlsXtrLoadResources}[1] [] {%
```

```

4232 \ifnum\glxtrresourcecount=0\relax
4233   \glxtrresourcefile[#1]{\jobname}%
4234 \else
4235   \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%
4236 \fi
4237 \advance\glxtrresourcecount by 1\relax
4238 }

glxtr@resource
4239 \newcommand*{\glxtr@resource}[2]{%

\glxtr@fields
4240 \newcommand*{\glxtr@fields}[1]{%

xtr@texencoding
4241 \newcommand*{\glxtr@texencoding}[1]{%

\glxtr@langtag
4242 \newcommand*{\glxtr@langtag}[1]{%

@pluralsuffixes
4243 \newcommand*{\glxtr@pluralsuffixes}[4]{%

tr@shortcutsval
4244 \newcommand*{\glxtr@shortcutsval}[1]{%

sxtr@linkprefix
4245 \newcommand*{\glxtr@linkprefix}[1]{%

xtr@writefields This information only needs to be written once, so disable it after it's been used.
4246 \newcommand*{\glxtr@writefields}{%

4247 \protected@write\@auxout{}%
4248   {\string\providecommand*{\string\glxtr@fields}[1]{}}%
4249 \protected@write\@auxout{}%
4250   {\string\providecommand*{\string\glxtr@resource}[2]{}}%
4251 \protected@write\@auxout{}%
4252   {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
4253 \protected@write\@auxout{}%
4254   {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
4255 \protected@write\@auxout{}%
4256   {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
4257 \protected@write\@auxout{ }\string\glxtr@fields{\@gls@keymap}}%

4258 \protected@write\@auxout{}%
4259   {\string\providecommand*{\string\glxtr@record}[5]{}}%

```

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 indicated in the sort key when using `\glxtrresourcefile`.

```

4260 \ifdef\CurrentTrackedLanguageTag
4261 {%
4262     \protected@write\@auxout{}\{%
4263         \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
4264     }%
4265 {%
4266     \protected@write\@auxout{}\{\string\glxtr@pluralsuffixes
4267         {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
4268         {\glxtrabbrvpluralsuffix}}%
4269     \ifdef\inputencodingname
4270     {%
4271         \protected@write\@auxout{}\{\string\glxtr@texencoding{\inputencodingname}}%
4272     }%
4273     {%

```

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.)

```

4274     \@ifpackageloaded{fontspec}%
4275     {\protected@write\@auxout{}\{\string\glxtr@texencoding{utf8}}}%
4276     {%
4277 }%
4278 \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.

```

4279 \AtBeginDocument
4280 {\protected@write\@auxout{}\{\string\glxtr@linkprefix{\glolinkprefix}}}%
4281 \let\glxtr@writefields\relax

```

If the `automake` option is on, try running `bib2gls` if the aux file exists. The double-quotes around `\jobname` have been removed (v1.19) since `\jobname` will include double-quotes if the file name has spaces.

```

4282 \ifglautomake
4283 \IfFileExists{\jobname.aux}%
4284 {\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`.

```

4285 \ifx\gls@doautomake\gls@doautomake@err
4286 \let\gls@doautomake\relax
4287 \fi
4288 \fi
4289 }

```

`do@automake@err`

```

4290 \newcommand*{\@gls@doautomake@err}{%
4291 \PackageError{glossaries}{You must use

```

```

4292 \string\makeglossaries\space with automake=true}
4293 {%
4294     Either remove the automake=true setting or
4295     add \string\makeglossaries\space to your document preamble.%
4296 }%
4297 }

```

Allow locations specific to a particular counter to be recorded.

\glxtr@record

```

4298 \newcommand*{\glxtr@record}[5]{}

```

r@counterrecord Aux file command.

```

4299 \newcommand*{\glxtr@counterrecord}[3]{%
4300   \glxtrfieldlistgadd{#1}{record.#2}{#3}%
4301 }

```

unterrecordhook Hook used by \@glxtr@dorecord.

```

4302 \newcommand*{\@glxtr@counterrecordhook}{}

```

trRecordCounter Activate recording for a particular counter (identified in the argument).

```

4303 \newcommand*{\GlsXtrRecordCounter}[1]{%
4304   \@glxtr@recordcounter{#1}%
4305 }
4306 \@onlypreamble\GlsXtrRecordCounter

```

docounterrecord

```

4307 \newcommand*{\@glxtr@docounterrecord}[1]{%
4308   \protected@write\@auxout{}\string\glxtr@counterrecord
4309     {\@glx@label}{#1}{\csuse{the#1}}}%
4310 }

```

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.

```

4311 \newcommand*{\glxtrglossentry}[1]{%
4312   \glxtrtitleorpdforheading
4313   {\@glxtrglossentry{#1}}%
4314   {\glsentryname{#1}}%
4315   {\glxtrheadname{#1}}%
4316 }

```

lsxtrglossentry Another test is needed in case \@glxtrglossentry has been written to the table of contents.

```

4317 \newrobustcmd*{\@glxtrglossentry}[1]{%

```

```

4318 \glstrtitleorpdforheading
4319 {%
4320   \glsoifexists{#1}%
4321   {%
4322     \beginngroup
4323     \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4324     \edef\currentglossary{\glsetentrytype{\glscurrententrylabel}}%
4325     \ifglshasparent{#1}%
4326     {\glssubentryitem{#1}}%
4327     {\glsetentryitem{#1}}%
4328     \glstarget{#1}{\glossentryname{#1}}%
4329   \endgroup
4330   }%
4331 }%
4332 {\glsetentryname{#1}}%
4333 {\glstrheadname{#1}}%
4334 }

```

`glossentryother` As `\glstrglossentry` but uses a different field. First argument is command 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.

```

4335 \newcommand*{\glstrglossentryother}[3]{%
4336   \ifstrempy{#1}%
4337   {%
4338     \ifcsdef{glstrhead#3}%
4339     {%
4340       \glstrtitleorpdforheading
4341       {\@glstrglossentryother{#2}{#3}{#1}}%
4342       {\@gls@entry@field{#2}{#3}}%
4343       {\csuse{glstrhead#3}{#2}}%
4344     }%
4345     {%
4346       \glstrtitleorpdforheading
4347       {\@glstrglossentryother{#2}{#3}{#1}}%
4348       {\@gls@entry@field{#2}{#3}}%
4349       {\@gls@entry@field{NoCaseChange{#2}}{#3}}%
4350     }%
4351   }%
4352   {%
4353     \glstrtitleorpdforheading
4354     {\@glstrglossentryother{#2}{#3}{#1}}%
4355     {\@gls@entry@field{#2}{#3}}%
4356     {#1}%
4357   }%
4358 }

```

`glossentryother` As `\@glstrglossentry` but uses a different field.

```

4359 \newrobustcmd*{\@glstrglossentryother}[3]{%

```

```

4360 \glstrtitleorpdforheading
4361 {%
4362   \glsdoifexists{#1}%
4363   {%
4364     \beginngroup
4365       \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4366       \edef\currentglossary{\glssentrytype{\glscurrententrylabel}}%
4367       \ifglshasparent{#1}%
4368         {\glssubentryitem{#1}}%
4369         {\glssentryitem{#1}}%
4370       \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4371     \endngroup
4372   }%
4373 }%
4374 {\@gls@entry@field{#1}{#2}}%
4375 {#3}%
4376 }

```

`\printunsrtglossary` Similar to `\printnoidxglossary` but it displays all entries defined for the given glossary without sorting.

```

4377 \newcommand*{\printunsrtglossary}{%
4378   \@ifstar\s@printunsrtglossary\@printunsrtglossary
4379 }

```

`\printunsrtglossary` Unstarred version.

```

4380 \newcommand*{\@printunsrtglossary}[1][ ]{%
4381   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4382 }

```

`\printunsrtglossary` Starred version.

```

4383 \newcommand*{\s@printunsrtglossary}[2][ ]{%
4384   \beginngroup
4385     #2%
4386     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4387   \endngroup
4388 }

```

`\printunsrtglossaries` Similar to `\printnoidxglossaries` but it displays all entries defined for the given glossary without sorting.

```

4389 \newcommand*{\printunsrtglossaries}{%
4390   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4391 }

```

`\@unsrt@glossary`

```

4392 \newcommand*{\@print@unsrt@glossary}{%
4393   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4394   \glossarypreamble

```

check for empty list

```

4395 \glxtrifemptyglossary{\@glo@type}%
4396 {%
4397   \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4398 }%
4399 {%

4400   \key@ifundefined{glossentry}{group}%
4401   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4402   {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%
4403   \def\@gls@currentlettergroup{}}%

```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```

4404 \def\@glxtr@doglossary{%
4405   \begin{theglossary}%
4406   \glossaryheader
4407   \glsresetentrylist
4408 }%
4409 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4410   :\expandafter=\csname glist@\@glo@type\endcsname\do{%
4411   \ifdefempty{\glscurrententrylabel}
4412   }%
4413   {%

```

Provide a hook (for example to measure width).

```

4414   \let\glxtr@process\@firstofone
4415   \let\printunsrtglossaryskipentry
4416       \@glxtr@printunsrtglossaryskipentry
4417   \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

4418   \glxtr@process
4419   {%
4420     \ifglshasparent{\glscurrententrylabel}{}%
4421     {%
4422       \@glxtr@checkgroup\glscurrententrylabel
4423       \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
4424         {\@glxtr@groupheading}%
4425     }%
4426     \eappto\@glxtr@doglossary{%
4427       \noexpand\@printunsrtglossary@handler{\glscurrententrylabel}}%
4428   }%
4429 }%
4430 }%
4431 \appto\@glxtr@doglossary{\end{theglossary}}%
4432 \printunsrtglossarypredoglossary
4433 \@glxtr@doglossary
4434 }%
4435 \glossarypostamble
4436 }

```



entryprocesshook

```
4437 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}
```

entryprocesshook

```
4438 \newcommand*{\printunsrtglossaryskipentry}{%
4439   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4440 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4441 }
```

entryprocesshook

```
4442 \newcommand*{@glxstr@printunsrtglossaryskipentry}{%
4443   \let\glxstr@process\gobble
4444 }
```

entrypredoglossary

```
4445 \newcommand*{\printunsrtglossarypredoglossary}{}%
```

glossary@handler

```
4446 \newcommand{\@printunsrtglossary@handler}[1]{%
4447   \xdef\glscurrententrylabel{#1}%
4448   \printunsrtglossaryhandler\glscurrententrylabel
4449 }
```

glossaryhandler

```
4450 \newcommand{\printunsrtglossaryhandler}[1]{%
4451   \glxstrunsrtdo{#1}%
4452 }
```

triflabelinlist 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 \@glx@ifinlist which ensures the label and list are fully expanded.

```
4453 \newrobustcmd*{\glxstriflabelinlist}[4]{%
4454   \protected@edef\@glxstr@doiflabelinlist{\noexpand\@glx@ifinlist{#1}{#2}}%
4455   \@glxstr@doiflabelinlist{#3}{#4}%
4456 }
```

unsrtglossaryunit

```
4457 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
4458   \s@printunsrtglossary[type=\gldefaulttype,#1]{%
4459     \printunsrtglossaryunitsetup{#2}%
4460   }%
4461 }
```

glossaryunitsetup

```
4462 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4463   \renewcommand{\printunsrtglossaryhandler}[1]{%
4464     \glxstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
```

```

4465 {\glxtrunsrtdo{##1}}%
4466 {}%
4467 }%

```

Only the target names should have the prefixes adjusted as \gls etc need the original \glolinkprefix. The \@gobble part discards \glolinkprefix.

```

4468 \ifcsundef{theH#1}%
4469 {%
4470 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4471 }%
4472 {%
4473 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4474 }%
4475 \renewcommand*{\glossarysection}[2][{}]{%
4476 \appto\glossarypostamble{\glspar\medskip\glspar}%
4477 }

```

srtglossaryunit

```

4478 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
4479 \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4480 requires the record=only or record=alsoindex package option}{}%
4481 }

```

t@getgrouptitle

```

4482 \newrobustcmd*{\@glxtr@unsrt@getgrouptitle}[2]{%
4483 \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4484 \@onelevel@sanitize\@glxtr@titlelabel
4485 \ifcsdef{\@glxtr@titlelabel}
4486 {\letcs{#2}{\@glxtr@titlelabel}}%
4487 {\def#2{#1}}%
4488 }

```

\glxtrunsrtdo Provide a user-level call to \@glxtr@noidx@do to make it easier to define a new handler.

```

4489 \newcommand{\glxtrunsrtdo}{\@glxtr@noidx@do}

```

lsxtrgroupfield 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).

```

4490 \newcommand*{\lsxtrgroupfield}{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.

lsxtr@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.

```

4491 \newcommand*{\@glsxtr@checkgroup}[1]{%
4492   \def\@glsxtr@groupheading{}%
4493   \key@ifundefined{glossentry}{group}%
4494   {%
4495     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4496     \expandafter\glo@grabfirst\@gls@sort{}\@nil
4497   }%
4498   {%
4499     \protected@edef\@glo@thislettergrp{%
4500       \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
4501     }%
4502     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4503     {}%
4504     {%
4505       \ifdefempty{\@gls@currentlettergroup}{}%
4506       {\def\@glsxtr@groupheading{\gls@groupskip}}%
4507       \eappto\@glsxtr@groupheading{%
4508         \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
4509       }%
4510     }%
4511     \let\@gls@currentlettergroup\@glo@thislettergrp
4512 }

```

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.

```

4513 \newcommand{\@glsxtr@noidx@do}[1]{%
4514   \ifglsentryexists{#1}%
4515   {%
4516     \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4517     \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
4518     \ifgls@hasparent{#1}%
4519     {%
4520       \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
4521       \ifdefvoid{\@gls@location}%
4522       {%
4523         \ifdefvoid{\@gls@loclist}%
4524         {%
4525           \subglossentry{\gls@level}{#1}{}%
4526         }%
4527         {%
4528           \subglossentry{\gls@level}{#1}%
4529           {%
4530             \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4531           }%
4532         }%
4533       }%
4534     }%
4535     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4536   }%

```

```

4537 }%
4538 {%

4539 \ifdefvoid{\@gls@location}%
4540 {%
4541 \ifdefvoid{\@gls@loclist}
4542 {%
4543 \glossentry{#1}{}%
4544 }%
4545 {%
4546 \glossentry{#1}%
4547 {%
4548 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4549 }%
4550 }%
4551 }%
4552 {%
4553 \glossentry{#1}%
4554 {%
4555 \glossaryentrynumbers{\@gls@location}%
4556 }%
4557 }%
4558 }%
4559 }%
4560 {}%
4561 }

```

### 1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4562 \newcommand*{\glshex}{\string\u}
```

`\glxtrresourceinit` Code used during the protected write operation.

```
4563 \newcommand*{\glxtrresourceinit}{}%
```

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.

`\glxtrnewgls`

```
4564 \newcount\@glxtrnewgls@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.)

`\@glstrnewgls` `\glstrnewgls[<options>]{<prefix>}{<cs>}{<inner cs name>}`

```

4565 \newcommand*{\@glstrnewgls}[4]{%
4566   \ifdef{#3}%
4567   {%
4568     \PackageError{glossaries-extra}{Command \string#3\space already
4569 defined}{}%
4570   }%
4571   {%
4572     \ifcsdef{@#4like@#2}%
4573     {%
4574       \advance\@glstrnewgls@inner by \@ne
4575       \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
4576     }%
4577     {\def\@glstrnewgls@innercsname{@#4like@#2}}%
4578     \expandafter\newrobustcmd\expandafter*\expandafter
4579       #3\expandafter{\expandafter\@glshyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
4580     \ifstrempy{#1}%
4581     {%
4582       \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
4583         \new@ifnextchar[%
4584           {\csname @#4@\endcsname{##1}{#2##2}}%
4585           {\csname @#4@\endcsname{##1}{#2##2}[]}%
4586         }%
4587       }%
4588     }%
4589     \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
4590       \new@ifnextchar[%
4591         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4592         {\csname @#4@\endcsname{#1,##1}{#2##2}[]}%
4593       }%
4594     }%
4595   }%
4596 }

```

`\glstrnewgls` `\glstrnewgls[<options>]{<prefix>}{<cs>}`

The first argument prepends to the options and the second argument is the prefix.

```

4597 \newrobustcmd*{\glstrnewgls}[3][]{%
4598   \@glstrnewgls{#1}{#2}{#3}{gls}%
4599 }

```

`lsxtrnewglslike` Provide a way to conveniently define commands that behave like `\gls`, `\glspl`, `\Gls` and

`\GLspl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
4600 \newrobustcmd*{\glxtrnewglslike}[6] [] {%
4601   \@glxtrnewgls{#1}{#2}{#3}{gls}%
4602   \@glxtrnewgls{#1}{#2}{#4}{glspl}%
4603   \@glxtrnewgls{#1}{#2}{#5}{Gls}%
4604   \@glxtrnewgls{#1}{#2}{#6}{GLspl}%
4605 }
```

`\glxtrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSpl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
4606 \newrobustcmd*{\glxtrnewGLSlike}[4] [] {%
4607   \@glxtrnewgls{#1}{#2}{#3}{GLS}%
4608   \@glxtrnewgls{#1}{#2}{#4}{GLSpl}%
4609 }
```

`\glxtrnewrgls` As `\glxtrnewgls` but for `\rgls`.

```
4610 \newrobustcmd*{\glxtrnewrgls}[3] [] {%
4611   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4612 }
```

`\glxtrnewrglslike` As `\glxtrnewglslike` but for `\rgls` etc.

```
4613 \newrobustcmd*{\glxtrnewrglslike}[6] [] {%
4614   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4615   \@glxtrnewgls{#1}{#2}{#4}{rglspl}%
4616   \@glxtrnewgls{#1}{#2}{#5}{rGls}%
4617   \@glxtrnewgls{#1}{#2}{#6}{rGLspl}%
4618 }
```

`\glxtrnewGLSlike` As `\glxtrnewGLSlike` but for `\rGLS` etc.

```
4619 \newrobustcmd*{\glxtrnewrGLSlike}[4] [] {%
4620   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4621   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4622 }
```

Provide easy access to record count fields.

`\totalRecordCount` Access total record count. This is designed to be expandable. The argument is the label.

```
4623 \newcommand*{\GlsXtrTotalRecordCount}[1] {%
4624   \ifcsdef{glo@\glsdetoklabel{#1}@recordcount}%
4625   {\csname glo@\glsdetoklabel{#1}@recordcount\endcsname}%
4626   {0}%
4627 }
```

`\sXtrRecordCount` Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
4628 \newcommand*{\GlsXtrRecordCount}[2] {%
4629   \ifcsdef{glo@\glsdetoklabel{#1}@recordcount.#2}%
4630   {\csname glo@\glsdetoklabel{#1}@recordcount.#2\endcsname}%
4631   {0}%
4632 }
```

```

4630 {\csname glo@\glxtrdetoklabel{#1}@recordcount.#2\endcsname}%
4631 {0}%
4632 }

```

**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 `\glxtrdetoklocation` can be set to something sensible.

```

4633 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
4634 \ifcsdef{glo@\glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
4635 {\csname glo@\glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
4636 {0}%
4637 }

```

**trdetoklocation**

```

4638 \newcommand*{\glxtrdetoklocation}[1]{#1}

```

**ablerecordcount**

```

4639 \newcommand*{\glxtrenablerecordcount}{%
4640 \renewcommand*{\gls}{\rgls}%
4641 \renewcommand*{\Gls}{\rGls}%
4642 \renewcommand*{\glspl}{\rglspl}%
4643 \renewcommand*{\Glspl}{\rGlspl}%
4644 \renewcommand*{\GLS}{\rGLS}%
4645 \renewcommand*{\GLSpl}{\rGLSpl}%
4646 }

```

**ordtriggervalue** The value used by the record trigger test. The argument is the entry's label.

```

4647 \newcommand*{\glxtrrecordtriggervalue}[1]{%
4648 \GlsXtrTotalRecordCount{#1}%
4649 }

```

**dCountAttribute**

```

4650 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4651 \@for\@glxtr@cat:=#1\do
4652 {%
4653 \ifdefempty{\@glxtr@cat}{}%
4654 {%
4655 \glsssetcategoryattribute{\@glxtr@cat}{recordcount}{#2}%
4656 }%
4657 }%
4658 }

```

**rifrecordtrigger** `\glxtrifrecordtrigger{<label>}{<trigger format>}{<normal>}`

```

4659 \newcommand*{\glxstrifrecordtrigger}[3]{%
4660   \glshasattribute{#1}{recordcount}%
4661   {%
4662     \ifnum\glxstrrecordtriggervalue{#1}>\glsetattribute{#1}{recordcount}\relax
4663     #3%
4664   \else
4665     #2%
4666   \fi
4667 }%
4668 {#3}%
4669 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

4670 \newcommand*{\@glxstr@rglstrigger@record}[3]{%
4671   \edef\glslabel{\glsdetoklabel{#2}}%
4672   \let\@gls@link@label\glslabel
4673   \def\@glxstr@thevalue{%
4674     \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4675     \def\@glsnumberformat{glstriggerrecordformat}%
4676     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4677     \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4678     \def\@glxstr@thevalue{%
4679       \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4680       \glxstrinitwrgloss
4681       \setkeys{glslink}{#1}%
4682       \glslinkpostsetkeys
4683       \ifdefempty{\@glxstr@thevalue}%
4684       {%
4685         \@gls@saveentrycounter
4686       }%
4687       {%
4688         \let\thegl Sentrycounter\@glxstr@thevalue
4689         \def\theHgl Sentrycounter{\@glxstr@theHvalue}%
4690       }%
4691       \ifglxstrinitwrglossbefore
4692         \@do@wrglossary{#2}%
4693       \fi
4694       #3%
4695       \ifglxstrinitwrglossbefore
4696       \else
4697         \@do@wrglossary{#2}%
4698       \fi
4699       \ifKV@glslink@local
4700         \glslocalunset{#2}%
4701       \else
4702         \glsunset{#2}%
4703       \fi
4704 }

```



gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```
4705 \newcommand*{\glstriggerrecordformat}[1]{}
```

\rgls

```
4706 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}
```

\@rgls

```
4707 \newcommand*{\@rgls}[2][{}]{%
4708   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4709 }
```

\@rgls@

```
4710 \def\@rgls@#1#2[#3]{%
4711   \glstrifrecordtrigger{#2}%
4712   {%
4713     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4714   }%
4715   {%
4716     \@gls@{#1}{#2}[#3]%
4717   }%
4718 }%
```

\rglsp1

```
4719 \newrobustcmd*{\rglsp1}{\@gls@hyp@opt\@rglsp1}
```

\@rglsp1

```
4720 \newcommand*{\@rglsp1}[2][{}]{%
4721   \new@ifnextchar[{\@rglsp1@{#1}{#2}}{\@rglsp1@{#1}{#2}[]}%
4722 }
```

\@rglsp1@

```
4723 \def\@rglsp1@#1#2[#3]{%
4724   \glstrifrecordtrigger{#2}%
4725   {%
4726     \@glstr@rglstrigger@record{#1}{#2}{\rglsp1format{#2}{#3}}%
4727   }%
4728   {%
4729     \@glspl@{#1}{#2}[#3]%
4730   }%
4731 }%
```

\rGls

```
4732 \newrobustcmd*{\rGls}{\@gls@hyp@opt\@rGls}
```

\@rGls

```
4733 \newcommand*{\@rGls}[2][{}]{%
4734   \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4735 }
```

```

\@rGls@
4736 \def\@rGls@#1#2[#3]{%
4737   \glxtrifrecordtrigger{#2}%
4738   {%
4739     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4740   }%
4741   {%
4742     \@Gls@{#1}{#2}[#3]%
4743   }%
4744 }%

\rGlspl
4745 \newrobustcmd*{\rGlspl}{\@glshyp@opt\rGlspl}

\@rGlspl
4746 \newcommand*{\@rGlspl}[2][{}]{%
4747   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4748 }

\@rGlspl@
4749 \def\@rGlspl@#1#2[#3]{%
4750   \glxtrifrecordtrigger{#2}%
4751   {%
4752     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4753   }%
4754   {%
4755     \@Glspl@{#1}{#2}[#3]%
4756   }%
4757 }%

\rGLS
4758 \newrobustcmd*{\rGLS}{\@glshyp@opt\rGLS}

\@rGLS
4759 \newcommand*{\@rGLS}[2][{}]{%
4760   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}%
4761 }

\@rGLS@
4762 \def\@rGLS@#1#2[#3]{%
4763   \glxtrifrecordtrigger{#2}%
4764   {%
4765     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
4766   }%
4767   {%
4768     \@GLS@{#1}{#2}[#3]%
4769   }%
4770 }%

```

```
\rGLSpl
4771 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}
```

```
\@rGLSpl
4772 \newcommand*{\@rGLSpl}[2][\%
4773   \new@ifnextchar[\@rGLSpl@{#1}{#2}]{\@rGLSpl@{#1}{#2}[\%
4774 }
```

```
\@rGLSpl@
4775 \def\rGLSpl@#1#2[#3]{\%
4776   \glstriferecordtrigger{#2}%
4777   {\%
4778     \@glstr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
4779   }%
4780   {\%
4781     \rGLSpl@{#1}{#2}[#3]%
4782   }%
4783 }%
```

```
\rglsformat
4784 \newcommand*{\rglsformat}[2]{\%
4785   \glsifregular{#1}
4786   {\glsentryfirst{#1}}%
4787   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4788 }
```

```
\rglsplformat
4789 \newcommand*{\rglsplformat}[2]{\%
4790   \glsifregular{#1}
4791   {\glsentryfirstplural{#1}}%
4792   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4793 }
```

```
\rGlsformat
4794 \newcommand*{\rGlsformat}[2]{\%
4795   \glsifregular{#1}
4796   {\Glsentryfirst{#1}}%
4797   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4798 }
```

```
\rGlsplformat
4799 \newcommand*{\rGlsplformat}[2]{\%
4800   \glsifregular{#1}
4801   {\Glsentryfirstplural{#1}}%
4802   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4803 }
```

`\rGLSformat`

```
4804 \newcommand*{\rGLSformat}[2]{%
4805   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSformat{#1}{#2}}%
4806 }
```

`\rGLSplformat`

```
4807 \newcommand*{\rGLSplformat}[2]{%
4808   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSplformat{#1}{#2}}%
4809 }
```

## 1.4 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.

```
4810 \@ifpackageloaded{glossaries-accsupp}
4811 {
```

Define (or redefine) commands to use the accessibility information.

`\glsaccessname` Display the name value (no link and no check for existence).

```
4812   \newcommand*{\glsaccessname}[1]{%
4813     \glsnameaccessdisplay
4814     {%
4815       \glsentryname{#1}%
4816     }%
4817     {#1}%
4818   }
```

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.

```
4819   \newcommand*{\Glsaccessname}[1]{%
4820     \glsnameaccessdisplay
4821     {%
4822       \Glsentryname{#1}%
4823     }%
4824     {#1}%
4825   }
```

`\GLSaccessname` Display the name value (no link and no check for existence) converted to upper case.

```
4826   \newcommand*{\GLSaccessname}[1]{%
4827     \glsnameaccessdisplay
4828     {%
4829       \mfirstucMakeUppercase{\glsentryname{#1}}%
4830     }%
4831     {#1}%
4832   }
```

`\glsaccesstext` Display the text value (no link and no check for existence).

```
4833 \newcommand*{\glsaccesstext}[1]{%
4834   \glstextaccessdisplay
4835   {%
4836     \glstrytext{#1}%
4837   }%
4838   {#1}%
4839 }
```

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.

```
4840 \newcommand*{\Glsaccesstext}[1]{%
4841   \glstextaccessdisplay
4842   {%
4843     \Glsstrytext{#1}%
4844   }%
4845   {#1}%
4846 }
```

`\GLSaccesstext` Display the text value (no link and no check for existence) converted to upper case.

```
4847 \newcommand*{\GLSaccesstext}[1]{%
4848   \glstextaccessdisplay
4849   {%
4850     \mfirstucMakeUppercase{\glstrytext{#1}}%
4851   }%
4852   {#1}%
4853 }
```

`\glsaccessplural` Display the plural value (no link and no check for existence).

```
4854 \newcommand*{\glsaccessplural}[1]{%
4855   \glspluralaccessdisplay
4856   {%
4857     \glstryplural{#1}%
4858   }%
4859   {#1}%
4860 }
```

`\Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```
4861 \newcommand*{\Glsaccessplural}[1]{%
4862   \glspluralaccessdisplay
4863   {%
4864     \Glsstryplural{#1}%
4865   }%
4866   {#1}%
4867 }
```

`\GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```

4868 \newcommand*{\GLSaccessplural}[1]{%
4869   \glspluralaccessdisplay
4870   {%
4871     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4872   }%
4873   {#1}%
4874 }

```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

4875 \newcommand*{\glsaccessfirst}[1]{%
4876   \glsfirstaccessdisplay
4877   {%
4878     \glsentryfirst{#1}%
4879   }%
4880   {#1}%
4881 }

```

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

4882 \newcommand*{\Glsaccessfirst}[1]{%
4883   \glsfirstaccessdisplay
4884   {%
4885     \Glsentryfirst{#1}%
4886   }%
4887   {#1}%
4888 }

```

`\GLSaccessfirst` Display the first value (no link and no check for existence) converted to upper case.

```

4889 \newcommand*{\GLSaccessfirst}[1]{%
4890   \glsfirstaccessdisplay
4891   {%
4892     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4893   }%
4894   {#1}%
4895 }

```

`\glsaccessfirstplural` Display the firstplural value (no link and no check for existence).

```

4896 \newcommand*{\glsaccessfirstplural}[1]{%
4897   \glsfirstpluralaccessdisplay
4898   {%
4899     \glsentryfirstplural{#1}%
4900   }%
4901   {#1}%
4902 }

```

`\Glsaccessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

4903 \newcommand*{\Glsaccessfirstplural}[1]{%

```

```

4904   \glsfirstpluralaccessdisplay
4905   {%
4906       \Glsentryfirstplural{#1}%
4907   }%
4908   {#1}%
4909   }

```

**glsaccessfirstplural** Display the firstplural value (no link and no check for existence) converted to upper case.

```

4910   \newcommand*{\Glsaccessfirstplural}[1]{%
4911       \glsfirstpluralaccessdisplay
4912       {%
4913           \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4914       }%
4915       {#1}%
4916   }

```

**glsaccesssymbol** Display the symbol value (no link and no check for existence).

```

4917   \newcommand*{\glsaccesssymbol}[1]{%
4918       \glssymbolaccessdisplay
4919       {%
4920           \glsentrysymbol{#1}%
4921       }%
4922       {#1}%
4923   }

```

**Glsaccesssymbol** Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4924   \newcommand*{\Glsaccesssymbol}[1]{%
4925       \glssymbolaccessdisplay
4926       {%
4927           \Glsentrysymbol{#1}%
4928       }%
4929       {#1}%
4930   }

```

**GLSaccesssymbol** Display the symbol value (no link and no check for existence) converted to upper case.

```

4931   \newcommand*{\GLSaccesssymbol}[1]{%
4932       \glssymbolaccessdisplay
4933       {%
4934           \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
4935       }%
4936       {#1}%
4937   }

```

**glsaccesssymbolplural** Display the symbolplural value (no link and no check for existence).

```

4938   \newcommand*{\glsaccesssymbolplural}[1]{%
4939       \glssymbolpluralaccessdisplay
4940       {%

```

```

4941     \glentrysymbolplural{#1}%
4942   }%
4943   {#1}%
4944 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4945 \newcommand*{\Glsaccesssymbolplural}[1]{%
4946   \glssymbolpluralaccessdisplay
4947   {%
4948     \glentrysymbolplural{#1}%
4949   }%
4950   {#1}%
4951 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) converted to upper case.

```

4952 \newcommand*{\GLSaccesssymbolplural}[1]{%
4953   \glssymbolpluralaccessdisplay
4954   {%
4955     \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
4956   }%
4957   {#1}%
4958 }

```

\glsaccessdesc Display the desc value (no link and no check for existence).

```

4959 \newcommand*{\glsaccessdesc}[1]{%
4960   \glsdescriptionaccessdisplay
4961   {%
4962     \glentrydesc{#1}%
4963   }%
4964   {#1}%
4965 }

```

\GLSaccessdesc Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

4966 \newcommand*{\GLSaccessdesc}[1]{%
4967   \glsdescriptionaccessdisplay
4968   {%
4969     \GLSentrydesc{#1}%
4970   }%
4971   {#1}%
4972 }

```

\GLSAccessdesc Display the desc value (no link and no check for existence) converted to upper case.

```

4973 \newcommand*{\GLSAccessdesc}[1]{%
4974   \glsdescriptionaccessdisplay
4975   {%
4976     \mfirstucMakeUppercase{\GLSentrydesc{#1}}%

```



```

4977     }%
4978     {#1}%
4979 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence).

```

4980 \newcommand*{\glsaccessdescplural}[1]{%
4981   \glsdescriptionpluralaccessdisplay
4982   {%
4983     \glsentrydescplural{#1}%
4984   }%
4985   {#1}%
4986 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

```

4987 \newcommand*{\Glsaccessdescplural}[1]{%
4988   \glsdescriptionpluralaccessdisplay
4989   {%
4990     \Glsentrydescplural{#1}%
4991   }%
4992   {#1}%
4993 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

4994 \newcommand*{\GLSaccessdescplural}[1]{%
4995   \glsdescriptionpluralaccessdisplay
4996   {%
4997     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4998   }%
4999   {#1}%
5000 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5001 \newcommand*{\glsaccessshort}[1]{%
5002   \glsshortaccessdisplay
5003   {%
5004     \glsentryshort{#1}%
5005   }%
5006   {#1}%
5007 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5008 \newcommand*{\GLSaccessshort}[1]{%
5009   \glsshortaccessdisplay
5010   {%
5011     \Glsentryshort{#1}%
5012   }%

```

```

5013     {#1}%
5014 }

```

`\GLSaccessshort` Display the short value (no link and no check for existence) converted to upper case.

```

5015 \newcommand*{\GLSaccessshort}[1]{%
5016   \glsshortaccessdisplay
5017   {%
5018     \mfirstucMakeUppercase{\glsentryshort{#1}}%
5019   }%
5020   {#1}%
5021 }

```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

5022 \newcommand*{\lsaccessshortpl}[1]{%
5023   \glsshortpluralaccessdisplay
5024   {%
5025     \glsentryshortpl{#1}%
5026   }%
5027   {#1}%
5028 }

```

`lSaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5029 \newcommand*{\lSaccessshortpl}[1]{%
5030   \glsshortpluralaccessdisplay
5031   {%
5032     \lSentryshortpl{#1}%
5033   }%
5034   {#1}%
5035 }

```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence) converted to upper case.

```

5036 \newcommand*{\LSaccessshortpl}[1]{%
5037   \glsshortpluralaccessdisplay
5038   {%
5039     \mfirstucMakeUppercase{\lSentryshortpl{#1}}%
5040   }%
5041   {#1}%
5042 }

```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

5043 \newcommand*{\glsaccesslong}[1]{%
5044   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
5045 }

```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```

5046
5047 \newcommand*{\Glsaccesslong}[1]{%

```

```

5048     \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
5049 }

```

`\GLSaccesslong` Display the long value (no link and no check for existence) converted to upper case.

```

5050 \newcommand*{\GLSaccesslong}[1]{%
5051     \glslongaccessdisplay
5052     {%
5053         \mfirstucMakeUppercase{\glsentrylong{#1}}%
5054     }%
5055     {#1}%
5056 }

```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5057 \newcommand*{\glsaccesslongpl}[1]{%
5058     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5059 }

```

`GLsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5060
5061 \newcommand*{\GLsaccesslongpl}[1]{%
5062     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5063 }

```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```

5064 \newcommand*{\GLSaccesslongpl}[1]{%
5065     \glslongpluralaccessdisplay
5066     {%
5067         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5068     }%
5069     {#1}%
5070 }

```

End of if part

```

5071 }
5072 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

`\glsaccessname` Display the name value (no link and no check for existence).

```

5073 \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.

```

5074 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

`\GLSAccessname` Display the name value (no link and no check for existence). converted to upper case.

```

5075 \newcommand*{\GLSAccessname}[1]{%
5076     \protect\mfirstucMakeUppercase{\glsentryname{#1}}%

```

`\glsaccesstext` Display the text value (no link and no check for existence).  
5077 `\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.  
5078 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.  
5079 `\newcommand*{\GLSaccesstext}[1]{%`  
5080 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).  
5081 `\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.  
5082 `\newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.  
5083 `\newcommand*{\GLSaccessplural}[1]{%`  
5084 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).  
5085 `\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.  
5086 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.  
5087 `\newcommand*{\GLSaccessfirst}[1]{%`  
5088 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).  
5089 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`Glscessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.  
5090 `\newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

`GLScessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.  
5091 `\newcommand*{\GLSaccessfirstplural}[1]{%`  
5092 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).  
5093 `\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.

5094 `\newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

`\GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

5095 `\newcommand*{\GLSaccesssymbol}[1]{%`  
5096 `\protect\mfirstucMakeUppercase{\Glsentrysymbol{#1}}}`

`\esssymbolplural` Display the symbolplural value (no link and no check for existence).

5097 `\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.

5098 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`\esssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.

5099 `\newcommand*{\GLSaccesssymbolplural}[1]{%`  
5100 `\protect\mfirstucMakeUppercase{\Glsentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).

5101 `\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.

5102 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.

5103 `\newcommand*{\GLSaccessdesc}[1]{%`  
5104 `\protect\mfirstucMakeUppercase{\Glsentrydesc{#1}}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence).

5105 `\newcommand*{\glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

5106 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.

5107 `\newcommand*{\GLSaccessdescplural}[1]{%`  
5108 `\protect\mfirstucMakeUppercase{\Glsentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).

5109 `\newcommand*{\glsaccessshort}[1]{\Glsentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```
5110 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}
```

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.

```
5111 \newcommand*{\GLSaccessshort}[1]{%
5112 \protect\mfirstucMakeUppercase{\Glsentryshort{#1}}}
```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```
5113 \newcommand*{\lsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

`lSaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5114 \newcommand*{\lSaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.

```
5115 \newcommand*{\LSaccessshortpl}[1]{%
5116 \protect\mfirstucMakeUppercase{\Glsentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5117 \newcommand*{\glsaccesslong}[1]{\Glsentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5118 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

`\GLSaccesslong` Display the long value (no link and no check for existence). converted to upper case.

```
5119 \newcommand*{\GLSaccesslong}[1]{%
5120 \protect\mfirstucMakeUppercase{\Glsentrylong{#1}}}
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5121 \newcommand*{\glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5122 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.

```
5123 \newcommand*{\GLSaccesslongpl}[1]{%
5124 \protect\mfirstucMakeUppercase{\Glsentrylongpl{#1}}}
```

End of else part

```
5125 }
```

## 1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5126 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5127 \newcommand{\glsifcategory}[4]{%
5128   \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5129 }
```

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
5130 \newcommand*\glssetcategoryattribute[3]{%
5131   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
5132 }
```

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5133 \newcommand*\glsgetcategoryattribute[2]{%
5134   \csuse{@glxtr@categoryattr@@#1@#2}%
5135 }
```

`categoryattribute` `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5136 \newcommand*\glshascategoryattribute[4]{%
5137   \ifcvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
5138 }
```

`\glssetattribute` `\glssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5139 \newcommand*\glssetattribute[3]{%
```

```

5140 \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5141 }

```

`\glsggetattribute` `\glsggetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```

5142 \newcommand*{\glsggetattribute}[2]{%
5143   \glsssetcategoryattribute{\glscategory{#1}}{#2}%
5144 }

```

`\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.

```

5145 \newcommand*{\glshasattribute}[4]{%
5146   \ifglssentryexists{#1}%
5147   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5148   {#4}%
5149 }

```

`categoryattribute` `\glssifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```

5150 \newcommand{\glssifcategoryattribute}[5]{%
5151   \ifcsundef{@glssxtr@categoryattr@#1@#2}%
5152   {#5}%
5153   {\ifcsstring{@glssxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5154 }

```

`\glssifattribute` `\glssifattribute{<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.

```

5155 \newcommand{\glssifattribute}[5]{%
5156   \ifglssentryexists{#1}%
5157   {\glssifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5158   {#5}%
5159 }

```



Set attributes for the default general category:

```
5160 \glsssetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5161 \glsssetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```
5162 \newcommand*{\glsssetregularcategory}[1]{%
5163   \glsssetcategoryattribute{#1}{regular}{true}%
5164 }
```

`ifregularcategory` `\glssifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5165 \newcommand{\glssifregularcategory}[3]{%
5166   \glssifcategoryattribute{#1}{regular}{true}{#2}{#3}%
5167 }
```

`ifnotregularcategory` `\glssifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5168 \newcommand{\glssifnotregularcategory}[3]{%
5169   \glssifcategoryattribute{#1}{regular}{false}{#2}{#3}%
5170 }
```

`\glssifregular` `\glssifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5171 \newcommand{\glssifregular}[3]{%
5172   \glssifregularcategory{\glsscategor{#1}}{#2}{#3}%
5173 }
```

`\glssifnotregular` `\glssifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5174 \newcommand{\glssifnotregular}[3]{%
5175   \glssifnotregularcategory{\glsscategor{#1}}{#2}{#3}%
5176 }
```

oreachincategory

```
\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.

```
5177 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
5178   \forallglossaries[#1]{#3}%
5179   {%
5180     \forlgsentries[#3]{#4}%
5181     {%
5182       \glsifcategory{#4}{#2}{#5}{}%
5183     }%
5184   }%
5185 }
```

achwithattribute

```
\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.

```
5186 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5187   \forallglossaries[#1]{#4}%
5188   {%
5189     \forlgsentries[#4]{#5}%
5190     {%
5191       \glsifattribute{#5}{#2}{#3}{#6}{}%
5192     }%
5193   }%
5194 }
```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to `index` and add `\glstrpostdescription`.

```
5195 \ifdef\newterm
5196 {%
```

`\newterm`

```
5197   \renewcommand*{\newterm}[2][ ]{%
5198     \newglossaryentry{#2}%
5199     {type={index},category=index,name={#2},%
```

```

5200     description={\glxtrpostdescription\nopostdesc},#1}%
5201 }

```

Indexed terms are regular by default.

```

5202 \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5203 \newcommand*{\glxtrpostdescindex}{}

5204 }
5205 {}

```

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.

```

5206 \ifdef\printsymbols
5207 {%

```

glxtrnewsymbol Unlike \newterm, this has a separate argument for the label (since the symbol will likely contain commands).

```

5208 \newcommand*{\glxtrnewsymbol}[3] [] {%
5209     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5210 }

```

Symbols are regular by default.

```

5211 \glsssetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

5212 \newcommand*{\glxtrpostdescsymbol}{}

5213 }
5214 {}

```

Similar for the numbers option.

```

5215 \ifdef\printnumbers
5216 {%

```

glxtrnewnumber

```

5217 \ifdef\printnumbers
5218 \newcommand*{\glxtrnewnumber}[3] [] {%
5219     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5220 }

```

Numbers are regular by default.

```

5221 \glsssetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```

5222 \newcommand*{\glxtrpostdescnumber}{}

```

```
5223 }
5224 {}
```

`\glstrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
5225 \newcommand*{\glstrsetcategory}[2]{%
5226   \@for\@glstr@label:=#1\do
5227   {%
5228     \glsfieldxdef{\@glstr@label}{category}{#2}%
5229   }%
5230 }
```

`\glstrcategoryforall` 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.

```
5231 \newcommand*{\glstrcategoryforall}[2]{%
5232   \forallglossaries[#1]{\@glstr@type}{%
5233     \forallsentries[\@glstr@type]{\@glstr@label}%
5234     {%
5235       \glsfieldxdef{\@glstr@label}{category}{#2}%
5236     }%
5237   }%
5238 }
```

`\glstrfieldtitlecase` `\glstrfieldtitlecase{\langle label \rangle}{\langle field \rangle}`

Apply title casing to the contents of the given field.

```
5239 \newcommand*{\glstrfieldtitlecase}[2]{%
5240   \expandafter\glstrfieldtitlecasesecs\expandafter
5241   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
5242 }
```

`\glstrfieldtitlecasesecs` The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```
5243 \newcommand*{\glstrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}
```

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.

```
5244 \@ifpackageloaded{glossaries-accsupp}
5245 {
5246   \renewcommand*{\glossentrydesc}[1]{%
5247     \glsdoifexistsorwarn{#1}%
5248     {%
5249       \glssetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the glossdescfont attribute to determine the font applied.

```

5250 \glshasattribute{#1}{glossdescfont}%
5251 {%
5252 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5253 \ifcsdef{\@glxtr@attrval}%
5254 {%
5255 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5256 }%
5257 {%
5258 \GlossariesExtraWarning{Unknown control sequence name
5259 '\@glxtr@attrval' supplied in glossdescfont attribute
5260 for entry '#1'. Ignoring}%
5261 \let\@glxtr@glossdescfont\@firstofone
5262 }%
5263 }%
5264 {\let\@glxtr@glossdescfont\@firstofone}%
5265 \glusifattribute{#1}{glossdesc}{firstuc}%
5266 {%
5267 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5268 }%
5269 {%
5270 \glusifattribute{#1}{glossdesc}{title}%
5271 {%
5272 \@glxtr@do@titlecaps@warn
5273 \glsdescriptionaccessdisplay
5274 {%
5275 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5276 }%
5277 {#1}%
5278 }%
5279 {%
5280 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5281 }%
5282 }%
5283 }%
5284 }
5285 }
5286 {
5287 \renewcommand*{\glossentrydesc}[1]{%
5288 \glsdoifexistsorwarn{#1}%
5289 {%
5290 \glissetabbrvfmt{\glscategory{#1}}%
5291 \glshasattribute{#1}{glossdescfont}%
5292 {%
5293 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5294 \ifcsdef{\@glxtr@attrval}%
5295 {%
5296 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5297 }%

```

```

5298      {%
5299      \GlossariesExtraWarning{Unknown control sequence name
5300      '\@glsxtr@attrval' supplied in glossdescfont attribute
5301      for entry '#1'. Ignoring}%
5302      \let\@glsxtr@glossdescfont\@firstofone
5303      }%
5304      }%
5305      {\let\@glsxtr@glossdescfont\@firstofone}%
5306      \glsifattribute{#1}{glossdesc}{firstuc}%
5307      {%
5308      \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5309      }%
5310      {%
5311      \glsifattribute{#1}{glossdesc}{title}%
5312      {%
5313      \@glsxtr@do@titlecaps@warn
5314      \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5315      }%
5316      {%
5317      \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5318      }%
5319      }%
5320      }%
5321      }
5322      }

```

`\glossentryname` If the glossname attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5323 \ifpackageloaded{glossaries-accsupp}
5324 {
5325   \renewcommand*{\glossentryname}[1]{%
5326     \@glsdoifexistsorwarn{#1}%
5327     {%
5328       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5329   \glshasattribute{#1}{glossnamefont}%
5330   {%
5331   \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5332   \ifcsdef{\@glsxtr@attrval}%
5333   {%
5334   \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5335   }%
5336   {%
5337   \GlossariesExtraWarning{Unknown control sequence name
5338   '\@glsxtr@attrval' supplied in glossnamefont attribute
5339   for entry '#1'. Reverting to default \string\glsnamefont}%
5340   \let\@glsxtr@glossnamefont\glsnamefont
5341   }%
5342   }%

```

```

5343 {\let\@glsxtr@glossnamefont\glsnamefont}%
5344 \glsifattribute{#1}{glossname}{firstuc}%
5345 {%
5346   \glsnameaccessdisplay
5347   {%
5348     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5349   }%
5350   {#1}%
5351 }%
5352 {%
5353   \glsifattribute{#1}{glossname}{title}%
5354   {%
5355     \@glsxtr@do@titlecaps@warn
5356     \glsnameaccessdisplay
5357     {%
5358       \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
5359     }%
5360     {#1}%
5361   }%
5362   {%
5363     \glsifattribute{#1}{glossname}{uc}%
5364     {%
5365       \glsnameaccessdisplay
5366       {%

```

Hide the label from the upper-casing command.

```

5367       \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5368       \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5369     }%
5370     {#1}%
5371   }%
5372   {%
5373     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5374     \glsnameaccessdisplay
5375     {%
5376       \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
5377     }%
5378     {#1}%
5379   }%
5380 }%
5381 }%

```

Do post-name hook:

```

5382   \glsxtrpostnamehook{#1}%
5383 }%
5384 }
5385 }
5386 {
5387   \renewcommand*{\glossentryname}[1]{%
5388     \@glsdoifexistsorwarn{#1}%

```

```

5389  {%
5390      \glsetabbrvfmt{\glscategory{#1}}%
5391      \glshasattribute{#1}{glossnamefont}%
5392  {%
5393      \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5394      \ifcsdef{\@glxtr@attrval}%
5395      {%
5396          \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5397      }%
5398      {%
5399          \GlossariesExtraWarning{Unknown control sequence name
5400              '\@glxtr@attrval' supplied in glossnamefont attribute
5401              for entry '#1'. Reverting to default \string\glnamefont}%
5402          \let\@glxtr@glossnamefont\glnamefont
5403      }%
5404  }%
5405  {\let\@glxtr@glossnamefont\glnamefont}%
5406  \gl@ifattribute{#1}{glossname}{firstuc}%
5407  {%
5408      \@glxtr@glossnamefont{\Glsentryname{#1}}%
5409  }%
5410  {%
5411      \gl@ifattribute{#1}{glossname}{title}%
5412      {%
5413          \@glxtr@do@titlecaps@warn
5414          \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
5415      }%
5416      {%
5417          \gl@ifattribute{#1}{glossname}{uc}%
5418          {%

```

Hide the label from the upper-casing command.

```

5419      \letcs{\glo@name}{\glo@glsetoklabel{#1}@name}%
5420      \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5421      }%
5422      {%

```

This little trick is used by glossaries to allow the user to redefine `\glnamefont` to use `\makefirstuc`. Support it even though they can now use the `firstuc` attribute.

```

5423      \letcs{\glo@name}{\glo@glsetoklabel{#1}@name}%
5424      \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5425      }%
5426      }%
5427      }%

```

Do post-name hook.

```

5428      \glxtrpostnamehook{#1}%
5429      }%
5430  }
5431 }

```



`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
5432 \@ifpackageloaded{glossaries-accsupp}
5433 {
5434   \renewcommand*{\Glossentryname}[1]{%
5435     \@glsdoifexistsorwarn{#1}%
5436     {%
5437       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
5438     \glshasattribute{#1}{glossnamefont}%
5439     {%
5440       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5441       \ifcsdef{\@glstr@attrval}%
5442       {%
5443         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5444       }%
5445       {%
5446         \GlossariesExtraWarning{Unknown control sequence name
5447           '\@glstr@attrval' supplied in glossnamefont attribute
5448           for entry '#1'. Reverting to default \string\glnamefont}%
5449         \let\@glstr@glossnamefont\glnamefont
5450       }%
5451     }%
5452     {\let\@glstr@glossnamefont\glnamefont}%
5453     \glnameaccessdisplay
5454     {%
5455       \@glstr@glossnamefont{\Glsentryname{#1}}%
5456     }%
5457     {#1}%
```

Do post-name hook:

```
5458     \glstrpostnamehook{#1}%
5459   }%
5460 }
5461 }
5462 {
5463   \renewcommand*{\Glossentryname}[1]{%
5464     \@glsdoifexistsorwarn{#1}%
5465     {%
5466       \glsetabbrvfmt{\glscategory{#1}}%
5467       \glshasattribute{#1}{glossnamefont}%
5468       {%
5469         \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5470         \ifcsdef{\@glstr@attrval}%
5471         {%
5472           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5473         }%
5474         {%
5475           \GlossariesExtraWarning{Unknown control sequence name
5476             '\@glstr@attrval' supplied in glossnamefont attribute
```

```

5477         for entry '#1'. Reverting to default \string\glsnamefont}%
5478         \let\@glsxtr@glossnamefont\glsnamefont
5479     }%
5480 }%
5481 {\let\@glsxtr@glossnamefont\glsnamefont}%
5482 \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5483     \glsxtrpostnamehook{#1}%
5484 }%
5485 }
5486 }

```

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.

`xtrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```

5487 \newcommand*{\glsxtrpostnamehook}[1]{%
5488   \let\@glsnumberformat\@glsxtr@defaultnumberformat
5489   \glsxtrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```

5490   \csuse{glsxtrpostname\glscategory{#1}}%
5491 }

```

`etaccessdisplay`

```

5492 \@ifpackageloaded{glossaries-accsupp}
5493 {
5494   \newcommand*{\glsxtr@setaccessdisplay}[1]{%
5495     \ifcsdef{gls#1accessdisplay}%
5496     {\letcs\@glsxtr@accessdisplay{gls#1accessdisplay}}%
5497     {%

```

This is essentially the reverse of `\@gls@fetchfield`, since the field supplied to `\glossentryname` has to be the internal label, but the `\gls<field>accessdisplay` commands use the key name.

```

5498     \edef\@gls@thisval{#1}%
5499     \@for\@gls@map:=\@gls@keymap\do{%
5500       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5501       \ifdefequal{\@this@key}{\@gls@thisval}%
5502       {%
5503         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5504         \@endfortrue
5505       }%
5506     }%
5507 }%
5508 \ifcsdef{gls\@gls@thisval accessdisplay}%
5509 {\letcs\@glsxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5510 {\let\@glsxtr@accessdisplay\@firstoftwo}%

```

```

5511     }%
5512   }
5513 }
5514 {%
5515   \newcommand*{\glxtr@setaccessdisplay}[1]{%
5516     \let\@glxtr@accessdisplay\@firstoftwo}
5517 }

```

sentrynameother Provide a command that works like \glossentryname but accesses a different field (which must be supplied using its internal field label).

```

5518 \newrobustcmd*{\glossentrynameother}[2]{%
5519   \@glsoifexistsorwarn{#1}%
5520   {%

```

Accessibility support:

```

5521   \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5522   \glsssetabbrfmt{\glscategory{#1}}%
5523   \glshasattribute{#1}{glossnamefont}%
5524   {%
5525     \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5526     \ifcsdef{\@glxtr@attrval}%
5527     {%
5528       \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5529     }%
5530     {%
5531       \GlossariesExtraWarning{Unknown control sequence name
5532         '\@glxtr@attrval' supplied in glossnamefont attribute
5533         for entry '#1'. Reverting to default \string\glsgnamefont}%
5534       \let\@glxtr@glossnamefont\glsgnamefont
5535     }%
5536     }%
5537     {\let\@glxtr@glossnamefont\glsgnamefont}%
5538     \glsgifattribute{#1}{glossname}{firstuc}%
5539     {%
5540       \glxtr@accessdisplay
5541       {\@glxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5542       {#1}%
5543     }%
5544     {%
5545       \glsgifattribute{#1}{glossname}{title}%
5546       {%
5547         \@glxtr@do@titlecaps@warn
5548         \@glxtr@accessdisplay
5549         {\@glxtr@glossnamefont{\glxtr@field@titlecase{#1}{#2}}}%
5550         {#1}%
5551       }%
5552       {%
5553         \glsgifattribute{#1}{glossname}{uc}%

```

```

5554      {%
5555      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5556      \@glsxtr@accessdisplay
5557      {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5558      {#1}%
5559      }%
5560      {%
5561      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5562      \@glsxtr@accessdisplay
5563      {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
5564      {#1}%
5565      }%
5566      }%
5567      }%

Do post-name hook.
5568      \glsxtrpostnamehook{#1}%
5569      }%
5570 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

5571 \newif\if@glsxtr@format@override
5572 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5573 \@ifpackageloaded{hyperref}
5574 {
  If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
  don't add it.
5575   \ifHy@hyperindex
5576     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5577       \@glsxtr@format@override true
5578       \appto\theindex{\let\glshypernumber\@firstofone}%
5579     }
5580   \else
5581     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5582       \@glsxtr@format@override true
5583       \appto\theindex{\let\glshypernumber\hyperpage}%
5584     }
5585   \fi
5586 }
5587 {
5588   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5589     \@glsxtr@format@override true
5590   }
5591 }

```

```
5592 \@onlypreamble\GlsXtrEnableIndexFormatOverride
```

doautoindexname

```
5593 \newcommand*{\glstrdoautoindexname}[2]{%
5594   \glshasattribute{#1}{#2}%
5595   {%
```

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.

```
5596   \@glstr@autoindex@setname{#1}%
```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```
5597   \protected@edef\@glstr@attrval{\glsggetattribute{#1}{#2}}%
5598   \if@glstr@format@override

5599     \ifx\@glsnnumberformat\@glstr@defaultnumberformat
5600     \else
5601       \let\@glstr@attrval\@glsnnumberformat
5602     \fi
5603   \fi
5604   \ifdefstring{\@glstr@attrval}{true}%
5605   {}%
5606   {\eappto\@glo@name{\@glstr@autoindex@encap\@glstr@attrval}}%
5607   \expandafter\glstrautoindex\expandafter{\@glo@name}%
5608 }%
5609 {}%
5610 }
```

glstrautoindex

```
5611 \newcommand*{\glstrautoindex}{\index}
```

toindex@setname Assign \@glo@name for use with indexname attribute.

```
5612 \newcommand*{\@glstr@autoindex@setname}[1]{%
5613   \protected@edef\@glo@name{\glstrautoindexentry{#1}}%
5614   \glstrautoindexassignsort{\@glo@sort}{#1}%
5615   \@glsccheckmkidxchars\@glo@sort
5616   \@glstr@autoindex@doextra@esc\@glo@sort
5617   \epreto\@glo@name{\@glo@sort\@glstr@autoindex@at}%
5618 }
```

rautoindexentry Command used for the actual part when auto-indexing.

```
5619 \newcommand*{\glstrautoindexentry}[1]{\string\glstryname{#1}}
```

trautoindexsort Used to assign the sort value when auto-indexing.

```
5620 \newcommand*{\glstrautoindexassignsort}[2]{%
5621   \glslsetentryfield{#1}{#2}{sort}%
5622 }
```

dex@doextra@esc

```
5623 \newcommand*{\@glstr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5624 \ifx\@glsxtr@autoindex@esc\@gls@quotechar
5625 \else
5626 \def\@gls@checkedmkidx{}%
5627 \edef\@@glsxtr@checkspch{%
5628 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
5629 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
5630 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5631 @@glsxtr@checkspch
5632 \let#1\@gls@checkedmkidx\relax
5633 \fi
```

Escape actual character unless it has already been escaped.

```
5634 \ifx\@glsxtr@autoindex@at\@gls@actualchar
5635 \else
5636 \def\@gls@checkedmkidx{}%
5637 \edef\@@glsxtr@checkspch{%
5638 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
5639 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
5640 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5641 @@glsxtr@checkspch
5642 \let#1\@gls@checkedmkidx\relax
5643 \fi
```

Escape level character unless it has already been escaped.

```
5644 \ifx\@glsxtr@autoindex@level\@gls@levelchar
5645 \else
5646 \def\@gls@checkedmkidx{}%
5647 \edef\@@glsxtr@checkspch{%
5648 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
5649 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
5650 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5651 @@glsxtr@checkspch
5652 \let#1\@gls@checkedmkidx\relax
5653 \fi
```

Escape encap character unless it has already been escaped.

```
5654 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
5655 \else
5656 \def\@gls@checkedmkidx{}%
5657 \edef\@@glsxtr@checkspch{%
5658 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
5659 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
5660 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5661 @@glsxtr@checkspch
5662 \let#1\@gls@checkedmkidx\relax
5663 \fi
5664 }
```

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.

```
5665 \newcommand*{\@glxtr@autoindex@at}{}
```

trSetActualChar Set the actual character.

```
5666 \newcommand*{\GlsXtrSetActualChar}[1]{%
5667   \gdef\@glxtr@autoindex@at{#1}%
5668   \def\@glxtr@autoindex@escat##1#1##2#1##3\@glxtr@endescspch{%
5669     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
5670   }%
5671 }
5672 \@onlypreamble\GlsXtrSetActualChar
5673 \makeatother
5674 \GlsXtrSetActualChar{@}
5675 \makeatletter
```

autoindex@encap Encap character for use with \index.

```
5676 \newcommand*{\@glxtr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
5677 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5678   \gdef\@glxtr@autoindex@encap{#1}%
5679   \def\@glxtr@autoindex@escencap##1#1##2#1##3\@glxtr@endescspch{%
5680     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
5681   }%
5682 }
5683 \GlsXtrSetEncapChar{||}
5684 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
5685 \newcommand*{\@glxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
5686 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5687   \gdef\@glxtr@autoindex@level{#1}%
5688   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
5689     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
5690   }%
5691 }
5692 \GlsXtrSetLevelChar{!}
5693 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
5694 \newcommand*{\@glxtr@autoindex@esc}{"}
```

lsXtrSetEscChar Set the escape character.

```
5695 \newcommand*{\GlsXtrSetEscChar}[1]{%
5696   \gdef\@glxtr@autoindex@esc{#1}%
5697   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
```

```

5698 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escquote}{##1}{##2}{##3}%
5699 }%
5700 }
5701 \GlsXtrSetEscChar{"}
5702 \@onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```

5703 \ifdef\actualchar
5704 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5705 {}

```

Quote character \quotechar:

```

5706 \ifdef\quotechar
5707 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5708 {}

```

Level character \levelchar:

```

5709 \ifdef\levelchar
5710 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5711 {}

```

Encap character \encapchar:

```

5712 \ifdef\encapchar
5713 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5714 {}

```

leto@endescspch

```

5715 \def\@glsxtr@gobbleto@endescspch#1\@glsxtr@endescspch{}

```

toindex@esc@spch

<pre> \@@glsxtr@autoindex@escspch{&lt;char&gt;}{&lt;cs&gt;}{&lt;pre&gt;}{&lt;mid&gt;}{&lt;post&gt;} </pre>
--

```

5716 \newcommand*{\@glsxtr@autoindex@escspch}[5]{%
5717 \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
5718 \toks@={#3}%
5719 \ifx\@nnil#3\relax
5720 \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch#5\@glsxtr@endescspch}%
5721 \else
5722 \ifx\@nnil#4\relax
5723 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
5724 \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch
5725 #4#5\@glsxtr@endescspch}%
5726 \else
5727 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
5728 \@glsxtr@autoindex@esc#1}%
5729 \def\@glsxtr@checkspch{#2#5#1\@nnil#1\@glsxtr@endescspch}%
5730 \fi
5731 \fi

```



```

5732 \@@glxtr@checkspch
5733 }

```

`\Glossentrydesc` Redefine to set the abbreviation format and accessibility support.

```

5734 \renewcommand*{\Glossentrydesc}[1]{%
5735   \glsoifexistsorwarn{#1}%
5736   {%
5737     \glsetabbrvfmt{\glscategory{#1}}%
5738     \Glsaccessdesc{#1}%
5739   }%
5740 }

```

`\lossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

5741 \renewcommand*{\lossentrysymbol}[1]{%
5742   \glsoifexistsorwarn{#1}%
5743   {%
5744     \glsetabbrvfmt{\glscategory{#1}}%
5745     \Glsaccesssymbol{#1}%
5746   }%
5747 }

```

`\lossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

5748 \renewcommand*{\lossentrysymbol}[1]{%
5749   \glsoifexistsorwarn{#1}%
5750   {%
5751     \glsetabbrvfmt{\glscategory{#1}}%
5752     \Glsaccesssymbol{#1}%
5753   }%
5754 }

```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\eInitialTagging` 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.

```

5755 \newcommand*{\GlsXtrEnableInitialTagging}{%
5756   \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
5757 }
5758 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

5759 \newcommand*{\s@glxtr@enabletagging}[2]{%
5760   \undef#2%
5761   \@glxtr@enabletagging{#1}{#2}%
5762 }

```

`\r@enabletagging` Internal command.

```

5763 \newcommand*{\@glxtr@enabletagging}[2]{%

```

Set attributes for categories given in the first argument.

```

5764 \@for\@glstr@cat:=#1\do
5765 {%
5766   \ifdefempty\@glstr@cat
5767   {%
5768     {\glsetcategoryattribute{\@glstr@cat}{tagging}{true}}%
5769   }%
5770   \newrobustcmd*#2[1]{##1}%
5771   \def\@glstr@taggingcs{#2}%
5772   \renewcommand*\@glstr@activate@initialtagging{%
5773     \let#2\@glstr@tag
5774   }%
5775   \ifundef\@gl@preglossaryhook
5776   {\GlossariesExtraWarning{Initial tagging requires at least
5777     glossaries.sty v4.19 to work correctly}}%
5778   {%
5779 }
```

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.

fu@checkword@do If this command hasn't been defined, then we have pre v2.02 of mfirstuc

```

5780 \ifundef\mfu@checkword@do
5781 {
5782   \newcommand*{\mfu@checkword@do}[1]{%
5783     \ifdefstring{\mfu@checkword@arg}{#1}%
5784     {%
5785       \let\@mfu@domakefirstuc\@firstofone
5786       \listbreak
5787     }%
5788     {}%
5789   }
```

\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.

```

5790 \ifundef\mfu@checkword
5791 {
5792   \newcommand{\@glstr@do@titlecaps@warn}{%
5793     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5794       support not available}%
5795   }
```

One warning should suffice.

```

5795     \let\@glstr@do@titlecaps@warn\relax
5796   }
5797 }
5798 {
5799   \renewcommand*{\mfu@checkword}[1]{%
5800     \def\mfu@checkword@arg{#1}%
5801     \let\@mfu@domakefirstuc\makefirstuc
```

```

5802      \forlistloop\mfu@checkword@do\@mfu@nocaplist
5803    }
5804  }
5805 }
5806 {}% no patch required

@titlecaps@warn  Do warning if title case not supported.
5807 \newcommand*{\@glxtr@do@titlecaps@warn}{}

@initialtagging  Used in \printglossary but at least v4.19 of glossaries required.
5808 \newcommand*{\glxtr@activate@initialtagging}{}

\@glxtr@tag  Definition of tagging command when used in glossary.
5809 \newrobustcmd*{\@glxtr@tag}[1]{%
5810   \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%
5811   {\glxtrtagfont{#1}}{#1}%
5812 }

\glxtrtagfont  Used in the glossary.
5813 \newcommand*{\glxtrtagfont}[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.
5814 \ifdef\@gl@preglossaryhook
5815 {
5816   \renewcommand*{\@gl@preglossaryhook}{%
5817     \@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.
5818     \ifundef\@glxtr@org@postdescription
5819     {%
5820       \let\@glxtr@org@postdescription\glspostdescription
5821       \renewcommand*{\glspostdescription}{%
5822         \ifglentryexists{\glscurrententrylabel}%
5823         {%
5824           \glxtrpostdescription
5825           \@glxtr@org@postdescription
5826         }%
5827       }%
5828     }%
5829   }%
5830 }%

    Enable the options used by \@glxtrp:
5831   \glossxtrsetpopts
5832 }%

```

```

5833 }
5834 {}

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 prede-
fined styles that don't include it.
5835 \newcommand*{\glstrpostdescription}{%
5836   \csuse{glstrpostdesc\glscategory{\glscurrententrylabel}}%
5837 }

postdescgeneral
5838 \newcommand*{\glstrpostdescgeneral}{}

xtrpostdescterm
5839 \newcommand*{\glstrpostdescterm}{}

postdescacronym
5840 \newcommand*{\glstrpostdescacronym}{}

descabbreviation
5841 \newcommand*{\glstrpostdescabbreviation}{}

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.
5842 \renewcommand*{\glspostlinkhook}{%
5843   \ifglstryexists{\glslabel}{\glstrpostlinkhook}{}%
5844 }

xtrpostlinkhook The entry label should already be stored in \glslabel by \@gls@link.
5845 \newcommand*{\glstrpostlinkhook}{%
5846   \glstrdiscardperiod{\glslabel}%
5847   {\glstrpostlinkendsentence}%
5848   {\glstrifcustomdiscardperiod
5849     {\glstrifperiod{\glstrpostlinkendsentence}{\glstrpostlink}}}%
5850   {\glstrpostlink}%
5851   }%
5852 }

omdiscardperiod Allow user to provide a custom check. Should expand to #2 if no check is required otherwise
expand to #1.
5853 \newcommand*{\glstrifcustomdiscardperiod}[2]{#2}

\glstrpostlink
5854 \newcommand*{\glstrpostlink}{%
5855   \csuse{glstrpostlink\glscategory{\glslabel}}%
5856 }

```

linkendsentence Done by \glxtrpostlinkhook if a full stop is discarded.

```
5857 \newcommand*{\glxtrpostlinkendsentence}{%
5858   \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
5859   {%
5860     \csuse{glxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
5861     .\spacefactor\sfcode'\. \relax
5862   }%
5863   {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

```
5864     \spacefactor\sfcode'\. \relax
5865   }%
5866 }
```

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.

```
5867 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
5868   \glxtrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
5869 }
```

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.

```
5870 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5871   \glxtrifwasfirstuse
5872   {%
5873     \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
5874   }%
5875   {}%
5876 }
```

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).

```
5877 \newcommand*{\glxtrdiscardperiod}[3]{%
5878   \glxtrifwasfirstuse
5879   {%
5880     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5881     {#3}%
5882   }%
5883     \glusifattribute{#1}{discardperiod}{true}%
5884     {%
5885       \glusifplural
5886       {%
5887         \glusifattribute{#1}{pluraldiscardperiod}{true}%
5888         {\glxtrifperiod{#2}{#3}}%
```

```

5889         {#3}%
5890     }%
5891     {%
5892         \glxtrifperiod{#2}{#3}%
5893     }%
5894 }%
5895 {#3}%
5896 }%
5897 }%
5898 {%
5899     \glusifattribute{#1}{discardperiod}{true}%
5900     {%
5901         \glusifplural
5902         {%
5903             \glusifattribute{#1}{pluraldiscardperiod}{true}%
5904             {\glxtrifperiod{#2}{#3}}%
5905             {#3}%
5906         }%
5907         {%
5908             \glxtrifperiod{#2}{#3}%
5909         }%
5910     }%
5911     {#3}%
5912 }%
5913 }

```

`\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`

```
5914 \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 ' ').

```
5915 \newcommand*{\glxtr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
5916 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
5917 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

`\glxtrifpunc` `\glxtrifnextpunc{<true part>}{<false part>}`

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```
5918 \newcommand*{\glxtrifnextpunc}[2]{%
```

```

5919 \def\reserved@a{#1}%
5920 \def\reserved@b{#2}%
5921 \futurelet\@glspunc@token\glxtr@ifnextpunc
5922 }

```

`glxtr@ifnextpunc`

```

5923 \newcommand*\glxtr@ifnextpunc{%
5924 \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5925 \reserved@b
5926 }

```

`glxtr@ifpunctoken` Test if the token given in the first argument is in the punctuation list.

```

5927 \newcommand*\glxtr@ifpunctoken[1]{%
5928 \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5929 }

```

`glxtr@ifpunctoken`

```

5930 \def\@glxtr@ifpunctoken#1#2{%
5931 \let\reserved@d=#2%
5932 \ifx\reserved@d\@nnil
5933 \let\glxtr@next\@glxtr@notfoundinlist
5934 \else
5935 \ifx#1\reserved@d
5936 \let\glxtr@next\@glxtr@foundinlist
5937 \else
5938 \let\glxtr@next\@glxtr@ifpunctoken
5939 \fi
5940 \fi
5941 \glxtr@next#1%
5942 }

```

`glxtr@foundinlist`

```

5943 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}

```

`@notfoundinlist`

```

5944 \def\@glxtr@notfoundinlist#1{\@secondoftwo}

```

`glxtrdopostpunc`

```
\glxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do *<code>* after the character otherwise do *<code>* before whatever comes next.

```

5945 \newcommand{\glxtrdopostpunc}[1]{%
5946 \glxtr@ifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
5947 }

```

`@glxtr@swaptwo`

```

5948 \newcommand{\@glxtr@swaptwo}[2]{#2#1}

```

## 1.6 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, apply it.

```
5949 \define@key{glstrabbrv}{category}{%
5950 \edef\glscategorylabel{#1}%
5951 \ifcsdef{@glssabbrv@current@#1}%
5952 {%
```

Warning should already have been issued.

```
5953 \let\@glstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5954 \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5955 \glstr@applyabbrvstyle{\csname @glssabbrv@current@#1\endcsname}%
5956 \let\GlsXtrWarnDeprecatedAbbrStyle\@glstr@orgwarndep
5957 }%
5958 }%
5959 }
```

Save the short plural form. This may be needed before the entry is defined.

```
5960 \define@key{glstrabbrv}{shortplural}{%
5961 \def\@gls@shortpl{#1}%
5962 }
```

Similarly for the long plural form.

```
5963 \define@key{glstrabbrv}{longplural}{%
5964 \def\@gls@longpl{#1}%
5965 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

```
\glsshortpltok
5966 \newtoks\glsshortpltok

\glslongpltok
5967 \newtoks\glslongpltok
```

`\glstr@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.

```
5968 \newcommand*{\@glstr@insertdots}[2]{%
5969 \def#1{%
5970 \@glstr@insert@dots#1#2\@nnil
5971 }
```



xtr@insert@dots

```
5972 \newcommand*{\@glsxtr@insert@dots}[2]{%
5973   \ifx\@nnil#2\relax
5974   \let\@glsxtr@insert@dots@next\@gobble
5975   \else
5976   \ifx\relax#2\relax
5977   \else
5978     \appto#1{#2.}%
5979   \fi
5980   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
5981   \fi
5982   \@glsxtr@insert@dots@next#1%
5983 }
```

Similarly provide a way of replacing spaces with `\glsxtrwordsep`, which first needs to be defined:

`\glsxtrwordsep`

```
5984 \newcommand*{\glsxtrwordsep}{\space}
```

Each word is marked with

`\glsxtrword`

```
5985 \newcommand*{\glsxtrword}[1]{#1}
```

tr@markwordseps

```
5986 \newcommand*{\@glsxtr@markwordseps}[2]{%
5987   \def#1{}%
5988   \@glsxtr@mark@wordseps#1#2 \@nnil
5989 }
```

r@mark@wordseps

```
5990 \def\@glsxtr@mark@wordseps#1#2 #3{%
5991   \ifdefempty{#1}%
5992   {\def#1{\protect\glsxtrword{#2}}}%
5993   {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
5994   \ifx\@nnil#3\relax
5995   \let\@glsxtr@mark@wordseps@next\relax
5996   \else
5997     \def\@glsxtr@mark@wordseps@next{%
5998       \@glsxtr@mark@wordseps#1#3}%
5999   \fi
6000   \@glsxtr@mark@wordseps@next
6001 }
```

newabbreviation Define a new generic abbreviation.

```
6002 \newcommand*{\newabbreviation}[4][ ]{%
6003   \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
6004 }
```

`\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.)

```
6005 \newcommand*{\glxtr@newabbreviation}[4]{%
6006   \glskeylisttok{#1}%
6007   \glslabeltok{#2}%
6008   \glsshorttok{#3}%
6009   \glslongtok{#4}%
```

Save the original short and long values (before attribute settings modify them).

```
6010   \def\glxtrorgshort{#3}%
6011   \def\glxtrorglong{#4}%
```

Get the category.

```
6012   \def\glscategorylabel{abbreviation}%
6013   \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
```

Ignore the shortplural and longplural keys.

```
6014   \setkeys*{\glxtrabbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
6015   \def\@gls@longpl{#4\glspluralsuffix}%
6016   \let\@gls@default@longpl\@gls@longpl
```

Has the markwords attribute been set?

```
6017   \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6018   {%
6019     \@glxtr@markwordseps\@gls@long{#4}%
6020     \expandafter\def\expandafter\@gls@longpl\expandafter
6021       {\@gls@long\glspluralsuffix}%
6022     \let\@gls@default@longpl\@gls@longpl
```

Update `\glslongtok`.

```
6023     \expandafter\glslongtok\expandafter{\@gls@long}%
6024   }%
6025   {}%
```

Has the markshortwords attribute been set? (Not compatible with `insertdots`.)

```
6026   \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
6027   {%
6028     \@glxtr@markwordseps\@gls@short{#3}%
6029   }%
6030   {}%
```

Has the `insertdots` attribute been set?

```
6031   \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
6032   {%
6033     \@glxtr@insertdots\@gls@short{#3}%
6034     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6035   }%
6036   {\def\@gls@short{#3}}%
6037   }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
6038 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%  
6039 {%  
6040 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
6041 '\abbrvpluralsuffix}%  
6042 }%  
6043 {%
```

Has the noshortplural attribute been set?

```
6044 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%  
6045 {%  
6046 \let\@gls@shortpl\@gls@short  
6047 }%  
6048 {%  
6049 \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
6050 \abbrvpluralsuffix}%  
6051 }%  
6052 }%
```

Update \glsshorttok:

```
6053 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
6054 \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).

```
6055 \setkeys*{\glsxtrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
6056 \ifx\@gls@default@longpl\@gls@longpl  
6057 \else
```

Has the markwords attribute been set?

```
6058 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%  
6059 {%  
6060 \expandafter\@glxtr@keywordseps\expandafter\@gls@longpl\expandafter  
6061 {\@gls@longpl}%  
6062 }%  
6063 {}%  
6064 \fi
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
6065 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%  
6066 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```
6067 \newabbreviationhook
```

Define this entry:

```
6068 \protected@edef\@do@newglossaryentry{%  
6069 \noexpand\newglossaryentry{\the\glslabeltok}%  
6070 {%
```

```

6071     type=\glstrabbrvtype,%
6072     category=abbreviation,%
6073     short={\the\glsshorttok},%
6074     shortplural={\the\glsshortpltok},%
6075     long={\the\glslongtok},%
6076     longplural={\the\glslongpltok},%
6077     name={\the\glsshorttok},%
6078     \CustomAbbreviationFields,%
6079     \the\glskeylisttok
6080   }%
6081 }%
6082 \do@newglossaryentry
6083 \GlsXtrPostNewAbbreviation
6084 }

```

`\evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```
6085 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```
6086 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`\bbreviationhook` Hook for use with `\newabbreviation`.

```
6087 \newcommand*{\newabbreviationhook}{}

```

`\reivationFields`

```
6088 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrparen` For the parenthetical styles.

```
6089 \newcommand*{\glstrparen}[1]{(#1)}

```

`\lsxtrfullformat` Full format without case change.

```

6090 \newcommand*{\glxtrfullformat}[2]{%
6091   \glsfirstlongfont{\glsaccesslong{#1}}#2\glxtrfullsep{#1}%
6092   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6093 }

```

`\lsxtrfullformat` Full format with case change.

```

6094 \newcommand*{\Glsxtrfullformat}[2]{%
6095   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glxtrfullsep{#1}%
6096   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6097 }

```

`\xtrfullplformat` Plural full format without case change.

```

6098 \newcommand*{\glxtrfullplformat}[2]{%
6099   \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6100   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
6101 }

```

`\glstrfullplformat` Plural full format with case change.

```
6102 \newcommand*{\Glsxtrfullplformat}[2]{%
6103   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
6104   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6105 }
```

`\glstrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
6106 \newcommand*{\glstrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glstrinlinefullformat` Full format without case change.

```
6107 \newcommand*{\Glsxtrinlinefullformat}{\glstrfullformat}
```

`\glstrinlinefullformat` Full format with case change.

```
6108 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glstrfullplformat` Plural full format without case change.

```
6109 \newcommand*{\Glsxtrinlinefullplformat}{\glstrfullplformat}
```

`\glstrinlinefullplformat` Plural full format with case change.

```
6110 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

Redefine `\glentryfull` 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 `\glstrfull` set of commands instead.

`\glentryfull`

```
6111 \renewcommand*{\glentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
6112 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\glentryfullpl`

```
6113 \renewcommand*{\glentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
6114 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
6115 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
6116 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
6117 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

bbvrdefaultfont

```
6118 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
6119 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
6120 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
6121 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
6122 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
6123 \newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
6124 \newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
6125 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
6126 \newcommand*{\ns@glsxtrfull}[2][\%
```

```
6127 \new@ifnextchar[\@glsxtr@full{#1}{#2}}%
```

```
6128 \@glsxtr@full{#1}{#2}[]}%
```

```
6129 }
```

`\@glsxtr@full` Low-level macro:

```
6130 \def\@glsxtr@full#1#2[#3]{%
```

```
6131 \glsdoifexists{#2}}%
```

```
6132 {%
```

```
6133 \glssetabbrvfmt{\glscategory{#2}}%
```

```
6134 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
6135 \let\glsifplural\@secondoftwo
```

```
6136 \let\glsupcase\@firstofthree
```

```
6137 \let\glsinsert\@empty
```

```
6138 \def\glscustomtext{\glsxtrinlinefullformat{#2}{#3}}%
```

What should `\glsxtrifwasfirstuse` 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.

```
6139 \glsxtrsetupfulldefs
```

```
6140 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
```

```
6141 }%
```

```
6142 \glspostlinkhook
```

```
6143 }
```

trsetupfulldefs

```
6144 \newcommand*{\glxtrsetupfulldefs}{%
6145   \let\glxtrifwasfirstuse\@firstoftwo
6146 }
```

\Glsxtrfull Full form (first letter uppercase).

```
6147 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
6148 \newcommand*\ns@Glsxtrfull[2][]{%
6149   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
6150     {\@Glsxtr@full{#1}{#2}[]}%
6151 }
```

\@Glsxtr@full Low-level macro:

```
6152 \def\@Glsxtr@full#1#2[#3]{%
6153   \glsdoifexists{#2}%
6154   {%
6155     \glssetabbrvfmt{\glscategory{#2}}%
6156     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6157     \let\glsifplural\@secondoftwo
6158     \let\glscapscase\@secondofthree
6159     \let\glsinsert\@empty
6160     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6161     \glxtrsetupfulldefs
6162     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6163   }%
6164   \glspostlinkhook
6165 }
```

\GLSxtrfull Full form (all uppercase).

```
6166 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
6167 \newcommand*\ns@GLSxtrfull[2][]{%
6168   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}{%
6169     {\@GLSxtr@full{#1}{#2}[]}%
6170 }
```

\@GLSxtr@full Low-level macro:

```
6171 \def\@GLSxtr@full#1#2[#3]{%
6172   \glsdoifexists{#2}%
6173   {%
6174     \glssetabbrvfmt{\glscategory{#2}}%
6175     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6176     \let\glsifplural\@secondoftwo
6177     \let\glscapscase\@thirdofthree
6178     \let\glsinsert\@empty
6179     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
6180     \glxtrsetupfulldefs
6181     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6182   }%
6183   \glspostlinkhook
```

6184 }

\glstrfullpl Plural full form (no case-change).

```
6185 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\@ns@glstrfullpl}
6186 \newcommand*\ns@glstrfullpl[2][]{%
6187   \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}%
6188     {\@glstr@fullpl{#1}{#2}[]}%
6189 }
```

\@glstr@fullpl Low-level macro:

```
6190 \def\@glstr@fullpl#1#2[#3]{%
6191   \glstoifexists{#2}%
6192   {%
6193     \glsssetabbrvfmt{\glscategory{#2}}%
6194     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6195     \let\gl@ifplural\@firstoftwo
6196     \let\glscapscase\@firstofthree
6197     \let\glinsert\@empty
6198     \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
6199     \glstrsetupfulldefs
6200     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6201   }%
6202   \glspostlinkhook
6203 }
```

\Glsxtrfullpl Plural full form (first letter uppercase).

```
6204 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6205 \newcommand*\ns@Glsxtrfullpl[2][]{%
6206   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6207     {\@Glsxtr@fullpl{#1}{#2}[]}%
6208 }
```

\@Glsxtr@fullpl Low-level macro:

```
6209 \def\@Glsxtr@fullpl#1#2[#3]{%
6210   \glstoifexists{#2}%
6211   {%
6212     \glsssetabbrvfmt{\glscategory{#2}}%
6213     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6214     \let\gl@ifplural\@firstoftwo
6215     \let\glscapscase\@secondofthree
6216     \let\glinsert\@empty
6217     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6218     \glstrsetupfulldefs
6219     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6220   }%
6221   \glspostlinkhook
6222 }
```

\GLSxtrfullpl Plural full form (all upper case).



```

6223 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
6224 \newcommand*\ns@GLSxtrfullpl[2][{}]{%
6225   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
6226     {\@GLSxtr@fullpl{#1}{#2}[]}%
6227 }

```

\@GLSxtr@fullpl Low-level macro:

```

6228 \def\@GLSxtr@fullpl#1#2[#3]{%
6229   \glsdoifexists{#2}%
6230   {%
6231     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6232     \let\glsifplural\@firstoftwo
6233     \let\glscapscase\@thirdofthree
6234     \let\glsinsert\@empty
6235     \def\glscustomtext{%
6236       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6237     \glsxtrsetupfulldefs
6238     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6239   }%
6240   \glspostlinkhook
6241 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

6242 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6243 \newcommand*{\ns@glsxtrshort}[2][{}]{%
6244   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}]{\@glsxtrshort{#1}{#2}[]}%
6245 }

```

Read in the final optional argument:

```

6246 \def\@glsxtrshort#1#2[#3]{%
6247   \glsdoifexists{#2}%
6248   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

6249   \glssetabbrvfmt{\glscategory{#2}}%
6250   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6251   \let\glsxtrifwasfirstuse\@secondoftwo
6252   \let\glsifplural\@secondoftwo
6253   \let\glscapscase\@firstofthree
6254   \let\glsinsert\@empty
6255   \def\glscustomtext{%
6256     \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
6257     \ifglsxtrininsertinside\else#3\fi
6258   }%
6259   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6260 }%
6261 \glspostlinkhook

```

6262 }

\Glsxtrshort

6263 \newrobustcmd\*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

6264 \newcommand\*{\ns@Glsxtrshort}[2][\%

6265 \new@ifnextchar[\@Glsxtrshort{#1}{#2}]{\@Glsxtrshort{#1}{#2}[]}%

6266 }

Read in the final optional argument:

6267 \def\@Glsxtrshort#1#2[#3]{\%

6268 \glsdoifexists{#2}%

6269 {\%

6270 \glssetabbrvfmt{\glscategory{#2}}%

6271 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

6272 \let\glsxtrifwasfirstuse\@secondoftwo

6273 \let\glsifplural\@secondoftwo

6274 \let\glsapscase\@secondofthree

6275 \let\glsinsert\@empty

6276 \def\glscustomtext{\%

6277 \glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%

6278 \ifglsxtrininsertinside\else#3\fi

6279 }%

6280 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

6281 }%

6282 \glspostlinkhook

6283 }

\GLSxtrshort

6284 \newrobustcmd\*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

6285 \newcommand\*{\ns@GLSxtrshort}[2][\%

6286 \new@ifnextchar[\@GLSxtrshort{#1}{#2}]{\@GLSxtrshort{#1}{#2}[]}%

6287 }

Read in the final optional argument:

6288 \def\@GLSxtrshort#1#2[#3]{\%

6289 \glsdoifexists{#2}%

6290 {\%

6291 \glssetabbrvfmt{\glscategory{#2}}%

6292 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

6293 \let\glsxtrifwasfirstuse\@secondoftwo

6294 \let\glsifplural\@secondoftwo

6295 \let\glsapscase\@thirdofthree

6296 \let\glsinsert\@empty

6297 \def\glscustomtext{\%

6298 \mfirstucMakeUppercase

6299 {\glsabbrvfont{\Glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%

6300 \ifglsxtrininsertinside\else#3\fi

```

6301     }%
6302     }%
6303     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6304     }%
6305     \glspostlinkhook
6306 }

```

`\glsxtrlong`

```

6307 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6308 \newcommand*{\ns@glsxtrlong}[2] [] {%
6309   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}%
6310 }

    Read in the final optional argument:
6311 \def\@glsxtrlong#1#2[#3]{%
6312   \glsdoifexists{#2}%
6313   {%
6314     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6315     \let\glsxtrifwasfirstuse\@secondoftwo
6316     \let\glsifplural\@secondoftwo
6317     \let\glscapscase\@firstofthree
6318     \let\glsinsert\@empty
6319     \def\glscustomtext{%
6320       \glsfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
6321       \ifglsxtrininsertinside\else#3\fi
6322     }%
6323     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6324     }%
6325     \glspostlinkhook
6326 }

```

`\Glsxtrlong`

```

6327 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6328 \newcommand*{\ns@Glsxtrlong}[2] [] {%
6329   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
6330 }

    Read in the final optional argument:
6331 \def\@Glsxtrlong#1#2[#3]{%
6332   \glsdoifexists{#2}%
6333   {%
6334     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6335     \let\glsxtrifwasfirstuse\@secondoftwo
6336     \let\glsifplural\@secondoftwo
6337     \let\glscapscase\@secondofthree
6338     \let\glsinsert\@empty
6339     \def\glscustomtext{%

```

```

6340     \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6341     \ifglxtrinsertinside\else#3\fi
6342 }%
6343     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6344 }%
6345 \glspostlinkhook
6346 }

```

\GLSxtrlong

```

6347 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

Define the un-starred form. Need to determine if there is a final optional argument
6348 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6349   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6350 }

Read in the final optional argument:
6351 \def\@GLSxtrlong#1#2[#3] {%
6352   \glsdoifexists{#2}%
6353   {%
6354     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6355     \let\glxtrifwasfirstuse\@secondoftwo
6356     \let\glsifplural\@secondoftwo
6357     \let\glscapscase\@thirdofthree
6358     \let\glsinsert\@empty
6359     \def\glscustomtext{%
6360       \mfirstucMakeUppercase
6361       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6362       \ifglxtrinsertinside\else#3\fi
6363     }%
6364   }%
6365   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6366 }%
6367 \glspostlinkhook
6368 }

```

Plural short forms:

\glxtrshortpl

```

6369 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

Define the un-starred form. Need to determine if there is a final optional argument
6370 \newcommand*{\ns@glxtrshortpl}[2] [] {%
6371   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
6372 }

Read in the final optional argument:
6373 \def\@glxtrshortpl#1#2[#3] {%
6374   \glsdoifexists{#2}%
6375   {%
6376     \glsssetabbrvfmt{\glscategory{#2}}%

```

```

6377 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6378 \let\glxtrifwasfirstuse\@secondoftwo
6379 \let\gl@sifplural\@firstoftwo
6380 \let\glscapscase\@firstofthree
6381 \let\gl$insert\@empty
6382 \def\glscustomtext{%
6383 \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6384 \ifglxtrininsertinside\else#3\fi
6385 }%
6386 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6387 }%
6388 \glspostlinkhook
6389 }

```

\Glsxtrshortpl

```

6390 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6391 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
6392 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
6393 }
    Read in the final optional argument:
6394 \def\@Glsxtrshortpl#1#2[#3] {%
6395 \glsoifexists{#2}%
6396 {%
6397 \glsetabbrvfmt{\glscategory{#2}}%
6398 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6399 \let\glxtrifwasfirstuse\@secondoftwo
6400 \let\gl@sifplural\@firstoftwo
6401 \let\glscapscase\@secondofthree
6402 \let\gl$insert\@empty
6403 \def\glscustomtext{%
6404 \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6405 \ifglxtrininsertinside\else#3\fi
6406 }%
6407 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6408 }%
6409 \glspostlinkhook
6410 }

```

\GLSxtrshortpl

```

6411 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6412 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6413 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6414 }
    Read in the final optional argument:
6415 \def\@GLSxtrshortpl#1#2[#3] {%

```

```

6416 \glsdoifexists{#2}%
6417 {%
6418   \glssetabbrvfmt{\glscategory{#2}}%
6419   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6420   \let\glsxtrifwasfirstuse\@secondoftwo
6421   \let\glsifplural\@firstoftwo
6422   \let\glsapscase\@thirdofthree
6423   \let\glsinsert\@empty
6424   \def\glscustomtext{%
6425     \mfirstucMakeUppercase
6426     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6427     \ifglsxtrininsertinside\else#3\fi
6428   }%
6429 }%
6430 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6431 }%
6432 \glspostlinkhook
6433 }

```

Plural long forms:

\glsxtrlongpl

```

6434 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6435 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
6436   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
6437 }

```

Read in the final optional argument:

```

6438 \def\@glsxtrlongpl#1#2[#3]{%
6439   \glsdoifexists{#2}%
6440   {%
6441     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6442     \let\glsxtrifwasfirstuse\@secondoftwo
6443     \let\glsifplural\@firstoftwo
6444     \let\glsapscase\@firstofthree
6445     \let\glsinsert\@empty
6446     \def\glscustomtext{%
6447       \glsfont{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
6448       \ifglsxtrininsertinside\else#3\fi
6449     }%
6450     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6451   }%
6452   \glspostlinkhook
6453 }

```

\Glsxtrlongpl

```

6454 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
6455 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
6456   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
6457 }
```

Read in the final optional argument:

```
6458 \def\@Glsxtrlongpl#1#2[#3]{%
6459   \glsdoifexists{#2}%
6460   {%
6461     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6462     \let\glxtrifwasfirstuse\@secondoftwo
6463     \let\glsifplural\@firstoftwo
6464     \let\glscapscase\@secondofthree
6465     \let\glsinsert\@empty
6466     \def\glscustomtext{%
6467       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6468       \ifglxtrininsertinside\else#3\fi
6469     }%
6470     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6471   }%
6472   \glspostlinkhook
6473 }
```

\GLSxtrlongpl

```
6474 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6475 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
6476   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
6477 }
```

Read in the final optional argument:

```
6478 \def\@GLSxtrlongpl#1#2[#3]{%
6479   \glsdoifexists{#2}%
6480   {%
6481     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6482     \let\glxtrifwasfirstuse\@secondoftwo
6483     \let\glsifplural\@firstoftwo
6484     \let\glscapscase\@thirdofthree
6485     \let\glsinsert\@empty
6486     \def\glscustomtext{%
6487       \mfirstucMakeUppercase
6488       {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6489       \ifglxtrininsertinside\else#3\fi
6490     }%
6491   }%
6492   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6493 }%
6494   \glspostlinkhook
6495 }
```

`\glsssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6496 \newcommand*{\glsssetabbrvfmt}[1]{%
6497   \ifcsdef{@glsabbrv@current@#1}%
6498     {\glstr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6499     {\glstr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
6500 }

```

`\glssuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6501 \newrobustcmd*{\glssuseabbrvfont}[2]{\glsssetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\glssuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6502 \newrobustcmd*{\glssuselongfont}[2]{\glsssetabbrvfmt{#2}\glslongfont{#1}}

```

`\glstrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6503 \newcommand*{\glstrgenabbrvfmt}{%
6504   \ifdefempty\glscustomtext
6505     {%
6506       \ifglssused\glslabel
6507         {%

```

Subsequent use:

```

6508       \glsifplural
6509       {%

```

Subsequent plural form:

```

6510       \glscapscase
6511       {%

```

Subsequent plural form, don't adjust case:

```

6512       \glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6513       }%
6514       {%

```

Subsequent plural form, make first letter upper case:

```

6515       \Glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6516       }%
6517       {%

```

Subsequent plural form, all caps:

```

6518       \mfirstucMakeUppercase
6519       {\glstrsubsequentplfmt{\glslabel}{\glsinsert}}%
6520       }%
6521       }%
6522       {%

```

Subsequent singular form

```

6523       \glscapscase
6524       {%

```



Subsequent singular form, don't adjust case:

```
6525      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6526      }%  
6527      {%
```

Subsequent singular form, make first letter upper case:

```
6528      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6529      }%  
6530      {%
```

Subsequent singular form, all caps:

```
6531      \mfirstucMakeUppercase  
6532      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6533      }%  
6534      }%  
6535      }%  
6536      {%
```

First use:

```
6537      \glsifplural  
6538      {%
```

First use plural form:

```
6539      \glscapscase  
6540      {%
```

First use plural form, don't adjust case:

```
6541      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6542      }%  
6543      {%
```

First use plural form, make first letter upper case:

```
6544      \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
6545      }%  
6546      {%
```

First use plural form, all caps:

```
6547      \mfirstucMakeUppercase  
6548      {\glxtrfullplformat{\glslabel}{\glsinsert}}%  
6549      }%  
6550      }%  
6551      {%
```

First use singular form

```
6552      \glscapscase  
6553      {%
```

First use singular form, don't adjust case:

```
6554      \glxtrfullformat{\glslabel}{\glsinsert}%  
6555      }%  
6556      {%
```

First use singular form, make first letter upper case:

```
6557      \Glsxtrfullformat{\glslabel}{\glsinsert}%  
6558      }%  
6559      {%
```

First use singular form, all caps:

```
6560      \mfirstucMakeUppercase  
6561      {\glsxtrfullformat{\glslabel}{\glsinsert}}%  
6562      }%  
6563      }%  
6564      }%  
6565      }%  
6566      {%
```

User supplied text.

```
6567      \glscustomtext  
6568      }%  
6569      }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6570 \newcommand*{\glsxtrsubsequentfmt}[2]{%  
6571   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%  
6572   \ifglsxtrininsertinside \else#2\fi  
6573 }  
6574 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6575 \newcommand*{\glsxtrsubsequentplfmt}[2]{%  
6576   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%  
6577   \ifglsxtrininsertinside \else#2\fi  
6578 }  
6579 \let\glsxtrdefaultsubsequentplfmt\glsxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
6580 \newcommand*{\Glsxtrsubsequentfmt}[2]{%  
6581   \glsabbrvfont{\Glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%  
6582   \ifglsxtrininsertinside \else#2\fi  
6583 }  
6584 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```
6585 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%  
6586   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%  
6587   \ifglsxtrininsertinside \else#2\fi  
6588 }  
6589 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt
```

## 1.6.1 Abbreviation Styles Setup

abbreviationstyle

```
6590 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6591   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6592   {%
6593     \PackageError{glossaries-extra}{Undefined abbreviation style '#2'}{}}%
6594   }%
6595   {%
```

Have abbreviations already been defined for this category?

```
6596   \ifcsstring{@glsabbrv@current@#1}{#2}%
6597   {%
```

Style already set.

```
6598   }%
6599   {%
6600     \def\@glsxtr@dostylewarn{%
6601       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6602       {%
6603         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
6604           style has been switched \MessageBreak
6605           for category '#1', \MessageBreak
6606           but there have already been entries \MessageBreak
6607           defined for this category. Unwanted \MessageBreak
6608           side-effects may result}}}%
6609       \@endfortrue
6610     }%
6611     \@glsxtr@dostylewarn
```

Set up the style for the given category.

```
6612     \csdef{@glsabbrv@current@#1}{#2}%
6613     \glsxtr@applyabbrvstyle{#2}%
6614   }%
6615 }%
6616 }
```

applyabbrvstyle Apply the abbreviation style without existence check.

```
6617 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
6618   \csuse{@glsabbrv@dispstyle@setup@#1}%
6619   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6620 }
```

r@applyabbrvfmt Only apply the style formats.

```
6621 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
6622   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6623 }
```

abbreviationstyle 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.

```

6624 \newcommand*{\newabbreviationstyle}[3]{%
6625   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6626   {%
6627     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6628       defined}{}%
6629   }%
6630   {%
6631     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6632       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6633       #2}%
6634     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6635       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6636       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6637       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6638       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
        Reset \glsxtrsubsequentfmt etc in case a style changes this.
6639       \let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt
6640       \let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt
6641       \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6642       \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6643       #3}%
6644     }%
6645 }

```

breivationstyle

```

6646 \newcommand*{\renewabbreviationstyle}[3]{%
6647   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
6648   {%
6649     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6650   }%
6651   {%
6652     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6653       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6654       #2}%
6655     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6656       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6657       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6658       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6659       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6660       #3}%
6661     }%
6662 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```
6663 \newcommand*{\letabbreviationstyle}[2]{%
6664   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6665   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6666 }
```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```
6667 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6668   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6669     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6670     \csuse{@glsabbrv@dispstyle@setup@#2}%
6671   }%
6672   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6673 }
```

deprecatedAbbrStyle Generate warning for deprecated style use.

```
6674 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6675   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6676   use ‘#2’ instead}%
6677 }
```

useAbbrStyleSetup

```
6678 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6679   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
6680   {%
6681     \PackageError{glossaries-extra}%
6682     {Unknown abbreviation style definitions ‘#1’}{}%
6683   }%
6684   {%
6685     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6686   }%
6687 }
```

useAbbrStyleFmt

```
6688 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
6689   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6690   {%
6691     \PackageError{glossaries-extra}%
6692     {Unknown abbreviation style formats ‘#1’}{}%
6693   }%
6694   {%
6695     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6696   }%
6697 }
```

## 1.6.2 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, `\glstrfullformat` 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.

```
6698 \newif\ifglstrinsertinside
6699 \glstrinsertinsidefalse
```

`long-short`

```
6700 \newabbreviationstyle{long-short}%
6701 {%
6702   \renewcommand*{\CustomAbbreviationFields}{%
6703     name={\protect\glsabbrvfont{\the\glsshorttok}},
6704     sort={\the\glsshorttok},
6705     first={\protect\glsfirstlongfont{\the\glslongtok}%
6706       \protect\glstrfullsep{\the\glslabeltok}%
6707       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6708     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6709       \protect\glstrfullsep{\the\glslabeltok}%
6710       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
6711     plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
6712     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
6713 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6714   \glshasattribute{\the\glslabeltok}{regular}%
6715   {%
6716     \glissetattribute{\the\glslabeltok}{regular}{false}%
6717   }%
6718   {}%
6719 }%
6720 }%
6721 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6722 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6723 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6724 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6725 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6726 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

6727 \renewcommand*{\glxtrfullformat}[2]{%
6728   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6729   \ifglxtrininsertinside\else##2\fi
6730   \glxtrfullsep{##1}%
6731   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6732 }%
6733 \renewcommand*{\glxtrfullplformat}[2]{%
6734   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6735   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6736   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6737 }%
6738 \renewcommand*{\Glsxtrfullformat}[2]{%
6739   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
6740   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6741   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6742 }%
6743 \renewcommand*{\Glsxtrfullplformat}[2]{%
6744   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
6745   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6746   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6747 }%
6748 }

```

Set this as the default style for general abbreviations:

```

6749 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

6750 \newcommand*{\glxtrlongshortdescsort}{%
6751   \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6752 }

```

ngshortdescname

```

6753 \newcommand*{\glxtrlongshortdescname}{%
6754   \protect\glslongfont{\the\glslongtok}
6755   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6756 }

```

long-short-desc User supplies description. The long form is included in the name.

```

6757 \newabbreviationstyle{long-short-desc}%
6758 {%
6759   \renewcommand*{\CustomAbbreviationFields}{%
6760     name={\glxtrlongshortdescname},
6761     sort={\glxtrlongshortdescsort},%
6762     first={\protect\glsfirstlongfont{\the\glslongtok}%
6763       \protect\glxtrfullsep{\the\glslabeltok}%
6764       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6765     firstplural={\protect\glsfirstlongfont{\the\glslongtok}%
6766       \protect\glxtrfullsep{\the\glslabeltok}%

```

```
6767 \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
6768 text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```
6769 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6770 }%
```

Unset the regular attribute if it has been set.

```
6771 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6772 \glshasattribute{\the\glslabeltok}{regular}%
6773 {%
6774 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6775 }%
6776 {}%
6777 }%
6778 }%
6779 {%
6780 \GlsXtrUseAbbrStyleFmts{long-short}%
6781 }
```

**short-long** Short form followed by long form in parenthesis on first use.

```
6782 \newabbreviationstyle{short-long}%
6783 {%
6784 \renewcommand*{\CustomAbbreviationFields}{%
6785 name={\protect\glsabbrvfont{\the\glsshorttok}},
6786 sort={\the\glsshorttok},
6787 description={\the\glslongtok},%
6788 first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6789 \protect\glstrfullsep{\the\glslabeltok}%
6790 \glstrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6791 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6792 \protect\glstrfullsep{\the\glslabeltok}%
6793 \glstrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6794 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
6795 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6796 \glshasattribute{\the\glslabeltok}{regular}%
6797 {%
6798 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6799 }%
6800 {}%
6801 }%
6802 }%
6803 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
6804 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6805 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%
```



```

6806 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6807 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6808 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6809 \renewcommand*{\glsxtrfullformat}[2]{%
6810   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6811   \ifglsxtrininsertinside\else##2\fi
6812   \glsxtrfullsep{##1}%
6813   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6814 }%
6815 \renewcommand*{\glsxtrfullplformat}[2]{%
6816   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6817   \ifglsxtrininsertinside\else##2\fi
6818   \glsxtrfullsep{##1}%
6819   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6820 }%
6821 \renewcommand*{\Glsxtrfullformat}[2]{%
6822   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6823   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6824   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6825 }%
6826 \renewcommand*{\Glsxtrfullplformat}[2]{%
6827   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6828   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6829   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6830 }%
6831 }

```

ortlongdescsort

```

6832 \newcommand*{\glsxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

6833 \newcommand*{\glsxtrshortlongdescname}{%
6834   \protect\glsabbrvfont{\the\glsshorttok}
6835   \glsxtrparen{\protect\glslongfont{\the\glslongtok}}}%
6836 }

```

short-long-desc User supplies description. The long form is included in the name.

```

6837 \newabbreviationstyle{short-long-desc}%
6838 {%
6839   \renewcommand*{\CustomAbbreviationFields}{%
6840     name={\glsxtrshortlongdescname},
6841     sort={\glsxtrshortlongdescsort},
6842     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6843     \protect\glsxtrfullsep{\the\glslabeltok}}%
6844     \glsxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6845     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6846     \protect\glsxtrfullsep{\the\glslabeltok}}%
6847     \glsxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%

```

```

6848     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6849     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6850 }%
    Unset the regular attribute if it has been set.
6851 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6852   \glshasattribute{\the\glslabeltok}{regular}%
6853   {%
6854     \glissetattribute{\the\glslabeltok}{regular}{false}%
6855   }%
6856   {}}%
6857 }%
6858 }%
6859 {%
6860   \GlsXtrUseAbbrStyleFmts{short-long}%
6861 }

```

ongfootnotefont Only used by the “footnote” styles.

```
6862 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6863 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote `\glsxtrabbrvfootnote{<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, `\gls` or `\glspl`).

```
6864 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```

6865 \newabbreviationstyle{footnote}%
6866 {%
6867   \renewcommand*{\CustomAbbreviationFields}{%
6868     name={\protect\glsabbrvfont{\the\glsshorttok}},
6869     sort={\the\glsshorttok},
6870     description={\the\glslongtok},%
6871     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6872     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6873     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6874     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6875     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6876     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

```

```
6877 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.

```
6878 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6879   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6880   \glsattribute{\the\glslabeltok}{regular}%
6881   {%
6882     \glssetattribute{\the\glslabeltok}{regular}{false}%
6883   }%
6884   {}%
6885 }%
6886 }%
6887 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6888 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6889 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6890 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6891 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6892 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
6893 \renewcommand*{\glsxtrfullformat}[2]{%
6894   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6895   \ifglsxtrininsertinside\else##2\fi
6896   \protect\glsxtrabbrvfootnote{##1}%
6897   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6898 }%
6899 \renewcommand*{\glsxtrfullplformat}[2]{%
6900   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6901   \ifglsxtrininsertinside\else##2\fi
6902   \protect\glsxtrabbrvfootnote{##1}%
6903   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6904 }%
6905 \renewcommand*{\Glsxtrfullformat}[2]{%
6906   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6907   \ifglsxtrininsertinside\else##2\fi
6908   \protect\glsxtrabbrvfootnote{##1}%
6909   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6910 }%
6911 \renewcommand*{\Glsxtrfullplformat}[2]{%
6912   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6913   \ifglsxtrininsertinside\else##2\fi
6914   \protect\glsxtrabbrvfootnote{##1}%
6915   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6916 }%
```

The first use full form and the inline full form use the short (long) style.

```
6917 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6918   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6919   \ifglsxtrininsertinside\else##2\fi
6920   \protect\glsxtrabbrvfootnote{##1}%
6921   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6922 }%
```

```

6919     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6920     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
6921 }%
6922 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6923     \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6924     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6925     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
6926 }%
6927 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6928     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6929     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6930     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
6931 }%
6932 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6933     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6934     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6935     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
6936 }%
6937 }

```

#### short-footnote

```
6938 \letabbreviationstyle{short-footnote}{footnote}
```

postfootnote Similar to the above 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.

```

6939 \newabbreviationstyle{postfootnote}%
6940 {%
6941     \renewcommand*{\CustomAbbreviationFields}{%
6942         name={\protect\glssabbrvfont{\the\glssshorttok}},
6943         sort={\the\glssshorttok},
6944         description={\the\glslongtok},%
6945         first={\protect\glsfirstabbrvfont{\the\glssshorttok}},%
6946         firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},%
6947         plural={\protect\glssabbrvfont{\the\glssshortpltok}}}%

```

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.

```

6948 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6949     \csdef{glxtrpostlink\glscategorylabel}{%
6950         \glxtrifwasfirstuse
6951         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6952         \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
6953             {\glsfirstlongfootnotefont{\glssentrylong{\glslabel}}}}%
6954     }%

```

```

6955     {}%
6956   }%
6957   \glshasattribute{\the\glslabeltok}{regular}%
6958   {%
6959     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
6960   }%
6961   {}%
6962 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

6963 \renewcommand*\glxtrsetupfulldefs{%
6964   \let\glxtrifwasfirstuse\@secondoftwo
6965 }%
6966 }%
6967 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6968 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6969 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6970 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6971 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6972 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6973 \renewcommand*\glxtrfullformat[2]{%
6974   \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6975   \ifglxtrininsertinside\else##2\fi
6976 }%
6977 \renewcommand*\glxtrfullplformat[2]{%
6978   \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6979   \ifglxtrininsertinside\else##2\fi
6980 }%
6981 \renewcommand*\Glsxtrfullformat[2]{%
6982   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6983   \ifglxtrininsertinside\else##2\fi
6984 }%
6985 \renewcommand*\Glsxtrfullplformat[2]{%
6986   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6987   \ifglxtrininsertinside\else##2\fi
6988 }%

```

The first use full form and the inline full form use the short (long) style.

```

6989 \renewcommand*\glxtrininlinefullformat[2]{%
6990   \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6991   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6992   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
6993 }%
6994 \renewcommand*\glxtrininlinefullplformat[2]{%
6995   \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6996   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

6997 \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6998 }%
6999 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7000 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7001 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7002 \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7003 }%
7004 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7005 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7006 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7007 \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7008 }%
7009 }

```

rt-postfootnote

```
7010 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

**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.

```

7011 \newabbreviationstyle{short}%
7012 {%
7013 \renewcommand*{\CustomAbbreviationFields}{%
7014 name={\protect\glssabbrvfont{\the\glssshorttok}},
7015 sort={\the\glssshorttok},
7016 first={\protect\glsfirstabbrvfont{\the\glssshorttok}},
7017 firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},
7018 text={\protect\glssabbrvfont{\the\glssshorttok}},
7019 plural={\protect\glssabbrvfont{\the\glssshortpltok}},
7020 description={\the\glslongtok}}%
7021 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7022 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7023 }%
7024 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7025 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7026 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
7027 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7028 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7029 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7030 \renewcommand*{\glxtrinlinefullformat}[2]{%
7031 \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7032 \ifglxtrinsertinside##2\fi}%
7033 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7034 \glxstrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7035 }%

```

```

7036 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7037   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
7038     \ifglxtrininsertinside##2\fi}%
7039   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7040   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7041 }%
7042 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7043   \protect\glsfirstabbrvfont{\glsaccessshort{##1}%
7044     \ifglxtrininsertinside##2\fi}%
7045   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7046   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7047 }%
7048 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7049   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
7050     \ifglxtrininsertinside##2\fi}%
7051   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7052   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7053 }%

```

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.

```

7054 \renewcommand*{\glxtrfullformat}[2]{%
7055   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7056   \ifglxtrininsertinside\else##2\fi
7057 }%
7058 \renewcommand*{\glxtrfullplformat}[2]{%
7059   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7060   \ifglxtrininsertinside\else##2\fi
7061 }%
7062 \renewcommand*{\Glsxtrfullformat}[2]{%
7063   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7064   \ifglxtrininsertinside\else##2\fi
7065 }%
7066 \renewcommand*{\Glsxtrfullplformat}[2]{%
7067   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7068   \ifglxtrininsertinside\else##2\fi
7069 }%
7070 }

```

Set this as the default style for acronyms:

```

7071 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7072 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7073 \newabbreviationstyle{short-nolong-noreg}%
7074 {%
7075   \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7076 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7077   \glshasattribute{\the\glslabeltok}{regular}%
7078   {%
7079     \glsssetattribute{\the\glslabeltok}{regular}{false}%
7080   }%
7081   {}%
7082 }%
7083}%
7084{%
7085   \GlsXtrUseAbbrStyleFmts{short-nolong}%
7086 }

```

trshortdescname

```

7087 \newcommand*{\glxtrshortdescname}{%
7088   \protect\glsabbrvfont{\the\glssshorttok}%
7089 }

```

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.

```

7090 \newabbreviationstyle{short-desc}%
7091 {%
7092   \renewcommand*{\CustomAbbreviationFields}{%
7093     name={\glxtrshortdescname},
7094     sort={\the\glssshorttok},
7095     first={\protect\glsfirstabbrvfont{\the\glssshorttok}},
7096     firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},
7097     text={\protect\glsabbrvfont{\the\glssshorttok}},
7098     plural={\protect\glsabbrvfont{\the\glssshortpltok}},
7099     description={\the\glslongtok}}%
7100   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7101     \glsssetattribute{\the\glslabeltok}{regular}{true}}%
7102 }%
7103 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7104 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7105 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7106 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7107 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7108 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7109 \renewcommand*{\glxtrinlinefullformat}[2]{%
7110   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7111   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7112   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7113 }%
7114 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7115   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```



```

7116 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7117 \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
7118 }%
7119 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7120 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7121 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7122 \glxtrparen{\glsfirstlongfont{\glssaccesslong{##1}}}%
7123 }%
7124 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7125 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7126 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7127 \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
7128 }%

```

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.

```

7129 \renewcommand*{\glxtrfullformat}[2]{%
7130 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7131 \ifglxtrinsertinside\else##2\fi
7132 }%
7133 \renewcommand*{\glxtrfullplformat}[2]{%
7134 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7135 \ifglxtrinsertinside\else##2\fi
7136 }%
7137 \renewcommand*{\Glsxtrfullformat}[2]{%
7138 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7139 \ifglxtrinsertinside\else##2\fi
7140 }%
7141 \renewcommand*{\Glsxtrfullplformat}[2]{%
7142 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7143 \ifglxtrinsertinside\else##2\fi
7144 }%
7145 }

```

ort-nolong-desc

```
7146 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7147 \newabbreviationstyle{short-nolong-desc-noreg}%
7148 {%
7149 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

7150 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7151 \glshasattribute{\the\glslabeltok}{regular}%
7152 {%
7153 \glsssetattribute{\the\glslabeltok}{regular}{false}%
7154 }%
7155 {}%
7156 }%

```

```

7157 }%
7158 {%
7159   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7160 }

```

**nolong-short** Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7161 \newabbreviationstyle{nolong-short}%
7162 {%
7163   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7164 }%
7165 {%
7166   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7167 \renewcommand*{\glxtrinlinefullformat}[2]{%
7168   \protect\glstlongfont{\glsaccesslong{##1}%
7169     \ifglxtrinsertinside##2\fi}%
7170   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7171   \glxtrparen{\glstabbrfont{\glsaccessshort{##1}}}%
7172 }%
7173 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7174   \protect\glstlongfont{\glsaccesslongpl{##1}%
7175     \ifglxtrinsertinside##2\fi}%
7176   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7177   \glxtrparen{\glstabbrfont{\glsaccessshortpl{##1}}}%
7178 }%
7179 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7180   \protect\glstlongfont{\glsaccesslong{##1}%
7181     \ifglxtrinsertinside##2\fi}%
7182   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7183   \glxtrparen{\glstabbrfont{\Glsaccessshort{##1}}}%
7184 }%
7185 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7186   \protect\glstlongfont{\glsaccesslongpl{##1}%
7187     \ifglxtrinsertinside##2\fi}%
7188   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7189   \glxtrparen{\glstabbrfont{\Glsaccessshortpl{##1}}}%
7190 }%
7191 }

```

**nolong-short-noreg** Like nolong-short but doesn't set the regular attribute.

```

7192 \newabbreviationstyle{nolong-short-noreg}%
7193 {%
7194   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7195 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7196   \glshasattribute{\the\glstlabeltok}{regular}%
7197   {%

```

```

7198     \glssetattribute{\the\glslabeltok}{regular}{false}%
7199   }%
7200   {}%
7201 }%
7202 }%
7203 {%
7204   \GlsXtrUseAbbrStyleFmts{nolong-short}%
7205 }

```

**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.

```

7206 \newabbreviationstyle{long-desc}%
7207 {%
7208   \renewcommand*{\CustomAbbreviationFields}{%
7209     name={\protect\protect\glslongfont{\the\glslongtok}},
7210     sort={\the\glslongtok},
7211     first={\protect\glsfirstlongfont{\the\glslongtok}},
7212     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7213     text={\glslongfont{\the\glslongtok}},
7214     plural={\glslongfont{\the\glslongpltok}}}%
7215   }%
7216   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7217     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7218 }%
7219 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7220 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
7221 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7222 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7223 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7224 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7225 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
7226   \glslongfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7227   \ifglsxtrininsertinside \else##2\fi
7228 }%
7229 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
7230   \glslongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7231   \ifglsxtrininsertinside \else##2\fi
7232 }%
7233 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7234   \glslongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7235   \ifglsxtrininsertinside \else##2\fi
7236 }%
7237 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7238   \glslongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7239   \ifglsxtrininsertinside \else##2\fi

```

7240 }%

The inline full form displays the long format followed by the short form in parentheses.

```

7241 \renewcommand*{\glxtrinlinefullformat}[2]{%
7242   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7243   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7244   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7245 }%
7246 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7247   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7248   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7249   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7250 }%
7251 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7252   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7253   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7254   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7255 }%
7256 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7257   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7258   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7259   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7260 }%

```

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.

```

7261 \renewcommand*{\glxtrfullformat}[2]{%
7262   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7263   \ifglxtrinsertinside\else##2\fi
7264 }%
7265 \renewcommand*{\glxtrfullplformat}[2]{%
7266   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7267   \ifglxtrinsertinside\else##2\fi
7268 }%
7269 \renewcommand*{\Glsxtrfullformat}[2]{%
7270   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7271   \ifglxtrinsertinside\else##2\fi
7272 }%
7273 \renewcommand*{\Glsxtrfullplformat}[2]{%
7274   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7275   \ifglxtrinsertinside\else##2\fi
7276 }%
7277 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
7278 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

hort-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```

7279 \newabbreviationstyle{long-noshort-desc-noreg}%
7280 {}

```

```

7281 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
      Unset the regular attribute if it has been set.
7282 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7283   \glshasattribute{\the\glslabeltok}{regular}%
7284   {%
7285     \glissetattribute{\the\glslabeltok}{regular}{false}%
7286   }%
7287   {}%
7288 }%
7289}%
7290{%
7291 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7292 }

```

**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.

```

7293 \newabbreviationstyle{long}%
7294 {%
7295   \renewcommand*{\CustomAbbreviationFields}{%
7296     name={\protect\glsabbrvfont{\the\glsshorttok}},
7297     sort={\the\glsshorttok},
7298     first={\protect\glsfirstlongfont{\the\glslongtok}},
7299     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7300     text={\glslongfont{\the\glslongtok}},
7301     plural={\glslongfont{\the\glslongpltok}},%
7302     description={\the\glslongtok}%
7303   }%
7304   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7305     \glissetattribute{\the\glslabeltok}{regular}{true}}%
7306 }%
7307 {%
7308   \GlsXtrUseAbbrStyleFmts{long-desc}%
7309 }

```

**long-noshort** Provide a synonym that matches similar styles.

```

7310 \letabbreviationstyle{long-noshort}{long}

```

**g-noshort-noreg** Like long-noshort but doesn't set the regular attribute.

```

7311 \newabbreviationstyle{long-noshort-noreg}%
7312 {%
7313   \GlsXtrUseAbbrStyleSetup{long-noshort}%
      Unset the regular attribute if it has been set.
7314   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7315     \glshasattribute{\the\glslabeltok}{regular}%
7316     {%
7317       \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

7318 }%
7319 {}%
7320 }%
7321 }%
7322 {%
7323 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7324 }

```

### 1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

```

\glxtrscfont    Maintained for backward-compatibility.
7325 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}

\glsabrvscfont  Added for consistent naming.
7326 \newcommand*{\glsabrvscfont}{\glxtrscfont}

\glxtrfirstscfont  Maintained for backward-compatibility.
7327 \newcommand*{\glxtrfirstscfont}[1]{\glsabrvscfont{#1}}

\glxtrfirstabrvscfont  Added for consistent naming.
7328 \newcommand*{\glxtrfirstabrvscfont}{\glxtrfirstscfont}

```

and for the default short form suffix:

```

\glxtrscsuffix
7329 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabrvpluralsuffix}}

long-short-sc
7330 \newabbreviationstyle{long-short-sc}%
7331 {%
7332 \renewcommand*{\CustomAbbreviationFields}{%
7333   name={\protect\glsabrvscfont{\the\glsshorttok}},
7334   sort={\the\glsshorttok},
7335   first={\protect\glxtrfirstlongdefaultfont{\the\glslongtok}%
7336     \protect\glxtrfullsep{\the\glslabeltok}}%
7337   \glxtrparen{\protect\glxtrfirstabrvscfont{\the\glsshorttok}}},%
7338   firstplural={\protect\glxtrfirstlongdefaultfont{\the\glslongpltok}%
7339     \protect\glxtrfullsep{\the\glslabeltok}}%
7340   \glxtrparen{\protect\glxtrfirstabrvscfont{\the\glsshortpltok}}},%
7341   plural={\protect\glsabrvscfont{\the\glsshortpltok}}},%
7342   description={\the\glslongtok}}%
7343 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7344   \glshasattribute{\the\glslabeltok}{regular}%
7345   {%
7346     \glissetattribute{\the\glslabeltok}{regular}{false}%
7347   }}%
7348 {}%

```

```

7349 }%
7350 }%
7351 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7352 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7353 \renewcommand*{\glssabrvfont[1]{\glssabrvscfont{##1}}}%
7354 \renewcommand*{\glssfirstabrvfont[1]{\glssfirstabrvscfont{##1}}}%

```

Use the default long fonts.

```

7355 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
7356 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7357 \renewcommand*{\glxtrfullformat}[2]{%
7358   \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7359   \ifglxtrininsertinside\else##2\fi
7360   \glxtrfullsep{##1}%
7361   \glxtrparen{\glssfirstabrvscfont{\glssaccessshort{##1}}}%
7362 }%
7363 \renewcommand*{\glxtrfullplformat}[2]{%
7364   \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7365   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7366   \glxtrparen{\glssfirstabrvscfont{\glssaccessshortpl{##1}}}%
7367 }%
7368 \renewcommand*{\Glsxtrfullformat}[2]{%
7369   \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7370   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7371   \glxtrparen{\glssfirstabrvscfont{\glssaccessshort{##1}}}%
7372 }%
7373 \renewcommand*{\Glsxtrfullplformat}[2]{%
7374   \glssfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7375   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7376   \glxtrparen{\glssfirstabrvscfont{\glssaccessshortpl{##1}}}%
7377 }%
7378 }

```

g-short-sc-desc

```

7379 \newabbreviationstyle{long-short-sc-desc}%
7380 {%
7381   \renewcommand*{\CustomAbbreviationFields}{%
7382     name={\glsxtrlongshortdescname},
7383     sort={\glsxtrlongshortdescsort},%
7384     first={\protect\glssfirstlongdefaultfont{\the\glsslongtok}%
7385       \protect\glxtrfullsep{\the\glsslabeltok}%
7386       \glxtrparen{\protect\glssfirstabrvscfont{\the\glssshorttok}}},%
7387     firstplural={\protect\glssfirstlongdefaultfont{\the\glsslongpltok}%
7388       \protect\glxtrfullsep{\the\glsslabeltok}%
7389       \glxtrparen{\protect\glssfirstabrvscfont{\the\glssshortpltok}}},%
7390     text={\protect\glssabrvscfont{\the\glssshorttok}},%

```

```

7391 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7392 }%

```

Unset the regular attribute if it has been set.

```

7393 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7394   \glshasattribute{\the\glslabeltok}{regular}%
7395   {%
7396     \glissetattribute{\the\glslabeltok}{regular}{false}%
7397   }%
7398   {}%
7399 }%
7400 }%
7401 {%

```

As long-short-sc style:

```

7402 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7403 }

```

Now the short (long) version

```

7404 \newabbreviationstyle{short-sc-long}%
7405 {%
7406   \renewcommand*{\CustomAbbreviationFields}{%
7407     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7408     sort={\the\glsshorttok},
7409     description={\the\glslongtok},%
7410     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7411     \protect\glxtrfullsep{\the\glslabeltok}%
7412     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7413     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7414     \protect\glxtrfullsep{\the\glslabeltok}%
7415     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7416     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7417 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7418   \glshasattribute{\the\glslabeltok}{regular}%
7419   {%
7420     \glissetattribute{\the\glslabeltok}{regular}{false}%
7421   }%
7422   {}%
7423 }%
7424 }%
7425 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7426 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7427 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7428 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7429 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7430 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.



```

7431 \renewcommand*{\glxtrfullformat}[2]{%
7432   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7433   \ifglxtrininsertinside\else##2\fi
7434   \glxtrfullsep{##1}%
7435   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7436 }%
7437 \renewcommand*{\glxtrfullplformat}[2]{%
7438   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7439   \ifglxtrininsertinside\else##2\fi
7440   \glxtrfullsep{##1}%
7441   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7442 }%
7443 \renewcommand*{\Glsxtrfullformat}[2]{%
7444   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7445   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7446   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7447 }%
7448 \renewcommand*{\Glsxtrfullplformat}[2]{%
7449   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7450   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7451   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7452 }%
7453 }

```

As before but user provides description

```

7454 \newabbreviationstyle{short-sc-long-desc}%
7455 {%
7456   \renewcommand*{\CustomAbbreviationFields}{%
7457     name={\glxtrshortlongdescname},
7458     sort={\glxtrshortlongdescsort},
7459     first={\protect\glsfirstabbrvscfont{\the\glssshorttok}%
7460       \protect\glxtrfullsep{\the\glslabeltok}%
7461       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7462     firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}%
7463       \protect\glxtrfullsep{\the\glslabeltok}%
7464       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7465     text={\protect\glssabbrvscfont{\the\glssshorttok}},%
7466     plural={\protect\glssabbrvscfont{\the\glssshortpltok}}%
7467   }%

```

Unset the regular attribute if it has been set.

```

7468 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7469   \glshasattribute{\the\glslabeltok}{regular}%
7470   {%
7471     \glissetattribute{\the\glslabeltok}{regular}{false}%
7472   }%
7473   {}%
7474 }%
7475 }%
7476 {%

```

As short-sc-long style:

```
7477 \GlsXtrUseAbbrStyleFmts{short-sc-long}%  
7478 }
```

short-sc

```
7479 \newabbreviationstyle{short-sc}%  
7480 {%  
7481   \renewcommand*{\CustomAbbreviationFields}{%  
7482     name={\protect\glsabbrvscfont{\the\glsshorttok}},  
7483     sort={\the\glsshorttok},  
7484     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},  
7485     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},  
7486     text={\protect\glsabbrvscfont{\the\glsshorttok}},  
7487     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},  
7488     description={\the\glslongtok}}%  
7489   \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7490     \glssetattribute{\the\glslabeltok}{regular}{true}}%  
7491 }%  
7492 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7493 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
7494 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%  
7495 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%  
7496 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%  
7497 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7498 \renewcommand*{\glxtrinlinefullformat}[2]{%  
7499   \protect\glsfirstabbrvscfont{\glsaccesssshort{##1}}%  
7500   \ifglxtrininsertinside##2\fi}%  
7501   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7502   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%  
7503 }%  
7504 \renewcommand*{\glxtrinlinefullplformat}[2]{%  
7505   \protect\glsfirstabbrvscfont{\glsaccesssshortpl{##1}}%  
7506   \ifglxtrininsertinside##2\fi}%  
7507   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7508   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%  
7509 }%  
  
7510 \renewcommand*{\Glsxtrinlinefullformat}[2]{%  
7511   \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}}%  
7512   \ifglxtrininsertinside##2\fi}%  
7513   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7514   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%  
7515 }%  
7516 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%  
7517   \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}}%  
7518   \ifglxtrininsertinside##2\fi}%  
7519 }
```

```

7519 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7520 \glxtrparen{\glsfirstlongdefaultfont{\glaccesslongpl{##1}}}%
7521 }%

```

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.

```

7522 \renewcommand*{\glxtrfullformat}[2]{%
7523 \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7524 \ifglxtrinsertinside\else##2\fi
7525 }%
7526 \renewcommand*{\glxtrfullplformat}[2]{%
7527 \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7528 \ifglxtrinsertinside\else##2\fi
7529 }%
7530 \renewcommand*{\Glsxtrfullformat}[2]{%
7531 \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7532 \ifglxtrinsertinside\else##2\fi
7533 }%
7534 \renewcommand*{\Glsxtrfullplformat}[2]{%
7535 \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7536 \ifglxtrinsertinside\else##2\fi
7537 }%
7538 }

```

short-sc-nolong

```

7539 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7540 \newabbreviationstyle{short-sc-desc}{%
7541 {%
7542 \renewcommand*{\CustomAbbreviationFields}{%
7543 name={\glxtrshortdescname},
7544 sort={\the\glsshorttok},
7545 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7546 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7547 text={\protect\glsabbrvscfont{\the\glsshorttok}},
7548 plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7549 description={\the\glslongtok}}%
7550 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7551 \glsetattribute{\the\glslabeltok}{regular}{true}}%
7552 }%
7553 }%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7554 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7555 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7556 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7557 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7558 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7559 \renewcommand*{\glxtrinlinefullformat}[2]{%
7560   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7561   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7562   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7563 }%
7564 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7565   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7566   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7567   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7568 }%
7569 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7570   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7571   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7572   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7573 }%
7574 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7575   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7576   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7577   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7578 }%

```

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.

```

7579 \renewcommand*{\glxtrfullformat}[2]{%
7580   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7581   \ifglxtrinsertinside\else##2\fi
7582 }%
7583 \renewcommand*{\glxtrfullplformat}[2]{%
7584   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7585   \ifglxtrinsertinside\else##2\fi
7586 }%
7587 \renewcommand*{\Glsxtrfullformat}[2]{%
7588   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7589   \ifglxtrinsertinside\else##2\fi
7590 }%
7591 \renewcommand*{\Glsxtrfullplformat}[2]{%
7592   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7593   \ifglxtrinsertinside\else##2\fi
7594 }%
7595 }

```

-sc-nolong-desc

```

7596 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7597 \newabbreviationstyle{nolong-short-sc}%
7598 {%
7599   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%

```

```

7600 }%
7601 {%
7602   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7603   \renewcommand*{\glsxtrinlinefullformat}[2]{%
7604     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%
7605     \ifglsxtrininsertinside##2\fi}%
7606   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7607   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7608 }%
7609   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7610     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%
7611     \ifglsxtrininsertinside##2\fi}%
7612   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7613   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7614 }%
7615   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7616     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%
7617     \ifglsxtrininsertinside##2\fi}%
7618   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7619   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7620 }%
7621   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7622     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%
7623     \ifglsxtrininsertinside##2\fi}%
7624   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7625   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7626 }%
7627 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsxtrshort`.

```

7628 \newabbreviationstyle{long-noshort-sc}%
7629 {%
7630   \renewcommand*{\CustomAbbreviationFields}{%
7631     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7632     sort={\the\glsshorttok},
7633     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7634     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7635     text={\protect\glslongdefaultfont{\the\glslongtok}},
7636     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7637     description={\the\glslongtok}%
7638   }%
7639   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7640     \glssetAttribute{\the\glslabeltok}{regular}{true}}%
7641 }%
7642 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7643 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7644 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7645 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7646 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7647 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7648 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7649   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7650   \ifglxtrininsertinside \else##2\fi
7651 }%
7652 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7653   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7654   \ifglxtrininsertinside \else##2\fi
7655 }%
7656 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7657   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7658   \ifglxtrininsertinside \else##2\fi
7659 }%
7660 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7661   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7662   \ifglxtrininsertinside \else##2\fi
7663 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7664 \renewcommand*{\glxtrininlinefullformat}[2]{%
7665   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7666   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7667   \glxtrparen{\protect\glsfirstabbrvscfont{\glssaccessshort{##1}}}%
7668 }%
7669 \renewcommand*{\glxtrininlinefullplformat}[2]{%
7670   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7671   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7672   \glxtrparen{\protect\glsfirstabbrvscfont{\glssaccessshortpl{##1}}}%
7673 }%
7674 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
7675   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7676   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7677   \glxtrparen{\protect\glsfirstabbrvscfont{\glssaccessshort{##1}}}%
7678 }%
7679 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
7680   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7681   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7682   \glxtrparen{\protect\glsfirstabbrvscfont{\glssaccessshortpl{##1}}}%
7683 }%

```

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.

```

7684 \renewcommand*{\glxtrfullformat}[2]{%
7685   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7686   \ifglxtrininsertinside\else##2\fi

```

```

7687 }%
7688 \renewcommand*{\glxtrfullplformat}[2]{%
7689   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7690   \ifglxtrinsertinside\else##2\fi
7691 }%
7692 \renewcommand*{\Glsxtrfullformat}[2]{%
7693   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7694   \ifglxtrinsertinside\else##2\fi
7695 }%
7696 \renewcommand*{\Glsxtrfullplformat}[2]{%
7697   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7698   \ifglxtrinsertinside\else##2\fi
7699 }%
7700 }

```

long-sc Backward compatibility:

```

7701 \@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.

```

7702 \newabbreviationstyle{long-noshort-sc-desc}%
7703 {%
7704   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7705 }%
7706 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7707 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7708 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7709 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7710 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7711 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7712 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7713   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7714   \ifglxtrinsertinside \else##2\fi
7715 }%
7716 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7717   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7718   \ifglxtrinsertinside \else##2\fi
7719 }%
7720 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7721   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7722   \ifglxtrinsertinside \else##2\fi
7723 }%
7724 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7725   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7726   \ifglxtrinsertinside \else##2\fi
7727 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7728 \renewcommand*{\glxtrinlinefullformat}[2]{%
7729   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7730   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7731   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7732 }%
7733 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7734   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7735   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7736   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7737 }%
7738 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7739   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7740   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7741   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7742 }%
7743 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7744   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7745   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7746   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7747 }%

```

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.

```

7748 \renewcommand*{\glxtrfullformat}[2]{%
7749   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7750   \ifglxtrininsertinside\else##2\fi
7751 }%
7752 \renewcommand*{\glxtrfullplformat}[2]{%
7753   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7754   \ifglxtrininsertinside\else##2\fi
7755 }%
7756 \renewcommand*{\Glsxtrfullformat}[2]{%
7757   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7758   \ifglxtrininsertinside\else##2\fi
7759 }%
7760 \renewcommand*{\Glsxtrfullplformat}[2]{%
7761   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7762   \ifglxtrininsertinside\else##2\fi
7763 }%
7764 }

```

long-desc-sc Backward compatibility:

```

7765 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7766 \newabbreviationstyle{short-sc-footnote}%
7767 {%
7768   \renewcommand*{\CustomAbbreviationFields}{%

```



```

7769 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7770 sort={\the\glsshorttok},
7771 description={\the\glslongtok},%
7772 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7773 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7774 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7775 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7776 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7777 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7778 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.

```

7779 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7780 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7781 \glsattribute{\the\glslabeltok}{regular}%
7782 {%
7783 \glssetattribute{\the\glslabeltok}{regular}{false}%
7784 }%
7785 {}%
7786 }%
7787 }%
7788 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7789 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7790 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7791 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7792 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7793 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7794 \renewcommand*{\glsxtrfullformat}[2]{%
7795 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7796 \ifglsxtrininsertinside\else##2\fi
7797 \protect\glsxtrabbrvfootnote{##1}%
7798 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7799 }%
7800 \renewcommand*{\glsxtrfullplformat}[2]{%
7801 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7802 \ifglsxtrininsertinside\else##2\fi
7803 \protect\glsxtrabbrvfootnote{##1}%
7804 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7805 }%
7806 \renewcommand*{\Glsxtrfullformat}[2]{%
7807 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7808 \ifglsxtrininsertinside\else##2\fi
7809 \protect\glsxtrabbrvfootnote{##1}%
7810 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7811 }%
7812 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

7813 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7814 \ifglxtrinsertinside\else##2\fi
7815 \protect\glxtrabbrvfootnote{##1}%
7816 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7817 }%

```

The first use full form and the inline full form use the short (long) style.

```

7818 \renewcommand*{\glxtrinlinefullformat}[2]{%
7819 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7820 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7821 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7822 }%
7823 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7824 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7825 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7826 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7827 }%
7828 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7829 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7830 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7831 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7832 }%
7833 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7834 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7835 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7836 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7837 }%
7838 }

```

footnote-sc Backward compatibility:

```

7839 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

7840 \newabbreviationstyle{short-sc-postfootnote}%
7841 {%
7842 \renewcommand*{\CustomAbbreviationFields}{%
7843 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7844 sort={\the\glsshorttok},
7845 description={\the\glslongtok},%
7846 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7847 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7848 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.

```

7849 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7850 \csdef{glxtrpostlink\glscategorylabel}{%
7851 \glxtrifwasfirstuse
7852 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7853     \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7854     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
7855     }%
7856     {}%
7857     }%
7858     \glshasattribute{\the\glslabeltok}{regular}%
7859     {%
7860     \glissetattribute{\the\glslabeltok}{regular}{false}%
7861     }%
7862     {}%
7863     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7864 \renewcommand*{\glxtrsetupfulldefs}{%
7865     \let\glxtrifwasfirstuse\@secondoftwo
7866 }%
7867 }%
7868 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7869 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7870 \renewcommand*{\glxtrabbrvfont}[1]{\glxtrabbrvscfont{##1}}%
7871 \renewcommand*{\glxtrfirstabbrvfont}[1]{\glxtrfirstabbrvscfont{##1}}%
7872 \renewcommand*{\glxtrfirstlongfont}[1]{\glxtrfirstlongfootnotefont{##1}}%
7873 \renewcommand*{\glxtrlongfont}[1]{\glxtrlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7874 \renewcommand*{\glxtrfullformat}[2]{%
7875     \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7876     \ifglxtrininsertinside\else##2\fi
7877 }%
7878 \renewcommand*{\glxtrfullplformat}[2]{%
7879     \glxtrfirstabbrvscfont{\glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7880     \ifglxtrininsertinside\else##2\fi
7881 }%
7882 \renewcommand*{\Glsxtrfullformat}[2]{%
7883     \glxtrfirstabbrvscfont{\Glsxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7884     \ifglxtrininsertinside\else##2\fi
7885 }%
7886 \renewcommand*{\Glsxtrfullplformat}[2]{%
7887     \glxtrfirstabbrvscfont{\Glsxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7888     \ifglxtrininsertinside\else##2\fi
7889 }%

```

The first use full form and the inline full form use the short (long) style.

```

7890 \renewcommand*{\glxtrininlinefullformat}[2]{%
7891     \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7892     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7893 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7894 }%
7895 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7896 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
7897 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7898 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7899 }%
7900 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7901 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
7902 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7903 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7904 }%
7905 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7906 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
7907 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7908 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7909 }%
7910 }

```

postfootnote-sc Backward compatibility:

```

7911 \@glsxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

## 1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the `relsize` package, which must be loaded by the user. These styles all use:

```

\glsxtrsmfont Maintained for backward compatibility.
7912 \newcommand*{\glsxtrsmfont}[1]{\textsmaller{#1}}

\glsabbrvsmfont Added for consistent naming.
7913 \newcommand*{\glsabbrvsmfont}{\glsxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
7914 \newcommand*{\glsxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
7915 \newcommand*{\glsfirstabbrvsmfont}{\glsxtrfirstsmfont}

and for the default short form suffix:

\glsxtrsmsuffix
7916 \newcommand*{\glsxtrsmsuffix}{\glsxtrabbrvpluralsuffix}

long-short-sm
7917 \newabbreviationstyle{long-short-sm}%
7918 {%
7919 \renewcommand*{\CustomAbbreviationFields}{%

```

```

7920 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7921 sort={\the\glsshorttok},
7922 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7923 \protect\glsxtrfullsep{\the\glslabeltok}%
7924 \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7925 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7926 \protect\glsxtrfullsep{\the\glslabeltok}%
7927 \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7928 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
7929 description={\the\glslongtok}}%
7930 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7931 \glsattribute{\the\glslabeltok}{regular}%
7932 {%
7933 \glssetAttribute{\the\glslabeltok}{regular}{false}%
7934 }%
7935 {}%
7936 }%
7937 }%
7938 {%
7939 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7940 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7941 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%

```

Use the default long fonts.

```

7942 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7943 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7944 \renewcommand*\glsxtrfullformat[2]{%
7945 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7946 \ifglsxtrininsertinside\else##2\fi
7947 \glsxtrfullsep{##1}%
7948 \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7949 }%
7950 \renewcommand*\glsxtrfullplformat[2]{%
7951 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7952 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7953 \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7954 }%
7955 \renewcommand*\Glsxtrfullformat[2]{%
7956 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7957 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7958 \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7959 }%
7960 \renewcommand*\Glsxtrfullplformat[2]{%
7961 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7962 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7963 \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7964 }%
7965 }

```

g-short-sm-desc

```
7966 \newabbreviationstyle{long-short-sm-desc}%
7967 {%
7968   \renewcommand*{\CustomAbbreviationFields}{%
7969     name={\glxtrlongshortdescname},
7970     sort={\glxtrlongshortdescsort},%
7971     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7972       \protect\glxtrfullsep{\the\glslabeltok}%
7973       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7974     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7975       \protect\glxtrfullsep{\the\glslabeltok}%
7976       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7977     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7978     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7979   }%
```

Unset the regular attribute if it has been set.

```
7980 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7981   \glshasattribute{\the\glslabeltok}{regular}%
7982   {%
7983     \glissetattribute{\the\glslabeltok}{regular}{false}%
7984   }%
7985   {}%
7986 }%
7987 }%
7988 {%
```

As long-short-sm style:

```
7989 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
7990 }
```

short-sm-long Now the short (long) version

```
7991 \newabbreviationstyle{short-sm-long}%
7992 {%
7993   \renewcommand*{\CustomAbbreviationFields}{%
7994     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7995     sort={\the\glsshorttok},
7996     description={\the\glslongtok},%
7997     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7998       \protect\glxtrfullsep{\the\glslabeltok}%
7999       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8000     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8001       \protect\glxtrfullsep{\the\glslabeltok}%
8002       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8003     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7999   }%
```

Unset the regular attribute if it has been set.

```
8004 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8005   \glshasattribute{\the\glslabeltok}{regular}%
8006   {%
```

```

8007     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8008   }%
8009   {}%
8010 }%
8011}%
8012{%
8013  \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8014  \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8015  \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8016  \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8017  \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8018  \renewcommand*\glsxtrfullformat[2]{%
8019    \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8020    \ifglsxtrininsertinside\else##2\fi
8021    \glsxtrfullsep{##1}%
8022    \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8023  }%
8024  \renewcommand*\glsxtrfullplformat[2]{%
8025    \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8026    \ifglsxtrininsertinside\else##2\fi
8027    \glsxtrfullsep{##1}%
8028    \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8029  }%
8030  \renewcommand*\Glsxtrfullformat[2]{%
8031    \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8032    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8033    \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8034  }%
8035  \renewcommand*\Glsxtrfullplformat[2]{%
8036    \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8037    \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8038    \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8039  }%
8040 }

```

rt-sm-long-desc As before but user provides description

```

8041 \newabbreviationstyle{short-sm-long-desc}%
8042 {%
8043   \renewcommand*\CustomAbbreviationFields{%
8044     name={\glsxtrshortlongdescname},
8045     sort={\glsxtrshortlongdescsort},
8046     first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}%
8047       \protect\glsxtrfullsep{\the\glslabeltok}%
8048       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8049     firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}%
8050       \protect\glsxtrfullsep{\the\glslabeltok}%
8051       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8052     text={\protect\glsabbrvsmfont{\the\glssshorttok}},%

```

```

8053 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
8054 }%

```

Unset the regular attribute if it has been set.

```

8055 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8056   \glshasattribute{\the\glslabeltok}{regular}%
8057   {%
8058     \glissetattribute{\the\glslabeltok}{regular}{false}%
8059   }%
8060   {}%
8061 }%
8062 }%
8063 {%

```

As short-sm-long style:

```

8064 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8065 }

```

short-sm

```

8066 \newabbreviationstyle{short-sm}%
8067 {%
8068   \renewcommand*{\CustomAbbreviationFields}{%
8069     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8070     sort={\the\glsshorttok},
8071     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8072     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8073     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8074     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8075     description={\the\glslongtok}}%
8076   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8077     \glissetattribute{\the\glslabeltok}{regular}{true}}%
8078 }%
8079 {%
8080   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8081   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8082   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrmsuffix}%
8083   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8084   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8085 \renewcommand*{\glxtrinlinefullformat}[2]{%
8086   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8087   \ifglxtrininsertinside##2\fi}%
8088   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8089   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8090 }%
8091 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8092   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
8093   \ifglxtrininsertinside##2\fi}%
8094   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

8095 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8096 }%

8097 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8098 \protect\glfirstabbrvsmfont{\Glsaccessshort{##1}%
8099 \ifglxtrinsertinside##2\fi}%
8100 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8101 \glstrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
8102 }%
8103 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8104 \protect\glfirstabbrvsmfont{\Glsaccessshortpl{##1}%
8105 \ifglxtrinsertinside##2\fi}%
8106 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8107 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8108 }%

```

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.

```

8109 \renewcommand*{\glxtrfullformat}[2]{%
8110 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8111 \ifglxtrinsertinside\else##2\fi
8112 }%
8113 \renewcommand*{\glxtrfullplformat}[2]{%
8114 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8115 \ifglxtrinsertinside\else##2\fi
8116 }%
8117 \renewcommand*{\Glsxtrfullformat}[2]{%
8118 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8119 \ifglxtrinsertinside\else##2\fi
8120 }%
8121 \renewcommand*{\Glsxtrfullplformat}[2]{%
8122 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8123 \ifglxtrinsertinside\else##2\fi
8124 }%
8125 }

```

short-sm-nolong

```

8126 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

8127 \newabbreviationstyle{short-sm-desc}%
8128 {%
8129 \renewcommand*{\CustomAbbreviationFields}{%
8130 name={\glxtrshortdescname},
8131 sort={\the\glsshorttok},
8132 first={\protect\glfirstabbrvsmfont{\the\glsshorttok}},
8133 firstplural={\protect\glfirstabbrvsmfont{\the\glsshortpltok}},
8134 text={\protect\glabbrvsmfont{\the\glsshorttok}},
8135 plural={\protect\glabbrvsmfont{\the\glsshortpltok}},

```

```

8136     description={\the\glslongtok}}%
8137 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8138   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8139 }%
8140 {%
8141   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8142   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8143   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
8144   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8145   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8146 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8147   \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8148   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8149   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8150 }%
8151 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8152   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8153   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8154   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8155 }%
8156 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8157   \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8158   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8159   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8160 }%
8161 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8162   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8163   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8164   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8165 }%

```

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.

```

8166 \renewcommand*{\glsxtrfullformat}[2]{%
8167   \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8168   \ifglsxtrininsertinside\else##2\fi
8169 }%
8170 \renewcommand*{\glsxtrfullplformat}[2]{%
8171   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8172   \ifglsxtrininsertinside\else##2\fi
8173 }%
8174 \renewcommand*{\Glsxtrfullformat}[2]{%
8175   \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8176   \ifglsxtrininsertinside\else##2\fi
8177 }%
8178 \renewcommand*{\Glsxtrfullplformat}[2]{%
8179   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8180   \ifglsxtrininsertinside\else##2\fi

```

```

8181 }%
8182 }

```

-sm-nolong-desc

```

8183 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8184 \newabbreviationstyle{nolong-short-sm}%
8185 {%
8186   \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8187 }%
8188 {%
8189   \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8190 \renewcommand*{\glxtrinlinefullformat}[2]{%
8191   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8192     \ifglxtrinsertinside##2\fi}%
8193   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8194   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8195 }%
8196 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8197   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8198     \ifglxtrinsertinside##2\fi}%
8199   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8200   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8201 }%
8202 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8203   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8204     \ifglxtrinsertinside##2\fi}%
8205   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8206   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8207 }%
8208 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8209   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8210     \ifglxtrinsertinside##2\fi}%
8211   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8212   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8213 }%
8214 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

8215 \newabbreviationstyle{long-noshort-sm}%
8216 {%
8217   \renewcommand*{\CustomAbbreviationFields}{%
8218     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8219     sort={\the\glsshorttok},
8220     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},

```

```

8221 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8222 text={\protect\glslongdefaultfont{\the\glslongtok}},
8223 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8224 description={\the\glslongtok}%
8225 }%
8226 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8227   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8228 }%
8229 {%
8230 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8231 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8232 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8233 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8234 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8235 \renewcommand*\glsxtrsubsequentfmt}[2]{%
8236   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8237   \ifglsxtrininsertinside \else##2\fi
8238 }%
8239 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
8240   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8241   \ifglsxtrininsertinside \else##2\fi
8242 }%
8243 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8244   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8245   \ifglsxtrininsertinside \else##2\fi
8246 }%
8247 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8248   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8249   \ifglsxtrininsertinside \else##2\fi
8250 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8251 \renewcommand*\glsxtrinlinefullformat}[2]{%
8252   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8253   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8254   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8255 }%
8256 \renewcommand*\glsxtrinlinefullplformat}[2]{%
8257   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8258   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8259   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8260 }%
8261 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8262   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8263   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8264   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8265 }%
8266 \renewcommand*\Glsxtrinlinefullplformat}[2]{%

```

```

8267 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8268 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8269 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8270 }%

```

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.

```

8271 \renewcommand*{\glxtrfullformat}[2]{%
8272 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8273 \ifglxtrinsertinside\else##2\fi
8274 }%
8275 \renewcommand*{\glxtrfullplformat}[2]{%
8276 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8277 \ifglxtrinsertinside\else##2\fi
8278 }%
8279 \renewcommand*{\Glsxtrfullformat}[2]{%
8280 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8281 \ifglxtrinsertinside\else##2\fi
8282 }%
8283 \renewcommand*{\Glsxtrfullplformat}[2]{%
8284 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8285 \ifglxtrinsertinside\else##2\fi
8286 }%
8287 }

```

`long-sm` Backward compatibility:

```

8288 \@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`.

```

8289 \newabbreviationstyle{long-noshort-sm-desc}%
8290 {%
8291 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8292 }%
8293 {%
8294 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8295 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8296 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8297 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8298 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8299 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8300 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8301 \ifglxtrinsertinside \else##2\fi
8302 }%
8303 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8304 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8305 \ifglxtrinsertinside \else##2\fi
8306 }%

```

```

8307 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8308   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8309   \ifglxtrinsertinside \else##2\fi
8310 }%
8311 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8312   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8313   \ifglxtrinsertinside \else##2\fi
8314 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8315 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8316   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8317   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
8318   \Glsxtrparen{\protect\Glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8319 }%
8320 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8321   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8322   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
8323   \Glsxtrparen{\protect\Glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8324 }%
8325 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8326   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8327   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
8328   \Glsxtrparen{\protect\Glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8329 }%
8330 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8331   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8332   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
8333   \Glsxtrparen{\protect\Glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8334 }%

```

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.

```

8335 \renewcommand*{\Glsxtrfullformat}[2]{%
8336   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8337   \ifglxtrinsertinside\else##2\fi
8338 }%
8339 \renewcommand*{\Glsxtrfullplformat}[2]{%
8340   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8341   \ifglxtrinsertinside\else##2\fi
8342 }%
8343 \renewcommand*{\Glsxtrfullformat}[2]{%
8344   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8345   \ifglxtrinsertinside\else##2\fi
8346 }%
8347 \renewcommand*{\Glsxtrfullplformat}[2]{%
8348   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8349   \ifglxtrinsertinside\else##2\fi
8350 }%
8351 }

```

long-desc-sm    Backward compatibility:

```
8352 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

ort-sm-footnote

```
8353 \newabbreviationstyle{short-sm-footnote}%
8354 {%
8355   \renewcommand*{\CustomAbbreviationFields}{%
8356     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8357     sort={\the\glsshorttok},
8358     description={\the\glslongtok},%
8359     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8360       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8361       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8362     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8363       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8364       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8365     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.

```
8366 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8367   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8368   \glsattribute{\the\glslabeltok}{regular}%
8369   {%
8370     \glssetattribute{\the\glslabeltok}{regular}{false}%
8371   }%
8372   {}%
8373 }%
8374 }%
8375 {%
8376 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8377 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8378 \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
8379 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8380 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```
8381 \renewcommand*\glxtrfullformat[2]{%
8382   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8383   \ifglxtrinsertinside\else##2\fi
8384   \protect\glxtrabbrvfootnote{##1}%
8385   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8386 }%
8387 \renewcommand*\glxtrfullplformat[2]{%
8388   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8389   \ifglxtrinsertinside\else##2\fi
8390   \protect\glxtrabbrvfootnote{##1}%
8391   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8392 }%
8393 \renewcommand*\Glsxtrfullformat[2]{%

```

```

8394 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8395 \ifglxtrinsertinside\else##2\fi
8396 \protect\glxtrabbrvfootnote{##1}%
8397 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8398 }%
8399 \renewcommand*{\Glsxtrfullplformat}[2]{%
8400 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8401 \ifglxtrinsertinside\else##2\fi
8402 \protect\glxtrabbrvfootnote{##1}%
8403 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8404 }%

```

The first use full form and the inline full form use the short (long) style.

```

8405 \renewcommand*{\glxtrinlinefullformat}[2]{%
8406 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8407 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8408 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8409 }%
8410 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8411 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8412 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8413 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8414 }%
8415 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8416 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8417 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8418 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8419 }%
8420 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8421 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8422 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8423 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8424 }%
8425 }

```

footnote-sm Backward compatibility:

```

8426 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8427 \newabbreviationstyle{short-sm-postfootnote}%
8428 {%
8429 \renewcommand*{\CustomAbbreviationFields}{%
8430 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8431 sort={\the\glsshorttok},
8432 description={\the\glslongtok},%
8433 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8434 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8435 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.

```
8436 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8437   \csdef{glxstrpostlink\glscategorylabel}{%
8438     \glxstrifwasfirstuse
8439     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
8440       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8441       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}}%
8442   }%
8443   {%
8444   }%
8445   \glshasattribute{\the\glslabeltok}{regular}%
8446   {%
8447     \glissetattribute{\the\glslabeltok}{regular}{false}%
8448   }%
8449   {%
8450   }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
8451 \renewcommand*{\glxtrsetupfulldefs}{%
8452   \let\glxstrifwasfirstuse\@secondoftwo
8453 }%
8454 }%
8455 {%
8456 \renewcommand*\glxtrfont[1]{\glxtrfont{##1}}%
8457 \renewcommand*\glxtrfirstfont[1]{\glxtrfirstfont{##1}}%
8458 \renewcommand*\glxtrpluralsuffix{\protect\glxtrpluralsuffix}%
8459 \renewcommand*\glxtrfirstlongfont[1]{\glxtrfirstlongfootnotefont{##1}}%
8460 \renewcommand*\glxtrlongfont[1]{\glxtrlongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
8461 \renewcommand*\glxtrfullformat[2]{%
8462   \glxtrfirstfont{\glxtrfirstfont{##1}\ifglxtrinsertinside##2\fi}%
8463   \ifglxtrinsertinside\else##2\fi
8464 }%
8465 \renewcommand*\glxtrfullplformat[2]{%
8466   \glxtrfirstfont{\glxtrfirstfont{##1}\ifglxtrinsertinside##2\fi}%
8467   \ifglxtrinsertinside\else##2\fi
8468 }%
8469 \renewcommand*\glxtrfullformat[2]{%
8470   \glxtrfirstfont{\glxtrfirstfont{##1}\ifglxtrinsertinside##2\fi}%
8471   \ifglxtrinsertinside\else##2\fi
8472 }%
8473 \renewcommand*\glxtrfullplformat[2]{%
8474   \glxtrfirstfont{\glxtrfirstfont{##1}\ifglxtrinsertinside##2\fi}%
8475   \ifglxtrinsertinside\else##2\fi
```

8476 }%

The first use full form and the inline full form use the short (long) style.

```
8477 \renewcommand*{\glxtrinlinefullformat}[2]{%
8478   \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8479   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8480   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
8481 }%
8482 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8483   \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8484   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8485   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8486 }%
8487 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8488   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8489   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8490   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
8491 }%
8492 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8493   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8494   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8495   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8496 }%
8497 }
```

postfootnote-sm Backward compatibility:

```
8498 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}
```

## 1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```
8499 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%
```

`\glsfirstabbrvemfont`

```
8500 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%
```

The default short form suffix:

`\glxxtremsuffix`

```
8501 \newcommand*{\glxxtremsuffix}{\glxtrabbrvpluralsuffix}
```

`\glsfirstlongemfont`

Only used by the “long-em” styles.

```
8502 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%
```

`\glslongemfont`

Only used by the “long-em” styles.

```
8503 \newcommand*{\glslongemfont}[1]{\emph{##1}}%
```

long-short-em The long form is just set in the default long font.

```
8504 \newabbreviationstyle{long-short-em}%
8505 {%
8506   \renewcommand*{\CustomAbbreviationFields}{%
8507     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8508     sort={\the\glsshorttok},
8509     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8510       \protect\glsxtrfullsep{\the\glslabeltok}%
8511       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8512     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8513       \protect\glsxtrfullsep{\the\glslabeltok}%
8514       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8515     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8516     description={\the\glslongtok}}%
8517   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8518     \glsattribute{\the\glslabeltok}{regular}%
8519     {%
8520       \glssetattribute{\the\glslabeltok}{regular}{false}%
8521     }%
8522   }%
8523 }%
8524 }%
8525 {%
8526   \renewcommand*{\glsabbrvfont[1]{\glsabbrvemfont{##1}}}%
8527   \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}}%
8528   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8529 }
```

Use the default long fonts.

```
8529 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8530 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8531 \renewcommand*{\glsxtrfullformat}[2]{%
8532   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8533   \ifglsxtrininsertinside\else##2\fi
8534   \glsxtrfullsep{##1}%
8535   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8536 }%
8537 \renewcommand*{\glsxtrfullplformat}[2]{%
8538   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8539   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8540   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8541 }%
8542 \renewcommand*{\Glsxtrfullformat}[2]{%
8543   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8544   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8545   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8546 }%
8547 \renewcommand*{\Glsxtrfullplformat}[2]{%
8548   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8549 }
```

```

8549 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8550 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8551 }%
8552 }

```

g-short-em-desc

```

8553 \newabbreviationstyle{long-short-em-desc}%
8554 {%
8555 \renewcommand*{\CustomAbbreviationFields}{%
8556 name={\glxtrlongshortdescname},
8557 sort={\glxtrlongshortdescsort},%
8558 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8559 \protect\glxtrfullsep{\the\glslabeltok}%
8560 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8561 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8562 \protect\glxtrfullsep{\the\glslabeltok}%
8563 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8564 text={\protect\glssabbrvemfont{\the\glsshorttok}},%
8565 plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
8566 }%

```

Unset the regular attribute if it has been set.

```

8567 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8568 \glshasattribute{\the\glslabeltok}{regular}%
8569 {%
8570 \glissetattribute{\the\glslabeltok}{regular}{false}%
8571 }%
8572 {}%
8573 }%
8574 }%
8575 {%

```

As long-short-em style:

```

8576 \GlsXtrUseAbbrStyleFmts{long-short-em}%
8577 }

```

ong-em-short-em

```

8578 \newabbreviationstyle{long-em-short-em}%
8579 {%
8580 \renewcommand*{\CustomAbbreviationFields}{%
8581 name={\protect\glssabbrvemfont{\the\glsshorttok}},
8582 sort={\the\glsshorttok},
8583 first={\protect\glsfirstlongemfont{\the\glslongtok}%
8584 \protect\glxtrfullsep{\the\glslabeltok}%
8585 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8586 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8587 \protect\glxtrfullsep{\the\glslabeltok}%
8588 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

```

```

8589 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8590 description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8591 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8592   \glsattribute{\the\glslabeltok}{regular}%
8593   {%
8594     \glsssetattribute{\the\glslabeltok}{regular}{false}%
8595   }%
8596   {}}%
8597 }%
8598 }%
8599 {%
8600 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8601 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8602 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8603 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8604 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8605 \renewcommand*{\glsxtrfullformat}[2]{%
8606   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8607   \ifglsxtrininsertinside\else##2\fi
8608   \glsxtrfullsep{##1}%
8609   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8610 }%
8611 \renewcommand*{\glsxtrfullplformat}[2]{%
8612   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8613   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8614   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8615 }%
8616 \renewcommand*{\Glsxtrfullformat}[2]{%
8617   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8618   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8619   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8620 }%
8621 \renewcommand*{\Glsxtrfullplformat}[2]{%
8622   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8623   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8624   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8625 }%
8626 }

```

m-short-em-desc

```

8627 \newabbreviationstyle{long-em-short-em-desc}%
8628 {%
8629 \renewcommand*{\CustomAbbreviationFields}{%
8630   name={\glsxtrlongshortdescname},
8631   sort={\glsxtrlongshortdescsort},%
8632   first={\protect\glsfirstlongemfont{\the\glslongtok}}%

```

```

8633 \protect\glxtrfullsep{\the\glslabeltok}%
8634 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8635 firstplural={\protect\glsfirstlongemfont{\the\gslongpltok}%
8636 \protect\glxtrfullsep{\the\glslabeltok}%
8637 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8638 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
8639 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8640 }%

```

Unset the regular attribute if it has been set.

```

8641 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8642 \glshasattribute{\the\glslabeltok}{regular}%
8643 {%
8644 \glissetattribute{\the\glslabeltok}{regular}{false}%
8645 }%
8646 {}%
8647 }%
8648 }%
8649 {%
8650 \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8651 }

```

short-em-long Now the short (long) version

```

8652 \newabbreviationstyle{short-em-long}%
8653 {%
8654 \renewcommand*{\CustomAbbreviationFields}{%
8655 name={\protect\glsabbrvemfont{\the\glsshorttok}},
8656 sort={\the\glsshorttok},
8657 description={\the\gslongtok},%
8658 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8659 \protect\glxtrfullsep{\the\glslabeltok}%
8660 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
8661 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8662 \protect\glxtrfullsep{\the\glslabeltok}%
8663 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
8664 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8665 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8666 \glshasattribute{\the\glslabeltok}{regular}%
8667 {%
8668 \glissetattribute{\the\glslabeltok}{regular}{false}%
8669 }%
8670 {}%
8671 }%
8672 }%
8673 {%

```

Mostly as short-long style:

```

8674 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%

```

```

8675 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8676 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8677 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8678 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8679 \renewcommand*\glsxtrfullformat}[2]{%
8680   \glsfirstabbrvemfont{\glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
8681   \ifglsxtrininsertinside\else##2\fi
8682   \glsxtrfullsep{##1}%
8683   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8684 }%
8685 \renewcommand*\glsxtrfullplformat}[2]{%
8686   \glsfirstabbrvemfont{\glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8687   \ifglsxtrininsertinside\else##2\fi
8688   \glsxtrfullsep{##1}%
8689   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8690 }%
8691 \renewcommand*\Glsxtrfullformat}[2]{%
8692   \glsfirstabbrvemfont{\Glsaccesssshort{##1}\ifglsxtrininsertinside##2\fi}%
8693   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8694   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8695 }%
8696 \renewcommand*\Glsxtrfullplformat}[2]{%
8697   \glsfirstabbrvemfont{\Glsaccesssshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8698   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8699   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8700 }%
8701 }

```

rt-em-long-desc As before but user provides description

```

8702 \newabbreviationstyle{short-em-long-desc}%
8703 {%
8704   \renewcommand*\CustomAbbreviationFields{%
8705     name={\glsxtrshortlongdescname},
8706     sort={\glsxtrshortlongdescsort},
8707     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8708     \protect\glsxtrfullsep{\the\glslabeltok}}%
8709     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8710     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8711     \protect\glsxtrfullsep{\the\glslabeltok}}%
8712     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8713     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8714     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8715 }%

```

Unset the regular attribute if it has been set.

```

8716 \renewcommand*\GlsXtrPostNewAbbreviation{%
8717   \glshasattribute{\the\glslabeltok}{regular}%
8718   {%

```

```

8719     \glsetattribute{\the\glslabeltok}{regular}{false}%
8720   }%
8721   {}%
8722 }%
8723 }%
8724 {%
8725   \GlsXtrUseAbbrStyleFmts{short-em-long}%
8726 }

```

hort-em-long-em

```

8727 \newabbreviationstyle{short-em-long-em}%
8728 {%
    \glslongemfont is used in the description since \glsgdesc doesn't set the style.
8729   \renewcommand*{\CustomAbbreviationFields}{%
8730     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8731     sort={\the\glsshorttok},
8732     description={\protect\glslongemfont{\the\glslongtok}},%
8733     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8734     \protect\glsxtrfullsep{\the\glslabeltok}%
8735     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8736     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8737     \protect\glsxtrfullsep{\the\glslabeltok}%
8738     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8739     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

    Unset the regular attribute if it has been set.
8740   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8741     \glshasattribute{\the\glslabeltok}{regular}%
8742     {%
8743       \glsetattribute{\the\glslabeltok}{regular}{false}%
8744     }%
8745     {}%
8746   }%
8747 }%
8748 {%
8749   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8750   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8751   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8752   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8753   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8754   \renewcommand*{\glsxtrfullformat}[2]{%
8755     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8756     \ifglsxtrininsertinside\else##2\fi
8757     \glsxtrfullsep{##1}%
8758     \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8759   }%
8760   \renewcommand*{\glsxtrfullplformat}[2]{%

```



```

8761 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8762 \ifglxtrinsertinside\else##2\fi
8763 \glxtrfullsep{##1}%
8764 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8765 }%
8766 \renewcommand*{\Glsxtrfullformat}[2]{%
8767 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8768 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8769 \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8770 }%
8771 \renewcommand*{\Glsxtrfullplformat}[2]{%
8772 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8773 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8774 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8775 }%
8776 }

```

em-long-em-desc

```

8777 \newabbreviationstyle{short-em-long-em-desc}%
8778 {%
8779 \renewcommand*{\CustomAbbreviationFields}{%
8780 name={\glxtrshortlongdescname},%
8781 sort={\glxtrshortlongdescsort},%
8782 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8783 \protect\glxtrfullsep{\the\glslabeltok}%
8784 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8785 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8786 \protect\glxtrfullsep{\the\glslabeltok}%
8787 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8788 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8789 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8790 }%

```

Unset the regular attribute if it has been set.

```

8791 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8792 \glshasattribute{\the\glslabeltok}{regular}%
8793 {%
8794 \glissetattribute{\the\glslabeltok}{regular}{false}%
8795 }%
8796 {}%
8797 }%
8798 }%
8799 {%
8800 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8801 }

```

short-em

```

8802 \newabbreviationstyle{short-em}%
8803 {%
8804 \renewcommand*{\CustomAbbreviationFields}{%

```

```

8805     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8806     sort={\the\glsshorttok},
8807     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8808     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8809     text={\protect\glsabbrvemfont{\the\glsshorttok}},
8810     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8811     description={\the\glslongtok}}%
8812 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8813   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8814 }%
8815 {%
8816 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8817 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8818 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8819 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8820 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8821 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8822   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
8823   \ifglsxtrininsertinside##2\fi}%
8824 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8825 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8826 }%
8827 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8828   \protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}%
8829   \ifglsxtrininsertinside##2\fi}%
8830 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8831 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8832 }%
8833 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8834   \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}%
8835   \ifglsxtrininsertinside##2\fi}%
8836 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8837 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8838 }%
8839 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8840   \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}%
8841   \ifglsxtrininsertinside##2\fi}%
8842 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8843 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8844 }%

```

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.

```

8845 \renewcommand*{\glsxtrfullformat}[2]{%
8846   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8847   \ifglsxtrininsertinside\else##2\fi
8848 }%

```

```

8849 \renewcommand*{\glxtrfullplformat}[2]{%
8850   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8851   \ifglxtrinsertinside\else##2\fi
8852 }%
8853 \renewcommand*{\Glsxtrfullformat}[2]{%
8854   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8855   \ifglxtrinsertinside\else##2\fi
8856 }%
8857 \renewcommand*{\Glsxtrfullplformat}[2]{%
8858   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8859   \ifglxtrinsertinside\else##2\fi
8860 }%
8861 }

```

short-em-nolong

```

8862 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8863 \newabbreviationstyle{short-em-desc}%
8864 {%
8865   \renewcommand*{\CustomAbbreviationFields}{%
8866     name={\glxtrshortdescname},
8867     sort={\the\glssshorttok},
8868     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},
8869     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},
8870     text={\protect\glsabbrvemfont{\the\glssshorttok}},
8871     plural={\protect\glsabbrvemfont{\the\glssshortpltok}},
8872     description={\the\glslongtok}}%
8873   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8874     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8875 }%
8876 {%
8877   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8878   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8879   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8880   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8881   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8882 \renewcommand*{\glxtrinlinefullformat}[2]{%
8883   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8884   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8885   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
8886 }%
8887 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8888   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8889   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8890   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8891 }%
8892 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

8893 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8894 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8895 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8896 }%
8897 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8898 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8899 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8900 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8901 }%

```

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.

```

8902 \renewcommand*{\glxtrfullformat}[2]{%
8903 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8904 \ifglxtrinsertinside\else##2\fi
8905 }%
8906 \renewcommand*{\glxtrfullplformat}[2]{%
8907 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8908 \ifglxtrinsertinside\else##2\fi
8909 }%
8910 \renewcommand*{\Glsxtrfullformat}[2]{%
8911 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8912 \ifglxtrinsertinside\else##2\fi
8913 }%
8914 \renewcommand*{\Glsxtrfullplformat}[2]{%
8915 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8916 \ifglxtrinsertinside\else##2\fi
8917 }%
8918 }

```

-em-nolong-desc

```

8919 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8920 \newabbreviationstyle{nolong-short-em}%
8921 {%
8922 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8923 }%
8924 {%
8925 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8926 \renewcommand*{\glxtrinlinefullformat}[2]{%
8927 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8928 \ifglxtrinsertinside##2\fi}%
8929 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8930 \glxtrparen{\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8931 }%
8932 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8933 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%

```

```

8934     \ifglxtrinsertinside##2\fi}%
8935     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8936     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8937 }%
8938 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8939     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8940         \ifglxtrinsertinside##2\fi}%
8941     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8942     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8943 }%
8944 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8945     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8946         \ifglxtrinsertinside##2\fi}%
8947     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8948     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8949 }%
8950 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8951 \newabbreviationstyle{long-noshort-em}%
8952 {%
8953     \renewcommand*{\CustomAbbreviationFields}{%
8954         name={\protect\glsabbrvemfont{\the\glsshorttok}},
8955         sort={\the\glsshorttok},
8956         first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8957         firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8958         text={\protect\glslongdefaultfont{\the\glslongtok}},
8959         plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8960         description={\the\glslongtok}%
8961     }%
8962     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8963         \glssetattribute{\the\glslabeltok}{regular}{true}}%
8964 }%
8965 {%
8966     \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8967     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8968     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8969     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8970     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8971 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8972     \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8973     \ifglxtrinsertinside \else##2\fi
8974 }%
8975 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8976     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8977     \ifglxtrinsertinside \else##2\fi
8978 }%
8979 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%

```

```

8980 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8981 \ifglxtrinsertinside \else##2\fi
8982 }%
8983 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8984 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8985 \ifglxtrinsertinside \else##2\fi
8986 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8987 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8988 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8989 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8990 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8991 }%
8992 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8993 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8994 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8995 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8996 }%
8997 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8998 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8999 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9000 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9001 }%
9002 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9003 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9004 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9005 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9006 }%

```

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.

```

9007 \renewcommand*{\glsxtrfullformat}[2]{%
9008 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9009 \ifglxtrinsertinside\else##2\fi
9010 }%
9011 \renewcommand*{\glsxtrfullplformat}[2]{%
9012 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9013 \ifglxtrinsertinside\else##2\fi
9014 }%
9015 \renewcommand*{\Glsxtrfullformat}[2]{%
9016 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9017 \ifglxtrinsertinside\else##2\fi
9018 }%
9019 \renewcommand*{\Glsxtrfullplformat}[2]{%
9020 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9021 \ifglxtrinsertinside\else##2\fi
9022 }%
9023 }

```

long-em    Backward compatibility:

```
9024 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em    The short form is explicitly invoked through commands like `\glsshort`.

```
9025 \newabbreviationstyle{long-em-noshort-em}%
9026 {%
9027   \renewcommand*{\CustomAbbreviationFields}{%
9028     name={\protect\glsabbrvemfont{\the\glsshorttok}},
9029     sort={\the\glsshorttok},
9030     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9031     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9032     text={\protect\glslongemfont{\the\glslongtok}},
9033     plural={\protect\glslongemfont{\the\glslongpltok}},%
9034     description={\protect\glslongemfont{\the\glslongtok}}%
9035   }%
9036   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9037     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9038 }%
9039 {%
9040   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9041   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9042   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9043   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
9044   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9045 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9046   \glslongemfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9047   \ifglxtrininsertinside \else##2\fi
9048 }%
9049 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9050   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9051   \ifglxtrininsertinside \else##2\fi
9052 }%
9053 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9054   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9055   \ifglxtrininsertinside \else##2\fi
9056 }%
9057 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9058   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9059   \ifglxtrininsertinside \else##2\fi
9060 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
9061 \renewcommand*{\glxtrinlinefullformat}[2]{%
9062   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9063   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9064   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9065 }%
9066 \renewcommand*{\glxtrinlinefullplformat}[2]{%
```

```

9067 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9068 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9069 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9070 }%
9071 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9072 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9073 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9074 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9075 }%
9076 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9077 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9078 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9079 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9080 }%

```

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.

```

9081 \renewcommand*{\glxtrfullformat}[2]{%
9082 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9083 \ifglxtrinsertinside\else##2\fi
9084 }%
9085 \renewcommand*{\glxtrfullplformat}[2]{%
9086 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9087 \ifglxtrinsertinside\else##2\fi
9088 }%
9089 \renewcommand*{\Glsxtrfullformat}[2]{%
9090 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9091 \ifglxtrinsertinside\else##2\fi
9092 }%
9093 \renewcommand*{\Glsxtrfullplformat}[2]{%
9094 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9095 \ifglxtrinsertinside\else##2\fi
9096 }%
9097 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9098 \newabbreviationstyle{long-em-noshort-em-noreg}%
9099 {%
9100 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9101 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9102 \glshasattribute{\the\glslabeltok}{regular}%
9103 {%
9104 \glssetattribute{\the\glslabeltok}{regular}{false}%
9105 }%
9106 {}%
9107 }%
9108 }%
9109 {%

```



```

9110 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9111 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

9112 \newabbreviationstyle{long-noshort-em-desc}%
9113 {%
9114 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9115 }%
9116 {%
9117 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9118 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
9119 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvemfont{##1}}%
9120 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
9121 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9122 \renewcommand*{\glssxtrsubsequentfmt}[2]{%
9123 \glsslongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9124 \ifglssxtrininsertinside \else##2\fi
9125 }%
9126 \renewcommand*{\glssxtrsubsequentplfmt}[2]{%
9127 \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9128 \ifglssxtrininsertinside \else##2\fi
9129 }%
9130 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9131 \glsslongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9132 \ifglssxtrininsertinside \else##2\fi
9133 }%
9134 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9135 \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9136 \ifglssxtrininsertinside \else##2\fi
9137 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9138 \renewcommand*{\glssxtrinlinefullformat}[2]{%
9139 \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
9140 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9141 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
9142 }%
9143 \renewcommand*{\glssxtrinlinefullplformat}[2]{%
9144 \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
9145 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9146 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshortpl{##1}}}%
9147 }%
9148 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9149 \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
9150 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9151 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
9152 }%

```

```

9153 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9154   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9155   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9156   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9157 }%

```

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.

```

9158 \renewcommand*{\glxtrfullformat}[2]{%
9159   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9160   \ifglxtrinsertinside\else##2\fi
9161 }%
9162 \renewcommand*{\glxtrfullplformat}[2]{%
9163   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9164   \ifglxtrinsertinside\else##2\fi
9165 }%
9166 \renewcommand*{\Glsxtrfullformat}[2]{%
9167   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9168   \ifglxtrinsertinside\else##2\fi
9169 }%
9170 \renewcommand*{\Glsxtrfullplformat}[2]{%
9171   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9172   \ifglxtrinsertinside\else##2\fi
9173 }%
9174 }

```

long-desc-em Backward compatibility:

```

9175 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like \glsshort. The long form is emphasized.

```

9176 \newabbreviationstyle{long-em-noshort-em-desc}%
9177 {%
9178   \renewcommand*{\CustomAbbreviationFields}{%
9179     name={\protect\protect\glslongemfont{\the\glslongtok}},
9180     sort={\the\glslongtok},
9181     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9182     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9183     text={\glslongemfont{\the\glslongtok}},
9184     plural={\glslongemfont{\the\glslongpltok}}}%
9185 }%
9186 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9187   \glssetattribute{\the\glslabeltok}{regular}{true}}%
9188 }%
9189 {%
9190   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9191   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9192   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9193   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
9194   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9195 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9196   \glslongemfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9197   \ifglxtrinsertinside \else##2\fi
9198 }%
9199 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9200   \glslongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9201   \ifglxtrinsertinside \else##2\fi
9202 }%
9203 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9204   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9205   \ifglxtrinsertinside \else##2\fi
9206 }%
9207 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9208   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9209   \ifglxtrinsertinside \else##2\fi
9210 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9211 \renewcommand*{\glxtrinlinefullformat}[2]{%
9212   \glslongemfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9213   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9214   \glxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
9215 }%
9216 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9217   \glslongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9218   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9219   \glxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9220 }%
9221 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9222   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9223   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9224   \glxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
9225 }%
9226 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9227   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9228   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9229   \glxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9230 }%

```

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.

```

9231 \renewcommand*{\glxtrfullformat}[2]{%
9232   \glslongemfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9233   \ifglxtrinsertinside\else##2\fi
9234 }%
9235 \renewcommand*{\glxtrfullplformat}[2]{%
9236   \glslongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9237   \ifglxtrinsertinside\else##2\fi
9238 }%

```

```

9239 \renewcommand*{\GlsXtrFullFormat}[2]{%
9240   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9241   \ifglxtrinsertinside\else##2\fi
9242 }%
9243 \renewcommand*{\GlsXtrFullPlFormat}[2]{%
9244   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9245   \ifglxtrinsertinside\else##2\fi
9246 }%
9247 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9248 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9249 {%
9250   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
9251   \UnsetTheRegularAttributeIfSet
9252   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9253     \glshasattribute{\the\glslabeltok}{regular}%
9254     {%
9255       \glissetattribute{\the\glslabeltok}{regular}{false}%
9256     }%
9257   }%
9258 }%
9259 {%
9260   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9261 }

```

ort-em-footnote

```

9262 \newabbreviationstyle{short-em-footnote}%
9263 {%
9264   \renewcommand*{\CustomAbbreviationFields}{%
9265     name={\protect\glsabbrvemfont{\the\glsshorttok}},
9266     sort={\the\glsshorttok},
9267     description={\the\glslongtok},%
9268     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
9269       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9270       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9271     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
9272       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9273       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9274     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}
9275   \SwitchOffHyperlinksOnFirstUse
9276   \glissetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9277   \glshasattribute{\the\glslabeltok}{regular}%
9278   {%
9279     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9280 }%
9281 {}%
9282 }%
9283 }%
9284 {%
9285 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9286 \renewcommand*{\glssabrvfont}[1]{\glssabrvfont{##1}}%
9287 \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvfont{##1}}%
9288 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9289 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9290 \renewcommand*{\glsxtrfullformat}[2]{%
9291   \glsfirstabrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9292   \ifglsxtrininsertinside\else##2\fi
9293   \protect\glsxtrabrvfootnote{##1}%
9294   {\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9295 }%
9296 \renewcommand*{\glsxtrfullplformat}[2]{%
9297   \glsfirstabrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9298   \ifglsxtrininsertinside\else##2\fi
9299   \protect\glsxtrabrvfootnote{##1}%
9300   {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9301 }%
9302 \renewcommand*{\Glsxtrfullformat}[2]{%
9303   \glsfirstabrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9304   \ifglsxtrininsertinside\else##2\fi
9305   \protect\glsxtrabrvfootnote{##1}%
9306   {\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9307 }%
9308 \renewcommand*{\Glsxtrfullplformat}[2]{%
9309   \glsfirstabrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9310   \ifglsxtrininsertinside\else##2\fi
9311   \protect\glsxtrabrvfootnote{##1}%
9312   {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9313 }%

```

The first use full form and the inline full form use the short (long) style.

```

9314 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9315   \glsfirstabrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9316   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9317   \glsxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9318 }%
9319 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9320   \glsfirstabrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9321   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9322   \glsxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9323 }%
9324 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9325   \glsfirstabrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```

```

9326     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9327     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9328 }%
9329 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9330     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9331     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9332     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9333 }%
9334 }

```

footnote-em Backward compatibility:

```

9335 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9336 \newabbreviationstyle{short-em-postfootnote}%
9337 {%
9338   \renewcommand*{\CustomAbbreviationFields}{%
9339     name={\protect\glssabbrvemfont{\the\glssshorttok}},
9340     sort={\the\glssshorttok},
9341     description={\the\glslongtok},%
9342     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},%
9343     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},%
9344     plural={\protect\glssabbrvemfont{\the\glssshortpltok}}}%

```

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.

```

9345 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9346   \csdef{glxtrpostlink\glscategorylabel}{%
9347     \glxtrifwasfirstuse
9348     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9349     \glxtrdopostpunc{\protect\glxtrabbrvf footnote{\glslabel}%
9350     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
9351     }%
9352     {}%
9353     }%
9354     \glshasattribute{\the\glslabeltok}{regular}%
9355     {%
9356     \glissetattribute{\the\glslabeltok}{regular}{false}%
9357     }%
9358     {}%
9359     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9360 \renewcommand*{\glxtrsetupfulldefs}{%
9361   \let\glxtrifwasfirstuse\@secondoftwo
9362   }%

```

```

9363 }%
9364 {%
9365   \renewcommand*{\abbrvpluralsuffix}{\protect\glstremssuffix}%
9366   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9367   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
9368   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
9369   \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9370   \renewcommand*{\glssxtrfullformat}[2]{%
9371     \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9372     \ifglssxtrininsertinside\else##2\fi
9373   }%
9374   \renewcommand*{\glssxtrfullplformat}[2]{%
9375     \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9376     \ifglssxtrininsertinside\else##2\fi
9377   }%
9378   \renewcommand*{\Glsxtrfullformat}[2]{%
9379     \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9380     \ifglssxtrininsertinside\else##2\fi
9381   }%
9382   \renewcommand*{\Glsxtrfullplformat}[2]{%
9383     \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9384     \ifglssxtrininsertinside\else##2\fi
9385   }%

```

The first use full form and the inline full form use the short (long) style.

```

9386   \renewcommand*{\glssxtrininlinefullformat}[2]{%
9387     \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9388     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9389     \glssxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
9390   }%
9391   \renewcommand*{\glssxtrininlinefullplformat}[2]{%
9392     \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9393     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9394     \glssxtrparen{\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9395   }%
9396   \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9397     \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
9398     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9399     \glssxtrparen{\glssfirstlongfootnotefont{\Glsaccesslong{##1}}}%
9400   }%
9401   \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
9402     \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
9403     \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9404     \glssxtrparen{\glssfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
9405   }%
9406 }

```

postfootnote-em Backward compatibility:

9407 \@glxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

## 1.6.6 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.

9408 \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.

```
9409 \ifdef\glscurrentfieldvalue
9410 {
9411   \newcommand*{\glxtruserparen}[2]{%
9412     \glxtrfullsep{#2}%
9413     \glxtrparen
9414     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9415   }
9416 }
9417 {
9418   \newcommand*{\glxtruserparen}[2]{%
9419     \glxtrfullsep{#2}%
9420     \glxtrparen
9421     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9422   }
9423 }
```

Font used for short form:

lsabbrvuserfont

9424 \newcommand\*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

Font used for short form on first use:

stabbrvuserfont

9425 \newcommand\*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

Font used for long form:

glslonguserfont

9426 \newcommand\*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

Font used for long form on first use:

rstlonguserfont

9427 \newcommand\*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}

The default short form suffix:



lsxtrusersuffix

```
9428 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

long-short-user

```
9429 \newabbreviationstyle{long-short-user}%
9430 {%
9431   \renewcommand*{\CustomAbbreviationFields}{%
9432     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9433     sort={\the\glsshorttok},
9434     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9435       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9436     {\the\glslabeltok}},%
9437     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9438       \protect\glxtruserparen
9439       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9440     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9441     description={\protect\glslonguserfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```
9442 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9443   \glshasattribute{\the\glslabeltok}{regular}%
9444   {%
9445     \glissetattribute{\the\glslabeltok}{regular}{false}%
9446   }%
9447   {}%
9448 }%
9449 }%
9450 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
9451 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9452 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9453 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9454 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9455 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
9456 \renewcommand*{\glxtrfullformat}[2]{%
9457   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9458   \ifglxtrinsertinside\else##2\fi
9459   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9460 }%
9461 \renewcommand*{\glxtrfullplformat}[2]{%
9462   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9463   \ifglxtrinsertinside\else##2\fi
9464   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9465 }%
9466 \renewcommand*{\Glsxtrfullformat}[2]{%
9467   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%

```

```

9468     \ifglxtrinsertinside\else##2\fi
9469     \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9470 }%
9471 \renewcommand*{\Glsxtrfullplformat}[2]{%
9472     \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9473     \ifglxtrinsertinside\else##2\fi
9474     \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9475 }%
9476 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9477 \newabbreviationstyle{long-postshort-user}%
9478 {%
9479     \renewcommand*{\CustomAbbreviationFields}{%
9480         name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9481         sort={\the\glsshorttok},
9482         first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9483         firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9484         plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9485         description={\protect\glslonguserfont{\the\glslongtok}}}%
9486 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9487     \csdef{glxtrpostlink\glscategorylabel}{%
9488         \glxtrifwasfirstuse
9489         {%
9490             \glxtruserparen
9491             {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
9492             {\glslabel}%
9493         }%
9494     }%
9495 }%
9496 \glshasattribute{\the\glslabeltok}{regular}%
9497 {%
9498     \glissetattribute{\the\glslabeltok}{regular}{false}%
9499 }%
9500 {}%
9501 }%
9502 }%
9503 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9504 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9505 \renewcommand*{\glsabbrvfnt}[1]{\glsabbrvuserfont{##1}}%
9506 \renewcommand*{\glsfirstabbrvfnt}[1]{\glsfirstabbrvuserfont{##1}}%
9507 \renewcommand*{\glsfirstlongfnt}[1]{\glsfirstlonguserfont{##1}}%
9508 \renewcommand*{\glslongfnt}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9509 \renewcommand*{\glxtrfullformat}[2]{%
9510     \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%

```

```

9511 \ifglxtrinsertinside\else##2\fi
9512 }%
9513 \renewcommand*{\glxtrfullplformat}[2]{%
9514 \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9515 \ifglxtrinsertinside\else##2\fi
9516 }%
9517 \renewcommand*{\Glsxtrfullformat}[2]{%
9518 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9519 \ifglxtrinsertinside\else##2\fi
9520 }%
9521 \renewcommand*{\Glsxtrfullplformat}[2]{%
9522 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9523 \ifglxtrinsertinside\else##2\fi
9524 }%

```

In-line format:

```

9525 \renewcommand*{\glxtrinlinefullformat}[2]{%
9526 \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9527 \ifglxtrinsertinside\else##2\fi
9528 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9529 }%
9530 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9531 \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9532 \ifglxtrinsertinside\else##2\fi
9533 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9534 }%
9535 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9536 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9537 \ifglxtrinsertinside\else##2\fi
9538 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9539 }%
9540 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9541 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9542 \ifglxtrinsertinside\else##2\fi
9543 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9544 }%
9545 }

```

**short-user-desc** Like long-postshort-user but the user supplies the description.

```

9546 \newabbreviationstyle{long-postshort-user-desc}%
9547 {%
9548 \renewcommand*{\CustomAbbreviationFields}{%
9549 name={\protect\glslonguserfont{\the\glslongtok}}%
9550 \protect\glxtruserparen
9551 {\protect\glssabbrvuserfont{\the\glssshorttok}}{\the\glslabeltok}},
9552 sort={\the\glslongtok},
9553 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9554 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9555 text={\protect\glssabbrvuserfont{\the\glssshorttok}},%

```

```

9556 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9557 }%
9558 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9559   \csdef{glxtrpostlink\glscategorylabel}{%
9560     \glxtrifwasfirstuse
9561     {%
9562       \glxtruserparen
9563       {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9564       {\glslabel}%
9565     }%
9566   }%
9567 }%
9568 \glshasattribute{\the\glslabeltok}{regular}%
9569 {%
9570   \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9571 }%
9572 {}%
9573 }%
9574 }%
9575 {%
9576   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9577 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9578 \newabbreviationstyle{short-postlong-user}%
9579 {%
9580   \renewcommand*{\CustomAbbreviationFields}{%
9581     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9582     sort={\the\glsshorttok},
9583     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9584     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9585     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9586     description={\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9587 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9588   \csdef{glxtrpostlink\glscategorylabel}{%
9589     \glxtrifwasfirstuse
9590     {%
9591       \glxtruserparen
9592       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9593       {\glslabel}%
9594     }%
9595   }%
9596 }%
9597 \glshasattribute{\the\glslabeltok}{regular}%
9598 {%
9599   \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9600 }%
9601 {}%
9602 }%

```

9603 }%  
 9604 {%

In case the user wants to mix and match font styles, these are redefined here.

```
9605 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9606 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9607 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9608 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9609 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

First use full form:

```
9610 \renewcommand*{\glxtrfullformat}[2]{%
9611   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9612   \ifglxtrininsertinside\else##2\fi
9613 }%
9614 \renewcommand*{\glxtrfullplformat}[2]{%
9615   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9616   \ifglxtrininsertinside\else##2\fi
9617 }%
9618 \renewcommand*{\Glsxtrfullformat}[2]{%
9619   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9620   \ifglxtrininsertinside\else##2\fi
9621 }%
9622 \renewcommand*{\Glsxtrfullplformat}[2]{%
9623   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9624   \ifglxtrininsertinside\else##2\fi
9625 }%
```

In-line format:

```
9626 \renewcommand*{\glxtrinlinefullformat}[2]{%
9627   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9628   \ifglxtrininsertinside\else##2\fi
9629   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9630 }%
9631 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9632   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9633   \ifglxtrininsertinside\else##2\fi
9634   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9635 }%
9636 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9637   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9638   \ifglxtrininsertinside\else##2\fi
9639   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9640 }%
9641 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9642   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9643   \ifglxtrininsertinside\else##2\fi
9644   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9645 }%
9646 }
```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```
9647 \newabbreviationstyle{short-postlong-user-desc}%
9648 {%
9649   \renewcommand*{\CustomAbbreviationFields}{%
9650     name={\protect\glsabbrvuserfont{\the\glsshorttok}%
9651       \protect\glsxtruserparen
9652         {\protect\glslonguserfont{\the\glslongpltok}}}%
9653       {\the\glslabeltok}},
9654     sort={\the\glsshorttok},
9655     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9656     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9657     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9658     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9659   }%
9660   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9661     \csdef{glsxtrpostlink\glscategorylabel}{%
9662       \glsxtrifwasfirstuse
9663       {%
9664         \glsxtruserparen
9665         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9666         {\glslabel}%
9667       }%
9668     }%
9669   }%
9670   \glsattribute{\the\glslabeltok}{regular}%
9671   {%
9672     \glssetattribute{\the\glslabeltok}{regular}{false}%
9673   }%
9674   {}%
9675 }%
9676 }%
9677 {%
9678   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9679 }
```

short-user-desc

```
9680 \newabbreviationstyle{long-short-user-desc}%
9681 {%
9682   \renewcommand*{\CustomAbbreviationFields}{%
9683     name={\glsxtrlongshortdescname},
9684     sort={\glsxtrlongshortdescsort},%

9685     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9686       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9687     {\the\glslabeltok}},%
9688     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9689       \protect\glsxtruserparen
9690       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9691     text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```

9692 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
9693 }%

```

Unset the regular attribute if it has been set.

```

9694 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9695   \glshasattribute{\the\glslabeltok}{regular}%
9696   {%
9697     \glissetattribute{\the\glslabeltok}{regular}{false}%
9698   }%
9699   {}%
9700 }%
9701 }%
9702 {%
9703   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9704 }

```

short-long-user

```

9705 \newabbreviationstyle{short-long-user}%
9706 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

9707 \renewcommand*{\CustomAbbreviationFields}{%
9708   name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9709   sort={\the\glsshorttok},
9710   description={\protect\glslonguserfont{\the\glslongtok}},%
9711   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9712   \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9713   {\the\glslabeltok}},%
9714   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9715   \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
9716   {\the\glslabeltok}},%
9717   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9718 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9719   \glshasattribute{\the\glslabeltok}{regular}%
9720   {%
9721     \glissetattribute{\the\glslabeltok}{regular}{false}%
9722   }%
9723   {}%
9724 }%
9725 }%
9726 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9727 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9728 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9729 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9730 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9731 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline form are the same for this style.

```

9732 \renewcommand*{\glxtrfullformat}[2]{%
9733   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9734   \ifglxtrinsertinside\else##2\fi
9735   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9736 }%
9737 \renewcommand*{\glxtrfullplformat}[2]{%
9738   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9739   \ifglxtrinsertinside\else##2\fi
9740   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9741 }%
9742 \renewcommand*{\Glsxtrfullformat}[2]{%
9743   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9744   \ifglxtrinsertinside\else##2\fi
9745   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9746 }%
9747 \renewcommand*{\Glsxtrfullplformat}[2]{%
9748   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9749   \ifglxtrinsertinside\else##2\fi
9750   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9751 }%
9752 }

```

-long-user-desc

```

9753 \newabbreviationstyle{short-long-user-desc}%
9754 {%
9755   \renewcommand*{\CustomAbbreviationFields}{%
9756     name={\glxtrshortlongdescname},
9757     sort={\glxtrshortlongdescsort},%
9758     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9759     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9760     {\the\glslabeltok}},%
9761     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9762     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9763     {\the\glslabeltok}},%
9764     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9765     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9766   }%

```

Unset the regular attribute if it has been set.

```

9767 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9768   \glshasattribute{\the\glslabeltok}{regular}%
9769   {%
9770     \glssetattribute{\the\glslabeltok}{regular}{false}%
9771   }%
9772   {}%
9773 }%
9774 }%
9775 {%

```



```

9776 \GlsXtrUseAbbrStyleFmts{short-long-user}%
9777 }

```

### 1.6.7 Predefined Styles (Hyphen)

These styles are designed to work with the `markwords` 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 `\glstrlong` set `\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.

```

9778 \newrobustcmd*{\glstrtrifhyphenstart}[3]{%
9779   \ifx\glsinsert#1\relax
9780   \expandafter\@glstrtrifhyphenstart#1\relax\relax
9781   \@end@glstrtrifhyphenstart{#2}{#3}%
9782   \else
9783   \@glstrtrifhyphenstart#1\relax\relax\@end@glstrtrifhyphenstart{#2}{#3}%
9784   \fi
9785 }

```

`\trifhyphenstart`

```

9786 \def\@glstrtrifhyphenstart#1#2\@end@glstrtrifhyphenstart#3#4{%
9787   \ifx-#1\relax#3\else #4\fi
9788 }

```

`\trlonghyphenshort`

```
\glstrlonghyphenshort{<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.

```

9789 \newcommand*{\glstrlonghyphenshort}[4]{%

```

Grouping is needed to localise the redefinitions.

```

9790 {%

```

If `<insert>` starts with a hyphen, redefine `\glstrwordsep` 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 `\glstrwordsep` if `<insert>` doesn't start with a hyphen.

```

9791   \glstrtrifhyphenstart{#4}{\def\glstrwordsep{-}}{}%
9792   \glstrfirstlonghyphenfont{#2\ifglstrinsertinside{#4}\fi}%
9793   \ifglstrinsertinside\else{#4}\fi
9794   \glstrfullsep{#1}%
9795   \glstrparen{\glstrfirstabbrvhyphenfont{#3\ifglstrinsertinside{#4}\fi}%
9796   \ifglstrinsertinside\else{#4}\fi}%
9797 }%
9798 }

```

abbrvhyphenfont

```
9799 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
9800 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
9801 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%
```

tlonghyphenfont

```
9802 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
9803 \newcommand*{\glsxtrhyphensuffix}{\glsxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the markwords attribute.

```
9804 \newabbreviationstyle{long-hyphen-short-hyphen}%
9805 {%
9806   \renewcommand*{\CustomAbbreviationFields}{%
9807     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9808     sort={\the\glsshorttok},
9809     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
9810     \protect\glsxtrfullsep{\the\glslabeltok}%
9811     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9812     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
9813     \protect\glsxtrfullsep{\the\glslabeltok}%
9814     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9815     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9816     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
9817   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9818     \glshasattribute{\the\glslabeltok}{regular}%
9819     {%
9820       \glssetattribute{\the\glslabeltok}{regular}{false}%
9821       }%
9822     {}%
9823   }%
9824 }%
9825 {%
9826   \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9827   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9828   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9829   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9830   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

9831 \renewcommand*{\glxstrfullformat}[2]{%
9832   \glxstrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9833 }%
9834 \renewcommand*{\glxstrfullplformat}[2]{%
9835   \glxstrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
9836   {\glsaccessshortpl{##1}}{##2}%
9837 }%
9838 \renewcommand*{\Glsxstrfullformat}[2]{%
9839   \glxstrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9840 }%
9841 \renewcommand*{\Glsxstrfullplformat}[2]{%
9842   \glxstrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9843   {\glsaccessshortpl{##1}}{##2}%
9844 }%
9845 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9846 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9847 {%
9848   \renewcommand*{\CustomAbbreviationFields}{%
9849     name={\glxstrlongshortdescname},
9850     sort={\glxstrlongshortdescsort},
9851     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
9852     \protect\glxstrfullsep{\the\glslabeltok}}%
9853     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9854     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
9855     \protect\glxstrfullsep{\the\glslabeltok}}%
9856     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9857     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}}},%
9858     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9859 }%

```

Unset the regular attribute if it has been set.

```

9860 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9861   \glshasattribute{\the\glslabeltok}{regular}%
9862   {%
9863     \glissetattribute{\the\glslabeltok}{regular}{false}%
9864   }%
9865   {}%
9866 }%
9867 }%
9868 {%
9869   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9870 }

```

onghyphennoshort `\glxstrlonghyphennoshort{<label>}{<long>}{<insert>}`

```
9871 \newcommand*{\glxtrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
9872 {%
```

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.

```
9873 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
```

```
9874 \glsfirslonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
```

```
9875 \ifglxtrininsertinsideelse{#3}\fi
```

```
9876 }%
```

```
9877 }
```

`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.

```
9878 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
```

```
9879 {%
```

```
9880 \renewcommand*{\CustomAbbreviationFields}{%
```

```
9881 name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
```

```
9882 sort={\expandonce\glxtrorglong},
```

```
9883 first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
```

```
9884 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
```

```
9885 plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
```

```
9886 }%
```

Unset the regular attribute if it has been set.

```
9887 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
9888 \glshasattribute{\the\glslabeltok}{regular}%
```

```
9889 {%
```

```
9890 \glissetattribute{\the\glslabeltok}{regular}{false}%
```

```
9891 }%
```

```
9892 {}%
```

```
9893 }%
```

```
9894 }%
```

```
9895 {%
```

```
9896 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9897 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
```

```
9898 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
```

```
9899 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```

```
9900 \renewcommand*{\glsfirslongfont}[1]{\glsfirslonghyphenfont{##1}}%
```

```
9901 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9902 \renewcommand*{\glxtrsubsequentfmt}[2]{%
```

```
9903 \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
```

```
9904 }%
```

```

9905 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9906   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9907 }%
9908 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9909   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9910 }%
9911 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9912   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9913 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9914 \renewcommand*{\glxtrinlinefullformat}[2]{%
9915   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
9916   \glxtrfullsep{##1}%
9917   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
9918 }%
9919 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9920   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9921   \glxtrfullsep{##1}%
9922   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
9923 }%
9924 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9925   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9926   \glxtrfullsep{##1}%
9927   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
9928 }%
9929 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9930   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9931   \glxtrfullsep{##1}%
9932   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
9933 }%

```

The first use full form only displays the long form.

```

9934 \renewcommand*{\glxtrfullformat}[2]{%
9935   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
9936 }%
9937 \renewcommand*{\glxtrfullplformat}[2]{%
9938   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9939 }%
9940 \renewcommand*{\Glsxtrfullformat}[2]{%
9941   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9942 }%
9943 \renewcommand*{\Glsxtrfullplformat}[2]{%
9944   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9945 }%
9946 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a descriptionless 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.

```

9947 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9948 {%
9949   \renewcommand*{\CustomAbbreviationFields}{%
9950     name={\protect\glsabbrvfont{\the\glsshorttok}},
9951     sort={\the\glsshorttok},
9952     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9953     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9954     text={\protect\glslonghyphenfont{\the\glslongtok}},%
9955     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
9956     description={\the\glslongtok}%
9957   }%

```

Unset the regular attribute if it has been set.

```

9958   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9959     \glshasattribute{\the\glslabeltok}{regular}%
9960     {%
9961       \glssetattribute{\the\glslabeltok}{regular}{false}%
9962     }%
9963   }%
9964 }%
9965 }%
9966 {%
9967   \GlsXtrUseAbbrStyleFmts{long-desc}%
9968 }

```

glsxtrlonghyphen

`\glsxtrlonghyphen{<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.

```

9969 \newcommand*{\glsxtrlonghyphen}[3]{%
9970   {%
9971     \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9972     \glsfirstlonghyphenfont{#1}%
9973   }%
9974 }

```

rposthyphenshort

`\glsxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glsxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```

9975 \newcommand*{\glsxtrposthyphenshort}[2]{%
9976   {%

```

```

9977 \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
9978 \ifglxtrininsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
9979 \glxtrfullsep{#1}%
9980 \glxtrparen
9981 {\glsfirstabbrvhyphenfont{\glsentryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
9982 \ifglxtrininsertinside\else{#2}\fi
9983 }%
9984 }%
9985 }

```

hyphensubsequent `\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

9986 \newcommand*{\glxtrposthyphensubsequent}[2]{%
9987 \glsabbrvfont{\ifglxtrininsertinside {#2}\fi}%
9988 \ifglxtrininsertinside \else{#2}\fi
9989 }

```

postshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9990 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
9991 {%
9992 \renewcommand*{\CustomAbbreviationFields}{%
9993 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9994 sort={\the\glsshorttok},
9995 first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
9996 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
9997 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9998 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9999 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10000 \csdef{glxtrpostlink\glscategorylabel}{%
10001 \glxtrifwasfirstuse
10002 {%
10003 \glxtrposthyphenshort{\glslabel}{\glsinsert}%
10004 }%
10005 }%

```

Put the insertion into the post-link:

```

10006 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10007 }%
10008 }%
10009 \glshasattribute{\the\glslabeltok}{regular}%
10010 {%
10011 \glissetattribute{\the\glslabeltok}{regular}{false}%
10012 }%
10013 {}%
10014 }%

```

10015 }%  
 10016 {%

In case the user wants to mix and match font styles, these are redefined here.

10017 \renewcommand\*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%  
 10018 \renewcommand\*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%  
 10019 \renewcommand\*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%  
 10020 \renewcommand\*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%  
 10021 \renewcommand\*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

Subsequent use needs to omit the insertion:

10022 \renewcommand\*{\glxtrsubsequentfmt}[2]{%  
 10023 \glsabbrvfont{\glsaccessshort{##1}}%  
 10024 }%  
 10025 \renewcommand\*{\glxtrsubsequentplfmt}[2]{%  
 10026 \glsabbrvfont{\glsaccessshortpl{##1}}%  
 10027 }%  
 10028 \renewcommand\*{\Glsxtrsubsequentfmt}[2]{%  
 10029 \glsabbrvfont{\Glsaccessshort{##1}}%  
 10030 }%  
 10031 \renewcommand\*{\Glsxtrsubsequentplfmt}[2]{%  
 10032 \glsabbrvfont{\Glsaccessshortpl{##1}}%  
 10033 }%

First use full form:

10034 \renewcommand\*{\glxtrfullformat}[2]{%  
 10035 \glxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%  
 10036 }%  
 10037 \renewcommand\*{\glxtrfullplformat}[2]{%  
 10038 \glxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%  
 10039 }%  
 10040 \renewcommand\*{\Glsxtrfullformat}[2]{%  
 10041 \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%  
 10042 }%  
 10043 \renewcommand\*{\Glsxtrfullplformat}[2]{%  
 10044 \glxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%  
 10045 }%

In-line format.

10046 \renewcommand\*{\glxtrinilinefullformat}[2]{%  
 10047 \glsfirstlonghyphenfont{\glsaccesslong{##1}}%  
 10048 \ifglxtrininsertinside{##2}\fi}%  
 10049 \ifglxtrininsertinside \else{##2}\fi  
 10050 }%  
 10051 \renewcommand\*{\glxtrinilinefullplformat}[2]{%  
 10052 \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%  
 10053 \ifglxtrininsertinside{##2}\fi}%  
 10054 \ifglxtrininsertinside \else{##2}\fi  
 10055 }%  
 10056 \renewcommand\*{\Glsxtrinilinefullformat}[2]{%  
 10057 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%



```

10058     \ifglxtrinsertinside{##2}\fi}%
10059     \ifglxtrinsertinside \else{##2}\fi
10060 }%
10061 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10062     \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10063     \ifglxtrinsertinside{##2}\fi}%
10064     \ifglxtrinsertinside \else{##2}\fi
10065 }%
10066 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10067 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10068 {%
10069     \renewcommand*{\CustomAbbreviationFields}{%
10070         name={\glxtrlongshortdescname},
10071         sort={\glxtrlongshortdescsort},%
10072         first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10073         firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10074         text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10075         plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10076 }%
10077 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10078     \csdef{glxtrpostlink\glscategorylabel}{%
10079         \glxtrifwasfirstuse
10080         {%
10081             \glxtrposthyphenshort{\glslabel}{\glsinsert}%
10082         }%
10083         {%

```

Put the insertion into the post-link:

```

10084         \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10085     }%
10086 }%
10087 \glshasattribute{\the\glslabeltok}{regular}%
10088 {%
10089     \glissetattribute{\the\glslabeltok}{regular}{false}%
10090 }%
10091 {}%
10092 }%
10093 }%
10094 {%
10095     \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10096 }

```

`rshorthyphenlong` `\glxtrshorthyphenlong{<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.

```
10097 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10098 {%
```

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.)

```
10099 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
```

```
10100 \glsfirtabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
```

```
10101 \ifglxtrininsertinside\else{#4}\fi
```

```
10102 \glxtrfullsep{#1}%
```

```
10103 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
```

```
10104 \ifglxtrininsertinside\else{#4}\fi}%
```

```
10105 }%
```

```
10106 }
```

hen-long-hyphen Designed for use with the markwords attribute.

```
10107 \newabbreviationstyle{short-hyphen-long-hyphen}%
```

```
10108 {%
```

```
10109 \renewcommand*{\CustomAbbreviationFields}{%
```

```
10110 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
```

```
10111 sort={\the\glsshorttok},
```

```
10112 first={\protect\glsfirtabbrvhyphenfont{\the\glsshorttok}}%
```

```
10113 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
10114 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
```

```
10115 firstplural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
```

```
10116 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
10117 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
```

```
10118 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%
```

```
10119 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
10120 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
10121 \glshasattribute{\the\glslabeltok}{regular}}%
```

```
10122 {%
```

```
10123 \glissetattribute{\the\glslabeltok}{regular}{false}}%
```

```
10124 }%
```

```
10125 {}%
```

```
10126 }%
```

```
10127 }%
```

```
10128 {%
```

```
10129 \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
```

```
10130 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
```

```
10131 \renewcommand*{\glsfirtabbrvfont}[1]{\glsfirtabbrvhyphenfont{##1}}%
```

```
10132 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
```

```
10133 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

10134 \renewcommand*{\glxtrfullformat}[2]{%
10135   \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10136 }%
10137 \renewcommand*{\glxtrfullplformat}[2]{%
10138   \glxtrshorthyphenlong{##1}%
10139   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10140 }%
10141 \renewcommand*{\Glsxtrfullformat}[2]{%
10142   \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10143 }%
10144 \renewcommand*{\Glsxtrfullplformat}[2]{%
10145   \glxtrshorthyphenlong{##1}%
10146   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10147 }%
10148 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10149 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10150 {%
10151   \renewcommand*{\CustomAbbreviationFields}{%
10152     name={\glxtrshortlongdescname},
10153     sort={\glxtrshortlongdescsort},
10154     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
10155     \protect\glxtrfullsep{\the\glslabeltok}%
10156     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10157     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
10158     \protect\glxtrfullsep{\the\glslabeltok}%
10159     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10160     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10161     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10162 }%

```

Unset the regular attribute if it has been set.

```

10163 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10164   \glshasattribute{\the\glslabeltok}{regular}%
10165   {%
10166     \glissetattribute{\the\glslabeltok}{regular}{false}%
10167   }%
10168   {}%
10169 }%
10170 }%
10171 {%
10172   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10173 }

```

lsxtrshorthyphen `\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.

```
10174 \newcommand*{\glxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
10175 {%
10176   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
10177   \glsfirstabbrvhyphenfont{#1}%
10178 }%
10179 }
```

trposthyphenlong `\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.

```
10180 \newcommand*{\glxtrposthyphenlong}[2]{%
10181 {%
10182   \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
10183   \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10184   \glxtrfullsep{#1}%
10185   \glxtrparen
10186   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
10187   \ifglxtrininsertinside\else{#2}\fi
10188 }%
10189 }%
10190 }
```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
10191 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10192 {%
10193   \renewcommand*{\CustomAbbreviationFields}{%
10194     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
10195     sort={\the\glsshorttok},
10196     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10197     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10198     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10199     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10200   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10201     \csdef{glxtrpostlink\glscategorylabel}{%
10202       \glxtrifwasfirstuse
10203       {%
10204         \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10205       }%
10206     }%
```

Put the insertion into the post-link:

```

10207      \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10208    }%
10209  }%
10210  \glshasattribute{\the\glslabeltok}{regular}%
10211  {%
10212    \glsselattribute{\the\glslabeltok}{regular}{false}%
10213  }%
10214  {}%
10215 }%
10216}%
10217{%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10218 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10219 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10220 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10221 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10222 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10223 \renewcommand*{\glxtrs subsequentfmt}[2]{%
10224   \glsabbrvfont{\glsaccessshort{##1}}%
10225 }%
10226 \renewcommand*{\glxtrs subsequentplfmt}[2]{%
10227   \glsabbrvfont{\glsaccessshortpl{##1}}%
10228 }%
10229 \renewcommand*{\Glsxtrs subsequentfmt}[2]{%
10230   \glsabbrvfont{\Glsaccessshort{##1}}%
10231 }%
10232 \renewcommand*{\Glsxtrs subsequentplfmt}[2]{%
10233   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10234 }%

```

First use full form:

```

10235 \renewcommand*{\glxtrfullformat}[2]{%
10236   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10237 }%
10238 \renewcommand*{\glxtrfullplformat}[2]{%
10239   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10240 }%
10241 \renewcommand*{\Glsxtrfullformat}[2]{%
10242   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10243 }%
10244 \renewcommand*{\Glsxtrfullplformat}[2]{%
10245   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10246 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10247 \renewcommand*{\glxtrinlinefullformat}[2]{%

```

```

10248 \glsfirstabbrvhyphenfont{\glsaccessshort{##1}%
10249 \ifglxtrinsertinside{##2}\fi}%
10250 \ifglxtrinsertinside \else{##2}\fi
10251 }%
10252 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10253 \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}%
10254 \ifglxtrinsertinside{##2}\fi}%
10255 \ifglxtrinsertinside \else{##2}\fi
10256 }%
10257 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10258 \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}%
10259 \ifglxtrinsertinside{##2}\fi}%
10260 \ifglxtrinsertinside \else{##2}\fi
10261 }%
10262 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10263 \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
10264 \ifglxtrinsertinside{##2}\fi}%
10265 \ifglxtrinsertinside \else{##2}\fi
10266 }%
10267 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

10268 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10269 {%
10270 \renewcommand*{\CustomAbbreviationFields}{%
10271 name={\glxtrshortlongdescname},
10272 sort={\glxtrshortlongdescsort},%
10273 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10274 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10275 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10276 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10277 }%
10278 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10279 \csdef{glxtrpostlink\glscategorylabel}{%
10280 \glxtrifwasfirstuse
10281 {%
10282 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10283 }%
10284 {%

```

Put the insertion into the post-link:

```

10285 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10286 }%
10287 }%
10288 \glshasattribute{\the\glslabeltok}{regular}%
10289 {%
10290 \glissetattribute{\the\glslabeltok}{regular}{false}%
10291 }%
10292 {}%
10293 }%

```

```

10294 }%
10295 {%
10296   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10297 }

```

### 1.6.8 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

```

10298 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%

```

stabbrvonlyfont

```

10299 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%

```

glslongonlyfont

```

10300 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

rstlongonlyfont

```

10301 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

lsxtronlysuffix

```

10302 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}

```

only-short-only

```

10303 \newabbreviationstyle{long-only-short-only}%
10304 {%
10305   \renewcommand*{\CustomAbbreviationFields}{%
10306     name={\protect\glsabbrvonlyfont{\the\glsshorttok}},
10307     sort={\the\glsshorttok},
10308     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10309     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10310     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10311     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10312 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10313   \glshasattribute{\the\glslabeltok}{regular}%
10314   {%
10315     \glssetattribute{\the\glslabeltok}{regular}{false}%
10316   }%
10317   {}%
10318 }%
10319 }%
10320 {%
10321   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtronlysuffix}%
10322   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%

```

```

10323 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10324 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10325 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

10326 \renewcommand*{\glsxtrfullformat}[2]{%
10327   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10328   \ifglsxtrininsertinside\else##2\fi
10329 }%
10330 \renewcommand*{\glsxtrfullplformat}[2]{%
10331   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10332   \ifglsxtrininsertinside\else##2\fi
10333 }%
10334 \renewcommand*{\Glsxtrfullformat}[2]{%
10335   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10336   \ifglsxtrininsertinside\else##2\fi
10337 }%
10338 \renewcommand*{\Glsxtrfullplformat}[2]{%
10339   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10340   \ifglsxtrininsertinside\else##2\fi
10341 }%

```

The inline full form does show the short form.

```

10342 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10343   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10344   \ifglsxtrininsertinside\else##2\fi
10345   \glsxtrfullsep{##1}%
10346   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10347 }%
10348 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10349   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10350   \ifglsxtrininsertinside\else##2\fi
10351   \glsxtrfullsep{##1}%
10352   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10353 }%
10354 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10355   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10356   \ifglsxtrininsertinside\else##2\fi
10357   \glsxtrfullsep{##1}%
10358   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10359 }%
10360 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10361   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10362   \ifglsxtrininsertinside\else##2\fi
10363   \glsxtrfullsep{##1}%
10364   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10365 }%
10366 }

```

xtronlydescsort



```

10367 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

xtronlydescname
10368 \newcommand*{\glxtronlydescname}{%
10369   \protect\glslongfont{\the\glslongtok}%
10370 }

short-only-desc
10371 \newabbreviationstyle{long-only-short-only-desc}%
10372 {%
10373   \renewcommand*{\CustomAbbreviationFields}{%
10374     name={\glxtronlydescname},
10375     sort={\glxtronlydescsort},%
10376     first={\protect\glslongonlyfont{\the\glslongtok}},%
10377     firstplural={\protect\glslongonlyfont{\the\glslongpltok}},%
10378     text={\protect\glssabbrvonlyfont{\the\glsshorttok}},%
10379     plural={\protect\glssabbrvonlyfont{\the\glsshortpltok}}}%
10380   }%

   Unset the regular attribute if it has been set.
10381   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10382     \glshasattribute{\the\glslabeltok}{regular}%
10383     {%
10384       \glissetattribute{\the\glslabeltok}{regular}{false}%
10385     }%
10386     {}%
10387   }%
10388 }%
10389 {%
10390   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10391 }

```

## 1.7 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 `\glssabbrvfont`. 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 `\glxtrshort` 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:

```
10392 \let\@glxstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
10393 \renewcommand*{\markright}[1]{%
```

```
10394 \glxstrmarkhook
```

```
10395 \@glxstr@org@markright{\@glxstrinmark#1\@glxstrnotinmark}%
```

```
10396 \glxstrrestoremarkhook
```

```
10397 }
```

`\markboth` Save original definition:

```
10398 \let\@glxstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
10399 \renewcommand*{\markboth}[2]{%
```

```
10400 \glxstrmarkhook
```

```
10401 \@glxstr@org@markboth
```

```
10402 {\@glxstrinmark#1\@glxstrnotinmark}%
```

```
10403 {\@glxstrinmark#2\@glxstrnotinmark}%
```

```
10404 \glxstrrestoremarkhook
```

```
10405 }
```

Also do this for `\@starttoc`

`\@starttoc` Save original definition:

```
10406 \let\@glxstr@org@@starttoc\@starttoc
```

Redefine:

```
10407 \renewcommand*{\@starttoc}[1]{%
```

```
10408 \glxstrmarkhook
```

```
10409 \@glxstrinmark
```

```
10410 \@glxstr@org@@starttoc{#1}%
```

```
10411 \@glxstrnotinmark
```

```
10412 \glxstrrestoremarkhook
```

```
10413 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`sxtrRevertMarks`

```
10414 \newcommand*{\glxstrRevertMarks}{%
```

```
10415 \let\markright\@glxstr@org@markright
```

```

10416 \let\markboth\@glxtr@org@markboth
10417 \let\@starttoc\@glxtr@org\@starttoc
10418 }

```

\glxtrifinmark

```

10419 \newcommand*\glxtrifinmark}[2]{#2}

```

\@glxtrinmark

```

10420 \newrobustcmd*\@glxtrinmark){%
10421 \let\glxtrifinmark\@firstoftwo
10422 }

```

glxtrnotinmark

```

10423 \newrobustcmd*\glxtrnotinmark){%
10424 \let\glxtrifinmark\@secondoftwo
10425 }

```

eorpdforheading

```

10426 \ifdef\texorpdfstring
10427 {
10428 \newcommand*\glxtrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10429 }
10430 {
10431 \newcommand*\glxtrtitleorpdforheading}[3]{#1}
10432 }

```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

10433 \newcommand*\glxtrmarkhook){%

```

Save current definitions:

```

10434 \let\@glxtr@org@MakeUppercase\MakeUppercase
10435 \let\@glxtr@org@glxtrtitleorpdforheading\glxtrtitleorpdforheading
10436 \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
10437 \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
10438 \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10439 \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
10440 \let\@glxtr@org@glxtrtitlename\glxtrtitlename
10441 \let\@glxtr@org@Glsxtrtitlename\Glsxtrtitlename
10442 \let\@glxtr@org@glxtrtitletext\glxtrtitletext
10443 \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
10444 \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
10445 \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10446 \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
10447 \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10448 \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
10449 \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10450 \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
10451 \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl

```

```

10452 \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
10453 \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10454 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
10455 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
10456 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
10457 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

#### New definitions

```

10458 \let\glsxtrifinmark\@firstoftwo
10459 \let\MakeUppercase\MakeTextUppercase
10460 \let\glsxtrtitleorpdforheading\@thirdofthree
10461 \let\glsxtrtitleshort\glsxtrheadshort
10462 \let\glsxtrtitleshortpl\glsxtrheadshortpl
10463 \let\Glsxtrtitleshort\Glsxtrheadshort
10464 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
10465 \let\glsxtrtitlename\glsxtrheadname
10466 \let\Glsxtrtitlename\Glsxtrheadname
10467 \let\glsxtrtitletext\glsxtrheadtext
10468 \let\Glsxtrtitletext\Glsxtrheadtext
10469 \let\glsxtrtitleplural\glsxtrheadplural
10470 \let\Glsxtrtitleplural\Glsxtrheadplural
10471 \let\glsxtrtitlefirst\glsxtrheadfirst
10472 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10473 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
10474 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10475 \let\glsxtrtitlelong\glsxtrheadlong
10476 \let\glsxtrtitlelongpl\glsxtrheadlongpl
10477 \let\Glsxtrtitlelong\Glsxtrheadlong
10478 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10479 \let\glsxtrtitlefull\glsxtrheadfull
10480 \let\glsxtrtitlefullpl\glsxtrheadfullpl
10481 \let\Glsxtrtitlefull\Glsxtrheadfull
10482 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10483 }

```

`\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.)

```

10484 \newcommand*{\glsxtrrestoremarkhook}{%
10485 \let\glsxtrifinmark\@secondoftwo
10486 \let\MakeUppercase\@glsxtr@org@MakeUppercase
10487 \let\glsxtrtitleorpdforheading\@glsxtr@org@glsxtrtitleorpdforheading
10488 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
10489 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
10490 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
10491 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
10492 \let\glsxtrtitlename\@glsxtr@org@glsxtrtitlename
10493 \let\Glsxtrtitlename\@glsxtr@org@Glsxtrtitlename

```

```

10494 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
10495 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
10496 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
10497 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
10498 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
10499 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
10500 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
10501 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
10502 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
10503 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
10504 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
10505 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
10506 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
10507 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
10508 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
10509 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
10510 }

```

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.

```

10511 \newcommand*{\glsxtrheadshort}[1]{%
10512 \protect\NoCaseChange
10513 {%
10514 \glsifattribute{#1}{headuc}{true}%
10515 {%
10516 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10517 }%
10518 {%
10519 \glsxtrshort[noindex,hyper=false]{#1}[]%
10520 }%
10521 }%
10522 }

```

**lgsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents.

```

10523 \newrobustcmd*{\lgsxtrtitleshort}[1]{%
10524 \glsxtrshort[noindex,hyper=false]{#1}[]%
10525 }

```

**glsxtrheadshortpl** 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.

```

10526 \newcommand*{\glsxtrheadshortpl}[1]{%
10527 \protect\NoCaseChange
10528 {%
10529 \glsifattribute{#1}{headuc}{true}%
10530 {%
10531 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%

```

```

10532 }%
10533 {%
10534 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10535 }%
10536 }%
10537 }

```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents.

```

10538 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10539 \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10540 }

```

**Glsxtrheadshort** Command used to display short form in the page header with the first letter converted to upper case.

```

10541 \newcommand*{\Glsxtrheadshort}[1]{%
10542 \protect\NoCaseChange
10543 {%
10544 \glsifattribute{#1}{headuc}{true}%
10545 {%
10546 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10547 }%
10548 {%
10549 \Glsxtrshort[noindex,hyper=false]{#1}[]%
10550 }%
10551 }%
10552 }

```

**lsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10553 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10554 \Glsxtrshort[noindex,hyper=false]{#1}[]%
10555 }

```

**sxtrheadshortpl** Command used to display plural short form in the page header with the first letter converted to upper case.

```

10556 \newcommand*{\Glsxtrheadshortpl}[1]{%
10557 \protect\NoCaseChange
10558 {%
10559 \glsifattribute{#1}{headuc}{true}%
10560 {%
10561 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10562 }%
10563 {%
10564 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10565 }%
10566 }%
10567 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10568 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10569   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10570 }
```

`\glxtrheadname` As above but for the name value.

```
10571 \newcommand*{\glxtrheadname}[1]{%
10572   \protect\NoCaseChange
10573   {%
10574     \glsifattribute{#1}{headuc}{true}%
10575     {%
10576       \GLSname[noindex,hyper=false]{#1}[]%
10577     }%
10578     {%
10579       \glsname[noindex,hyper=false]{#1}[]%
10580     }%
10581   }%
10582 }
```

`glxtrtitlename` Command to display name value in section title and table of contents.

```
10583 \newrobustcmd*{\glxtrtitlename}[1]{%
10584   \glsname[noindex,hyper=false]{#1}[]%
10585 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10586 \newcommand*{\Glsxtrheadname}[1]{%
10587   \protect\NoCaseChange
10588   {%
10589     \glsifattribute{#1}{headuc}{true}%
10590     {%
10591       \GLSname[noindex,hyper=false]{#1}[]%
10592     }%
10593     {%
10594       \Glsname[noindex,hyper=false]{#1}[]%
10595     }%
10596   }%
10597 }
```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
10598 %\changes{1.21}{2017-11-03}{new}
10599 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10600   \Glsname[noindex,hyper=false]{#1}[]%
10601 }
```

`\glxtrheadtext` As above but for the text value.

```
10602 \newcommand*{\glxtrheadtext}[1]{%
```

```

10603 \protect\NoCaseChange
10604 {%
10605   \glsifattribute{#1}{headuc}{true}%
10606   {%
10607     \GLStext[noindex,hyper=false]{#1}[]%
10608   }%
10609   {%
10610     \glstext[noindex,hyper=false]{#1}[]%
10611   }%
10612 }%
10613 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

10614 \newrobustcmd*{\glsxtrtitletext}[1]{%
10615   \glstext[noindex,hyper=false]{#1}[]%
10616 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

10617 \newcommand*{\Glsxtrheadtext}[1]{%
10618   \protect\NoCaseChange
10619   {%
10620     \glsifattribute{#1}{headuc}{true}%
10621     {%
10622       \GLStext[noindex,hyper=false]{#1}[]%
10623     }%
10624     {%
10625       \Glstext[noindex,hyper=false]{#1}[]%
10626     }%
10627   }%
10628 }

```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

10629 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10630   \Glstext[noindex,hyper=false]{#1}[]%
10631 }

```

`lslxtrheadplural` As above but for the plural value.

```

10632 \newcommand*{\glsxtrheadplural}[1]{%
10633   \protect\NoCaseChange
10634   {%
10635     \glsifattribute{#1}{headuc}{true}%
10636     {%
10637       \GLSplural[noindex,hyper=false]{#1}[]%
10638     }%
10639     {%
10640       \glsplural[noindex,hyper=false]{#1}[]%
10641     }%
10642   }%

```



10643 }

**sxtrtitleplural** Command to display plural value in section title and table of contents.

```
10644 \newrobustcmd*{\glsxtrtitleplural}[1]{%
10645   \glsplural[noindex,hyper=false]{#1}[]%
10646 }
```

**lsxtrheadplural** Convert first letter to upper case.

```
10647 \newcommand*{\Glsxtrheadplural}[1]{%
10648   \protect\NoCaseChange
10649   {%
10650     \glusifattribute{#1}{headuc}{true}%
10651     {%
10652       \GLSplural[noindex,hyper=false]{#1}[]%
10653     }%
10654     {%
10655       \GLsplural[noindex,hyper=false]{#1}[]%
10656     }%
10657   }%
10658 }
```

**sxtrtitleplural** Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
10659 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10660   \GLsplural[noindex,hyper=false]{#1}[]%
10661 }
```

**glsxtrheadfirst** As above but for the first value.

```
10662 \newcommand*{\glsxtrheadfirst}[1]{%
10663   \protect\NoCaseChange
10664   {%
10665     \glusifattribute{#1}{headuc}{true}%
10666     {%
10667       \GLSfirst[noindex,hyper=false]{#1}[]%
10668     }%
10669     {%
10670       \glsfirst[noindex,hyper=false]{#1}[]%
10671     }%
10672   }%
10673 }
```

**lsxtrtitlefirst** Command to display first value in section title and table of contents.

```
10674 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10675   \glsfirst[noindex,hyper=false]{#1}[]%
10676 }
```

**Glsxtrheadfirst** First letter converted to upper case

```
10677 \newcommand*{\Glsxtrheadfirst}[1]{%
```

```

10678 \protect\NoCaseChange
10679 {%
10680 \glsifattribute{#1}{headuc}{true}%
10681 {%
10682 \GLSfirst[noindex,hyper=false]{#1}[]%
10683 }%
10684 {%
10685 \Glsfirst[noindex,hyper=false]{#1}[]%
10686 }%
10687 }%
10688 }

```

**lsxtrtitlefirst** Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10689 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10690 \GLSfirst[noindex,hyper=false]{#1}[]%
10691 }

```

**headfirstplural** As above but for the firstplural value.

```

10692 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10693 \protect\NoCaseChange
10694 {%
10695 \glsifattribute{#1}{headuc}{true}%
10696 {%
10697 \GLSfirstplural[noindex,hyper=false]{#1}[]%
10698 }%
10699 {%
10700 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10701 }%
10702 }%
10703 }

```

**itlefirstplural** Command to display firstplural value in section title and table of contents.

```

10704 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
10705 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10706 }

```

**headfirstplural** First letter converted to upper case

```

10707 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10708 \protect\NoCaseChange
10709 {%
10710 \glsifattribute{#1}{headuc}{true}%
10711 {%
10712 \GLSfirstplural[noindex,hyper=false]{#1}[]%
10713 }%
10714 {%
10715 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10716 }%
10717 }%

```

10718 }

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10719 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10720   \Glsfirstplural[noindex,hyper=false]{#1}[]%
10721 }
```

`\glxstrheadlong` Command used to display long form in the page header.

```
10722 \newcommand*{\glxstrheadlong}[1]{%
10723   \protect\NoCaseChange
10724   {%
10725     \glsifattribute{#1}{headuc}{true}%
10726     {%
10727       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10728     }%
10729     {%
10730       \glxstrlong[noindex,hyper=false]{#1}[]%
10731     }%
10732   }%
10733 }
```

`\glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
10734 \newrobustcmd*{\glxstrtitlelong}[1]{%
10735   \glxstrlong[noindex,hyper=false]{#1}[]%
10736 }
```

`\glxstrheadlongpl` 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.

```
10737 \newcommand*{\glxstrheadlongpl}[1]{%
10738   \protect\NoCaseChange
10739   {%
10740     \glsifattribute{#1}{headuc}{true}%
10741     {%
10742       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10743     }%
10744     {%
10745       \glxstrlongpl[noindex,hyper=false]{#1}[]%
10746     }%
10747   }%
10748 }
```

`\glxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
10749 \newrobustcmd*{\glxstrtitlelongpl}[1]{%
10750   \glxstrlongpl[noindex,hyper=false]{#1}[]%
10751 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

10752 \newcommand*{\Glsxtrheadlong}[1]{%
10753   \protect\NoCaseChange
10754   {%
10755     \glsifattribute{#1}{headuc}{true}%
10756     {%
10757       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10758     }%
10759     {%
10760       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10761     }%
10762   }%
10763 }

```

`\Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10764 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10765   \GLSxtrlong[noindex,hyper=false]{#1}[]%
10766 }

```

`\Glsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

10767 \newcommand*{\Glsxtrheadlongpl}[1]{%
10768   \protect\NoCaseChange
10769   {%
10770     \glsifattribute{#1}{headuc}{true}%
10771     {%
10772       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10773     }%
10774     {%
10775       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10776     }%
10777   }%
10778 }

```

`\Glsxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10779 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10780   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10781 }

```

`\Glsxtrheadfull` Command used to display full form in the page header.

```

10782 \newcommand*{\Glsxtrheadfull}[1]{%
10783   \protect\NoCaseChange
10784   {%
10785     \glsifattribute{#1}{headuc}{true}%
10786     {%

```

```

10787     \GLSxtrfull[noindex,hyper=false]{#1}[]%
10788 }%
10789 {%
10790     \glsxtrfull[noindex,hyper=false]{#1}[]%
10791 }%
10792 }%
10793 }

```

**glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents.

```

10794 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10795     \glsxtrfull[noindex,hyper=false]{#1}[]%
10796 }

```

**glsxtrheadfullpl** 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.

```

10797 \newcommand*{\glsxtrheadfullpl}[1]{%
10798     \protect\NoCaseChange
10799 {%
10800     \glsifattribute{#1}{headuc}{true}%
10801     {%
10802         \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10803     }%
10804     {%
10805         \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10806     }%
10807 }%
10808 }

```

**glsxtrtitlefullpl** Command to display plural full form of abbreviation in section title and table of contents.

```

10809 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
10810     \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10811 }

```

**\Glsxtrheadfull** Command used to display full form in the page header with the first letter converted to upper case.

```

10812 \newcommand*{\Glsxtrheadfull}[1]{%
10813     \protect\NoCaseChange
10814 {%
10815     \glsifattribute{#1}{headuc}{true}%
10816     {%
10817         \GLSxtrfull[noindex,hyper=false]{#1}[]%
10818     }%
10819     {%
10820         \Glsxtrfull[noindex,hyper=false]{#1}[]%
10821     }%
10822 }%
10823 }

```

`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10824 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10825   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10826 }
```

`\Glsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```
10827 \newcommand*{\Glsxtrheadfullpl}[1]{%
10828   \protect\NoCaseChange
10829   {%
10830     \glsifattribute{#1}{headuc}{true}%
10831     {%
10832       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10833     }%
10834     {%
10835       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10836     }%
10837   }%
10838 }
```

`\Glsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10839 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10840   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10841 }
```

`\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.

```
10842 \ifdef\texorpdfstring
10843 {
10844   \newcommand*{\glsfmtshort}[1]{%
10845     \texorpdfstring
10846       {\glsxtrtitleshort{#1}}%
10847       {\glsentryshort{#1}}%
10848   }
10849 }
10850 {
10851   \newcommand*{\glsfmtshort}[1]{%
10852     \glsxtrtitleshort{#1}}
10853 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
10854 \ifdef\texorpdfstring
10855 {
10856   \newcommand*{\glsfmtshortpl}[1]{%
```

```

10857 \texorpdfstring
10858   {\glstrtitleshortpl{#1}}}%
10859   {\glentryshortpl{#1}}}%
10860 }
10861 }
10862 {
10863 \newcommand*{\glfmtshortpl}[1]{%
10864   \glstrtitleshortpl{#1}}
10865 }

```

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).

```

10866 \ifdef\texorpdfstring
10867 {
10868   \newcommand*{\Glsfmtshort}[1]{%
10869     \texorpdfstring
10870       {\Glsxtrtitleshort{#1}}}%
10871     {\glentryshort{#1}}}%
10872   }
10873 }
10874 {
10875   \newcommand*{\Glsfmtshort}[1]{%
10876     \Glsxtrtitleshort{#1}}
10877 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10878 \ifdef\texorpdfstring
10879 {
10880   \newcommand*{\Glsfmtshortpl}[1]{%
10881     \texorpdfstring
10882       {\Glsxtrtitleshortpl{#1}}}%
10883     {\glentryshortpl{#1}}}%
10884   }
10885 }
10886 {
10887   \newcommand*{\Glsfmtshortpl}[1]{%
10888     \Glsxtrtitleshortpl{#1}}
10889 }

```

`\glfmtname` As above but for the name value.

```

10890 \ifdef\texorpdfstring
10891 {
10892   \newcommand*{\glfmtname}[1]{%
10893     \texorpdfstring
10894       {\glxtrtitlename{#1}}}%
10895     {\glentryname{#1}}}%
10896   }

```

```

10897 }
10898 {
10899   \newcommand*{\Glsfmtname}[1]{%
10900     \Glsxtrtitlename{#1}}
10901 }

```

`\Glsfmtname` First letter converted to upper case.

```

10902 \ifdef\teorpdfstring
10903 {
10904   \newcommand*{\Glsfmtname}[1]{%
10905     \teorpdfstring
10906     {\Glsxtrtitlename{#1}}%
10907     {\Glsentryname{#1}}%
10908   }
10909 }
10910 {
10911   \newcommand*{\Glsfmtname}[1]{%
10912     \Glsxtrtitlename{#1}}
10913 }

```

`\glsfmttext` As above but for the text value.

```

10914 \ifdef\teorpdfstring
10915 {
10916   \newcommand*{\glsfmttext}[1]{%
10917     \teorpdfstring
10918     {\Glsxtrtitletext{#1}}%
10919     {\Glsentrytext{#1}}%
10920   }
10921 }
10922 {
10923   \newcommand*{\glsfmttext}[1]{%
10924     \Glsxtrtitletext{#1}}
10925 }

```

`\Glsfmttext` First letter converted to upper case.

```

10926 \ifdef\teorpdfstring
10927 {
10928   \newcommand*{\Glsfmttext}[1]{%
10929     \teorpdfstring
10930     {\Glsxtrtitletext{#1}}%
10931     {\Glsentrytext{#1}}%
10932   }
10933 }
10934 {
10935   \newcommand*{\Glsfmttext}[1]{%
10936     \Glsxtrtitletext{#1}}
10937 }

```

`\glsfmtplural` As above but for the plural value.



```

10938 \ifdef\teorpdfstring
10939 {
10940   \newcommand*\glsfmtplural}[1]{%
10941     \teorpdfstring
10942     {\glsxrtrtitleplural{#1}}%
10943     {\glsentryplural{#1}}%
10944   }
10945 }
10946 {
10947   \newcommand*\glsfmtplural}[1]{%
10948     \glsxrtrtitleplural{#1}}
10949 }

```

`\Glsfmtplural` First letter converted to upper case.

```

10950 \ifdef\teorpdfstring
10951 {
10952   \newcommand*\Glsfmtplural}[1]{%
10953     \teorpdfstring
10954     {\Glsxrtrtitleplural{#1}}%
10955     {\glsentryplural{#1}}%
10956   }
10957 }
10958 {
10959   \newcommand*\Glsfmtplural}[1]{%
10960     \Glsxrtrtitleplural{#1}}
10961 }

```

`\glsfmtfirst` As above but for the first value.

```

10962 \ifdef\teorpdfstring
10963 {
10964   \newcommand*\glsfmtfirst}[1]{%
10965     \teorpdfstring
10966     {\glsxrtrtitlefirst{#1}}%
10967     {\glsentryfirst{#1}}%
10968   }
10969 }
10970 {
10971   \newcommand*\glsfmtfirst}[1]{%
10972     \glsxrtrtitlefirst{#1}}
10973 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

10974 \ifdef\teorpdfstring
10975 {
10976   \newcommand*\Glsfmtfirst}[1]{%
10977     \teorpdfstring
10978     {\Glsxrtrtitlefirst{#1}}%
10979     {\glsentryfirst{#1}}%
10980   }

```

```

10981 }
10982 {
10983   \newcommand*{\Glsfmtfirst}[1]{%
10984     \Glsxtrtitlefirst{#1}}
10985 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

10986 \ifdef\texorpdfstring
10987 {
10988   \newcommand*{\glsfmtfirstpl}[1]{%
10989     \texorpdfstring
10990     {\Glsxtrtitlefirstplural{#1}}%
10991     {\glsentryfirstplural{#1}}%
10992   }
10993 }
10994 {
10995   \newcommand*{\glsfmtfirstpl}[1]{%
10996     \Glsxtrtitlefirstplural{#1}}
10997 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

10998 \ifdef\texorpdfstring
10999 {
11000   \newcommand*{\Glsfmtfirstpl}[1]{%
11001     \texorpdfstring
11002     {\Glsxtrtitlefirstplural{#1}}%
11003     {\glsentryfirstplural{#1}}%
11004   }
11005 }
11006 {
11007   \newcommand*{\Glsfmtfirstpl}[1]{%
11008     \Glsxtrtitlefirstplural{#1}}
11009 }

```

`\glsfmtlong` As above but for the long value.

```

11010 \ifdef\texorpdfstring
11011 {
11012   \newcommand*{\glsfmtlong}[1]{%
11013     \texorpdfstring
11014     {\Glsxtrtitlelong{#1}}%
11015     {\glsentrylong{#1}}%
11016   }
11017 }
11018 {
11019   \newcommand*{\glsfmtlong}[1]{%
11020     \Glsxtrtitlelong{#1}}
11021 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11022 \ifdef\teorpdfstring
11023 {
11024   \newcommand*\Glsfmtlong}[1]{%
11025     \teorpdfstring
11026     {\Glsxtrtitlelong{#1}}%
11027     {\glsentrylong{#1}}%
11028   }
11029 }
11030 {
11031   \newcommand*\Glsfmtlong}[1]{%
11032     \Glsxtrtitlelong{#1}}
11033 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11034 \ifdef\teorpdfstring
11035 {
11036   \newcommand*\glsfmtlongpl}[1]{%
11037     \teorpdfstring
11038     {\glsxtrtitlelongpl{#1}}%
11039     {\glsentrylongpl{#1}}%
11040   }
11041 }
11042 {
11043   \newcommand*\glsfmtlongpl}[1]{%
11044     \glsxtrtitlelongpl{#1}}
11045 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11046 \ifdef\teorpdfstring
11047 {
11048   \newcommand*\Glsfmtlongpl}[1]{%
11049     \teorpdfstring
11050     {\Glsxtrtitlelongpl{#1}}%
11051     {\glsentrylongpl{#1}}%
11052   }
11053 }
11054 {
11055   \newcommand*\Glsfmtlongpl}[1]{%
11056     \Glsxtrtitlelongpl{#1}}
11057 }

```

`\glsfmtfull` In-line full format.

```

11058 \ifdef\teorpdfstring
11059 {
11060   \newcommand*\glsfmtfull}[1]{%
11061     \teorpdfstring
11062     {\glsxtrtitlefull{#1}}%
11063     {\glsxtrinlinefullformat{#1}{}}%
11064   }

```

```

11065 }
11066 {
11067   \newcommand*{\glsfmtfull}[1]{%
11068     \glsxtrtitlefull{#1}}
11069 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11070 \ifdef\texorpdfstring
11071 {
11072   \newcommand*{\Glsfmtfull}[1]{%
11073     \texorpdfstring
11074       {\Glsxtrtitlefull{#1}}%
11075       {\Glsxtrinlinefullformat{#1}{}}%
11076   }
11077 }
11078 {
11079   \newcommand*{\Glsfmtfull}[1]{%
11080     \Glsxtrtitlefull{#1}}
11081 }

```

`\glsfmtfullpl` In-line full plural format.

```

11082 \ifdef\texorpdfstring
11083 {
11084   \newcommand*{\glsfmtfullpl}[1]{%
11085     \texorpdfstring
11086       {\glsxtrtitlefullpl{#1}}%
11087       {\glsxtrinlinefullplformat{#1}{}}%
11088   }
11089 }
11090 {
11091   \newcommand*{\glsfmtfullpl}[1]{%
11092     \glsxtrtitlefullpl{#1}}
11093 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11094 \ifdef\texorpdfstring
11095 {
11096   \newcommand*{\Glsfmtfullpl}[1]{%
11097     \texorpdfstring
11098       {\Glsxtrtitlefullpl{#1}}%
11099       {\Glsxtrinlinefullplformat{#1}{}}%
11100   }
11101 }
11102 {
11103   \newcommand*{\Glsfmtfullpl}[1]{%
11104     \Glsxtrtitlefullpl{#1}}
11105 }

```

## 1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```
11106 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11107   \@ifundefined{ver@glossariesxtr-#1.1df}{\input{glossariesxtr-#1.1df}}{}%
11108 }
```

sariesExtraLang

```
11109 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11110   \ProvidesFile{glossariesxtr-#1.1df}%
11111 }
```

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.)

```
11112 \@ifpackageloaded{tracklang}
11113 {%
11114   \AnyTrackedLanguages
11115   {%
11116     \ForEachTrackedDialect{\this@dialect}{%
11117       \IfTrackedLanguageFileExists{\this@dialect}%
11118       {glossariesxtr-}% prefix
11119       {.1df}%
11120       {%
11121         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11122       }%
11123     }%
11124   }%
11125 }%
11126 }%
11127 {}%
11128 }
11129 {}
```

Load `glossaries-extra-stylemods` if required.

```
11130 \@glsxtr@redefstyles
```

and set the style:

```
11131 \@glsxtr@do@style
```

## 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:

```
11132 \NeedsTeXFormat{LaTeX2e}
11133 \ProvidesPackage{glossaries-extra-stylemods}[2017/11/14 v1.24 (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`

```
11134 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
11135 \DeclareOption{all}{%
11136   \appto\@glxtr@loadstyles{%
11137     \RequirePackage{glossary-inline}%
11138     \RequirePackage{glossary-list}%
11139     \RequirePackage{glossary-tree}%
11140     \RequirePackage{glossary-mcols}%
11141     \RequirePackage{glossary-long}%
11142     \RequirePackage{glossary-longragged}%
11143     \RequirePackage{glossary-longbooktabs}%
11144     \RequirePackage{glossary-super}%
11145     \RequirePackage{glossary-superragged}%
11146     \RequirePackage{glossary-bookindex}%
11147   }
11148 }

11149 \DeclareOption*{%
11150   \IfFileExists{glossary-\CurrentOption.sty}
11151   {\eappto\@glxtr@loadstyles{%
11152     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
11153   }%
11154   {%
11155     \PackageError{glossaries-extra-styles}%

```

```

11156     {Unknown option ‘\CurrentOption’}{}%
11157   }%
11158 }

```

Process the package options:

```
11159 \ProcessOptions
```

Load the required packages:

```
11160 \@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`.

```
11161 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

11162 \providecommand{\renewglossarystyle}[2]{%
11163   \ifcsundef{@glsstyle@#1}%
11164   {%
11165     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
11166   }%
11167   {%
11168     \csdef{@glsstyle@#1}{#2}%
11169   }%
11170 }

```

## 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.

```

11171 \ifdef{@glsstyle@listdotted}
11172 {%
11173   \renewglossarystyle{listdotted}{%
11174     \setglossarystyle{list}%
11175     \renewcommand*{\glossentry}[2]{%
11176       \item[]\makebox[\glslistdottedwidth][l]{%
11177         \glsentryitem{##1}%
11178         \glstarget{##1}{\glossentryname{##1}}%
11179         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11180         \glossentrydesc{##1}\glspostdescription}%
11181     \renewcommand*{\subglossentry}[3]{%
11182       \item[]\makebox[\glslistdottedwidth][l]{%
11183         \glssubentryitem{##2}%
11184         \glstarget{##2}{\glossentryname{##2}}%
11185         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11186         \glossentrydesc{##2}\glspostdescription}%
11187   }

```

```
11188 }
11189 {%
```

Assume the style isn't required if it hasn't already been defined.

```
11190 }
```

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.

```
11191 \ifdef{\@glsstyle@list}
11192 {%
```

listprelocation Space before number list for top-level entries.

```
11193 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11194 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11195 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11196 \renewglossarystyle{list}{%
11197 \renewenvironment{theglossary}%
11198 {\begin{description}}{\end{description}}%
11199 \renewcommand*\glossaryheader{}%
11200 \renewcommand*\glsgroupheading}[1]{%
11201 \renewcommand*\glossentry}[2]{%
11202 \item[\glsentryitem{##1}%
11203 \glstarget{##1}{\glossentryname{##1}}]
11204 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11205 \renewcommand*\subglossentry}[3]{%
11206 \glssubentryitem{##2}%
11207 \glstarget{##2}{\strut}\space
11208 \glossentrydesc{##2}\glspostdescription
11209 \glslistchildprelocation ##3\glslistchildpostlocation}%
11210 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else\indexspace\fi}%
11211 }
11212 }
11213 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11214 \ifdef{\@glsstyle@altlist}
11215 {%
11216 \renewglossarystyle{altlist}{%
11217 \setglossarystyle{list}%
11218 \renewcommand*\glossentry}[2]{%
11219 \item[\glsentryitem{##1}%
```



```

11220      \glstarget{##1}{\glossentryname{##1}}}%
11221      \mbox{}\par\nobreak\@afterheading
11222      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11223      \renewcommand{\subglossentry}[3]{%
11224        \par
11225        \glssubentryitem{##2}%
11226        \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11227        \glslistchildprelocation ##3}%
11228    }
11229 }
11230 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11231 \ifdef{\@glsstyle@listgroup}
11232 {%
11233   \renewglossarystyle{listgroup}{%
11234     \setglossarystyle{list}%
11235     \renewcommand*\glsgroupheading[1]{%
11236       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11237       \mbox{}\par\nobreak\@afterheading
11238     }%
11239   }
11240 }
11241 {}

```

Similarly for listhypergroup.

```

11242 \ifdef{\@glsstyle@listhypergroup}
11243 {%
11244   \renewglossarystyle{listhypergroup}{%
11245     \setglossarystyle{list}%
11246     \renewcommand*\glossaryheader{%
11247       \glslistnavigationitem{\glsnavigation}}%
11248     \renewcommand*\glsgroupheading[1]{%
11249       \item[\glslistgroupheaderfmt
11250         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11251       \mbox{}\par\nobreak\@afterheading
11252     }%
11253   }
11254 }
11255 {}

```

Similarly for altlistgroup.

```

11256 \ifdef{\@glsstyle@altlistgroup}
11257 {%
11258   \renewglossarystyle{altlistgroup}{%
11259     \setglossarystyle{altlist}%
11260     \renewcommand*\glsgroupheading[1]{%
11261       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11262       \mbox{}\par\nobreak\@afterheading
11263     }%
11264   }

```

```

11265 }
11266 {}

    Similarly for altlisthypergroup.
11267 \ifdef{\@glsstyle@altlisthypergroup}
11268 {%
11269   \renewglossarystyle{altlisthypergroup}{%
11270     \setglossarystyle{altlist}%
11271     \renewcommand*\{glossaryheader\}{%
11272       \glslistnavigationitem{\glsnavigation}}%
11273     \renewcommand*\{glsgroupheading\}[1]{%
11274       \item[\glslistgroupheaderfmt
11275         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11276       \mbox{}\par\nobreak\@afterheading
11277     }%
11278   }
11279 }
11280 {}

```

## 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`.

```

11281 \ifcsdef{@glsstyle@long}
11282 {%
11283   \renewglossarystyle{long}{%
11284     \renewenvironment{theglossary}%
11285       {\begin{longtable}{lp{\glsdescwidth}}}%
11286       {\end{longtable}}%
11287     \renewcommand*\{glossaryheader\}{}%
11288     \renewcommand*\{glsgroupheading\}[1]{}%
11289     \renewcommand{\glossentry}[2]{%
11290       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11291       \glossentrydesc{##1}\glspostdescription
11292       \glstrprelocation ##2\tabularnewline
11293     }%
11294     \renewcommand{\subglossentry}[3]{%
11295       &
11296       \glssubentryitem{##2}%
11297       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11298       \glstrprelocation ##3\tabularnewline
11299     }%
11300     \ifglsnogroupskip
11301       \renewcommand*\{glsgroupskip\}{}%
11302     \else
11303       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11304     \fi
11305   }

```

```
11306 }
11307 {}
```

Three column style:

```
11308 \ifcsdef{@glsstyle@long3col}
11309 {%
11310   \renewglossarystyle{long3col}{%
11311     \renewenvironment{theglossary}%
11312       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
11313       {\end{longtable}}}%
11314     \renewcommand*{\glossaryheader}{}%
11315     \renewcommand*{\glsgroupheading}[1]{}%
11316     \renewcommand{\glossentry}[2]{%
11317       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11318       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11319     }%
11320     \renewcommand{\subglossentry}[3]{%
11321       &
11322       \glssubentryitem{##2}%
11323       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11324       ##3\tabularnewline
11325     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11326   \ifglsnogroupskip
11327     \renewcommand*{\glsgroupskip}{}%
11328   \else
11329     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11330   \fi
11331 }
11332 }
11333 {}
```

Four column style:

```
11334 \ifcsdef{@glsstyle@long4col}
11335 {%
11336   \renewglossarystyle{long4col}{%
11337     \renewenvironment{theglossary}%
11338       {\begin{longtable}{llll}}}%
11339       {\end{longtable}}}%
11340     \renewcommand*{\glossaryheader}{}%
11341     \renewcommand*{\glsgroupheading}[1]{}%
11342     \renewcommand{\glossentry}[2]{%
11343       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11344       \glossentrydesc{##1}\glspostdescription &
11345       \glossentrysymbol{##1} &
11346       ##2\tabularnewline
11347     }%
11348     \renewcommand{\subglossentry}[3]{%
11349       &
```

```

11350     \glssubentryitem{##2}%
11351     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11352     \glossentrysymbol{##2} & ##3\tabularnewline
11353 }%

11354 \ifglsnogroupskip
11355     \renewcommand*{\glsgroupskip}{}%
11356 \else
11357     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11358 \fi
11359 }
11360 }
11361 {}

```

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`.

```

11362 \ifcsdef{@glstyle@longragged}
11363 {%
11364     \renewglossarystyle{longragged}{%
11365         \renewenvironment{theglossary}%
11366             {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
11367             {\end{longtable}}%
11368         \renewcommand*{\glossaryheader}{}%
11369         \renewcommand*{\glsgroupheading}[1]{}%
11370         \renewcommand{\glossentry}[2]{%
11371             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11372             \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11373             \tabularnewline
11374         }%
11375         \renewcommand{\subglossentry}[3]{%
11376             &
11377             \glssubentryitem{##2}%
11378             \glstarget{##2}{\strut}\glossentrydesc{##2}%
11379             \glspostdescription\glxtrprelocation ##3%
11380             \tabularnewline
11381         }%
11382         \ifglsnogroupskip
11383             \renewcommand*{\glsgroupskip}{}%
11384         \else
11385             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11386         \fi
11387     }
11388 }

```

11389 {}

Three and four column styles don't use \glxstrprelocation since the number list is in its own column.

```
11390 \ifcsdef{@glsstyle@longragged3col}
11391 {%
11392   \renewglossarystyle{longragged3col}{%
11393     \renewenvironment{theglossary}%
11394       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11395         >{\raggedright}p{\glspagelistwidth}}}%
11396       {\end{longtable}}}%
11397     \renewcommand*{\glossaryheader}{}%
11398     \renewcommand*{\glsgroupheading}[1]{}%
11399     \renewcommand{\glossentry}[2]{%
11400       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11401       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11402     }%
11403     \renewcommand{\subglossentry}[3]{%
11404       &
11405       \glssubentryitem{##2}%
11406       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11407       ##3\tabularnewline
11408     }%
11409     \ifglsgroupskip
11410       \renewcommand*{\glsgroupskip}{}%
11411     \else
11412       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11413     \fi
11414   }
11415 }
11416 {}
```

Four column style:

```
11417 \ifcsdef{@glsstyle@altlongragged4col}
11418 {%
11419   \renewglossarystyle{altlongragged4col}{%
11420     \renewenvironment{theglossary}%
11421       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11422         >{\raggedright}p{\glspagelistwidth}}}%
11423       {\end{longtable}}}%
11424     \renewcommand*{\glossaryheader}{}%
11425     \renewcommand*{\glsgroupheading}[1]{}%
11426     \renewcommand{\glossentry}[2]{%
11427       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11428       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11429       ##2\tabularnewline
11430     }%
11431     \renewcommand{\subglossentry}[3]{%
11432       &
```

```

11433     \glssubentryitem{##2}%
11434     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11435     \glossentrysymbol{##2} & ##3\tabularnewline
11436 }%

11437 \ifglsnogroupskip
11438     \renewcommand*{\glsgroupskip}{}%
11439 \else
11440     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11441 \fi
11442 }
11443 }
11444 {}

```

## 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 `\glstrprelocation`.

```

11445 \ifcsdef{@glstyle@super}
11446 {%
11447     \renewglossarystyle{super}{%
11448         \renewenvironment{theglossary}%
11449             {\tablehead{}\tabletail}%
11450             \begin{supertabular}{lp{\glsdescwidth}}%
11451             {\end{supertabular}}%
11452         \renewcommand*{\glossaryheader}{}%
11453         \renewcommand*{\glsgroupheading}[1]{}%
11454         \renewcommand{\glossentry}[2]{%
11455             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11456             \glossentrydesc{##1}\glspostdescription
11457             \glstrprelocation ##2\tabularnewline
11458         }%
11459         \renewcommand{\subglossentry}[3]{%
11460             &
11461             \glssubentryitem{##2}%
11462             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11463             \glstrprelocation ##3\tabularnewline
11464         }%
11465         \ifglsnogroupskip
11466             \renewcommand*{\glsgroupskip}{}%
11467         \else
11468             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11469         \fi
11470     }
11471 }
11472 {}

```

Three column style:

```

11473 \ifcsdef{@glstyle@super3col}

```

```

11474 {%
11475   \renewglossarystyle{super3col}{%
11476     \renewenvironment{theglossary}%
11477       {\tablehead{}\tabletail{}}%
11478       \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
11479       {\end{supertabular}}%
11480   \renewcommand*{\glossaryheader}{}%
11481   \renewcommand*{\glsgroupheading}[1]{}%
11482   \renewcommand{\glossentry}[2]{%
11483     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11484     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11485   }%
11486   \renewcommand{\subglossentry}[3]{%
11487     &
11488     \glssubentryitem{##2}%
11489     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11490     ##3\tabularnewline
11491   }%

11492   \ifglsnogroupskip
11493     \renewcommand*{\glsgroupskip}{}%
11494   \else
11495     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11496   \fi
11497 }
11498 }
11499 {}

```

Four column styles:

```

11500 \ifcsdef{@glstyle@super4col}
11501 {%
11502   \renewglossarystyle{super4col}{%
11503     \renewenvironment{theglossary}%
11504       {\tablehead{}\tabletail{}}%
11505       \begin{supertabular}{llll}}{%
11506       \end{supertabular}}%
11507   \renewcommand*{\glossaryheader}{}%
11508   \renewcommand*{\glsgroupheading}[1]{}%
11509   \renewcommand{\glossentry}[2]{%
11510     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11511     \glossentrydesc{##1}\glspostdescription &
11512     \glossentrysymbol{##1} & ##2\tabularnewline
11513   }%
11514   \renewcommand{\subglossentry}[3]{%
11515     &
11516     \glssubentryitem{##2}%
11517     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11518     \glossentrysymbol{##2} & ##3\tabularnewline
11519   }%

```

```

11520 \ifglsgroupskip
11521 \renewcommand*{\glsgroupskip}{}%
11522 \else
11523 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11524 \fi
11525 }
11526 }
11527 {}

```

## 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 `\glstrprelocation`.

```

11528 \ifcsdef{@glstyle@superragged}
11529 {%
11530 \renewglossarystyle{superragged}{%
11531 \renewenvironment{theglossary}%
11532 {\tablehead{}}\tabletail{}}%
11533 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11534 {\end{supertabular}}%
11535 \renewcommand*{\glossaryheader}{}%
11536 \renewcommand*{\glsgroupheading}[1]{}%
11537 \renewcommand{\glossentry}[2]{%
11538 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11539 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11540 \tabularnewline
11541 }%
11542 \renewcommand{\subglossentry}[3]{%
11543 &
11544 \glssubentryitem{##2}%
11545 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11546 \glstrprelocation ##3%
11547 \tabularnewline
11548 }%
11549 \ifglsgroupskip
11550 \renewcommand*{\glsgroupskip}{}%
11551 \else
11552 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11553 \fi
11554 }
11555 }
11556 {}

```

Three column style:

```

11557 \ifcsdef{@glstyle@superragged3col}
11558 {%
11559 \renewglossarystyle{superragged3col}{%

```



```

11560 \renewenvironment{theglossary}%
11561 {\tablehead{}\tabletail{}}%
11562 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
11563 >{\raggedright}p{\glspagelistwidth}}}%
11564 {\end{supertabular}}%
11565 \renewcommand*{\glossaryheader}{}%
11566 \renewcommand*{\glsgroupheading}[1]{}%
11567 \renewcommand{\glossentry}[2]{%
11568 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11569 \glossentrydesc{##1}\glspostdescription &
11570 ##2\tabularnewline
11571 }%
11572 \renewcommand{\subglossentry}[3]{%
11573 &
11574 \glssubentryitem{##2}%
11575 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11576 ##3\tabularnewline
11577 }%

11578 \ifglsnogroupskip
11579 \renewcommand*{\glsgroupskip}{}%
11580 \else
11581 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
11582 \fi
11583 }
11584 }
11585 {}

```

Four columns:

```

11586 \ifcsdef{@glsstyle@altsuperragged4col}
11587 {%
11588 \renewglossarystyle{altsuperragged4col}{%
11589 \renewenvironment{theglossary}%
11590 {\tablehead{}\tabletail{}}%
11591 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
11592 >{\raggedright}p{\glspagelistwidth}}}%
11593 {\end{supertabular}}%
11594 \renewcommand*{\glossaryheader}{}%
11595 \renewcommand{\glossentry}[2]{%
11596 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11597 \glossentrydesc{##1}\glspostdescription &
11598 \glossentrysymbol{##1} & ##2\tabularnewline
11599 }%
11600 \renewcommand{\subglossentry}[3]{%
11601 &
11602 \glssubentryitem{##2}%
11603 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11604 \glossentrysymbol{##2} & ##3\tabularnewline
11605 }%

```

```

11606 \ifglsnogroupskip
11607 \renewcommand*{\glsgroupskip}{}%
11608 \else
11609 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11610 \fi
11611 }
11612 }
11613 {}

```

## 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`.

```

11614 \ifdef{\@glsstyle@inline}
11615 {%
11616 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11617 \renewcommand*{\glsinlinedescformat}[3]{%
11618 \space#1\glxtrpostdescription}
11619 \renewcommand*{\glsinlinesubdescformat}[3]{%
11620 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11621 }
11622 {}

```

## 2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11623 \ifdef{\@glsstyle@index}
11624 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11625 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11626 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

```

```

11627 \renewglossarystyle{index}{%
11628 \renewenvironment{theglossary}%
11629 {\setlength{\parindent}{0pt}%
11630 \setlength{\parskip}{0pt plus 0.3pt}%
11631 \let\item\glstreeitem
11632 \let\subitem\glstreesubitem

```

```

11633     \let\subsubitem\glstreesubsubitem
11634 }%
11635 {\par}%
11636 \renewcommand*{\glossaryheader}{}%
11637 \renewcommand*{\glsgroupheading}[1]{}%
11638 \renewcommand*{\glossentry}[2]{%
11639     \item\glstentryitem{##1}%
11640     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11641     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11642     \glstreepredesc \glossentrydesc{##1}\glspostdescription
11643     \glstreeprelocation ##2%
11644 }%
11645 \renewcommand{\subglossentry}[3]{%
11646     \ifcase##1\relax
11647         \item
11648     \or
11649         \subitem
11650         \glssubentryitem{##2}%
11651     \else
11652         \subsubitem
11653     \fi
11654     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11655     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11656     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11657     \glstreechildprelocation ##3%
11658 }%
11659 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11660 }
11661 }
11662 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11663 \ifdef{\@glsstyle@indexgroup}
11664 {%
11665     \renewglossarystyle{indexgroup}{%
11666         \setglossarystyle{index}%
11667         \renewcommand*{\glsgroupheading}[1]{%
11668             \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}%
11669             \nopagebreak\indexspace
11670             \nobreak\@afterheading
11671         }%
11672     }
11673 }
11674 {}

```

Similarly for indexhypergroup.

```

11675 \ifdef{\@glsstyle@indexhypergroup}
11676 {%
11677     \renewglossarystyle{indexhypergroup}{%
11678         \setglossarystyle{index}%

```

```

11679 \renewcommand*{\glossaryheader}{%
11680 \item\glstreenavigationfmt{\glsnavigation}%
11681 \nobreak\@afterheading\indexspace}%
11682 \renewcommand*{\glsgroupheading}[1]{%
11683 \item\glstreegroupheaderfmt
11684 {\glssnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
11685 \nopagebreak\indexspace
11686 \nobreak\@afterheading}%
11687 }%
11688 }
11689 {}

```

Adjust tree style to remove hard coded space before number list.

```

11690 \ifdef{\@glsstyle@tree}
11691 {%
11692 \renewglossarystyle{tree}{%
11693 \renewenvironment{theglossary}%
11694 {\setlength{\parindent}{0pt}%
11695 \setlength{\parskip}{0pt plus 0.3pt}}%
11696 {}%
11697 \renewcommand*{\glossaryheader}{}%
11698 \renewcommand*{\glsgroupheading}[1]{}%
11699 \renewcommand{\glossentry}[2]{%
11700 \hangindent0pt\relax
11701 \parindent0pt\relax
11702 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11703 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11704 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11705 \glstreeprelocation##2\par
11706 }%
11707 \renewcommand{\subglossentry}[3]{%
11708 \hangindent##1\glstreeindent\relax
11709 \parindent##1\glstreeindent\relax
11710 \ifnum##1=1\relax
11711 \glssubentryitem{##2}%
11712 \fi
11713 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11714 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11715 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11716 \glstreechildprelocation ##3\par
11717 }%
11718 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11719 }%
11720 }
11721 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11722 \ifdef{\@glsstyle@treegroup}
11723 {%
11724 \renewglossarystyle{treegroup}{%

```

```

11725 \setglossarystyle{tree}%
11726 \renewcommand{\glsgroupheading}[1]{\par
11727 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11728 \nopagebreak\indexspace\nobreak\@afterheading}%
11729 }
11730 }
11731 {}

```

Similarly for treehypergroup

```

11732 \ifdef{\@glsstyle@treehypergroup}
11733 {%
11734 \renewglossarystyle{treehypergroup}{%
11735 \setglossarystyle{tree}%
11736 \renewcommand*\glossaryheader{%
11737 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11738 \nobreak\@afterheading\indexspace}%
11739 \renewcommand*\glsgroupheading}[1]{%
11740 \par\noindent
11741 \glstreegroupheaderfmt
11742 {\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
11743 \nopagebreak\indexspace\nobreak\@afterheading}%
11744 }
11745 }
11746 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11747 \ifdef{\@glsstyle@treenoname}
11748 {%
11749 \renewglossarystyle{treenoname}{%
11750 \renewenvironment{theglossary}%
11751 {\setlength{\parindent}{0pt}%
11752 \setlength{\parskip}{0pt plus 0.3pt}}%
11753 {}%
11754 \renewcommand*\glossaryheader{}%
11755 \renewcommand*\glsgroupheading}[1]{}%
11756 \renewcommand{\glossentry}[2]{%
11757 \hangindent0pt\relax
11758 \parindent0pt\relax
11759 \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11760 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11761 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11762 \glstreeprelocation##2\par
11763 }%
11764 \renewcommand{\subglossentry}[3]{%
11765 \hangindent##1\glstreeindent\relax
11766 \parindent##1\glstreeindent\relax
11767 \ifnum##1=1\relax
11768 \glssubentryitem{##2}%
11769 \fi
11770 \glstarget{##2}{\strut}%

```

```

11771     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11772 }%
11773 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11774 }
11775 }
11776 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11777 \ifdef{\@glsstyle@treenonamegroup}
11778 {%
11779   \renewglossarystyle{treenonamegroup}{%
11780     \setglossarystyle{treenoname}%
11781     \renewcommand{\glsgroupheading}[1]{\par
11782       \noindent\glstreegroupheaderfmt
11783       {\glsggetgrouptitle{##1}}}%
11784     \nopagebreak\indexspace\nobreak\@afterheading
11785   }%
11786 }
11787 }
11788 {}

```

Similarly for `treenonamehypergroup`

```

11789 \ifdef{\@glsstyle@treenonamehypergroup}
11790 {%
11791   \renewglossarystyle{treenonamehypergroup}{%
11792     \setglossarystyle{treenoname}%
11793     \renewcommand*{\glossaryheader}{%
11794       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11795       \nobreak\@afterheading\indexspace}%
11796     \renewcommand*{\glsgroupheading}[1]{%
11797       \par\noindent
11798       \glstreegroupheaderfmt
11799       {\glsnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
11800     \nopagebreak\indexspace\nobreak\@afterheading}%
11801   }
11802 }
11803 {}

```

The `almtree` style is redefined to make it easier to made minor adjustments.

```

11804 \ifdef{\@glsstyle@almtree}
11805 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtralmtreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

11806 \newcommand{\glxtralmtreeSymbolDescLocation}[2]{%

```

```

11807   {%
11808       \let\par\glxstrAltTreePar
11809       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11810       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11811   }%
11812 }

```

**trAltTreeIndent** Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11813 \newlength\glxstrAltTreeIndent

```

**lsxtrAltTreePar** Multi-paragraph descriptions need to keep the hanging indent.

```

11814 \newcommand{\glxstrAltTreePar}{%
11815     @@par
11816     \glxstrAltTreeSetHangIndent
11817     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11818 }

```

**symbolDescLocation** `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11819 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11820     \glxstralttreeSymbolDescLocation{#2}{#3}%
11821 }

```

**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.

```

11822 \newlength\glxstrtreetopindent

```

**sxtralttreeInit** User-level initialisation for the alttree style.

```

11823 \newcommand*{\glxstralttreeInit}{%
11824     \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
11825     \glxstrAltTreeIndent=\parindent
11826 }

```

**\glsglsetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\gdef`.

```

11827 \newcommand*{\glsglsetwidest}[2][0]{%
11828     \csgdef{@glswidestname\romannumeral#1}{#2}%
11829 }

```

**\eglssetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\protected@csdef`.

```

11830 \newcommand*{\eglssetwidest}[2][0]{%
11831     \protected@csdef{@glswidestname\romannumeral#1}{#2}%
11832 }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```
11833 \newcommand*\xglissetwidest}[2][0]{%
11834   \protected@csxdef{#1}{#2}%
11835 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
11836 \newcommand*\glsupdatewidest}[2][0]{%
11837   \ifcsundef{#1}%
11838     {\csdef{#1}{#2}}%
11839     {%
11840       \settowidth{#1}{\csuse{#1}}%
11841       \settowidth{#2}%
11842       \ifdim#1>#2%
11843         \csdef{#1}{#2}%
11844       \fi
11845     }%
11846 }
```

`glsupdatewidest` As above but global definition.

```
11847 \newcommand*\gglsupdatewidest}[2][0]{%
11848   \ifcsundef{#1}%
11849     {\csgdef{#1}{#2}}%
11850     {%
11851       \settowidth{#1}{\csuse{#1}}%
11852       \settowidth{#2}%
11853       \ifdim#1>#2%
11854         \csgdef{#1}{#2}%
11855       \fi
11856     }%
11857 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
11858 \newcommand*\eglsupdatewidest}[2][0]{%
11859   \ifcsundef{#1}%
11860     {\protected@csedef{#1}{#2}}%
11861     {%
11862       \settowidth{#1}{\csuse{#1}}%
11863       \settowidth{#2}%
11864       \ifdim#1>#2%
11865         \protected@csedef{#1}{#2}%
11866       \fi
11867     }%
11868 }
```

`glsupdatewidest` As above but global.

```
11869 \newcommand*\xglsupdatewidest}[2][0]{%
11870   \ifcsundef{#1}%
11871     {\protected@csxdef{#1}{#2}}%
11872     {%
```



```

11873      \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
11874      \settowidth{\dimen@ii}{#2}%
11875      \ifdim\dimen@ii>\dimen@
11876      \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
11877      \fi
11878  }%
11879  }

```

**glswidestname** Provide a user-level macro to obtain the widest top-level name.

```

11880  \newcommand*{\glswidestname}{\@glswidestname}

```

**glswidestsubname** Provide a user-level macro to obtain the widest sub-entry name.

```

11881  \newcommand*{\glswidestsubname}[1]{%
11882    \ifcsundef{@glswidestname\romannumeral#1}%
11883    {\@glswidestname}%
11884    {\csuse{@glswidestname\romannumeral#1}}%
11885  }

```

**glswidestTopLevelName** CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

11886  \let\glswidestTopLevelName\glswidestTopLevelName

```

**glswidestUsedTopLevelName** Like `\glswidestTopLevelName` 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.

```

11887  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
11888    \dimen@=0pt\relax
11889    \gls@tmplen=0pt\relax
11890    \foralllglossaries[#1]{\@gls@type}%
11891    {%
11892      \forglsentries[\@gls@type]{\@glo@label}%
11893      {%
11894        \ifglswidest{\@glo@label}%
11895        {%
11896          \ifglshasparent{\@glo@label}%
11897          {}%
11898          {%
11899            \settowidth{\dimen@}{%
11900              {\glstreenamfmt{\glstryname{\@glo@label}}}%
11901              \ifdim\dimen@>\gls@tmplen
11902              \gls@tmplen=\dimen@
11903              \eglssetwidest{\glstryname{\@glo@label}}%
11904            }%
11905          }%
11906        }%
11907      }%
11908    }%
11909  }%
11910  }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11911 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11912 \dimen@=0pt\relax
11913 \gls@tmplen=0pt\relax
11914 \foralllglossaries[#1]{\@gls@type}%
11915 {%
11916 \forallglsentries[\@gls@type]{\@glo@label}%
11917 {%
11918 \ifglsused{\@glo@label}%
11919 {%
11920 \settowidth{\dimen@}%
11921 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11922 \ifdim\dimen@>\gls@tmplen
11923 \gls@tmplen=\dimen@
11924 \eglssetwidest{\glsentryname{\@glo@label}}%
11925 \fi
11926 }%
11927 }%
11928 }%
11929 }%
11930 }

```

`ndWidestAnyName` Like the above but doesn't check is the entry has been used.

```

11931 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
11932 \dimen@=0pt\relax
11933 \gls@tmplen=0pt\relax
11934 \foralllglossaries[#1]{\@gls@type}%
11935 {%
11936 \forallglsentries[\@gls@type]{\@glo@label}%
11937 {%
11938 \settowidth{\dimen@}%
11939 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11940 \ifdim\dimen@>\gls@tmplen
11941 \gls@tmplen=\dimen@
11942 \eglssetwidest{\glsentryname{\@glo@label}}%
11943 \fi
11944 }%
11945 }%
11946 }

```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

11947 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
11948 \dimen@=0pt\relax
11949 \dimen@i=0pt\relax
11950 \dimen@ii=0pt\relax
11951 \foralllglossaries[#1]{\@gls@type}%
11952 {%

```

```

11953 \forglsentries[\@gls@type]{\@glo@label}%
11954 {%
11955 \ifglsused{\@glo@label}%
11956 {%
11957 \ifglsashasparent{\@glo@label}%
11958 {%
11959 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
11960 \ifglsashasparent{\@glo@parent}%
11961 {%
11962 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
11963 \ifglsashasparent{\@glo@parent}%
11964 {}%
11965 {%
11966 \settowidth{\gls@tmplen}%
11967 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11968 \ifdim\gls@tmplen>\dimen@ii
11969 \dimen@ii=\gls@tmplen
11970 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
11971 \fi
11972 }%
11973 }%
11974 {%
11975 \settowidth{\gls@tmplen}%
11976 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11977 \ifdim\gls@tmplen>\dimen@i
11978 \dimen@i=\gls@tmplen
11979 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
11980 \fi
11981 }%
11982 }%
11983 {%
11984 \settowidth{\gls@tmplen}%
11985 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11986 \ifdim\gls@tmplen>\dimen@
11987 \dimen@=\gls@tmplen
11988 \eglssetwidest{\glsentryname{\@glo@label}}%
11989 \fi
11990 }%
11991 }%
11992 {}%
11993 }%
11994 }%
11995 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

11996 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
11997 \dimen@=0pt\relax
11998 \dimen@i=0pt\relax
11999 \dimen@ii=0pt\relax

```

```

12000 \forall glossaries[#1]{\@gls@type}%
12001 {%
12002   \forall sentries[\@gls@type]{\@glo@label}%
12003   {%
12004     \ifglshasparent{\@glo@label}%
12005     {%
12006       \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@label}@parent}}%
12007       \ifglshasparent{\@glo@parent}%
12008       {%
12009         \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@parent}@parent}}%
12010         \ifglshasparent{\@glo@parent}%
12011         {}%
12012         {%
12013           \settowidth{\gls@tmplen}%
12014             {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12015           \ifdim\gls@tmplen>\dimen@ii
12016             \dimen@ii=\gls@tmplen
12017             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
12018           \fi
12019         }%
12020       }%
12021     {%
12022       \settowidth{\gls@tmplen}%
12023         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12024       \ifdim\gls@tmplen>\dimen@i
12025         \dimen@i=\gls@tmplen
12026         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
12027       \fi
12028     }%
12029   }%
12030   {%
12031     \settowidth{\gls@tmplen}%
12032       {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12033     \ifdim\gls@tmplen>\dimen@
12034       \dimen@=\gls@tmplen
12035       \eglssetwidest{\glstentryname{\@glo@label}}%
12036     \fi
12037   }%
12038 }%
12039 }%
12040 }

```

**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.

```

12041 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
12042   \dimen@=0pt\relax
12043   \gls@tmplen=0pt\relax
12044   #2=0pt\relax
12045   \forall glossaries[#1]{\@gls@type}%

```

```

12046  {%
12047    \forglsentries[\@gls@type]{\@glo@label}%
12048  {%
12049    \ifglsused{\@glo@label}%
12050  {%
12051    \settowidth{\dimen@}%
12052      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12053    \ifdim\dimen@>\gls@tmplen
12054      \gls@tmplen=\dimen@
12055      \eglssetwidest{\glsentryname{\@glo@label}}%
12056    \fi
12057    \settowidth{\dimen@}%
12058      {\glsentrysymbol{\@glo@label}}%
12059    \ifdim\dimen@>#2\relax
12060      #2=\dimen@
12061    \fi
12062  }%
12063  {%
12064  }%
12065  }%
12066  }

```

**stAnyNameSymbol** Like the above but doesn't check if the entry has been used.

```

12067  \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
12068    \dimen@=0pt\relax
12069    \gls@tmplen=0pt\relax
12070    #2=0pt\relax
12071    \forallglossaries[#1]{\@gls@type}%
12072  {%
12073    \forglsentries[\@gls@type]{\@glo@label}%
12074  {%
12075    \settowidth{\dimen@}%
12076      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12077    \ifdim\dimen@>\gls@tmplen
12078      \gls@tmplen=\dimen@
12079      \eglssetwidest{\glsentryname{\@glo@label}}%
12080    \fi
12081    \settowidth{\dimen@}%
12082      {\glsentrysymbol{\@glo@label}}%
12083    \ifdim\dimen@>#2\relax
12084      #2=\dimen@
12085    \fi
12086  }%
12087  }%
12088  }

```

**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.

```

12089 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
12090   \dimen@=0pt\relax
12091   \gls@tmplen=0pt\relax
12092   #2=0pt\relax
12093   #3=0pt\relax
12094   \foralllglossaries[#1]{\@gls@type}%
12095   {%
12096     \forglsentries[\@gls@type]{\@glo@label}%
12097     {%
12098       \ifglsused{\@glo@label}%
12099       {%
12100         \settowidth{\dimen@}%
12101         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12102         \ifdim\dimen@>\gls@tmplen
12103           \gls@tmplen=\dimen@
12104           \eglssetwidest{\glsentryname{\@glo@label}}%
12105         \fi
12106         \settowidth{\dimen@}%
12107         {\glsentrysymbol{\@glo@label}}%
12108         \ifdim\dimen@>#2\relax
12109           #2=\dimen@
12110         \fi
12111         \settowidth{\dimen@}%
12112         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12113         \ifdim\dimen@>#3\relax
12114           #3=\dimen@
12115         \fi
12116       }%
12117     }%
12118   }%
12119 }%
12120 }
```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

12121 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
12122   \dimen@=0pt\relax
12123   \gls@tmplen=0pt\relax
12124   #2=0pt\relax
12125   #3=0pt\relax
12126   \foralllglossaries[#1]{\@gls@type}%
12127   {%
12128     \forglsentries[\@gls@type]{\@glo@label}%
12129     {%
12130       \settowidth{\dimen@}%
12131       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12132       \ifdim\dimen@>\gls@tmplen
12133         \gls@tmplen=\dimen@
12134         \eglssetwidest{\glsentryname{\@glo@label}}%
```

```

12135      \fi
12136      \settowidth{\dimen@}%
12137      {\glstrysymbol{\@glo@label}}}%
12138      \ifdim\dimen@>#2\relax
12139      #2=\dimen@
12140      \fi
12141      \settowidth{\dimen@}%
12142      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12143      \ifdim\dimen@>#3\relax
12144      #3=\dimen@
12145      \fi
12146    }%
12147  }%
12148 }

```

**AnyNameLocation** Like the `\glstryFindWidestUsedAnyNameSymbolLocation` 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.

```

12149 \newrobustcmd*{\glstryFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
12150   \dimen@=0pt\relax
12151   \glstry@tmplen=0pt\relax
12152   #2=0pt\relax
12153   \forallglossaries[#1]{\@gls@type}%
12154   {%
12155     \forallglsentries[\@gls@type]{\@glo@label}%
12156     {%
12157       \ifglstryused{\@glo@label}%
12158       {%
12159         \settowidth{\dimen@}%
12160         {\glstrynamefmt{\glstryname{\@glo@label}}}%
12161         \ifdim\dimen@>\glstry@tmplen
12162         \glstry@tmplen=\dimen@
12163         \eglstrysetwidest{\glstryname{\@glo@label}}%
12164         \fi
12165         \settowidth{\dimen@}%
12166         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12167         \ifdim\dimen@>#2\relax
12168         #2=\dimen@
12169         \fi
12170       }%
12171     }%
12172   }%
12173 }%
12174 }

```

**AnyNameLocation** Like the `\glstryFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

12175 \newrobustcmd*{\glstryFindWidestAnyNameLocation}[2][\@glo@types]{%
12176   \dimen@=0pt\relax
12177   \glstry@tmplen=0pt\relax

```

```

12178 #2=Opt\relax
12179 \foralllglossaries[#1]{\@gls@type}%
12180 {%
12181   \forglsentries[\@gls@type]{\@glo@label}%
12182   {%
12183     \settowidth{\dimen@}%
12184     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12185     \ifdim\dimen@>\gls@tmplen
12186       \gls@tmplen=\dimen@
12187       \eglssetwidest{\glsentryname{\@glo@label}}%
12188     \fi
12189     \settowidth{\dimen@}%
12190     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12191     \ifdim\dimen@>#2\relax
12192       #2=\dimen@
12193     \fi
12194   }%
12195 }%
12196 }

```

**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`.

```

12197 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
12198   \glstreeindent=\glsxtrtreetopindent\relax
12199 }

```

**computeTreeSubIndent** `\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.

```

12200 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
12201   \ifcsundef{@glswidestname\romannumeral#1}%
12202   {%
12203     \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
12204   }%
12205   {%
12206     \settowidth{#3}{\glstreenamefmt{%
12207       \csname @glswidestname\romannumeral#1\endcsname\space}}%
12208   }%
12209 }

```

**treeSetHangIndent** Set `\hangindent` for top-level entries:

```

12210 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```



etSubHangIndent Set \hangindent for sub-entries:

```
12211 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
12212 \renewglossarystyle{alttree}{%
12213   \renewenvironment{theglossary}%
12214     {%
12215       \glxtralttreeInit
12216       \def\@gls@prevlevel{-1}%
12217       \mbox{}\par}%
12218     {\par}%
12219   \renewcommand*{\glossaryheader}{}%
12220   \renewcommand*{\glsgroupheading}[1]{}%
12221   \renewcommand{\glossentry}[2]{%
12222     \ifnum\@gls@prevlevel=0\relax
12223     \else
12224       \glxtrComputeTreeIndent{##1}%
12225     \fi
12226     \parindent\glstreeindent
12227     \glxtrAltTreeSetHangIndent
12228     \makebox[0pt][r]%
12229     {%
12230       \glstreenamebox{\glstreeindent}%
12231       {%
12232         \glssentryitem{##1}%
12233         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
12234       }%
12235     }%
12236     \glxtralttreeSymbolDescLocation{##1}{##2}%
12237     \def\@gls@prevlevel{0}%
12238   }
12239   \renewcommand{\subglossentry}[3]{%
12240     \ifnum##1=1\relax
12241       \glssubentryitem{##2}%
12242     \fi
12243     \ifnum\@gls@prevlevel=##1\relax
12244     \else
12245       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12246       \ifnum\@gls@prevlevel<##1\relax
12247         \setlength\glstreeindent\gls@tmplen
12248         \addtolength\glstreeindent\parindent
12249         \parindent\glstreeindent
12250       \else
12251         \ifnum\@gls@prevlevel=0\relax
12252           \glxtrComputeTreeIndent{##2}%
12253         \else
12254           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12255         \fi
12256         \addtolength\parindent{-\glstreeindent}%

```

```

12257         \setlength\glstreeindent\parindent
12258     \fi
12259 \fi
12260 \glstrAltTreeSetSubHangIndent{##1}%
12261 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
12262     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12263 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12264 \def\@gls@prevlevel{##1}%
12265 }%
12266 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12267 }
12268 }%
12269 {%
12270 }

```

Redefine `alttreegroup` so that it discourages a break after group headings. Can't use `\@afterheading` here as it messes with the first item of the group.

```

12271 \ifdef{\@glsstyle@alttreegroup}
12272 {%
12273     \renewglossarystyle{alttreegroup}{%
12274         \setglossarystyle{alttree}%
12275         \renewcommand{\glsgroupheading}[1]{\par
12276             \def\@gls@prevlevel{-1}%
12277             \hangindentOpt\relax
12278             \parindentOpt\relax
12279             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%
12280         \nopagebreak\indexspace\nopagebreak
12281     }%
12282 }%
12283 }%
12284 {%
12285 }

```

Similarly for `alttreehypergroup`.

```

12286 \ifdef{\@glsstyle@alttreehypergroup}
12287 {%
12288     \renewglossarystyle{alttreehypergroup}{%
12289         \setglossarystyle{alttree}%
12290         \renewcommand*\{\glossaryheader}{%
12291             \par
12292             \def\@gls@prevlevel{-1}%
12293             \hangindentOpt\relax
12294             \parindentOpt\relax
12295             \glstreenavigationfmt{\glsnavigation}\par\indexspace
12296         }%
12297         \renewcommand*\{\glsgroupheading}[1]{%
12298             \par
12299             \def\@gls@prevlevel{-1}%
12300             \hangindentOpt\relax
12301             \parindentOpt\relax

```

```

12302      \glstreegroupheaderfmt
12303      {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12304      \nopagebreak\indexspace\nopagebreak
12305  }%
12306  }
12307 }%
12308 {%
12309 }

```

## 2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12310 \ifdef{\@glsstyle@mcolindexgroup}
12311 {%
12312   \renewglossarystyle{mcolindexgroup}{%
12313     \setglossarystyle{mcolindex}%
12314     \renewcommand*\{glsgroupheading}[1]{%
12315       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12316       \nopagebreak\indexspace\nobreak\@afterheading
12317     }%
12318   }
12319 }%
12320 {%
12321 }

```

Similarly for mcolindexhypergroup.

```

12322 \ifdef{\@glsstyle@mcolindexhypergroup}
12323 {%
12324   \renewglossarystyle{mcolindexhypergroup}{%
12325     \setglossarystyle{mcolindex}%
12326     \renewcommand*\{glossaryheader}{%
12327       \item\glstreenavigationfmt{\glsnavigation}%
12328       \indexspace
12329     }%
12330     \renewcommand*\{glsgroupheading}[1]{%
12331       \item\glstreegroupheaderfmt
12332       {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12333       \nopagebreak\indexspace\nobreak\@afterheading
12334     }%
12335   }
12336 }%
12337 {%
12338 }

```

Similarly for mcolindexspannav.

```

12339 \ifdef{\@glsstyle@mcolindexspannav}
12340 {%
12341   \renewglossarystyle{mcolindexspannav}{%
12342     \setglossarystyle{index}%

```

```

12343 \renewenvironment{theglossary}%
12344 {%
12345 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12346 \setlength{\parindent}{0pt}%
12347 \setlength{\parskip}{0pt plus 0.3pt}%
12348 \let\item\glstreeitem}%
12349 {\end{multicols}}}%
12350 \renewcommand*{\glsgroupheading}[1]{%
12351 \item\glstreegroupheaderfmt
12352 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12353 \nopagebreak\indexspace\nobreak\@afterheading
12354 }%
12355 }
12356 }%
12357 {%
12358 }

```

Similarly for mcoltreegroup.

```

12359 \ifdef{\@glstyle@mcoltreegroup}
12360 {%
12361 \renewglossarystyle{mcoltreegroup}{%
12362 \setglossarystyle{mcoltree}%
12363 \renewcommand{\glsgroupheading}[1]{\par
12364 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12365 \nopagebreak\indexspace\nobreak\@afterheading
12366 }%
12367 }
12368 }%
12369 {%
12370 }

```

Similarly for mcoltreehypergroup.

```

12371 \ifdef{\@glstyle@mcoltreehypergroup}
12372 {%
12373 \renewglossarystyle{mcoltreehypergroup}{%
12374 \setglossarystyle{mcoltree}%
12375 \renewcommand*{\glossaryheader}{%
12376 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12377 }%
12378 \renewcommand*{\glsgroupheading}[1]{%
12379 \par\noindent
12380 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12381 \nopagebreak\indexspace\nobreak\@afterheading
12382 }%
12383 }
12384 }%
12385 {%
12386 }

```

Similarly for mcoltreespannav.

```

12387 \ifdef{\@glstyle@mcoltreespannav}

```

```

12388 {%
12389   \renewglossarystyle{mcoltreespannav}{%
12390     \setglossarystyle{tree}%
12391     \renewenvironment{theglossary}%
12392     {%
12393       \begin{multicols}{\glsmcols}%
12394       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12395       \setlength{\parindent}{0pt}%
12396       \setlength{\parskip}{0pt plus 0.3pt}%
12397     }%
12398   {\end{multicols}}}%
12399   \renewcommand*{\glsgroupheading}[1]{%
12400     \par\noindent
12401     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12402     \nopagebreak\indexspace\nobreak\@afterheading
12403   }%
12404 }
12405 }%
12406 {%
12407 }

```

Similarly for mcoltreenonamegroup.

```

12408 \ifdef{\@glsstyle@mcoltreenonamegroup}
12409 {%
12410   \renewglossarystyle{mcoltreenonamegroup}{%
12411     \setglossarystyle{mcoltreenoname}%
12412     \renewcommand*{\glsgroupheading}[1]{\par
12413       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12414     \nopagebreak\indexspace\nobreak\@afterheading
12415   }%
12416 }
12417 }%
12418 {%
12419 }

```

Similarly for mcoltreenonamehypergroup.

```

12420 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12421 {%
12422   \renewglossarystyle{mcoltreenonamehypergroup}{%
12423     \setglossarystyle{mcoltreenoname}%
12424     \renewcommand*{\glossaryheader}{%
12425       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12426     \renewcommand*{\glsgroupheading}[1]{%
12427       \par\noindent
12428       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12429       \nopagebreak\indexspace\nobreak\@afterheading}%
12430   }
12431 }%
12432 {%
12433 }

```

Similarly for mcoltreenonamespannav.

```

12434 \ifdef{\@glsstyle@mcoltreenonamespannav}
12435 {%
12436   \renewglossarystyle{mcoltreenonamespannav}{%
12437     \setglossarystyle{treenoname}%
12438     \renewenvironment{theglossary}%
12439     {%
12440       \begin{multicols}{\glsmcols}%
12441       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12442       \setlength{\parindent}{0pt}%
12443       \setlength{\parskip}{0pt plus 0.3pt}%
12444     }%
12445     {\end{multicols}}}%
12446   \renewcommand*{\glsgroupheading}[1]{%
12447     \par\noindent
12448     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12449     \nopagebreak\indexspace\nobreak\@afterheading}%
12450   }
12451 }%
12452 {%
12453 }

```

mcolalmtree needs adjusting so that it uses \glsxtralmtreeInit This doesn't use \mbox{} \par which would unbalance the top of the columns.

```

12454 \ifdef{\@glsstyle@mcolalmtree}
12455 {%
12456   \renewglossarystyle{mcolalmtree}{%
12457     \setglossarystyle{almtree}%
12458     \renewenvironment{theglossary}%
12459     {%
12460       \glsxtralmtreeInit
12461       \def\@gls@prevlevel{-1}%
12462       \begin{multicols}{\glsmcols}%
12463     }%
12464     {\par\end{multicols}}}%
12465   }
12466 }%
12467 {%
12468 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12469 \ifdef{\@glsstyle@mcolalmtreegroup}
12470 {%
12471   \renewglossarystyle{mcolalmtreegroup}{%
12472     \setglossarystyle{mcolalmtree}%
12473     \renewcommand{\glsgroupheading}[1]{\par
12474       \def\@gls@prevlevel{-1}%
12475       \hangindent0pt\relax
12476       \parindent0pt\relax
12477       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

12478     \nopagebreak\indexspace\nopagebreak
12479   }%
12480 }
12481 }%
12482 {%
12483 }

```

Similarly for mcolalttreehypergroup.

```

12484 \ifdef{\@glsstyle@mcolalttreehypergroup}
12485 {%
12486   \renewglossarystyle{mcolalttreehypergroup}{%
12487     \setglossarystyle{mcolalttree}%
12488     \renewcommand*{\glossaryheader}{%
12489       \par
12490       \def\@gls@prevlevel{-1}%
12491       \hangindent0pt\relax
12492       \parindent0pt\relax
12493       \glstreenavigationfmt{\glsnavigation}%
12494       \par\indexspace
12495     }%
12496     \renewcommand*{\glsgroupheading}[1]{%
12497       \par
12498       \def\@gls@prevlevel{-1}%
12499       \hangindent0pt\relax
12500       \parindent0pt\relax
12501       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12502       \nopagebreak\indexspace\nopagebreak
12503     }%
12504   }
12505 }%
12506 {%
12507 }

```

Similarly for mcolalttreespannav.

```

12508 \ifdef{\@glsstyle@mcolalttreespannav}
12509 {%
12510   \renewglossarystyle{mcolalttreespannav}{%
12511     \setglossarystyle{alttree}%
12512     \renewenvironment{theglossary}%
12513     {%
12514       \glsextralmtreeInit
12515       \def\@gls@prevlevel{-1}%
12516       \begin{multicols}{\glsncols}%
12517         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12518     }%
12519     {\par\end{multicols}}%
12520     \renewcommand*{\glsgroupheading}[1]{%
12521       \par
12522       \def\@gls@prevlevel{-1}%
12523       \hangindent0pt\relax

```

```

12524      \parindent0pt\relax
12525      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12526      \nopagebreak\indexspace\nopagebreak
12527    }%
12528  }
12529 }%
12530 {%
12531 }

```

Reset the default style

```

12532 \ifx\@glossary@default@style\relax
12533 \else
12534   \setglossarystyle{\@glxtr@current@style}
12535 \fi

```



# 3 bookindex style (glossary-bookindex.sty)

## 3.1 Package Initialisation and Options

```
12536 \NeedsTeXFormat{LaTeX2e}
12537 \ProvidesPackage{glossary-bookindex}[2017/11/14 v1.24 (NLCT)]
```

Load required packages.

```
12538 \RequirePackage{multicol}
12539 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12540 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12541 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`ookindexsubname` Format used for sub entries.

```
12542 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`strprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12543 \providecommand*{\glstrprelocation}{\space}
```

`indexprelocation` 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.

```
12544 \newcommand*{\glstrbookindexprelocation}[1]{%
12545   \glstrifhasfield{location}{#1}%
12546   {,\glstrprelocation}%
12547   {\glstrprelocation}%
12548 }
```

`xsubprelocation` Separator used before location list for sub-entries.

```
12549 \newcommand*{\glstrbookindexsubprelocation}[1]{%
12550   \glstrbookindexprelocation{#1}%
12551 }
```

`xparentchildsep` Separator used between top-level parent and child entry.

```
12552 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`rentsubchildsep` Separator used between sub-level parent and child entry.

```
12553 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
12554 \newcommand{\glxstrbookindexbetween}[2] {}
```

`indexsubbetween` Between two level 1 entries identified by the labels in the arguments.

```
12555 \newcommand{\glxstrbookindexsubbetween}[2] {}
```

`exsubsubbetween` Between two level 2 entries identified by the labels in the arguments.

```
12556 \newcommand{\glxstrbookindexsubsubbetween}[2] {}
```

`indexatendgroup` At the end of a letter group. The argument is the index of the last top-level entry.

```
12557 \newcommand{\glxstrbookindexatendgroup}[1] {}
```

`exsubatendgroup` At the end of a letter group. The argument is the index of the last level 1 entry.

```
12558 \newcommand{\glxstrbookindexsubatendgroup}[1] {}
```

`subsubatendgroup` At the end of a letter group. The argument is the index of the last level 2 entry.

```
12559 \newcommand{\glxstrbookindexsubsubatendgroup}[1] {}
```

`bookindexgroupskip` Group separator.

```
12560 \newcommand{\glxstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

`indexformatheader` Group separator.

```
12561 \newcommand*{\glxstrbookindexformatheader}[1] {%
12562   \par{\centering\glstreegroupheaderfmt{#1}\par}%
12563 }
```

`bookindexbookmark` Book mark group heading if supported.

```
12564 \ifdef\pdfbookmark
12565 {%
12566   \newcommand*{\glxstrbookindexbookmark}[2] {%
12567     \ifdefstring{\@glossarysec}{chapter}%
12568     {\pdfbookmark[1]{#1}{#2}}%
12569     {\pdfbookmark[2]{#1}{#2}}%
12570   }
12571 }
12572 {%
12573   \newcommand*{\glxstrbookindexbookmark}[2] {}
12574 }
```

`bookindexcolspread`

```
12575 \newcommand*{\glxstrbookindexcolspread} {}
```

Define the style.

```
12576 \newglossarystyle{bookindex} {%
12577   \setglossarystyle{index}%
12578   \renewenvironment{theglossary}%
```

```

12579 {%
12580   \ifdefempty\glxstrbookindexcolspread
12581   {\begin{multicols}{\glxstrbookindexcols}}%
12582   {\begin{multicols}{\glxstrbookindexcols}[\glxstrbookindexcolspread]}%
12583   \setlength{\parindent}{0pt}%
12584   \setlength{\parskip}{0pt plus 0.3pt}%
12585   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12586   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12587   \let\@glxstr@bookindex@between\@gobble
12588   \let\@glxstr@bookindex@subbetween\@gobble
12589   \let\@glxstr@bookindex@subsubbetween\@gobble
12590   \let\@glxstr@bookindex@atendgroup\relax
12591   \let\@glxstr@bookindex@subatendgroup\relax
12592   \let\@glxstr@bookindex@subsubatendgroup\relax
12593   \let\@glxstr@bookindex@groupskip\relax
12594 }%
12595 {%

```

Do end group hooks.

```

12596   \@glxstr@bookindex@subsubatendgroup
12597   \@glxstr@bookindex@subatendgroup
12598   \@glxstr@bookindex@atendgroup

```

End multicols environment.

```

12599   \end{multicols}%
12600 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12601 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12602 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12603   \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12604   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12605   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12606   \let\@glxstr@bookindex@subbetween\@gobble
12607   \let\@glxstr@bookindex@subsubbetween\@gobble
12608   \edef\@glxstr@bookindex@between{%
12609     \noexpand\glxstrbookindexbetween{##1}%
12610   }%
12611   \edef\@glxstr@bookindex@atendgroup{%
12612     \noexpand\glxstrbookindexatendgroup{##1}%
12613   }%
12614   \let\@glxstr@bookindex@subatendgroup\relax
12615   \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12616   \glstreeitem

```

```

12617     \glsentryitem{##1}%
12618     \glstarget{##1}{\glstrbookindexname{##1}}%
12619     \glstrbookindexprelocation{##1}##2%
12620 }%
12621 \renewcommand{\subglossentry}[3]{%
12622     \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12623     \glstreeitem
12624     \or

```

Level 1.

```

12625     \@glstr@bookindex@sep
12626     \@glstr@bookindex@subbetween{##2}%
12627     \let\@glstr@bookindex@sep\relax

```

Update separators.

```

12628     \let\@glstr@bookindex@subsubbetween\@gobble
12629     \let\@glstr@bookindex@subsep\glstrbookindexparentschildsep
12630     \edef\@glstr@bookindex@subbetween{%
12631         \noexpand\glstrbookindexsubbetween{##2}%
12632     }%
12633     \edef\@glstr@bookindex@atsubendgroup{%
12634         \noexpand\glstrbookindexatsubendgroup{##1}%
12635     }%

```

Start sub-item.

```

12636     \glstreesubitem
12637     \glssubentryitem{##2}%
12638     \else

```

All other levels.

```

12639     \@glstr@bookindex@subsep
12640     \@glstr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12641     \let\@glstr@bookindex@subsep\relax
12642     \edef\@glstr@bookindex@subsubbetween{%
12643         \noexpand\glstrbookindexsubsubbetween{##2}%
12644     }%
12645     \edef\@glstr@bookindex@atsubsubendgroup{%
12646         \noexpand\glstrbookindexatsubsubendgroup{##1}%
12647     }%

```

Start sub-sub-item.

```

12648     \glstreesubsubitem
12649     \fi

```

Format entry.

```

12650     \glstarget{##2}{\glstrbookindexsubname{##2}}%
12651     \glstrbookindexsubprelocation{##2}##3%
12652 }%

```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12653 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12654 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12655 \glxstr@bookindex@subsubatendgroup
```

```
12656 \glxstr@bookindex@subatendgroup
```

```
12657 \glxstr@bookindex@atendgroup
```

```
12658 \glxstr@bookindexgroupskip
```

Update separators.

```
12659 \let\glxstr@bookindexgroupskip\glxstrbookindexgroupskip
```

```
12660 \let\glxstr@bookindex@between\@gobble
```

```
12661 \let\glxstr@bookindex@atendgroup\relax
```

```
12662 \let\glxstr@bookindex@subatendgroup\relax
```

```
12663 \let\glxstr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12664 \glxstrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12665 \glxstrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12666 \glxstrbookindexformatheader{\thisgrptitle}%
```

```
12667 \nopagebreak\indexspace\nopagebreak\@afterheading
```

```
12668 }%
```

```
12669 }
```

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 `\glxstrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`bookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
12670 \newcommand{\glxstrbookindexthepage}{%
```

```
12671 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
```

```
12672 }
```

`bookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
12673 \newcommand*{\glxstrbookindexmarkentry}[1]{%
```

```
12674 \protected@write\@auxout
```

```
12675 {\let\glxstrbookindexthepage\relax}%
```

```
12676 {\string\glxstr@setbookindexmark{\glxstrbookindexthepage}{#1}}%
```

```
12677 }
```

etbookindexmark

```
12678 \newcommand*{\glxtr@setbookindexmark}[2]{%
12679   \ifcsundef{glxtr@idxfirstmark@#1}%
12680   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
12681   {}%
12682   \csgdef{glxtr@idxlastmark@#1}{#2}%
12683 }
```

dexfirstmarkfmt

```
12684 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
12685   \glentryname{#1}%
12686 }
```

kindexfirstmark

```
12687 \newcommand*{\glxtrbookindexfirstmark}{%
12688   \letcs{glxtr@label}{glxtr@idxfirstmark@glxtrbookindexthepage}%
12689   \ifdefglxtr@label
12690   {\glxtrbookindexfirstmarkfmt{glxtr@label}}%
12691   {}%
12692 }
```

ndexlastmarkfmt

```
12693 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
12694   \glentryname{#1}%
12695 }
```

okindexlastmark

```
12696 \newcommand*{\glxtrbookindexlastmark}{%
12697   \letcs{glxtr@label}{glxtr@idxlastmark@glxtrbookindexthepage}%
12698   \ifdefglxtr@label
12699   {\glxtrbookindexlastmarkfmt{glxtr@label}}%
12700   {}%
12701 }
```

# Glossary

**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)		\@Glsymbol@: added redefinition	65
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	65
\Glsfmtshort: new	303	\@Glstext@: added redefinition	61
\glsfmtshort: new	302	\@Glsuseri@: added redefinition	66
\Glsfmtshortpl: new	303	\@Glsuserii@: added redefinition	66
\glsfmtshortpl: new	302	\@Glsuseriii@: added redefinition	66
short: switched inline full form to short		\@Glsuseriv@: added redefinition	66
(long)	206	\@Glsuserv@: added redefinition	67
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	67
\@ACRlong: added redefinition	70	\@acrlong: added redefinition	70
\@ACRlongpl: added redefinition	71	\@acrlongpl: added redefinition	71
\@ACRshort: added redefinition	68	\@acrshort: added redefinition	67
\@ACRshortpl: added redefinition	69	\@acrshortpl: added redefinition	68
\@Acrlong: added redefinition	70	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	71	argument	54
\@Acrshort: added redefinition	68	\@glsdescplural@: added redefinition	64
\@Acrshortpl: added redefinition	69	\@glsfirst@: added redefinition	61
\@GLSdesc@: added redefinition	64	\@glsfirstplural@: added redefinition	62
\@GLSdescplural@: added redefinition	64	\@glsplural@: added redefinition	62
\@GLSfirst@: added redefinition	61	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	63	redefinition	65
\@GLSname@: added redefinition	63	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	62	new	121
\@GLSSymbol@: added redefinition	65	\@glsxtr@field@linkdefs: new	60
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	176
redefinition	65	\@print@glossary: added redefinition	117
\@GLStext@: added redefinition	60	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	66	\abbrvdefaultfont	182
\@GLSuserii@: added redefinition	66	\glsaccessdesc: new	144
\@GLSuseriii@: added redefinition	66	\glsaccessdescplural: new	145
\@GLSuseriv@: added redefinition	67	\glsaccessfirst: new	142
\@GLSuserv@: added redefinition	67	\glsaccessfirstplural: new	142
\@GLSuservi@: added redefinition	67	\Glsaccesslong: new	146
\@Glsdesc@: added redefinition	64	\glsaccesslong: new	146
\@Glsdescplural@: added redefinition	64	\glsaccessname: new	140
\@Glsfirst@: added redefinition	61	\glsaccessplural: new	141
\@Glsfirstplural@: added redefinition	63	\Glsaccessshort: new	145
\@Glsname@: added redefinition	63	\glsaccessshort: new	145
\@Glsplural@: added redefinition	62	\Glsaccessshortpl: new	146



\glsaccesssshortpl: new .....	146	\cGLSpl: new .....	95
\glsaccessssymbol: new .....	143	\cGLSpl@: new .....	95
\glsaccessssymbolplural: new .....	143	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new .....	141	new .....	90
\glstentryfmt: added check for short ..	54	\cGLS: new .....	94
\glslongpltok: new .....	176	\cGLSformat: new .....	95
\glsshortpltok: new .....	176	\cGLSpl: new .....	95
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new .....	95
name in \setkeys .....	178	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new .....	15
for plural .....	173	\glsenableentrycount: new .....	90
\GLSxtrlongpl: new .....	191	\glsfirstabbrvdefaultfont: new ..	181
\Glsxtrlongpl: new .....	190	\glsfirstlongdefaultfont: new ...	182
\glsxtrlongpl: new .....	190	\Glsfmtfirst: new .....	305
\glsxtrNoGlossaryWarning: new ....	20	\glsfmtfirst: new .....	305
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new .....	306
new .....	173	\glsfmtfirstpl: new .....	306
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new .....	305
new .....	173	\glsfmtplural: new .....	304
\glsxtrpostlinkendsentence: new ..	173	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new .....	189	\Glsxtrtitleshort .....	303
\Glsxtrshortpl: new .....	189	renamed from \glstentryfmtshort ..	303
\glsxtrshortpl: new .....	188	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort .....	302
\glslabeltok .....	201	renamed from \glstentryfmtshort ..	302
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok .....	199	\Glsxtrtitleshortpl .....	303
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl .....	303
redefinition of \acronymtype .....	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl .....	302
\Glsxtrshort .....	303	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl .....	302
\glsxtrshort .....	302	\Glsfmttext: new .....	304
\Glsfmtshortpl: changed to use		\glsfmttext: new .....	304
\glsxtrshortpl .....	303	\glshasattribute: new .....	152
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	151
\glsxtrshortpl .....	302	\glsxtremsuffix: new .....	242
\glsxtrifemptyglossary: new .....	25	\GlsXtrEnableEntryCounting: new ..	89
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new .....	92
argument .....	155	\glsxtrscfont: new .....	214
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new .....	214
argument .....	155	\glsxtrsmfont: new .....	228
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new .....	228
default type to \acronymtype ....	103	short-em: new .....	249
\newterm: fixed name argument .....	154	short-em-desc: new .....	251
0.5 (2015-12-07)		short-em-footnote: new .....	260
\@cGLS: new .....	94	short-em-long: new .....	246
\@cGLS@: new .....	95	short-em-long-desc: new .....	247

short-em-postfootnote: new .....	262	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new .....	224	attribute .....	293
short-sc-postfootnote: new .....	226	\Glsxtrheadtext: now uses headuc	
short-sm: new .....	232	attribute .....	296
short-sm-desc: new .....	233	\glxtrheadtext: now uses headuc	
short-sm-footnote: new .....	239	attribute .....	295
short-sm-long: new .....	230	short-em-footnote: switch off regular	
short-sm-long-desc: new .....	231	attribute if set .....	260
short-sm-postfootnote: new .....	240	short-long: switch off regular attribute	
long-noshort-em: new .....	253	if set .....	200
long-noshort-em-desc: new .....	257	short-long-desc: switch off regular	
long-noshort-sm: new .....	235	attribute if set .....	202
long-noshort-sm-desc: new .....	237	short-sc-footnote: switch off regular	
long-short-em: new .....	243	attribute if set .....	225
long-short-em-desc: new .....	244	short-sm-footnote: switch off regular	
long-short-sm: new .....	228	attribute if set .....	239
long-short-sm-desc: new .....	230	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set .....	198
\Glsaccessstext: new .....	141	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set .....	200
\@glxtr@doaccsupp: new .....	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	293	attribute if set .....	216
\Glsaccessdesc: new .....	144	footnote: switch off regular attribute if	
\Glsaccessdescplural: new .....	145	set .....	203
\Glsaccessfirst: new .....	142	postfootnote: switch off regular	
\Glsaccessfirstplural: new .....	142	attribute if set .....	204
\Glsaccessname: new .....	140	0.5.2 (2015-12-08)	
\Glsaccessplural: new .....	141	\@GLSdesc@: added accessibility support	64
\Glsaccesssymbol: new .....	143	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new .....	144	support .....	64
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute .....	297	support .....	61
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute .....	297	support .....	63
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	63
headuc attribute .....	298	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support .....	62
headuc attribute .....	298	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support .....	65
attribute .....	297	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support .....	65
attribute .....	296	\@GLStext@: added accessibility support	60
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	64
attribute .....	294	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support .....	64
attribute .....	293	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support .....	61
attribute .....	294	\@GLSfirstplural@: added accessibility	
		support .....	63

\@Glsname@: add accessibility support ..	63	\GLSaccesssymbolplural: new ..	144, 149
\@Glsplural@: added accessibility support .....	62	\GLSaccessstext: new .....	141, 148
\@Glsymbol@: added accessibility support .....	65	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support .....	65	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	61	\glsxtrabbrvfmt to here .....	54
\@glsdesc@: added accessibility support	64	\GlsXtrEnableInitialTagging: new	169
\@glsdescplural@: added accessibility support .....	64	\glsxtrfieldtitlecase: new .....	156
\@glsfirst@: added accessibility support .....	61	\GlsXtrFormatLocationList: new ...	52
\@glsfirstplural@: added accessibility support .....	62	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	63	new .....	180
\@Glsplural@: added accessibility support .....	62	\glsxtrtagfont: new .....	171
\@Glsymbol@: added accessibility support .....	65	\KV@printgloss@nonumberlist: added	53
\@Glsymbolplural@: added accessibility support .....	65	\mfu@checkword@do: added .....	170
\@Glstext@: added accessibility support	60	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch ....	195
new .....	171	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	171	\@glsxtr@autoindex@at: new .....	167
\@glsxtr@tag: new .....	171	\@glsxtr@autoindex@encap: new ...	167
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new .....	167
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	167
\@ifpackageloaded .....	140	\@glsxtr@autoindex@setname: new ..	165
removed \glsxtrabbrvfmt .....	192	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added .....	51	General: removed	
\Glossentrydesc: added .....	169	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added .....	161	.....	120
\Glossentrysymbol: added .....	169	\glsdescwidth: added .....	51
\glossentrysymbol: added .....	169	\glspagelistwidth: added .....	51
\GLSaccessdesc: new .....	144, 149	\glsxtrdoautoindexname: new .....	165
\GLSaccessdescplural: new ...	145, 149	\glsxtrpostnamehook: new .....	162
\GLSaccessfirst: new .....	142, 148	\if@glsxtr@format@override: new ..	164
\GLSaccessfirstplural: new ..	143, 148	\ProvidesGlossariesExtraLang: new	309
\GLSaccesslong: new .....	147, 150	\RequireGlossariesExtraLang: new	309
\GLSaccesslongpl: new .....	147, 150	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new .....	147	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new .....	147	new .....	96
\GLSaccessname: new .....	140, 147	\@GLSxtr@p@acrlong@: new .....	83
\GLSaccessplural: new .....	141, 148	\@GLSxtr@p@acrlongpl@: new .....	83
\GLSaccessshort: new .....	146, 150	\@GLSxtr@p@acrshort@: new .....	83
\GLSaccessshortpl: new .....	146, 150	\@GLSxtr@p@acrshortpl@: new .....	83
\GLSaccesssymbol: new .....	143, 149	\@GLSxtr@p@long@: new .....	82
		\@GLSxtr@p@longpl@: new .....	83
		\@GLSxtr@p@plural@: new .....	81
		\@GLSxtr@p@short@: new .....	82
		\@GLSxtr@p@shortpl@: new .....	82
		\@GLSxtr@p@text@: new .....	81
		\@GlsXtrEnableOnTheFly: new .....	47
		\@Glsxtr: new .....	48
		\@Glsxtr@p@acrlong@: new .....	83
		\@Glsxtr@p@acrlongpl@: new .....	83

\@Glsxtr@p@acrshort@: new	83	\glsenableentryunitcount: new	98
\@Glsxtr@p@acrshortpl@: new	83	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	82	entry's existence	152
\@Glsxtr@p@longpl@: new	82	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	81	entry's existence	152
\@Glsxtr@p@short@: new	81	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	82	check	172
\@Glsxtr@p@text@: new	81	\Glsxtr: new	48
\@Glsxtrpl: new	48	\glxtr: new	47
\@alt@glshyp@opt: new	77	\glxtrcat: new	47
\@glscalt@hyp@opt: new	77	\glxtrdowrglossaryhook: new	77
\@glscalt@hyp@opt@char: new	77	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	77	new	101
\@glscincrement@currunitcount:		\GlsXtrEnableOnTheFly: new	46
new	97	\Glsxtrpl: new	48
\@glsclocal@increment@currunitcount:		\glxtrpl: new	48
new	97	\glxtrpostlocalreset: new	89
\@glscsetdefault@glslink@opts:		\glxtrpostlocalunset: new	89
new	74	\glxtrpostreset: new	89
\@glsxtr: new	47	\glxtrpostunset: new	89
\@glsxtr@addunitcounter: new	96	\glxtrprotectlinks: new	80
\@glsxtr@currunitcount: new	98	\GlsXtrSetAltModifier: new	77
\@glsxtr@ifunitcounter: new	96	\GlsXtrSetDefaultGlsOpts: new	76
\@glsxtr@p@acrlong@: new	83	\glxtrstarflywarn: new	47
\@glsxtr@p@acrlongpl@: new	83	\GlsXtrWarning: new	49
\@glsxtr@p@acrshort@: new	83	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	83	disables \setacronymstyle	103
\@glsxtr@p@long@: new	82	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	82	\@glsxtr@autoindexcrossrefs: new	15
\@glsxtr@p@plural@: new	81	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	81	new	111
\@glsxtr@p@shortpl@: new	82	\@glsxtr@idx@entrynumberlist: new	113
\@glsxtr@p@text@: new	81	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	98	new	112
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	101	new	113
\@glsxtr@unitcountlist: new	96	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	48	new	112
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	104
empty see value if not set and added		\makeglossaries: new	105
'see' to field key map	39	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	46	\glxtrdiscardperiod: added check	
\cGlsformat: added	95	for first use	173
\cGlsformat: added	95	short-desc: fixed typo in	
\cGlsplformat: added	96	\glxtrinlinfullformat and	
\cGlsplformat: added	95	added missing second argument	208
\glstdisablehyper: added	79	1.02 (2016-04-25)	
\glstdohyperlink: added	78	\@glsxtr@current@style: new	50
\glstdonohyperlink: added	80	\Glsfmtfull: new	308

\glsfmtfull: new .....	307	\@GLSdescplural@: set abbreviation and regular format .....	64
\Glsfmtfullpl: new .....	308	\@GLSfirst@: set abbreviation format ..	61
\glsfmtfullpl: new .....	308	\@GLSfirstplural@: set abbreviation and regular format .....	63
\Glsfmtlong: new .....	306	\@GLSname@: set abbreviation and regular format .....	63
\glsfmtlong: new .....	306	\@GLSplural@: set abbreviation and regular format .....	62
\Glsfmtlongpl: new .....	307	\@GLSsymbol@: set regular format .....	65
\glsfmtlongpl: new .....	307	\@GLSsymbolplural@: set regular format	65
\Glsxtrheadfull: new .....	301	\@GLStext@: set abbreviation and regular format .....	60
\glsxtrheadfull: new .....	300	\@GLSuseri@: set regular format .....	66
\Glsxtrheadfullpl: new .....	302	\@GLSuserii@: set regular format .....	66
\glsxtrheadfullpl: new .....	301	\@GLSuseriii@: set regular format .....	66
\Glsxtrheadlong: new .....	300	\@GLSuseriv@: set regular format .....	67
\glsxtrheadlong: new .....	299	\@GLSuservi@: set regular format .....	67
\Glsxtrheadlongpl: new .....	300	\@GLSdesc@: set abbreviation and regular format .....	64
\glsxtrheadlongpl: new .....	299	\@GLSdescplural@: set abbreviation and regular format .....	64
\Glsxtrtitlefull: new .....	302	\@GLSfirst@: set abbreviation and regular format .....	61
\glsxtrtitlefull: new .....	301	\@GLSfirstplural@: set abbreviation and regular format .....	63
\Glsxtrtitlefullpl: new .....	302	\@GLSname@: set abbreviation and regular format .....	63
\glsxtrtitlefullpl: new .....	301	\@GLSplural@: set abbreviation and regular format .....	62
\Glsxtrtitlelong: new .....	300	\@GLSsymbol@: set regular format .....	65
\glsxtrtitlelong: new .....	299	\@GLSsymbolplural@: set regular format	65
\Glsxtrtitlelongpl: new .....	300	\@GLStext@: set abbreviation and regular format .....	61
\glsxtrtitlelongpl: new .....	299	\@GLSuseri@: set regular format .....	66
\ifglxtrinsertinside: new .....	198	\@GLSuserii@: set regular format .....	66
postfootnote: added redef of		\@GLSuseriii@: set regular format .....	66
\glsxtrsetupfulldefs .....	205	\@GLSuseriv@: set regular format .....	67
stylemods: new .....	21	\@GLSuservi@: set regular format .....	67
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format .....	64
\@GLSfirstplural@: bug fix: misspelt cs name .....	63	\@GLSdescplural@: set abbreviation and regular format .....	62
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLSsymbol@: set regular format .....	65
\@GLSfirstplural@: bug fix: misspelt cs name .....	63	\@GLSsymbolplural@: set regular format	65
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLStext@: set abbreviation and regular format .....	61
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	62	\@GLSuseri@: set regular format .....	66
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	299	\@GLSuserii@: set regular format .....	66
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	294	\@GLSuseriii@: set regular format .....	66
1.04 (2015-04-30)		\@GLSuseriv@: set regular format .....	66
short-em-footnote: renamed from “footnote-em” .....	260	\@GLSuservi@: set regular format .....	67
1.04 (2016-05-02)		\@GLSdesc@: set abbreviation and regular format .....	64
\@glsxtrpostloctag: new .....	53	\@GLSdescplural@: set abbreviation and regular format .....	64
\@GLSdesc@: set abbreviation and regular format .....	64	\@GLSfirst@: set abbreviation and regular format .....	61

\@glsfirstplural@: set abbreviation and regular format .....	62	\glsxtruserparen: new .....	264
\@glsname@: set abbreviation and regular format .....	63	\glsxtrusersuffix: new .....	265
\@glsplural@: set abbreviation and regular format .....	62	\GlsXtrWarnDeprecatedAbbrStyle: new .....	197
\@glsymbol@: set regular format .....	65	short-em-long-em: new .....	248
\@glsymbolplural@: set regular format .....	65	short-em-long-em-desc: new .....	249
\@glstext@: set abbreviation and regular format .....	60	short-em-nolong: new .....	251
\@glsxtr@deprecated@abbrstyle: new .....	197	short-em-nolong-desc: new .....	252
\@glsxtr@do@style: new .....	21	short-em-postfootnote: renamed from “postfootnote-em” .....	262
\@glsxtr@dolocktag: new .....	53	short-footnote: new .....	204
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand .....	113	short-long-user: new .....	271
\@glsxtr@pagetag: new .....	53	short-long-user-desc: new .....	272
\@glsxtr@pagetag: new .....	53	short-nolong: new .....	207
\@glsxtr@prelocktag: new .....	53	short-nolong-desc: new .....	209
\@glsxtr@postlocktag: new .....	53	short-postfootnote: new .....	206
\@glsxtr@prelocktag: new .....	52, 53	short-sc-footnote: renamed from “footnote-sc” .....	224
\glossentrydesc: added glossdescfont attribute check .....	157	short-sc-nolong: new .....	219
\Glossentryname: added glossnamefont attribute check .....	161	short-sc-nolong-desc: new .....	220
\glossentryname: added glossnamefont attribute check .....	158	short-sc-postfootnote: renamed from “postfootnote-sc” .....	226
moved post name hook inside condition .....	160	short-sm-footnote: renamed from “footnote-sm” .....	239
\glsabbrvmfont: new .....	242	short-sm-nolong: new .....	233
\glsabbrvuserfont: new .....	264	short-sm-nolong-desc: new .....	235
\glsfirstabbrvmfont: new .....	242	short-sm-postfootnote: renamed from “postfootnote-sm” .....	240
\glsfirstabbrvuserfont: new .....	264	\letabbreviationstyle: new .....	197
\glsfirstlongemfont: new .....	242	\newabbreviationstyle: bug fix: corrected test for existence .....	195
\glsfirstlonguserfont: new .....	264	long-em-noshort-em: new .....	255
\glsifnotregularcategory: new ...	153	long-em-noshort-em-desc: new ...	258
\glslongdefaultfont: new .....	182	long-em-short-em: new .....	244
\glslongemfont: new .....	242	long-em-short-em-desc: new .....	245
\glslongfont: new .....	182	long-noshort: new .....	213
\glslonguserfont: new .....	264	long-noshort-desc: new .....	212
\glsxtrassignfieldfont: new .....	60	long-noshort-em: renamed from “long-em” .....	253
\GlsXtrEnablePreLocationTag: new .	52	long-noshort-em-desc: renamed from “long-desc-em” .....	257
\glsxtrfirstscfont: new .....	214	long-noshort-sc: renamed from “long-sc” .....	221
\glsxtrfirstsmfont: new .....	228	long-noshort-sc-desc: renamed from “long-desc-sc” .....	223
\glsxtrlongshortdescsort: new ...	199	long-noshort-sm: renamed from “long-sm” .....	235
\glsxtrpostnamehook: added category check .....	162	long-noshort-sm-desc: renamed from \long-desc-sm .....	237
\glsxtrregularfont: new .....	54		
\glsxtruserfield: new .....	264		

long-short-user: new .....	265	docdef option changed to choice .....	14
long-short-user-desc: new .....	270	\glstr@usesee: new .....	39
\renewabbreviationstyle: new ....	196	\glstrusesee: new .....	39
style: new .....	21	\glstruseseeformat: new .....	39
1.05 (2016-06-10)		\if@glstrdocdefrestricted: new ..	14
\eglssetwidest: new .....	327	1.07 (2016-08-15)	
\glFindWidestAnyName: new .....	330	\@@glstrp: new .....	84
\glFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new .....	335	nohyperfirst attribute .....	62
\glFindWidestAnyNameSymbol: new	333	\@GLSfirstplural@: added check for	
\glFindWidestAnyNameSymbolLocation:		nohyperfirst attribute .....	63
new .....	334	\@GLSxtrp: new .....	85
\glFindWidestLevelTwo: new ....	331	\@Glsfirst@: added check for	
\glFindWidestUsedAnyName: new ..	330	nohyperfirst attribute .....	61
\glFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new .....	335	nohyperfirst attribute .....	63
\glFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new .....	84
new .....	332	\@gls@preglossaryhook: added	
\glFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts .....	171
new .....	333	\@glsfirst@: added check for	
\glFindWidestUsedLevelTwo: new .	330	nohyperfirst attribute .....	61
\glFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new .....	329	nohyperfirst attribute .....	62
\glfirstlongfootnotefont: new ..	202	\@glstrinmark: new .....	291
\glsgetwidestname: new .....	329	\@glstrnotinmark: new .....	291
\glsgetwidestsubname: new .....	329	\@glstrp: new .....	84
\glslongfootnotefont: new .....	202	\@glstrp@opt: new .....	83
\glstrAltTreeIndent: new .....	327	\glossxtrsetpopts: new .....	84
\glstrAltTreeInit: new .....	327	\glsp: new .....	86
\glstrAltTreePar: new .....	327	\glsp: new .....	86
\glstrAltTreeSetHangIndent: new	336	\glstr@entry@p: new .....	85
\glstrAltTreeSetSubHangIndent:		\glstrabbrvfootnote: new .....	202
new .....	337	\glstrchecknohyperfirst: new ....	61
\glstrAltTreeSubSymbolDescLocation:		\glstrfieldtitlecasecs: new ....	156
new .....	327	\glstrifinmark: new .....	291
\glstrAltTreeSymbolDescLocation:		\GLSxtrp: new .....	87
new .....	326	\Glsxtrp: new .....	86
\glstrComputeTreeIndent: new ...	336	\glstrp: new .....	85
\glstrComputeTreeSubIndent: new	336	\glstrsetpopts: new .....	84
\glstrtreetopindent: new .....	327	short-long-desc: added text key ....	202
short-em-long: fixed incorrect font used		fixed misspelling of \glabbrvfont in	
by long form .....	246	plural key .....	202
\xglsetwidest: new .....	328	long-short-desc: added missing text	
1.06 (2016-06-18)		key .....	200
\@glsoifexistsorwarn: new .....	14	fixed misspelling of \glabbrvfont .	200
\@glstr@docdefval: new .....	14	footnote: changed first forms to use	
\@glstr@usesee: new .....	39	\glfirstlongfootnotefont ...	202
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document .....	25	from first keys .....	204

switched from \glsfirstlongfont to	1.10 (2016-12-17)
\glsfirstlongfootnotefont ... 205	\@GLSp1@: fixed bug caused by typo in
\RestoreAcronyms: modified	command name ..... 56
\@gls@link@checkfirsthyper to	1.11 (2017-01-19)
set \glsxtrifwasfirstuse ..... 104	\@glsxtr@do@redef@for@gl@sentries:
1.08 (2016-12-13)	new ..... 6
\@glsxtr@record: new ..... 8	\@glsxtr@noidx@do: new ..... 131
\@GLS@: added \@glsxtr@record ..... 55	\@glsxtr@redef@for@gl@sentries: new . 6
\@GLSp1@: added \@glsxtr@record ... 56	\@glsxtr@shortcutsval: new ..... 19
\@Gls@: added \@glsxtr@record ..... 55	\@glsxtr@unsrt@get@group@title: new 130
\@GLSp1@: added \@glsxtr@record ... 55	\@print@noidx@glossary: added
\@Gls@: added \@glsxtr@record ..... 55	redefinition ..... 115
\@gls@@link@: added	\glsxtr@add@loc@list@field: added
\@glsxtr@record ..... 56	group key ..... 12
\@gls@field@link: added	added location key ..... 12
\@glsxtr@record ..... 54	\glsxtr@fields: new ..... 123
\@gls@save@entry@counter: new ..... 25	\glsxtr@link@prefix: new ..... 123
\@glsdisp: added \@glsxtr@record .. 56	\glsxtr@org@new@ignored@glossary:
\@GLSp1@: added \@glsxtr@record ... 55	new ..... 34
\@glsxtr@dorecord: new ..... 10	\glsxtr@s@new@ignored@glossary: new 35
\@glsxtr@err@undefaction: new ..... 6	\glsxtr@shortcutsval: new ..... 123
\@glsxtr@record: new ..... 7	\glsxtr@texencoding: new ..... 123
\@glsxtr@warn@on@exists@ordo: new ... 6	\glsxtr@write@fields: new ..... 123
\@glsxtr@warn@undefaction: new .... 6	\GlsXtrLoadResources: new ..... 122
\@print@unsrt@glossary: new ..... 127	\glsxtr@resource@file: changed
General: added record package option ... 12	extension to .gls tex ..... 122
\glsadd: added \@glsxtr@record .... 59	\new@ignored@glossary: added starred
\glsdoifexists: now defines	version ..... 34
\glslabel ..... 37	1.12 (2017-02-03)
\glsxtr@do@wrglossary: new ..... 25	\@glsxtr@record@counter: new ..... 10
\glsxtr@add@loc@list@field: new ..... 11	\@gls@preglossary@hook: check for
\glsxtr@index@only@save@entry@counter:	definition ..... 171
new ..... 11	\@glsxtr@counter@record@hook: new . 125
\glsxtr@record: new ..... 125	\@glsxtr@display@loc: new ..... 115
\glsxtr@resource: new ..... 123	\@glsxtr@docounter@record: new ... 125
\glsxtr@save@entry@counter: new .... 25	\@glsxtr@long@new@glossary@entry:
\glsxtr@setup@record: new ..... 11	new ..... 33
\glsxtr@assign@field@font: added check	\@glsxtr@noop@record@counter: new . 11
for existence ..... 60	\@glsxtr@op@record@counter: new ... 11
\glsxtr@resource@file: new ..... 122	\@glsxtr@provide@storage@key: new . 26
\print@unsrt@glossaries: new ..... 127	\@glsxtr@s@long@new@glossary@entry:
\print@unsrt@glossary: new ..... 127	new ..... 33
1.09 (2016-12-16)	\@glsxtr@entry@fmt: new ..... 28
\@glsxtr@get@type: new ..... 111	\@glsxtr@index@aliased: new ..... 75
\@glsxtr@mixed@assign@sort@key:	\@glsxtr@set@alias@no@index: new .... 75
new ..... 111	\@new@glossary@entry@post@hook: added
\@print@glossary: redefined to save	check for alias key ..... 43
options ..... 110	\@no@glsxtr@index@aliased: new ..... 75
\glsxtr@make@glossaries: new ..... 111	\@print@unsrt@glossary: new ..... 127



General: added target key to printgloss family .....	110	\GlsXtrLoadResources: removed restriction on only one per document	122
\apptoglossarypreamble: new .....	32	\glxtrlocrangefmt: new .....	117
\csGlsXtrLetField: new .....	31	\glxtrpostlongdescription: new ..	34
\eGlsXtrSetField: new .....	32	\glxtrprovidestoragekey: new ....	26
\gGlsXtrSetField: new .....	31	\GlsXtrRecordCounter: new .....	125
\glsdohyperlink: added check for alias field .....	79	\glxtrresourcecount: new .....	122
\glsnoidxdisplayloc: added redefinition .....	115	\glxtrresourcefile: added catcode change for @ .....	122
\glissettoctitle: added patch .....	35	\glxtrsetaliasnoindex: new .....	75
\glxtr@counterrecord: new .....	125	\GlsXtrSetField: new .....	31
\glxtr@langtag: new .....	123	\glxtrsetfieldifexists: new ....	31
\glxtr@newabbreviation: new ....	178	\glxtrunsrtdo: new .....	130
\glxtr@org@newignoredglossary: Added check for existence .....	34	\Glsxtrusefield: new .....	30
\glxtr@pluralsuffixes: new .....	123	\glxtrusefield: new .....	30
\glxtr@provideignoredglossary: new .....	36	short-postlong-user: new .....	268
\glxtr@s@newignoredglossary: Added check for existence .....	35	short-postlong-user-desc: new ...	270
\glxtr@s@provideignoredglossary: new .....	36	\longnewglossaryentry: added starred version .....	33
\glxtrabbrvpluralsuffix: new ...	182	long-postshort-user: new .....	266
\glxtralias: new .....	43	long-postshort-user-desc: new ...	267
\glxtrcopytoglossary: new .....	37	postdot: new .....	15
\glxtrdeffield: new .....	31	\pretoglossarypreamble: new .....	32
\glxtrdisplayendloc: new .....	116	\print@noop@unsrtglossaryunit: new .....	130
\glxtrdisplayendlochook: new ...	116	\print@op@unsrtglossaryunit: new	129
\glxtrdisplaysingleloc: new ....	116	\printunsrtglossary: added starred form .....	127
\glxtrdisplaystartloc: new ....	116	\printunsrtglossaryhandler: new .	129
\glxtrdeffield: new .....	31	\printunsrtglossaryunit: new ....	11
\glxtrentryfmt: new .....	28	\printunsrtglossaryunitsetup: new	129
\glxtrfielddolistloop: new .....	29	\provideignoredglossary: new ....	36
\glxtrfieldforlistloop: new ....	29	\s@glxtr@provide@storagekey: new	26
\glxtrfielddifylist: new .....	29	\s@printunsrtglossary: new .....	127
\glxtrfieldlistadd: new .....	28	\xGlsXtrSetField: new .....	31
\glxtrfieldlistead: new .....	29	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new .....	29	\@glldisp: removed	
\glxtrfieldlistxadd: new .....	29	\@glxtr@org@glldisp .....	56
\glxtrfieldxifylist: new .....	29	\glxtrsetaliasnoindex: switched to \providecommand .....	75
\glxtrfmt: new .....	27	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new ....	27	\@glsl@link: added redefinition .....	57
\GlsXtrFmtField: new .....	27	\@glsl@noidx@getgrouptitle: new ..	113
\glxtrifkeydefined: new .....	26	\@glsl@removespaces: new .....	117
\glxtrindexaliased: new .....	76	\@glxtr@do@automake@err: new ...	124
\GlsXtrLetField: new .....	31	\@glxtr@org@gloautosee: new ....	23
\GlsXtrLetFieldToField: new .....	31	\@glxtr@record: added third arg ....	7
		\@glxtr@recordsee: new .....	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue .....	59	\@glo@autosee: added redefinition ....	24
added \glsadd option thevalue ....	59	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	79	bug .....	113
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@ .....	91	check for seealso field .....	44
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@ .....	99	instead of \csname .....	131
\glsnavigation: new .....	114	\@glsxtr@dorecordnodefer: new ....	10
\glsxtr@org@getgrouptitle: new ..	114	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new .....	7	misspelt command .....	128
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake .....	124	new .....	129
\glsxtrdisplayendloc: added check		General: added check for	
for empty format .....	116	\@gls@setupsort@none .....	13
\glsxtrgetgrouptitle: new .....	114	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new .....	56	redefinition .....	24
\glsxtrlocationhyperlink: new ...	117	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new .....	114	\providecommand lines .....	123
\glsxtrsupphypernumber: new ....	117	\@glsxtrautoindex: new .....	165
\ifglsxtrwrglossbefore: new .....	57	\@glsxtrautoindexentry: new .....	165
1.15 (2017-05-10)		\@glsxtrautoindexsort: new .....	165
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new .....	40
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new .....	43
short-em-long-em: fixed spelling of		\@glsxtrseelist: new .....	40
\glsabbrvfont .....	248	\@glsxtruseseealso: new .....	40
short-long: fixed spelling of		\@glsxtruseseealsoformat: new ....	40
\glsabbrvfont .....	200	\@seealso: new .....	40
short-long-user: fixed spelling of		autoseeindex: new .....	15
\glsabbrvfont .....	271	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new ....	177
\glsabbrvfont .....	268	\@glsxtr@markwordseps: new .....	177
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont .....	270	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag .....	112
\glsabbrvfont .....	245	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont .....	266	\glsxtrundeftag .....	113
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont .....	267	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag .....	112
\glsabbrvfont .....	198	\@glsxtrifhyphenstart: new .....	273
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont .....	265	in the abbreviation styles .....	198
footnote: fixed spelling of		\glsabbrvhyphenfont: new .....	274
\glsabbrvfont .....	203	\glsabbrvonlyfont: new .....	287
postfootnote: fixed spelling of		\glsabbrvscfont: new .....	214
\glsabbrvfont .....	204	\glsabbrvsmfont: new .....	228

\glsabbrvuserfont: initialised to default font .....	264	short-long-user-desc: corrected first forms .....	272
\glsfirstabbrvhyphenfont: new ...	274	short-nolong-desc-noreg: new ...	209
\glsfirstabbrvonlyfont: new .....	287	short-nolong-noreg: new .....	207
\glsfirstabbrvscfont: new .....	214	long-em-noshort-em-desc-noreg: new .....	260
\glsfirstabbrvsmfont: new .....	228	long-em-noshort-em-noreg: new ...	256
\glsfirstlonghyphenfont: new ...	274	long-hyphen-noshort-desc-noreg: new .....	276
\glsfirstlongonlyfont: new .....	287	long-hyphen-postshort-hyphen: new	279
\glslonghyphenfont: new .....	274	long-hyphen-postshort-hyphen-desc: new .....	281
\glslongonlyfont: new .....	287	long-hyphen-short-hyphen: new ...	274
\glslonguserfont: initialised to default font .....	264	long-hyphen-short-hyphen-desc: new .....	275
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong .....	178	long-noshort-desc-noreg: new ...	212
\GlsXtrDefineAcShortcuts: new ...	18	long-noshort-noreg: new .....	213
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside .....	192	long-only-short-only: new .....	287
\glsxtrhyphensuffix: new .....	274	long-only-short-only-desc: new ..	289
\glsxtrifhyphenstart: new .....	273	long-short-user-desc: corrected first forms .....	270
\glsxtrlonghyphen: new .....	278	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	275	stylemods: changed default value to "default" .....	21
\glsxtrlonghyphenshort: new ...	273	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	199	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new .....	289	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref .....	10
\glsxtronlydescsort: new .....	288	\@glsxtr@dorecordnodefer: Use \theglsentrycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new .....	287	\@glsxtr@record@setting: new .....	12
\glsxtrparen: new .....	180	\@glsxtr@record@setting@alsoindex: new .....	12
\glsxtrposthyphenlong: new .....	284	\@glsxtrifhasfield: new .....	30
\glsxtrposthyphenshort: new .....	278	General: added \glslink option theHvalue .....	57
\glsxtrposthyphensubsequent: new	279	added \glslink option thevalue ...	57
\glsxtrshortdescname: new .....	208	\glsxtr@writefields: removed double-quotes around \jobname ..	124
\glsxtrshorthyphen: new .....	283	\glsxtrdoautoindexname: changed format test .....	165
\glsxtrshorthyphenlong: new ...	281	\glsxtrhyperlink: new .....	79
\glsxtrshortlongdescname: new ...	201	\glsxtrifhasfield: new .....	30
\glsxtrshortlongdescsort: new ...	201	\GlsXtrSetDefaultNumberFormat: new .....	7
\Glsxtrsubsequentfmt: new .....	194	\s@glsxtrifhasfield: new .....	30
\glsxtrsubsequentfmt: new .....	194		
\Glsxtrsubsequentplfmt: new ...	194		
\glsxtrsubsequentplfmt: new .....	194		
\glsxtrword: new .....	177		
\glsxtrwordsep: new .....	177		
short-hyphen-long-hyphen: new ...	282		
short-hyphen-long-hyphen-desc: new .....	283		
short-hyphen-postlong-hyphen: new	284		
short-hyphen-postlong-hyphen-desc: new .....	286		



headings .....	341	\glstrbookindexbetween: new ....	346
redefined		\glstrbookindexbookmark: new ...	346
mcoltreenonamehypergroup to		\glstrbookindexcols: new .....	345
discourage breaks after group		\glstrbookindexcolspread: new ..	346
headings .....	341	\glstrbookindexfirstmark: new ..	350
redefined mcoltreenonamespannav to		\glstrbookindexfirstmarkfmt: new	350
discourage breaks after group		\glstrbookindexformatheader: new	346
headings .....	342	\glstrbookindexgroupskip: new ..	346
redefined mcoltreepannav to		\glstrbookindexlastmark: new ...	350
discourage breaks after group		\glstrbookindexlastmarkfmt: new	350
headings .....	340	\glstrbookindexmarkentry: new ..	349
redefined treegroup to discourage		\glstrbookindexname: new .....	345
breaks after group headings .....	324	\glstrbookindexparentchildsep:	
redefined treehypergroup to		new .....	345
discourage breaks after group		\glstrbookindexparentschildsep:	
headings .....	325	new .....	345
redefined treenonamegroup to		\glstrbookindexprelocation: new	345
discourage breaks after group		\glstrbookindexsubatendgroup:	
headings .....	326	new .....	346
redefined treenonamehypergroup to		\glstrbookindexsubbetween: new ..	346
discourage breaks after group		\glstrbookindexsubname: new ....	345
headings .....	326	\glstrbookindexsubprelocation:	
debug: new .....	22	new .....	345
\gglsetwidest: new .....	327	\glstrbookindexsubsubatendgroup:	
\gl:disablehyper: added check for		new .....	346
existence .....	79	\glstrbookindexsubsubbetween:	
changed to use \def rather than \let .	79	new .....	346
\gl:enablehyper: changed to use \def		\glstrbookindexthepage: new ....	349
rather than \let .....	80	\glstrdetoklocation: new .....	135
\gl:fmtname: new .....	304	\glxtrenablerecordcount: new ...	135
\gl:fmtname: new .....	303	\glstrglossentry: new .....	125
\gl:shex: new .....	132	\glstrgroupfield: new .....	130
\gl:listchildpostlocation: new ..	312	\glstrheadname: new .....	295
\gl:listchildprelocation: new ...	312	\glstrheadname: new .....	295
\gl:listprelocation: new .....	312	\GlsXtrIfFieldEqStr: new .....	32
\gl:navhyperlink: patched .....	78	\glstriflabelinlist: new .....	129
\gl:seeitemformat: new .....	39	\glstrifrecordtrigger: new ....	135
\gl:showtarget: new .....	23	\glstrindexseealso: added check	
\gl:treechildprelocation: new ...	322	that the entry exists .....	41
\gl:treeprelocation: new .....	322	\glstrinithyperoutside: new ....	57
\gl:triggerrecordformat: new ....	136	\GlsXtrLocationRecordCount: new ..	135
\gl:useabbrvfont: new .....	192	\glstrnewgls: new .....	132, 133
\gl:uselongfont: new .....	192	\glstrnewGLSlike: new .....	134
\glstr@do@alsoindex@wrglossary:		\glstrnewglslike: new .....	134
new .....	8	\glstrnewrgls: new .....	134
\glstr@org@@do@wrglossary: new ..	25	\glstrnewrGLSlike: new .....	134
\glstr@org@dohyperlink: new ....	77	\glstrnewrglslike: new .....	134
\glstr@setbookindexmark: new ...	350	\glstrprelocation: new ....	311, 345
\glstrbookindexatendgroup: new ..	346	\GlsXtrRecordCount: new .....	134

\glxtrrecordtriggervalue: new ..	135	\@glxtr@orgprintglossary: changed	
\glxtrresourcefile: now disables		explicit \let for \nopostdesc to	
record key .....	122	\glxtractivatenopost .....	109
\glxtrresourceinit: new .....	132	\@glxtrglossentryother: new ....	126
\GlsXtrSetRecordCountAttribute:		\glossentrynameother: new .....	163
new .....	135	\glseeitemformat: switched check	
\glxtrtitlename: new .....	295	from regular to short .....	39
\glxtrtitleorpdforheading: new ..	291	\glxtr@setaccessdisplay: new ...	162
\GlsXtrTotalRecordCount: new ....	134	\glxtr@writefields: provide	
\glxtrwrglossmark: new .....	22	\glxtr@record in aux file .....	123
short-em: new .....	250	\glxtractivatenopost: new .....	109
short-sc: corrected first letter		\glxtrbookindexprelocation:	
uppercasing .....	218	removed check for no post dot ....	345
short-sm: corrected first letter		\glxtrglossentryother: new .....	126
uppercasing .....	233	\glxtrnopostpunc: new .....	110
\ifglxtr@hyperoutside: new .....	57	1.23 (2017-11-12)	
all: new .....	310	\@glxtrfmt: added check for indexing	28
nolong-short: new .....	210	added grouping .....	27
nolong-short-em: new .....	252	new .....	27
nolong-short-noreg: new .....	210	\@glxtr@nopostpunc@postdesc: new	110
nolong-short-sc: new .....	220	\@glxtr@restore@postpunc: new ..	110
nolong-short-sm: new .....	235	\@glxtrentryfmt: fixed missing label	
nopostdot: new .....	15	argument .....	28
\printunsrtglossaryentryprocesshook:		\@glxtrfmt: new .....	27
new .....	129	\eglsupdatewidest: new .....	328
\printunsrtglossarypredoglossary:		\gglupdatewidest: new .....	328
new .....	129	\glsupdatewidest: new .....	328
\rGLS: new .....	138	\GlsXtrDefineAbbreviationShortcuts:	
\rGls: new .....	137	changed \newabbr definition to use	
\rgls: new .....	137	\providecommand .....	17
\rGLSformat: new .....	140	\GlsXtrDefineAcShortcuts: changed	
\rGlsformat: new .....	139	\newabbr definition to use	
\rglsformat: new .....	139	\providecommand .....	18
\rGLSpl: new .....	139	\glxtrfmtdisplay: new .....	28
\rGlspl: new .....	138	\glxtrifcustomdiscardperiod: new	172
\rglspl: new .....	137	\GlsXtrIfFieldUndef: new .....	30
\rGLSplformat: new .....	140	\glxtrrestorepostpunc: new .....	110
\rglsplformat: new .....	139	\s@glxtrfmt: new .....	27
\rglsplformat: new .....	139	\s@glxtrfmt: new .....	27
\s@glxtrifhasfield: switched from		\xglsupdatewidest: new .....	328
\ifdef to \ifundef .....	30	1.24 (2017-11-14)	
1.22 (2017-11-08)		\glsadd: added \@gls@setsort .....	60
\@glxtr@nopostpunc: new .....	110	\glxtrforcsvfield: new .....	29
		\glxtrlocalsetgrouptitle: new ..	114

# 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	
\.	15, 16, 173, 322
\@	122
\@cGLS@	91, 100
\@cGLSpl@	91, 100
\@cGls@	91, 99
\@cGlspl@	91, 100
\@cgls@	91, 99
\@cglspl@	91, 99
\@do@wrglossary	8, 9, 106
\@do@wrglossary	11, 13, 25, 60, 75
\@glo@assign@sortkey	111
\@glo@list	6
\@glo@type	127
\@glossarysec	346
\@gls@expand@field	26
\@glslocalreset	89
\@glslocalunset	89
\@glsreset	89
\@glsunset	89
\@glsxtr@autoindex@escspch	167, 168
\@glsxtr@checkspch	166, 168, 169
\@glsxtr@disabledflycommand	49
\@glsxtr@org@postdescription	110
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 125
\@glsxtrfmt	27
\@glsxtrp	84, 85
\@glsxtrpostloctag	52
\@glsxtrpreloctag	52
\@glsxtrwrglossmark	8, 11, 23, 25, 41, 45, 106
\@newglossaryentry@defcounters	90
\@newglossaryentry@defunitcounters	98
\@par	327
\@ACRlong	81
\@ACRlongpl	81
\@ACRshort	81
\@ACRshortpl	81
\@Acrlong	81
\@Acrlongpl	81
\@Acrshort	81
\@Acrshortpl	81
\@GLS@	80, 94, 95, 138
\@GLSdesc@	64
\@GLSpl@	80, 94, 95, 139
\@GLSplural@	81
\@GLSsymbol@	65
\@GLStext@	81
\@GLSxtr@full	183
\@GLSxtr@fullpl	185
\@GLSxtr@p@acrlong@	81
\@GLSxtr@p@acrlongpl@	81
\@GLSxtr@p@acrshort@	81
\@GLSxtr@p@acrshortpl@	81
\@GLSxtr@p@long@	80
\@GLSxtr@p@longpl@	80
\@GLSxtr@p@plural@	80
\@GLSxtr@p@short@	80
\@GLSxtr@p@shortpl@	80
\@GLSxtr@p@text@	80
\@GLSxtrlong	80, 188
\@GLSxtrlongpl	80, 191
\@GLSxtrp	88
\@GLSxtrshort	80, 186
\@GLSxtrshortpl	80, 189
\@Gls@	80, 93, 94, 138
\@Gls@acentryname	102
\@Gls@entry@field	72, 87, 163
\@Gls@entryname	102
\@GlsXtrEnableOnTheFly	46, 47
\@Glspl@	80, 93, 94, 138
\@Glsplural@	81
\@Glstext@	81
\@Glsxtr	48, 49

\@Glsxtr@full .....	183	\@firstofone .....	60, 128, 157, 158, 164, 170
\@Glsxtr@fullpl .....	184	\@firstofthree .....	56,
\@Glsxtr@p@acrlong@ .....	81	60, 68–71, 77, 182, 184, 185, 187, 189, 190	
\@Glsxtr@p@acrlongpl@ .....	81	\@firstoftwo 61–65, 69, 71, 74, 77, 104, 162,	
\@Glsxtr@p@acrshort@ .....	81	163, 174, 175, 183–185, 189–191, 291, 292	
\@Glsxtr@p@acrshortpl@ .....	81	\@for .....	6, 21, 30, 44, 90,
\@Glsxtr@p@long@ .....	80	102, 105, 108, 115, 128, 135, 156, 162, 170	
\@Glsxtr@p@longpl@ .....	80	\@glo@alias .....	41–43
\@Glsxtr@p@plural@ .....	80	\@glo@assign@sortkey .....	108
\@Glsxtr@p@short@ .....	80	\@glo@autosee .....	23
\@Glsxtr@p@shortpl@ .....	80	\@glo@autoseehook .....	42
\@Glsxtr@p@text@ .....	80	\@glo@category .....	96
\@Glsxtrlong .....	80, 187	\@glo@check@sortallowed .....	108
\@Glsxtrlongpl .....	80, 191	\@glo@counterprefix .....	10, 117
\@Glsxtrp .....	87	\@glo@countunit .....	96
\@Glsxtrpl .....	48, 49	\@glo@default@sorttype .....	108
\@Glsxtrshort .....	80, 186	\@glo@desc .....	33, 34
\@Glsxtrshortpl .....	80, 189	\@glo@descplural .....	33, 34
\@acrlong .....	81	\@glo@group .....	12
\@acrlongpl .....	81	\@glo@label .....	
\@acrshort .....	81	..... 11, 12, 26, 39, 42–44, 72, 79, 329–336	
\@acrshortpl .....	81	\@glo@location .....	12
\@afterheading .....		\@glo@loclist .....	11
..... 313, 314, 323–326, 339–342, 349		\@glo@name .....	165
\@alt@glshyp@opt .....	77	\@glo@no@assign@sortkey .....	111
\@auxout .....	10, 11,	\@glo@parent .....	331, 332
45, 53, 92, 100, 105, 106, 118, 122–125, 349		\@glo@see .....	39, 40, 42–44
\@bibgl@restoreat .....	122	\@glo@seealso .....	41, 42
\@cGLS .....	94	\@glo@sort .....	165
\@cGLS@ .....	91, 94, 100	\@glo@sorttype .....	108, 115
\@cGLSpl .....	95	\@glo@text .....	56
\@cGLSpl@ .....	91, 95, 100	\@glo@thislettergrp .....	131
\@cGls@ .....	91, 99	\@glo@thisvalue .....	264
\@cGlspl@ .....	91, 100	\@glo@tmp .....	26, 40, 72
\@cgls@ .....	91, 99	\@glo@type .....	44, 78, 102, 105,
\@cglspl@ .....	91, 99	108, 109, 111, 114, 115, 118, 121, 122, 128	
\@disable@onlypremakeg .....	106	\@glo@types .....	154, 329–335
\@do@auxoutstuff .....	118	\@glossary@default@style .....	50, 109, 344
\@do@gl@getcounterprefix .....	10	\@glossarystyle .....	109
\@do@gl@see .....	42, 43	\@gl@ .....	80, 93, 94, 137
\@do@newglossaryentry .. 102, 103, 179, 180		\@gl@link .....	56
\@do@seeglossary .....	13, 23, 45, 106	\@gl@ReturnAfterFi .....	117
\@do@wrglossary .....	58, 59, 136	\@gl@actualchar .....	166
\@empty .....	60, 68–71, 110, 166, 182–191	\@gl@adjustmode .....	59
\@end@glxtr@addunused .....	44	\@gl@alt@hyp@opt .....	77
\@end@glxtr@gettype .....	108, 111	\@gl@alt@hyp@opt@char .....	77
\@end@glxtr@usese .....	39	\@gl@alt@hyp@opt@keys .....	77
\@end@glxtrifhyphenstart .....	273	\@gl@automake .....	108
\@endfortrue .....	29, 162, 195	\@gl@between .....	114, 115



\@gls@checkedmkidx .....	166, 168	\@gls@noidxloclist@sep .....	112
\@gls@checkmkidxchars .....	41, 165	\@gls@noref@warn .....	106, 115
\@gls@codepage .....	118	\@gls@org@glsnoidxdisplayloc ..	112, 113
\@gls@counter .....	9, 10, 57, 59, 75, 136	\@gls@org@glsseeformat .....	112, 113
\@gls@currentlettergroup ...	115, 128, 131	\@gls@preglossaryhook .....	109, 170
\@gls@declareoption .....	5	\@gls@prevlevel .....	337, 338, 342, 343
\@gls@default@longpl .....	178, 179	\@gls@quotechar .....	166
\@gls@doautomake .....	108, 124	\@gls@reference .....	45, 105, 106
\@gls@doautomake@err .....	124	\@gls@saveentrycounter .	13, 25, 58, 59, 136
\@gls@encapchar .....	166	\@gls@see@noindex .....	24, 122
\@gls@entry@count .....	91, 92	\@gls@setdefault@glslink@opts .....	9, 28, 58, 76
\@gls@entry@field .....	26, 30, 42, 72, 85–88, 91, 126, 127	\@gls@setsort .....	58, 60
\@gls@entry@unitcount .....	100	\@gls@setupsort@none .....	13
\@gls@field@font .....	60–67	\@gls@short .....	178, 179
\@gls@field@link .....	60–67, 72, 73	\@gls@shortpl .....	176, 179
\@gls@getcounterprefix .....	10	\@gls@sort .....	131
\@gls@getgrouptitle .....	114, 128	\@gls@thisval .....	162
\@gls@grptitle .....	78, 115	\@gls@tmp .....	115
\@gls@hyp@opt .....	72, 73, 77, 94, 95, 133, 137–139, 182–191	\@gls@tmpb .....	168
\@gls@hyp@opt@cs .....	77	\@gls@type .....	106, 108, 195, 329–336
\@gls@ifinlist .....	129	\@gls@write@entrycounts .....	91
\@gls@increment@currcount .....	91	\@gls@write@entryunitcounts .....	100
\@gls@increment@currunitcount .....	99	\@gls@write@entryunitcounts@do ...	101
\@gls@keymap ..	11, 12, 26, 39, 42, 72, 123, 162	\@gls@xref .....	11, 41
\@gls@label ...	8–10, 45, 76, 77, 106, 125, 195	\@gls@abbrv@current@abbreviation	178, 192
\@gls@levelchar .....	166	\@gls@acronymlists .....	102
\@gls@link .....	27, 55, 56, 68–72, 182–191	\@gls@doifexistsorwarn	14, 158, 159, 161, 163
\@gls@link@checkfirsthyper .....	56, 104	\@gls@entry .....	92, 100, 101
\@gls@link@label .....	57, 136	\@gls@link .....	59, 78–80
\@gls@link@nocheckfirsthyper .....	55, 68–71, 182–191	\@gls@nextpages .....	109
\@gls@link@opts .....	57	\@gls@nonextpages .....	109
\@gls@list .....	114, 115	\@gls@numberformat .....	9, 10, 57, 59, 75, 136, 162, 165
\@gls@local@increment@currcount ....	91	\@gls@order .....	105
\@gls@local@increment@currunitcount	99	\@gls@pl@ .....	80, 93, 94, 137
\@gls@location .....	131, 132	\@gls@plural@ .....	81
\@gls@loclist .....	112, 113, 131, 132	\@gls@punc@token .....	175
\@gls@long .....	178	\@gls@recordlocref .....	10
\@gls@longpl .....	176, 178, 179	\@gls@showtarget .....	79
\@gls@map .....	162	\@gls@style@altlist .....	312
\@gls@nohyperlist .....	34, 36	\@gls@style@altlistgroup .....	313
\@gls@noidx@do .....	115	\@gls@style@altlisthypergroup .....	314
\@gls@noidx@getgrouptitle .....	128	\@gls@style@alttree .....	326
\@gls@noidx@nosanitizesort .....	107	\@gls@style@alttreegroup .....	338
\@gls@noidx@sanitizesort .....	107	\@gls@style@alttreehypergroup .....	338
\@gls@noidxloclist@finalsep .....	112	\@gls@style@index .....	322
\@gls@noidxloclist@prev .....	112	\@gls@style@indexgroup .....	323
		\@gls@style@indexhypergroup .....	323

<code>\@glsstyle@inline</code> .....	322	<code>\@glsxtr@autoseeindexfalse</code> .....	13
<code>\@glsstyle@list</code> .....	312	<code>\@glsxtr@autoseeindextrue</code> .....	15
<code>\@glsstyle@listdotted</code> .....	311	<code>\@glsxtr@bookindex@atendgroup</code> ..	347, 349
<code>\@glsstyle@listgroup</code> .....	313	<code>\@glsxtr@bookindex@atsubendgroup</code> ..	348
<code>\@glsstyle@listhypergroup</code> .....	313	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	348
<code>\@glsstyle@mcolalttree</code> .....	342	<code>\@glsxtr@bookindex@between</code> ....	347, 349
<code>\@glsstyle@mcolalttreegroup</code> .....	342	<code>\@glsxtr@bookindex@sep</code> .....	347, 348
<code>\@glsstyle@mcolalttreehypergroup</code> ..	343	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcolalttreespannav</code> ....	343	.....	347, 349
<code>\@glsstyle@mcolindexgroup</code> .....	339	<code>\@glsxtr@bookindex@subbetween</code> ..	347, 348
<code>\@glsstyle@mcolindexhypergroup</code> ....	339	<code>\@glsxtr@bookindex@subsep</code> ....	347, 348
<code>\@glsstyle@mcolindexspannav</code> ....	339	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcoltreegroup</code> .....	340	.....	347, 349
<code>\@glsstyle@mcoltreehypergroup</code> ....	340	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcoltreenamegroup</code> ....	341	.....	347, 348
<code>\@glsstyle@mcoltreenamehypergroup</code>	341	<code>\@glsxtr@bookindexgroupskip</code> ...	347, 349
<code>\@glsstyle@mcoltreename<span>spannav</span></code> ..	342	<code>\@glsxtr@cat</code> .....	90, 102, 135, 170
<code>\@glsstyle@mcoltreespannav</code> .....	340	<code>\@glsxtr@checkgroup</code> .....	128
<code>\@glsstyle@tree</code> .....	324	<code>\@glsxtr@counterrecordhook</code> .....	10, 11
<code>\@glsstyle@treegroup</code> .....	324	<code>\@glsxtr@csname</code> .....	97, 99
<code>\@glsstyle@treehypergroup</code> .....	325	<code>\@glsxtr@current@style</code> .....	50, 344
<code>\@glsstyle@treename</code> .....	325	<code>\@glsxtr@currentunitcount</code> .....	97, 99
<code>\@glsstyle@treenamegroup</code> .....	326	<code>\@glsxtr@currunitcount</code> .....	98, 100
<code>\@glsstyle@treenamehypergroup</code> ...	326	<code>\@glsxtr@declareoption</code> .....	5, 15, 17, 20
<code>\@glstarget</code> .....	80, 111	<code>\@glsxtr@defaultnoglossarywarning</code> ..	20
<code>\@glstext@</code> .....	81	<code>\@glsxtr@defaultnumberformat</code> .....	
<code>\@glswidestname</code> .....	329, 336	.....	7, 9, 57, 59, 75, 162, 165
<code>\@glsxtr</code> .....	47, 49	<code>\@glsxtr@defpostpunc</code> .....	15, 16, 23
<code>\@glsxtr@do@wrglossary</code> .....	106	<code>\@glsxtr@deprecated@abbrstyle</code> ....	
<code>\@glsxtr@abbreviationsdef</code> .....	17, 24	.....	223, 224, 226,
<code>\@glsxtr@accessdisplay</code> .....	162–164	228, 237, 239, 240, 242, 255, 258, 262, 264	
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@disabledflycommand</code> .....	49
.....	170, 171	<code>\@glsxtr@display@loc</code> .....	115
<code>\@glsxtr@addunitcounter</code> .....	96	<code>\@glsxtr@do@windex</code> .....	76
<code>\@glsxtr@addunused</code> .....	44	<code>\@glsxtr@do@gl:disablehyperinlist</code> ..	74
<code>\@glsxtr@addunusedxrefs</code> .....	44	<code>\@glsxtr@do@record@wrglossary</code> ....	8, 13
<code>\@glsxtr@attrval</code> .....		<code>\@glsxtr@do@redef@for@gl@sentries</code> ....	7
.....	58, 157, 158, 160, 161, 163, 165	<code>\@glsxtr@do@style</code> .....	21, 309
<code>\@glsxtr@autoindex@at</code> .....	165–167	<code>\@glsxtr@do@titlecaps@warn</code> .....	
<code>\@glsxtr@autoindex@doextra@esc</code> ....	165	.....	157–160, 163, 170
<code>\@glsxtr@autoindex@encap</code> .....	165–167	<code>\@glsxtr@doabbreviationsdef</code> .....	17
<code>\@glsxtr@autoindex@esc</code> .....	166–168	<code>\@glsxtr@doaccsupp</code> .....	20, 22
<code>\@glsxtr@autoindex@escat</code> .....	166, 167	<code>\@glsxtr@docdefval</code> .....	14, 46
<code>\@glsxtr@autoindex@escencap</code> ...	166, 167	<code>\@glsxtr@docounterrecord</code> .....	11
<code>\@glsxtr@autoindex@esclevel</code> ...	166, 167	<code>\@glsxtr@doglossary</code> .....	128
<code>\@glsxtr@autoindex@escquote</code> ...	166–168	<code>\@glsxtr@doiflabelinlist</code> .....	129
<code>\@glsxtr@autoindex@level</code> .....	166, 167	<code>\@glsxtr@doloctag</code> .....	52, 53
<code>\@glsxtr@autoindex@setname</code> .....	165	<code>\@glsxtr@dorecord</code> .....	8, 9
<code>\@glsxtr@autoindexcrossrefs</code> 13, 14, 39, 42		<code>\@glsxtr@dorecordnodefer</code> .....	8, 9

\@glsxtr@dosee@alsoindex@glossary ..	13	\@glsxtr@nopostpunc@postdesc .....	110
\@glsxtr@doseeglossary .....	13, 23	\@glsxtr@notfoundinlist .....	175
\@glsxtr@dostylewarn .....	195	\@glsxtr@op@recordcounter .....	13
\@glsxtr@enabletagging .....	169	\@glsxtr@optlist .....	49
\@glsxtr@end@ .....	47	\@glsxtr@org@@starttoc .....	290, 291
\@glsxtr@endescspch .....	166–168	\@glsxtr@org@GLS@ .....	55
\@glsxtr@entrycount@org@localreset ..	91	\@glsxtr@org@GLSpl@ .....	56
\@glsxtr@entrycount@org@localunset ..	91	\@glsxtr@org@Gls@ .....	55
\@glsxtr@entrycount@org@reset .....	91	\@glsxtr@org@Glspl@ .....	55
\@glsxtr@entrycount@org@unset .....	91	\@glsxtr@org@Glsxtrtitlefirst ..	291, 293
\@glsxtr@entryunitcount@org@localreset .....	99	\@glsxtr@org@Glsxtrtitlefirstplural .....	291, 293
\@glsxtr@entryunitcount@org@localunset .....	99	\@glsxtr@org@Glsxtrtitlefull ..	292, 293
\@glsxtr@entryunitcount@org@reset ..	99	\@glsxtr@org@Glsxtrtitlefullpl ..	292, 293
\@glsxtr@entryunitcount@org@unset ..	99	\@glsxtr@org@Glsxtrtitlelong ..	292, 293
\@glsxtr@err@undefaction .....	7, 13	\@glsxtr@org@Glsxtrtitlelongpl ..	292, 293
\@glsxtr@field@linkdefs .....	55	\@glsxtr@org@Glsxtrtitlename ..	291, 292
\@glsxtr@format@overridefalse .....	164	\@glsxtr@org@Glsxtrtitleplural ..	291, 293
\@glsxtr@format@overridefalse .....	164	\@glsxtr@org@Glsxtrtitleshort ..	291, 292
\@glsxtr@foundinlist .....	175	\@glsxtr@org@Glsxtrtitleshortpl ..	291, 292
\@glsxtr@full .....	182	\@glsxtr@org@Glsxtrtitletext ..	291, 293
\@glsxtr@fullpl .....	184	\@glsxtr@org@MakeUppercase ....	291, 292
\@glsxtr@gettype .....	108	\@glsxtr@org@checkfirsthyper ...	74, 104
\@glsxtr@glossdescfont .....	157, 158	\@glsxtr@org@delimN .....	52, 53
\@glsxtr@glossnamefont .....	158–164	\@glsxtr@org@delimR .....	52, 53
\@glsxtr@gobbleto@endescspch .....	168	\@glsxtr@org@doseeglossary .....	23, 106
\@glsxtr@groupheading .....	128, 131	\@glsxtr@org@gloautosee .....	24
\@glsxtr@idx@displaynumberlist ....	107	\@glsxtr@org@gls@ .....	55
\@glsxtr@idx@entrynumberlist .....	107	\@glsxtr@org@glsdohypertarget ....	111
\@glsxtr@ifcsstart .....	47	\@glsxtr@org@glsignore .....	52, 53
\@glsxtr@ifpunctoken .....	175	\@glsxtr@org@glspl@ .....	55
\@glsxtr@ifunitcounter .....	96	\@glsxtr@org@glsxtrtitlefirst ..	291, 293
\@glsxtr@insert@dots .....	176	\@glsxtr@org@glsxtrtitlefirstplural .....	291, 293
\@glsxtr@insert@dots@next .....	177	\@glsxtr@org@glsxtrtitlefull ..	292, 293
\@glsxtr@insertdots .....	178	\@glsxtr@org@glsxtrtitlefullpl ..	292, 293
\@glsxtr@label .....	30, 44, 156	\@glsxtr@org@glsxtrtitlelong ..	291, 293
\@glsxtr@loadstyles .....	310, 311	\@glsxtr@org@glsxtrtitlelongpl ..	291, 293
\@glsxtr@longnewglossaryentry .....	33	\@glsxtr@org@glsxtrtitlename ..	291, 292
\@glsxtr@mark@wordseps .....	177	\@glsxtr@org@glsxtrtitleorpdforheading .....	291, 292
\@glsxtr@mark@wordseps@next .....	177	\@glsxtr@org@glsxtrtitleplural ..	291, 293
\@glsxtr@markwordseps .....	178, 179	\@glsxtr@org@glsxtrtitleshort ..	291, 292
\@glsxtr@mixed@assign@sortkey .....	108	\@glsxtr@org@glsxtrtitleshortpl ..	291, 292
\@glsxtr@noidx@displaynumberlist ..	107	\@glsxtr@org@glsxtrtitletext ..	291, 293
\@glsxtr@noidx@do .....	130	\@glsxtr@org@makeglossaries .....	105
\@glsxtr@noidx@entrynumberlist ....	107	\@glsxtr@org@markboth .....	290, 291
\@glsxtr@noidx@numberlistloop .....	107	\@glsxtr@org@markright .....	290
\@glsxtr@noop@recordcounter .....	10, 13	\@glsxtr@org@newacronymstyle ..	103, 104
\@glsxtr@nopostpunc .....	109		

<code>\@glxtr@org@postdescription</code> ..	110, 171	<code>\@glxtr@thevalue</code> .....	8, 9, 57–59, 136
<code>\@glxtr@org@see@noindex</code> .....	122	<code>\@glxtr@thisloctag</code> .....	52, 53
<code>\@glxtr@org@setacronymstyle</code> ..	103, 104	<code>\@glxtr@titlelabel</code> .....	113, 114, 130
<code>\@glxtr@org@theHvalue</code> .....	8, 9	<code>\@glxtr@tmp</code> .....	21, 116
<code>\@glxtr@orgprefix</code> .....	10	<code>\@glxtr@type</code> .....	156
<code>\@glxtr@orgprintglossary</code> .....	49, 110	<code>\@glxtr@unitcountlist</code> .....	96
<code>\@glxtr@orgwarndep</code> .....	176	<code>\@glxtr@unsrt@getgrouptitle</code> .....	128
<code>\@glxtr@p@acrlong@</code> .....	81	<code>\@glxtr@usesee</code> .....	39
<code>\@glxtr@p@acrlongpl@</code> .....	81	<code>\@glxtr@warn@onexistsordo</code> .....	7, 13
<code>\@glxtr@p@acrshort@</code> .....	81	<code>\@glxtr@warn@undefaction</code> .....	7, 13
<code>\@glxtr@p@acrshortpl@</code> .....	81	<code>\@glxtr@docdeffalse</code> .....	45
<code>\@glxtr@p@long@</code> .....	80	<code>\@glxtr@entryfmt</code> .....	28
<code>\@glxtr@p@longpl@</code> .....	80	<code>\@glxtr@fmt</code> .....	27
<code>\@glxtr@p@plural@</code> .....	80	<code>\@glxtr@glossentry</code> .....	125
<code>\@glxtr@p@short@</code> .....	80	<code>\@glxtr@glossentryother</code> .....	126
<code>\@glxtr@p@shortpl@</code> .....	80	<code>\@glxtr@hypernameprefix</code> .....	111, 130
<code>\@glxtr@p@text@</code> .....	80	<code>\@glxtr@trifhasfield</code> .....	29, 30
<code>\@glxtr@pagestag</code> .....	52	<code>\@glxtr@trifhyphenstart</code> .....	273
<code>\@glxtr@pagetag</code> .....	52	<code>\@glxtr@indexaliased</code> .....	75
<code>\@glxtr@prevunitcount</code> .....	98	<code>\@glxtr@indexcrossrefsfalse</code> .....	14
<code>\@glxtr@printglossopts</code> .....	49, 108, 110	<code>\@glxtr@indexcrossrefstrue</code> .....	15
<code>\@glxtr@printunsrtglossaryskipentry</code> .....	128, 129	<code>\@glxtr@inmark</code> .....	290
<code>\@glxtr@provide@addstoragekey</code> .....	27	<code>\@glxtr@long</code> .....	80, 187
<code>\@glxtr@provide@storagekey</code> .....	26	<code>\@glxtr@longpl</code> .....	80, 190
<code>\@glxtr@record</code> .....	13, 54–56, 59	<code>\@glxtr@newgls</code> .....	133, 134
<code>\@glxtr@record@setting</code> .....	8, 9, 12, 41, 45, 46, 105	<code>\@glxtr@newgls@inner</code> .....	132, 133
<code>\@glxtr@record@setting@alsoindex</code> .....	8, 9, 41, 105	<code>\@glxtr@newgls@innercsname</code> .....	133
<code>\@glxtr@record@setting@off</code> .....	45	<code>\@glxtr@notinmark</code> .....	290
<code>\@glxtr@record@setting@only</code> .....	105	<code>\@glxtr@p</code> .....	86
<code>\@glxtr@recordsee</code> .....	13, 23, 41	<code>\@glxtr@p@opt</code> .....	84
<code>\@glxtr@redef@forglsentries</code> .....	7, 24	<code>\@glxtr@pl</code> .....	48, 49
<code>\@glxtr@redefstyles</code> .....	21, 309	<code>\@glxtr@postloctag</code> .....	51, 52, 54
<code>\@glxtr@reg@glosslist</code> .....	105–108, 111	<code>\@glxtr@preloctag</code> .....	51, 52, 54
<code>\@glxtr@restore@postpunc</code> .....	110	<code>\@glxtr@setaliasnoindex</code> .....	75, 76
<code>\@glxtr@rglstrigger@record</code> ...	137–139	<code>\@glxtr@short</code> .....	80, 185
<code>\@glxtr@s@longnewglossaryentry</code> ....	33	<code>\@glxtr@shortpl</code> .....	80, 188
<code>\@glxtr@savepreloctag</code> .....	52, 53	<code>\@glxtr@undeftag</code> .....	6, 25
<code>\@glxtr@setentrycountunsetattr</code> ....	90	<code>\@glxtr@wrglossmark</code> .....	22
<code>\@glxtr@setentryunitcountunsetattr</code>	101	<code>\@gobble</code> ..	7, 13, 15, 60, 129, 130, 177, 347–349
<code>\@glxtr@setupshortcuts</code> .....	19, 20, 24	<code>\@gobbletwo</code> .....	176
<code>\@glxtr@shortcutsval</code> .....	19, 124	<code>\@ifnextchar</code> .....	77
<code>\@glxtr@swaptwo</code> .....	175	<code>\@ifpackageloaded</code> .....	5, 16, 124, 140, 156, 158, 161, 162, 164, 309
<code>\@glxtr@tag</code> .....	170	<code>\@ifstar</code> ..	26, 27, 30, 33, 34, 36, 46, 77, 127, 169
<code>\@glxtr@taggingcs</code> .....	170	<code>\@ifundefined</code> .....	309
<code>\@glxtr@textformat</code> .....	58, 59	<code>\@ignored@glossaries</code> .....	34–36
<code>\@glxtr@theHvalue</code> .....	8, 9, 57–59, 136	<code>\@input</code> .....	122
		<code>\@input@</code> .....	118
		<code>\@istfilename</code> .....	105



\Aclp .....	18	\AtEndDocument .....	43, 91, 100, 118
\aclp .....	18		
\ACP .....	18	<b>B</b>	
\Acp .....	18	babel package .....	165, 166, 174
\acp .....	18	\begin .....	15, 16, 115, 119, 120, 128, 312, 314–321, 340–343, 347
\ACRfullfmt .....	103	\beginngroup .....	8, 9, 27, 28, 75, 126, 127
\Acrfullfmt .....	103	\bgroup .....	33, 109
\acrfullfmt .....	103	bib2gls .....	3, 28, 130, 132, 136, 137
\ACRfullplfmt .....	103		
\Acrfullplfmt .....	103	<b>C</b>	
\acrfullplfmt .....	103	\catcode .....	122
\acronymentry .....	102	category attributes:	
\acronymfont .....	68–71, 83, 103, 104	apospplural .....	179
\acronymname .....	16	discardperiod .....	173
\acronymsort .....	102	entrycount .....	89, 90, 92, 101
\acronymtype .....	17, 102, 103	firstuc .....	160
\acrpluralsuffix .....	102, 124	glossdesc .....	156
\ACS .....	18	glossdescfont .....	157
\Acs .....	18	glossname .....	158
\acs .....	18	glossnamefont .....	158, 161
\ACSP .....	18	headuc .....	293
\Acsp .....	18	indexname .....	165
\acsp .....	18	indexonlyfirst .....	76
\actualchar .....	168	insertdots .....	178
\addtolength .....	337	markshortwords .....	178
\advance .....	92, 101, 123, 133	markwords .....	178, 179, 273, 274, 282
\AF .....	17	nohyper .....	74
\Af .....	17	nohyperfirst .....	61–63
\af .....	17	noshortplural .....	179
\AFP .....	17	regular .....	54, 95, 198, 200, 202–204, 207–210, 212, 213, 216, 217, 219, 220, 222, 224– 226, 230, 232–234, 237–239, 241, 244– 250, 252, 254, 256, 258–260, 262, 265, 271, 272, 274–276, 278, 282, 283, 287, 289
\Afp .....	17	textformat .....	58
\afp .....	17	\cdot .....	22
\AL .....	17	\centering .....	346
\Al .....	17	\cGLS .....	17, 18, 90, 101
\alp .....	17	\cGls .....	17, 18, 90, 101
\AnyTrackedLanguages .....	309	\cglS .....	17, 18, 90, 101
\appto .....	11, 12, 21, 26, 39–43, 72, 76, 90, 98, 128, 130, 164, 174, 177, 310	\cGLSformat .....	94
\arabic .....	349	\cGlsformat .....	93
\AS .....	17	\cglSformat .....	93, 95
\As .....	17	\cGLSpl .....	17, 18, 90, 101
\as .....	17	\cGlspl .....	17, 18, 90, 101
\ASP .....	17	\cglSpl .....	17, 18, 90, 101
\Asp .....	17	\cGLSplformat .....	94
\asp .....	17	\cGlsplformat .....	93
\AtBeginDocument .....	22, 25, 50, 51, 124	\cglSplformat .....	93, 95





entry categories:		
abbreviation	192	
general	151, 153	
index	154	
\epreto	165	
\equal	121	
etoolbox package	5	
\expandafter		
.....	22, 26, 27, 30, 39, 40, 44, 47–49,	
.....	72, 73, 77, 84, 95, 96, 106–108, 111, 114,	
.....	116, 117, 128, 131, 133, 140, 156, 159,	
.....	160, 162, 164, 165, 168, 175, 178, 179, 273	
\expandonce	102, 131, 166, 199, 276	
<b>F</b>		
\fi	7–11, 13–15, 17, 19, 20, 22–24, 28, 32, 39,	
	41–43, 45–47, 50–52, 54, 56–59, 75, 76,	
	92, 100, 101, 104, 107–111, 113, 116–	
	124, 136, 164–166, 168, 175, 177, 179,	
	185–191, 194, 199, 201, 203–212, 215,	
	217–229, 231–245, 247–263, 265–267,	
	269, 272, 273, 276, 279–282, 284, 286,	
	288, 312, 314–326, 328–338, 344, 346, 348	
first use	351	
flag	351	
text	351	
\firstacronymfont	103, 104	
fontspec package	124	
\footnote	202	
\forall glossaries		
.....	44, 127, 154, 156, 329, 330, 332–336	
\forall gloss entries	92, 101	
\ForEachTrackedDialect	309	
\forall gloss entries	6, 44, 154, 156, 329–336	
\forall listcsloop	29, 101, 115	
\forall listloop	112, 171	
\futurelet	175	
<b>G</b>		
\gdef	52, 53, 167	
\Genacrfullformat	103	
\genacrfullformat	103	
\GenericAcronymFields	103	
\Genplacrfullformat	103	
\genplacrfullformat	103	
\glo@grabfirst	131	
\glo@name	159, 160, 164	
\gloaliaslabel	79	
\global	10, 33, 34, 109, 131	
\glolinkprefix	59, 79, 124	
glossaries package	13, 23, 24, 39–41, 43, 108, 311	
glossaries-accsupp package	20, 22, 140	
glossaries-extra package	2	
glossaries-extra-stylemods package	20, 172, 309	
glossaries-stylemods package	345	
glossaries.sty package	34	
\GlossariesExtraWarning	6,	
.....	15, 32, 33, 47, 49, 58, 104, 106, 116, 120,	
.....	122, 128, 157, 158, 160, 161, 163, 170, 197	
\GlossariesExtraWarningNoLine	15, 92, 101	
\GlossariesWarning	52, 107, 109, 112, 113, 195	
\GlossariesWarningNoLine	106, 118	
glossary styles:		
altlist	312	
altlistgroup	313	
altlisthypergroup	314	
alttree	326, 327, 337	
alttreegroup	338	
alttreehypergroup	338	
index	322	
indexgroup	323	
indexhypergroup	323	
inline	322	
list	312	
listdotted	311	
listdottedstyle	312	
listgroup	313	
listhypergroup	313	
mcolalttree	342	
mcolalttreegroup	342	
mcolalttreehypergroup	343	
mcolalttreespannav	343	
mcolindexgroup	339	
mcolindexhypergroup	339	
mcolindexspannav	339	
mcoltreegroup	340	
mcoltreehypergroup	340	
mcoltreenonamegroup	341	
mcoltreenonamehypergroup	341	
mcoltreenonamespannav	342	
mcoltreespannav	340	
sublistdotted	312	
tree	324	
treegroup	324	
treehypergroup	325	
treenoname	325	
treenonamegroup	326	
treenonamehypergroup	326	
glossary-bookindex package	311	



glossary-hypernav package .....	78	253, 255, 257, 258, 261, 263, 265, 266,	
glossary-long package .....	316	269–272, 274, 276, 278–280, 282, 285, 287	
glossary-longbooktabs package .....	316		
\glossaryentrynumbers ...	53, 109, 131, 132	\glsabbrvhyphenfont .... 274, 275, 279–286	
\glossaryheader .....	115,	\glsabbrvonlyfont .....	287, 289
128, 312–321, 323–326, 337–341, 343, 347		\glsabbrvscfont .	214–219, 221–223, 225–227
\glossaryname .....	108	\glsabbrvsmfont .....	228–237, 239–241
\glossarypostamble .....	115, 128, 130	\glsabbrvuserfont .....	264–271
\glossarypreamble .....	115, 127	\GLSaccessdesc .....	64
\glossarysection .....	115, 121, 127, 130	\Glsaccessdesc .....	64, 157, 169
\glossarytitle ...	35, 108, 109, 115, 121, 127	\glsaccessdesc .....	64, 157, 173
\glossarytoctitle ...	35, 108, 115, 121, 127	\GLSaccessdescplural .....	64
\glossentry .....	109,	\Glsaccessdescplural .....	64
132, 311, 312, 314–321, 323–325, 337, 347		\glsaccessdescplural .....	64
\glossentrydesc .....	311–321, 323–327	\GLSaccessfirst .....	62
\glossentryname .....		\Glsaccessfirst .....	61
.... 126, 311–321, 323–325, 337, 338, 345		\glsaccessfirst .....	61
\glossentrynameother .....	127	\GLSaccessfirstplural .....	63
\glossentrysymbol 315–319, 321, 323–325, 327		\Glsaccessfirstplural .....	63
\glossxtrsetpopts .....	171	\glsaccessfirstplural .....	62
\GLS .....	90, 101, 135	\Glsaccesslong .....	70,
\Gls .....	48, 90, 101, 135	180, 188, 199, 207, 211, 212, 215, 221–	
\gls .....	32, 48, 49, 90, 101, 107, 119, 135	224, 229, 235, 236, 238, 243, 245, 253–	
\gls@assign@desc .....	33, 34	257, 259, 265, 267, 275, 277, 280, 283, 288	
\gls@assign@field .....	11, 12, 26, 72	\glsaccesslong .....	
\gls@checkseeallowed .....	46, 106	.. 70, 180, 187, 188, 199, 201, 203–206,	
\gls@codepage .....	118	208–212, 215, 217, 218, 220–226, 228,	
\gls@defdocnewglossaryentry .....	90, 98	229, 231–240, 242, 243, 245, 247–263,	
\gls@defglossaryentry .....	33, 34, 47–49	265–267, 269, 272, 275–277, 280, 283, 288	
\gls@dotocitle .....	109	\Glsaccesslongpl .....	
\gls@glossary .....	41	.. 71, 181, 191, 199, 207, 211, 212, 215,	
\gls@grplabel .....	78	221–224, 229, 235–238, 243, 245, 253–	
\gls@level .....	131	259, 266, 267, 275, 277, 280, 281, 283, 288	
\gls@noidxglossary .....	106	\glsaccesslongpl .....	71, 180,
\gls@org@glossaryentryfield .....	109	190, 191, 199, 201, 203, 204, 206, 207,	
\gls@org@glossarysubentryfield ....	109	209–212, 215, 217–226, 228, 229, 231,	
\gls@save@numberlist .....	51, 53, 54	233–240, 242, 243, 245, 247, 249–263,	
\gls@set@xr@key .....	41	265, 267, 269, 272, 275, 277, 280, 283, 288	
\gls@tmplen .....	329–338	\GLSaccessname .....	63
\gls@type .....	106	\Glsaccessname .....	63
\glsabbrvdefaultfont ...	181, 198, 200,	\glsaccessname .....	40, 63
203, 205, 206, 208, 211, 264, 274, 276, 287		\GLSaccessplural .....	62
\glsabbrvmfont .....		\Glsaccessplural .....	62
.... 242–251, 253, 255, 257, 258, 260–263		\glsaccessplural .....	62
\glsabbrvfont .....	81, 82, 103, 181, 185,	\Glsaccessshort .....	68, 186, 194, 201,
186, 189, 190, 192, 194, 198–206, 208,		203–206, 209, 210, 217, 218, 220, 225–	
211, 213, 215, 216, 218, 219, 222, 223,		228, 231, 233, 234, 240–242, 247, 249,	
225, 227, 229, 231, 232, 234, 236, 237,		250, 252, 261, 263, 269, 272, 280, 285, 286	
239, 241, 243, 245, 247, 248, 250, 251,		\glsaccessshort .....	
		.... 68, 180, 185, 186, 194, 199, 201,	

203, 205–210, 212, 215, 217–222, 224–  
 227, 229, 231–236, 238–243, 245, 247,  
 248, 250–257, 259, 261, 263, 265–267,  
 269, 272, 275, 277, 280, 283, 285, 286, 288  
 \Glsaccesssshortpl 69, 189, 194, 201, 203–  
 206, 209, 210, 217, 218, 220, 226–228,  
 231, 233, 234, 240–242, 247, 249, 250,  
 252, 261–263, 269, 272, 280, 285, 286, 288  
 \glsaccesssshortpl 69, 180, 181, 189, 190,  
 194, 199, 201, 203–205, 207–210, 212,  
 215, 217–222, 224–229, 231–245, 247,  
 249–254, 256–259, 261, 263, 265–267,  
 269, 272, 275, 277, 280, 283, 285, 286, 288  
 \GLSaccessssymbol ..... 65  
 \Glsaccessssymbol ..... 65, 169  
 \glsaccessssymbol ..... 65, 169, 173  
 \GLSaccessssymbolplural ..... 65  
 \Glsaccessssymbolplural ..... 65  
 \glsaccessssymbolplural ..... 65  
 \GLSaccesstext ..... 61  
 \Glsaccesstext ..... 61  
 \glsaccesstext ..... 40, 60  
 \glsacrshortcutsTRUE ..... 19, 20  
 \glsacspacemax ..... 104  
 \glsadd ..... 28, 44, 119  
 \glsadd options  
   theHvalue ..... 9  
   thevalue ..... 9  
 \glsaddstoragekey ..... 43, 151  
 \glsbackslash ..... 47  
 \glsCAPS ..... 56, 60–71, 73, 182–193  
 \glsCATEGORY .....  
   .. 54, 60, 74, 81, 82, 152, 153, 156–158,  
   160–163, 169, 172, 173, 182–186, 188–190  
 \glsCATEGORYlabel .....  
   ..... 74, 176, 178, 179, 204, 226,  
   241, 262, 266, 268, 270, 279, 281, 284, 286  
 \glsCclosebrace ..... 41, 120, 121  
 \glsCURRENTentrylabel .....  
   ..... 51–53, 109, 117, 126–129, 171, 172  
 \glsCURRENTfieldvalue .. 27, 28, 30, 32, 264  
 \glsCcustomtext .. 55, 56, 68–71, 182–192, 194  
 \glsCdefaulttype . 6, 16, 32, 108, 119, 127, 129  
 \glsCdescriptionaccessdisplay .. 144, 157  
 \glsCdescriptionpluralaccessdisplay 145  
 \glsCdescwidth ..... 314–321  
 \glsDETOKlabel ..... 8,  
   9, 28–34, 37–40, 44–46, 57, 59, 75, 79,  
   91, 96–101, 106, 109, 112, 113, 126, 127,  
   131, 134–136, 156, 159, 160, 164, 331, 332  
 \glsdisplaynumberlist ..... 107, 111  
 \glsDOHhyperlink ..... 77, 78, 80  
 \glsDOHhypertarget ..... 80, 111  
 \glsDOIFexists .....  
   ..... 14, 23, 31, 37, 39–41, 54, 56, 59,  
   68–71, 79, 106, 112, 113, 126, 127, 182–191  
 \glsDOIFexistsORDO ..... 27, 28, 56  
 \glsDOIFexistsORWARN .... 14, 156, 157, 169  
 \glsDOIFNOexists ..... 33  
 \glsDONOHhyperlink ..... 59, 79, 80  
 \glsDOSanitizesort ..... 107  
 \glsENABLEentrycount ..... 90, 92, 100  
 \glsENABLEentryunitcount ..... 91, 101  
 \glsENTRYcounter ..... 117  
 \glsENTRYcurrcount ..... 91, 92, 98  
 \GlsENTRYdesc ..... 144, 149, 158  
 \glsENTRYdesc ..... 144, 149, 158  
 \GlsENTRYdescplural ..... 145, 149  
 \glsENTRYdescplural ..... 145, 149  
 \GlsENTRYfirst ..... 95, 139, 142, 148  
 \glsENTRYfirst ..... 95, 139, 142, 148, 305  
 \GlsENTRYfirstplural .... 96, 139, 143, 148  
 \glsENTRYfirstplural .....  
   ..... 95, 139, 142, 143, 148, 306  
 \glsENTRYfmt ..... 34–36  
 \GlsENTRYfull ..... 103  
 \glsENTRYfull ..... 103  
 \GlsENTRYfullpl ..... 103  
 \glsENTRYfullpl ..... 103  
 \glsENTRYitem ..... 126,  
   127, 311, 312, 314–321, 323–325, 337, 348  
 \GlsENTRYlong ..... 82, 83, 95, 139, 147, 150  
 \glsENTRYlong 82, 83, 95, 139, 146, 147, 150,  
   204, 227, 241, 262, 268, 270, 284, 306, 307  
 \GlsENTRYlongpl ..... 82, 83, 96, 147, 150  
 \glsENTRYlongpl .... 82, 83, 95, 147, 150, 307  
 \GlsENTRYlongplural ..... 139  
 \glsENTRYlongplural ..... 139  
 \GlsENTRYname ..... 140, 147, 159–162  
 \glsENTRYname ..... 125,  
   126, 140, 147, 165, 303, 304, 329–336, 350  
 \glsENTRYnumberlist .... 107, 113, 334–336  
 \GlsENTRYplural ..... 141, 148  
 \glsENTRYplural ..... 141, 142, 148, 305  
 \glsENTRYprevcount ..... 91, 92, 98  
 \glsENTRYprevmaxcount ..... 99  
 \glsENTRYprevtotalcount ..... 98

<code>\Glsentryshort</code> .....	81, 83, 145, 150	<code>\glsfirstlongemfont</code> .....	
<code>\glsentryshort</code> .....	81–83, 104,	.....	244–246, 248, 249, 255, 256, 258–260
145, 146, 149, 150, 266, 268, 279, 302, 303		<code>\glsfirstlongfont</code> ...	180, 181, 198–201,
<code>\Glsentryshortpl</code> .....	82, 83, 146, 150	203, 205–213, 215, 216, 218, 219, 222,	
<code>\glsentryshortpl</code> .....	82, 83, 146, 150, 303	223, 225, 227, 229, 231, 232, 234, 236,	
<code>\Glsentrysymbol</code> .....	143, 149	237, 239, 241, 243, 245, 247, 248, 250,	
<code>\glsentrysymbol</code> ....	143, 148, 149, 333–335	251, 253, 255, 257, 258, 261, 263, 265,	
<code>\Glsentrysymbolplural</code> .....	144, 149	266, 269, 271, 274, 276, 280, 282, 285, 288	
<code>\glsentrysymbolplural</code> .....	144, 149	<code>\glsfirstlongfootnotefont</code> .....	
<code>\Glsentrytext</code> .....	141, 148	.....	202–206, 225–228, 239–242, 260–263
<code>\glsentrytext</code> .....	79, 141, 148, 304	<code>\glsfirstlonghyphenfont</code>	273–276, 278–285
<code>\glsentrytype</code> .....	126, 127	<code>\glsfirstlongonlyfont</code> .....	287–289
<code>\Glsentryuseri</code> .....	66	<code>\glsfirstlonguserfont</code> .....	265–272
<code>\glsentryuseri</code> .....	66	<code>\GLSfirstplural</code> .....	298
<code>\Glsentryuserii</code> .....	66	<code>\Glsfirstplural</code> .....	298, 299
<code>\glsentryuserii</code> .....	66	<code>\glsfirstplural</code> .....	298
<code>\Glsentryuseriii</code> .....	66	<code>\glsfirstpluralaccessdisplay</code> ..	142, 143
<code>\glsentryuseriii</code> .....	66	<code>\glsforeachincategory</code> .....	195
<code>\Glsentryuseriv</code> .....	67	<code>\glsgenentryfmt</code> .....	54
<code>\glsentryuseriv</code> .....	67	<code>\glsgetattribute</code> .....	58, 78, 92,
<code>\Glsentryuserv</code> .....	67	96–98, 117, 136, 157, 158, 160, 161, 163, 165	
<code>\glsentryuserv</code> .....	67	<code>\glsgetcategoryattribute</code> .....	152
<code>\Glsentryuservi</code> .....	67	<code>\glsgetgrouptitle</code>	313, 314, 323–326, 338–344
<code>\glsentryuservi</code> .....	67	<code>\glsgetwidestname</code> .....	327
<code>\glsfieldfetch</code> .....	79	<code>\glsgroupheading</code> .....	
<code>\glsfieldxdef</code> .....	156	.....	131, 312–321, 323–326, 337–343, 349
<code>\glsfindwidesttoplevelname</code> .....	329	<code>\glsgroupskip</code>	131, 312, 314–324, 326, 338, 349
<code>\GLSfirst</code> .....	297, 298	<code>\glshasattribute</code> .....	58,
<code>\Glsfirst</code> .....	298	78, 92, 97, 99, 101, 117, 136, 157, 158,	
<code>\glsfirst</code> .....	297	160, 161, 163, 165, 198, 200, 202, 203,	
<code>\glsfirstabbrvdefaultfont</code> .....		205, 208–210, 213, 214, 216, 217, 225,	
181, 198, 201, 203, 205, 206, 208, 211, 276		227, 229, 230, 232, 239, 241, 243–249,	
<code>\glsfirstabbrvemfont</code> .....	243–263	256, 260, 262, 265, 266, 268, 270–272,	
<code>\glsfirstabbrvfont</code> ....	103, 180, 181,	274–276, 278, 279, 281–283, 285–287, 289	
198–212, 215, 216, 218, 219, 222, 223,		<code>\glshascategoryattribute</code> .....	152
225, 227, 229, 231, 232, 234, 236, 237,		<code>\glshyperlink</code> .....	79
239, 241, 243, 245, 247, 248, 250, 251,		<code>\glshypernavsep</code> .....	115
253, 255, 257, 258, 261, 263, 265, 266,		<code>\glshypernumber</code> .....	117, 164
269, 271, 274, 276, 277, 280, 282, 285, 288		<code>\glsifattribute</code> ...	56, 57, 61, 74, 76, 85,
<code>\glsfirstabbrvhyphenfont</code> .....		154, 157–160, 163, 171, 173, 174, 293–302	
.....	273–275, 279, 280, 282–286	<code>\glsifcategory</code> .....	154
<code>\glsfirstabbrvonlyfont</code> .....	288	<code>\glsifcategoryattribute</code> .....	
<code>\glsfirstabbrvscfont</code> .....	214–228	.....	74, 152, 153, 178, 179
<code>\glsfirstabbrvsmfont</code> .....	229–242	<code>\glsifnotregular</code> .....	60
<code>\glsfirstabbrvuserfont</code> .....	265–272	<code>\glsifnotregularcategory</code> .....	153
<code>\glsfirstaccessdisplay</code> .....	142	<code>\glsifplural</code> .....	
<code>\glsfirstlongdefaultfont</code> .....		..	56, 60, 62–65, 68–71, 173, 174, 182–193
198, 201, 206, 208, 211, 214–224, 229–		<code>\glsifregular</code> .....	54, 60, 95, 96, 139
238, 243, 244, 246, 247, 250–254, 257, 258		<code>\glsifregularcategory</code> .....	153

<code>\glsifusetranslator</code> .....	35	253, 255, 257, 258, 261, 263, 265, 266,
<code>\glsignore</code> .....	52, 53	269, 271, 274, 276, 280, 282, 285, 288, 289
<code>\glsinlinedescformat</code> .....	322	
<code>\glsinlinesubdescformat</code> .....	322	<code>\glslongfootnotefont</code> .....
<code>\glsinsert</code> .....	56,	202, 203, 205, 225, 227, 239, 241, 261, 263
	60, 68–71, 182–194, 273, 279, 281, 284–286	<code>\glslonghyphenfont</code> .....
<code>\glskeylisttok</code> .....	102, 103, 178, 180	..... 274, 276, 278–280, 282, 284, 285
<code>\glslabel</code> .....	8,	<code>\glslongonlyfont</code> .....
	9, 27, 37, 40, 54, 56–59, 74, 75, 78, 79,	287, 288
	104, 136, 172, 173, 192–194, 204, 227,	<code>\glslongpltok</code> .....
	241, 262, 266, 268, 270, 279, 281, 284–286	. 179, 180, 198–202, 211, 213–217, 221,
<code>\glslabeltok</code> .....		225, 229–231, 236, 239, 243, 244, 246–
	102, 178, 179, 198–203, 205, 206, 208–	249, 253, 255, 258, 260, 265–268, 270–
	211, 213–219, 221, 225, 227, 229–232,	272, 274–276, 278, 279, 281–283, 287, 289
	234, 236, 239, 241, 243–251, 253, 255,	<code>\glslongpluralaccessdisplay</code> .....
	256, 258, 260, 262, 265–268, 270–272,	147
	274–276, 278, 279, 281–283, 285–287, 289	<code>\glslongtok</code> .. 102, 178, 180, 198–202, 204,
<code>\glsletentryfield</code> .....	165	206, 208, 211, 213–219, 221, 225, 226,
<code>\glslink</code> .....	103	229–232, 234–236, 239, 240, 243–251,
<code>\glslink options</code>		253, 255, 258, 260, 262, 265–268, 270–
counter .....	9	272, 274–276, 278, 279, 281–284, 287, 289
format .....	164	<code>\glslonguserfont</code> .....
hyper .....	289	264–271
hyperoutside .....	57	<code>\glsmcols</code> .....
noindex .....	8, 74, 289	340–343
theHvalue .....	58	<code>\GLSname</code> .....
thevalue .....	58, 132	295
wrgloss .....	8, 56, 57	<code>\Glsname</code> .....
<code>\glslinkcheckfirsthyperhook</code> .....	74	295
<code>\glslinkpostsetkeys</code> .....	58, 136	<code>\glsnameaccessdisplay</code> .....
<code>\glslinkvar</code> .....	77	140, 159, 161
<code>\glslistchildpostlocation</code> .....	312	<code>\glsnamefont</code> .....
<code>\glslistchildprelocation</code> .....	312, 313	158–163
<code>\glslistdottedwidth</code> .....	311	<code>\glsnavhyperlink</code> .....
<code>\glslistgroupheaderfmt</code> .....	313, 314	115
<code>\glslistnavigationitem</code> .....	313, 314	<code>\glsnavhyperlinkname</code> .....
<code>\glslistprelocation</code> .....	312, 313	78
<code>\glslocalunset</code> .....	56, 136	<code>\glsnavhypertarget</code> .....
<code>\glslongaccessdisplay</code> .....	146, 147	..... 313, 314, 324–326, 339–344
<code>\glslongdefaultfont</code> .. 182, 198, 201, 202,		<code>\glsnavigation</code> .. 313, 314, 324–326, 338–343
	206, 208, 211, 215, 216, 218, 219, 221–	<code>\glsnextpages</code> .....
	223, 229, 231, 232, 234, 236–238, 243,	109
	247, 250, 251, 253, 254, 257, 264, 274, 287	<code>\glsnoidxdisplayloc</code> .....
<code>\glslongemfont</code> .. 242, 245, 248, 255, 258, 259		112, 113
<code>\glslongfont</code> .. 82, 83, 182, 187, 188, 190–		<code>\glsnoidxdisplaylocclsthandler</code> ....
	192, 198, 199, 201, 203, 205, 206, 208,	112
	211, 213, 215, 216, 218, 219, 222, 223,	<code>\glsnoidxloclist</code> .....
	225, 227, 229, 231, 232, 234, 236, 237,	113, 131, 132
	239, 241, 243, 245, 247, 248, 250, 251,	<code>\glsnoidxnumberlistloophandler</code> ....
		112
		<code>\glsnonextpages</code> .....
		109
		<code>\glsnonumberlistfalse</code> .....
		51
		<code>\glsnonumberlisttrue</code> .....
		51
		<code>\glsnopostdotfalse</code> .....
		110
		<code>\glsnopostdottrue</code> .....
		110
		<code>\glsnumberlistloop</code> .....
		107
		<code>\glsnumlistlastsep</code> .....
		112
		<code>\glsnumlistsep</code> .....
		112
		<code>\glsopenbrace</code> .....
		40, 120, 121
		<code>\glsorder</code> .....
		105
		<code>\glspagelistwidth</code> .....
		315, 317, 319, 321
		<code>\glspar</code> .....
		130
		<code>\GLSpl</code> .....
		90, 101, 135
		<code>\Glspl</code> .....
		49, 90, 101, 135

<code>\glsp</code>	48, 90, 101, 135	<code>\glstextformat</code>	56, 58
<code>\GLSplural</code>	296, 297	<code>\glstextup</code>	214
<code>\Glsplural</code>	297	<code>\glstreechildpredesc</code>	323, 324
<code>\glsplural</code>	296, 297	<code>\glstreechildprelocation</code>	323, 324, 326
<code>\glspluralaccessdisplay</code>	141, 142	<code>\glstreegroupheaderfmt</code>	323–326, 338–344, 346
<code>\glspluralsuffix</code>	124, 178, 182	<code>\glstreeindent</code>	324, 325, 336–338
<code>\glspostdescription</code>	15, 16, 110, 171, 311–321, 323–327	<code>\glstreeitem</code>	322, 340, 347, 348
<code>\glspostinline</code>	322	<code>\glstreenamebox</code>	337, 338
<code>\glspostlinkhook</code>	55, 56, 68–72, 84, 182–191	<code>\glstreenamefmt</code>	323–325, 327, 329–338
<code>\glsprestandardsort</code>	107	<code>\glstreenavigationfmt</code>	324–326, 338–343
<code>\glsresetentrylist</code>	115, 128	<code>\glstreepredesc</code>	323–325
<code>\glsee</code>	41, 43	<code>\glstreeprelocation</code>	322–325, 327
<code>\glseeformat</code>	39, 40, 45, 106, 112, 113	<code>\glstreesubitem</code>	322, 348
<code>\glseelist</code>	40	<code>\glstreesubsubitem</code>	323, 348
<code>\glsetabbrvfmt</code>	54, 60, 81, 82, 156–158, 160, 161, 163, 169, 182–186, 188–190, 192	<code>\GLstrLetField</code>	31
<code>\glsetattribute</code>	198, 200, 202, 203, 205, 206, 208, 209, 211, 213, 214, 216–219, 221, 225, 227, 229–232, 234, 236, 239, 241, 243–246, 248–251, 253, 255, 256, 258, 260, 262, 265, 266, 268, 270–272, 274–276, 278, 279, 281–283, 285–287, 289	<code>\glstype</code>	56, 57, 68–72, 136, 182–191
<code>\glsetcategoryattribute</code>	90, 102, 104, 135, 152, 153, 155, 170	<code>\glset</code>	44, 56, 93, 94, 136
<code>\glsetnoexpandfield</code>	11, 12	<code>\glswrite</code>	40, 105
<code>\glsettoctitle</code>	109	<code>\glswriteentry</code>	8, 9
<code>\glsshortaccessdisplay</code>	145, 146	<code>\Glsxtr</code>	49
<code>\glsshortpltok</code>	179, 180, 198, 200–204, 206, 208, 214–219, 225, 226, 229–233, 239, 240, 243–251, 260, 262, 265, 266, 268, 270–272, 274, 275, 279, 281–284, 286, 287, 289	<code>\glxtr</code>	49
<code>\glsshortpluralaccessdisplay</code>	146	<code>\glxtr@@do@wrglossary</code>	8, 9, 11, 13
<code>\glsshorttok</code>	102, 178–180, 198–202, 204, 206, 208, 213–219, 221, 225, 226, 229–233, 235, 239, 240, 243, 244, 246–251, 253, 255, 260, 262, 265–268, 270–272, 274, 275, 278, 279, 281–284, 286, 287, 289	<code>\glxtr@addloclistfield</code>	13
<code>\glssubentryitem</code>	126, 127, 311–321, 323–325, 337, 348	<code>\glxtr@addunused</code>	44
<code>\glssymbolaccessdisplay</code>	143	<code>\glxtr@applyabbrvfmt</code>	192
<code>\glssymbolpluralaccessdisplay</code>	143, 144	<code>\glxtr@applyabbrvstyle</code>	176, 178, 195
<code>\glstarget</code>	126, 127, 311–321, 323–325, 337, 338, 348	<code>\glxtr@counterrecord</code>	125
<code>\GLStext</code>	296	<code>\glxtr@do@alsoindex@wrglossary</code>	13
<code>\Glstext</code>	296	<code>\glxtr@doption</code>	5, 15, 16, 22, 24
<code>\glstext</code>	296	<code>\glxtr@fields</code>	123
<code>\glstextaccessdisplay</code>	141	<code>\glxtr@headentry@p</code>	85, 86
		<code>\glxtr@hyperoutsidefalse</code>	57
		<code>\glxtr@hyperoutsidettrue</code>	57
		<code>\glxtr@ifnextpunc</code>	175
		<code>\glxtr@ifpunctoken</code>	175
		<code>\glxtr@indexonly@saveentrycounter</code>	13, 25
		<code>\glxtr@keylist</code>	47–49
		<code>\glxtr@label</code>	350
		<code>\glxtr@langtag</code>	124
		<code>\glxtr@linkprefix</code>	123, 124
		<code>\glxtr@makeglossaries</code>	105
		<code>\glxtr@newabbreviation</code>	103, 177
		<code>\glxtr@next</code>	175
		<code>\glxtr@org@@do@wrglossary</code>	25
		<code>\glxtr@org@dohyperlink</code>	78
		<code>\glxtr@org@getgrouptitle</code>	114
		<code>\glxtr@org@newignoredglossary</code>	34

<code>\glxtr@orgmakenoidxglossaries</code> . . . . .	45	<code>\glxtrbookindexgroupskip</code> . . . . .	349
<code>\glxtr@pluralsuffixes</code> . . . . .	123, 124	<code>\glxtrbookindexlastmarkfmt</code> . . . . .	350
<code>\glxtr@process</code> . . . . .	128, 129	<code>\glxtrbookindexname</code> . . . . .	345, 348
<code>\glxtr@provideignoredglossary</code> . . . . .	36	<code>\glxtrbookindexparentchildsep</code> . . . . .	345, 347
<code>\glxtr@punclist</code> . . . . .	174, 175	<code>\glxtrbookindexparentschildsep</code> . . . . .	347, 348
<code>\glxtr@record</code> . . . . .	10, 123	<code>\glxtrbookindexprelocation</code> . . . . .	345, 348
<code>\glxtr@recordsee</code> . . . . .	11	<code>\glxtrbookindexsubbetween</code> . . . . .	348
<code>\glxtr@resource</code> . . . . .	122, 123	<code>\glxtrbookindexsubname</code> . . . . .	348
<code>\glxtr@s@newignoredglossary</code> . . . . .	34	<code>\glxtrbookindexsubprelocation</code> . . . . .	348
<code>\glxtr@s@provideignoredglossary</code> . . . . .	36	<code>\glxtrbookindexsubsubbetween</code> . . . . .	348
<code>\glxtr@saveentrycounter</code> . . . . .	8, 9, 11, 75	<code>\glxtrbookindexthepage</code> . . . . .	349, 350
<code>\glxtr@setaccessdisplay</code> . . . . .	163	<code>\glxtrcat</code> . . . . .	47–49
<code>\glxtr@setbookindexmark</code> . . . . .	349	<code>\glxtrchecknohyperfirst</code> . . . . .	61–63
<code>\glxtr@setup@record</code> . . . . .	12, 13, 24, 25	<code>\glxtrComputeTreeIndent</code> . . . . .	337
<code>\glxtr@shortcutsval</code> . . . . .	123, 124	<code>\glxtrComputeTreeSubIndent</code> . . . . .	337
<code>\glxtr@texencoding</code> . . . . .	124	<code>\GlsXtrdefaultsubsequentfmt</code> . . . . .	194, 196
<code>\glxtr@usesee</code> . . . . .	39	<code>\glxtrdefaultsubsequentfmt</code> . . . . .	194, 196
<code>\glxtr@warnonexistsordo</code> . . . . .	7, 13, 38	<code>\GlsXtrdefaultsubsequentplfmt</code> . . . . .	194, 196
<code>\glxtr@writefields</code> . . . . .	122	<code>\glxtrdefaultsubsequentplfmt</code> . . . . .	194, 196
<code>\glxtrabbrvfootnote</code> . . . . .	202–204, 225–227, 239–241, 260–262	<code>\GlsXtrDefineAbbreviationShortcuts</code> . . . . .	19, 20
<code>\glxtrabbrvpluralsuffix</code> . . . . .	124, 182, 198, 200, 203, 205, 206, 208, 211, 214, 228, 242, 265, 274, 276, 280, 285, 287	<code>\GlsXtrDefineAcShortcuts</code> . . . . .	19, 20
<code>\glxtrabbrvtype</code> . . . . .	16, 17, 180	<code>\GlsXtrDefineOtherShortcuts</code> . . . . .	19, 20
<code>\glxtractivatenopost</code> . . . . .	109	<code>\glxtrdetoklocation</code> . . . . .	135
<code>\glxtraddallcrossrefs</code> . . . . .	43	<code>\glxtrdiscardperiod</code> . . . . .	172
<code>\glxtralias</code> . . . . .	75	<code>\glxtrdisplayendloc</code> . . . . .	116
<code>\glxtrAltTreeIndent</code> . . . . .	327	<code>\glxtrdisplayendlochook</code> . . . . .	116
<code>\glxtralttreeInit</code> . . . . .	337, 342, 343	<code>\glxtrdisplayingleloc</code> . . . . .	116
<code>\glxtrAltTreePar</code> . . . . .	327	<code>\glxtrdisplaystartloc</code> . . . . .	116
<code>\glxtrAltTreeSetHangIndent</code> . . . . .	327, 337	<code>\glxtrdoautoindexname</code> . . . . .	76, 77, 162
<code>\glxtrAltTreeSetSubHangIndent</code> . . . . .	338	<code>\glxtrdopostpunc</code> . . . . .	204, 227, 241, 262
<code>\glxtralttreeSubSymbolDescLocation</code> . . . . .	338	<code>\glxtrdowrglossaryhook</code> . . . . .	76
<code>\glxtralttreeSymbolDescLocation</code> . . . . .	327, 337	<code>\glxtremsuffix</code> . . . . .	243, 245, 246, 248, 250, 251, 253, 255, 257, 258, 261, 263
<code>\glxtrassignfieldfont</code> . . . . .	60–67	<code>\GlsXtrEnableEntryCounting</code> . . . . .	101
<code>\glxtrautoindex</code> . . . . .	165	<code>\GlsXtrEnableEntryUnitCounting</code> . . . . .	90
<code>\glxtrautoindexassignsort</code> . . . . .	165	<code>\GlsXtrEnableOnTheFly</code> . . . . .	47, 49
<code>\glxtrautoindexentry</code> . . . . .	165	<code>\glxtrendfor</code> . . . . .	29
<code>\glxtrbookindexatendgroup</code> . . . . .	347	<code>\glxtrfieldlistgadd</code> . . . . .	125
<code>\glxtrbookindexatsubendgroup</code> . . . . .	348	<code>\glxtrfieldtitlecase</code> . . . . .	157–160, 163
<code>\glxtrbookindexatsubsubendgroup</code> . . . . .	348	<code>\glxtrfieldtitlecasecs</code> . . . . .	156
<code>\glxtrbookindexbetween</code> . . . . .	347	<code>\glxtrfieldxifinlist</code> . . . . .	129
<code>\glxtrbookindexbookmark</code> . . . . .	349	<code>\glxtrfirstscfont</code> . . . . .	214
<code>\glxtrbookindexcols</code> . . . . .	347	<code>\glxtrfirstsmfont</code> . . . . .	228
<code>\glxtrbookindexcolspread</code> . . . . .	347	<code>\GlsXtrFmtDefaultOptions</code> . . . . .	27, 28
<code>\glxtrbookindexfirstmarkfmt</code> . . . . .	350	<code>\glxtrfmtdisplay</code> . . . . .	27, 28
<code>\glxtrbookindexformatheader</code> . . . . .	349	<code>\GlsXtrFmtField</code> . . . . .	27, 28
		<code>\GlsXtrFormatLocationList</code> . . . . .	51, 54, 334–336



<code>\GLSxtrfull</code> .....	17, 18, 301	<code>\Glsxtrheadlongpl</code> .....	292
<code>\Glsxtrfull</code> .....	17, 18, 301, 302	<code>\glsxtrheadlongpl</code> .....	292
<code>\glsxtrfull</code> .....	17, 18, 301	<code>\Glsxtrheadname</code> .....	292
<code>\Glsxtrfullformat</code> 181, 194, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 249, 251, 252, 254, 256, 258, 260, 261, 263, 265, 267, 269, 272, 275, 277, 280, 283, 285, 288		<code>\glsxtrheadname</code> .....	125, 126, 292
<code>\glsxtrfullformat</code> .....		<code>\Glsxtrheadplural</code> .....	292
.... 181, 193, 194, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 222, 224, 225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 248, 250, 252, 254, 256, 258, 259, 261, 263, 265, 266, 269, 272, 275, 277, 280, 283, 285, 288		<code>\glsxtrheadplural</code> .....	292
<code>\GLSxtrfullpl</code> .....	17, 18, 301, 302	<code>\Glsxtrheadshort</code> .....	292
<code>\Glsxtrfullpl</code> .....	17, 18, 302	<code>\glsxtrheadshort</code> .....	292
<code>\glsxtrfullpl</code> .....	17, 18, 301	<code>\Glsxtrheadshortpl</code> .....	292
<code>\Glsxtrfullplformat</code> .....		<code>\glsxtrheadshortpl</code> .....	292
..... 181, 193, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237, 238, 240, 241, 243, 245, 247, 249, 251, 252, 254, 256, 258, 260, 261, 263, 266, 267, 269, 272, 275, 277, 280, 283, 285, 288		<code>\Glsxtrheadtext</code> .....	292
<code>\glsxtrfullplformat</code> . 193, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 248, 251, 252, 254, 256, 258, 259, 261, 263, 265, 267, 269, 272, 275, 277, 280, 283, 285, 288		<code>\glsxtrheadtext</code> .....	292
<code>\glsxtrfullsep</code> .....		<code>\glsxtrhyperlink</code> .....	79, 117
.... 180, 181, 198–201, 204–210, 212, 214–222, 224, 226–238, 240, 242–259, 261–264, 273–275, 277, 279, 282–284, 288		<code>\glsxtrhyphensuffix</code> .....	274, 282
<code>\glsxtrgenabbrvfmt</code> .....	54	<code>\glsxtrifcounttrigger</code> .....	93, 94
<code>\glsxtrgetgroup title</code> .....	115, 349	<code>\glsxtrifcustomdiscardperiod</code> .....	172
<code>\glsxtrgroupfield</code> .....	131	<code>\glsxtrifemptyglossary</code> .....	115, 121, 128
<code>\Glsxtrheadfirst</code> .....	292	<code>\glsxtrifhasfield</code> .....	32, 75, 345
<code>\glsxtrheadfirst</code> .....	292	<code>\glsxtrifhyphenstart</code> .....	
<code>\Glsxtrheadfirstplural</code> .....	292	..... 273, 276, 278, 279, 282, 284	
<code>\glsxtrheadfirstplural</code> .....	292	<code>\glsxtrifindexing</code> .....	76
<code>\Glsxtrheadfull</code> .....	292	<code>\glsxtrifinmark</code> .....	59, 85–88, 291, 292
<code>\glsxtrheadfull</code> .....	292	<code>\glsxtrifnextpunc</code> .....	174, 175
<code>\Glsxtrheadfullpl</code> .....	292	<code>\glsxtrifperiod</code> .....	172–174
<code>\glsxtrheadfullpl</code> .....	292	<code>\glsxtrifrecordtrigger</code> .....	137–139
<code>\Glsxtrheadlong</code> .....	292	<code>\glsxtrifwasfirstuse</code> .....	
<code>\glsxtrheadlong</code> .....	292	..... 60–63, 68–71, 74, 104, 173, 183, 185–191, 204, 205, 226, 227, 241, 262, 266, 268, 270, 279, 281, 284, 286	
		<code>\glsxtrindexaliased</code> .....	75
		<code>\glsxtrindexseealso</code> .....	42, 43
		<code>\glsxtrinithyperoutside</code> .....	58
		<code>\glsxtrinitwrgloss</code> .....	58, 136
		<code>\glsxtrinitwrglossbeforefalse</code> ...	56, 57
		<code>\glsxtrinitwrglossbeforetrue</code> .....	57
		<code>\Glsxtrinlinefullformat</code> 181, 183, 196, 204, 206, 207, 209, 210, 212, 218, 220– 222, 224, 226, 228, 233–236, 238, 240, 242, 250, 251, 253, 254, 256, 257, 259, 261, 263, 267, 269, 277, 280, 286, 288, 308	
		<code>\glsxtrinlinefullformat</code> 181–183, 196, 203, 205, 206, 208, 210, 212, 218, 220– 222, 224, 226, 227, 232, 234–236, 238, 240, 242, 250–252, 254, 255, 257, 259, 261, 263, 267, 269, 277, 280, 285, 288, 307	
		<code>\Glsxtrinlinefullplformat</code> .. 181, 184, 196, 204, 206, 207, 209, 210, 212, 218, 220–222, 224, 226, 228, 233–236, 238,	

240, 242, 250, 252–254, 256, 258, 259, 262, 263, 267, 269, 277, 281, 286, 288, 308	
\glsxtrinlinefullplformat .....	181, 184, 185, 196, 204, 205, 207, 208, 210, 212, 218, 220– 222, 224, 226, 228, 232, 234–236, 238, 240, 242, 250–252, 254, 255, 257, 259, 261, 263, 267, 269, 277, 280, 286, 288, 308
\glsxtrininsertinsidefalse .....	198
\glsxtrlocationhyperlink .....	117
\glsxtrlocrangefmt .....	116
\GLSxtrlong .....	17, 18, 299, 300
\Glsxtrlong .....	17, 18, 300
\glsxtrlong .....	17, 18, 299
\glsxtrlonghyphen .....	280
\glsxtrlonghyphennohook .....	276, 277
\glsxtrlonghyphenshort .....	275
\GLSxtrlongpl .....	17, 18, 299, 300
\Glsxtrlongpl .....	17, 18, 300
\glsxtrlongpl .....	17, 18, 299
\glsxtrlongshortdescname .....	199, 215, 230, 244, 245, 270, 275, 281
\glsxtrlongshortdescsort .....	199, 215, 230, 244, 245, 270, 275, 281
\glsxtrmarkhook .....	290
\glsxtrnewabbrevpresetkeyhook .....	179
\glsxtrnewnumber .....	18
\glsxtrnewsymbol .....	18
\glsxtrNoGlossaryWarning .....	20, 118
\GlsXtrNoGlsWarningAutoMake .....	121
\GlsXtrNoGlsWarningBuildInfo .....	122
\GlsXtrNoGlsWarningCheckFile .....	121
\GlsXtrNoGlsWarningEmptyMain .....	121
\GlsXtrNoGlsWarningEmptyNotMain .....	121
\GlsXtrNoGlsWarningEmptyStart .....	121
\GlsXtrNoGlsWarningHead .....	121
\GlsXtrNoGlsWarningMismatch .....	122
\GlsXtrNoGlsWarningNoOut .....	122
\GlsXtrNoGlsWarningTail .....	122
\glsxtrnopostpunc .....	109
\glsxtronlydescname .....	289
\glsxtronlydescsort .....	289
\glsxtronlysuffix .....	287
\glsxtrorg@ifKV@glslink@hyper .....	55
\glsxtrorglong .....	178, 199, 276
\glsxtrorgshort .....	178, 199
\GLSxtrp .....	85
\Glsxtrp .....	85
\glsxtrp .....	84, 86
\glsxtrparen .....	180, 181, 198–201, 204–210, 212, 214– 222, 224, 226, 228–238, 240, 242–259, 261–264, 273–275, 277, 279, 282–284, 288
\Glsxtrpl .....	49
\glsxtrpl .....	49
\glsxtrpostdescription .....	110, 155, 171, 322
\glsxtrposthyphenlong .....	284, 286
\glsxtrposthyphenshort .....	279, 281
\glsxtrposthyphensubsequent .....	279, 281, 285, 286
\glsxtrpostlink .....	172
\glsxtrpostlinkendsentence .....	172
\glsxtrpostlinkhook .....	172
\glsxtrpostlocalreset .....	89, 91, 99
\glsxtrpostlocalunset .....	89, 91, 99
\glsxtrpostlongdescription .....	33
\glsxtrpostnamehook .....	159–162, 164
\GlsXtrPostNewAbbreviation .....	180, 196, 198, 200, 202–204, 206, 208–211, 213, 214, 216– 219, 221, 225, 226, 229, 230, 232, 234, 236, 239, 241, 243–251, 253, 255, 256, 258, 260, 262, 265, 266, 268, 270–272, 274–276, 278, 279, 281–284, 286, 287, 289
\glsxtrpostreset .....	89, 91, 99
\glsxtrpostunset .....	89, 91, 99
\glsxtrprelocation .....	312, 314, 316, 318, 320, 322, 345
\glsxtrprotectlinks .....	78–80
\GlsXtrRecordCounter .....	11
\glsxtrrecordtriggervalue .....	136
\glsxtrregularfont .....	54, 60
\glsxtrresourcecount .....	123
\glsxtrresourcefile .....	123
\glsxtrresourceinit .....	122
\glsxtrrestoremarkhook .....	290
\glsxtrrestorepostpunc .....	110
\glsxtrscfont .....	214
\glsxtrscsuffix .....	215, 216, 218, 219, 222, 223, 225, 227
\GlsXtrSetActualChar .....	168
\glsxtrsetaliasnoindex .....	13, 75
\GlsXtrSetEncapChar .....	168
\GlsXtrSetEscChar .....	168
\glsxtrsetfieldifexists .....	31, 32
\GlsXtrSetLevelChar .....	168
\glsxtrsetpopts .....	84



<code>\glxtrsetupfulldefs</code> .....	<code>\glxtrtitleplural</code> .....
..... 182–185, 205, 227, 241, 262	291–293, 305
<code>\GLSxtrshort</code> .....	<code>\Glsxtrtitleshort</code> .....
17, 18, 88, 293, 294	291, 292, 303
<code>\Glsxtrshort</code> .....	<code>\glxtrtitleshort</code> .....
17, 18, 294	291, 292, 302
<code>\glxtrshort</code> .....	<code>\Glsxtrtitleshortpl</code> .....
17, 18, 293	291, 292, 303
<code>\glxtrshortdescname</code> ...	<code>\glxtrtitleshortpl</code> .....
208, 219, 233, 251	291, 292, 303
<code>\glxtrshorthyphen</code> .....	<code>\Glsxtrtitletext</code> .....
285	291–293, 304
<code>\glxtrshorthyphenlong</code> .....	<code>\glxtrtitletext</code> .....
283	291–293, 304
<code>\glxtrshortlongdescname</code> .....	<code>\GlsXtrTotalRecordCount</code> .....
..... 201, 217, 231, 247, 249, 272, 283, 286	135
<code>\glxtrshortlongdescsort</code> .....	<code>\glxtrtreetopindent</code> .....
..... 201, 217, 231, 247, 249, 272, 283, 286	327, 336
<code>\GLSxtrshortpl</code> .....	<code>\glxtrundefaction</code> ..
17, 18, 293, 294	7, 13, 25, 34, 35, 37, 38
<code>\Glsxtrshortpl</code> .....	<code>\glxtrundeftag</code> .....
17, 18, 294, 295	25, 112, 113
<code>\glxtrshortpl</code> .....	<code>\glxtrunsrtdo</code> .....
17, 18, 294	129, 130
<code>\glxtrsmfont</code> .....	<code>\GlsXtrUseAbbrStyleFmts</code> .....
228	..... 200, 202, 208, 210, 211, 213,
<code>\glxtrsmsuffix</code> .....	214, 216, 218, 221, 230, 232, 235, 244,
..... 229, 231, 232, 234, 236, 237, 239, 241	246, 248, 249, 252, 257, 260, 268, 270,
<code>\Glsxtrsubsequentfmt</code> .....	271, 273, 275, 276, 278, 281, 283, 287, 289
..... 193, 196, 211, 222, 223,	<code>\GlsXtrUseAbbrStyleSetup</code> 207, 209, 210,
236, 238, 253, 255, 257, 259, 277, 280, 285	213, 220, 223, 235, 237, 252, 256, 257, 260
<code>\glxtrsubsequentfmt</code> .....	<code>\glxtruserfield</code> .....
..... 193, 196, 211, 222, 223,	264
236, 237, 253, 255, 257, 259, 276, 280, 285	<code>\glxtruserparen</code> .....
<code>\Glsxtrsubsequentplfmt</code> .....	265–272
..... 192, 196, 211, 222, 223,	<code>\glxtrusersuffix</code> .....
236, 238, 254, 255, 257, 259, 277, 280, 285	265, 266, 269, 271
<code>\glxtrsubsequentplfmt</code> .....	<code>\glxtruseseealsoformat</code> .....
..... 192, 196, 211, 222, 223,	40
236, 237, 253, 255, 257, 259, 277, 280, 285	<code>\glxtruseseeeformat</code> .....
<code>\glxtrsupplocationurl</code> .....	39
117	<code>\GlsXtrWarnDeprecatedAbbrStyle</code> 176, 197
<code>\glxtrtagfont</code> .....	<code>\GlsXtrWarning</code> .....
171	47–49
<code>\Glsxtrtitlefirst</code> .....	<code>\glxtrword</code> .....
291–293, 305, 306	177
<code>\glxtrtitlefirst</code> .....	<code>\glxtrwordsep</code> 177, 273, 276, 278, 279, 282, 284
291–293, 305	<code>\glxtrwrglossmark</code> .....
<code>\Glsxtrtitlefirstplural</code> ....	22
291–293, 306	
<code>\glxtrtitlefirstplural</code> ....	
291–293, 306	
<code>\Glsxtrtitlefull</code> .....	
292, 293, 308	
<code>\glxtrtitlefull</code> .....	
292, 293, 307, 308	
<code>\Glsxtrtitlefullpl</code> .....	
292, 293, 308	
<code>\glxtrtitlefullpl</code> .....	
292, 293, 308	
<code>\Glsxtrtitlelong</code> .....	
292, 293, 307	
<code>\glxtrtitlelong</code> .....	
291–293, 306	
<code>\Glsxtrtitlelongpl</code> .....	
292, 293, 307	
<code>\glxtrtitlelongpl</code> .....	
291–293, 307	
<code>\Glsxtrtitlename</code> .....	
291, 292, 304	
<code>\glxtrtitlename</code> .....	
291, 292, 303, 304	
<code>\glxtrtitleorpdforheading</code> .....	
..... 23, 125–127, 291, 292	
<code>\Glsxtrtitleplural</code> .....	
291–293, 305	

## H

<code>\hangindent</code> ..	324, 325, 327, 336–338, 342, 343
<code>\hbox</code> .....	311
<code>\hfill</code> .....	311
<code>\href</code> .....	78
<code>\hsize</code> .....	50, 51
<code>\hss</code> .....	311
<code>\hyperlink</code> .....	79
<code>\hyperpage</code> .....	164
<code>\hyperref</code> .....	78, 117
<code>hyperref package</code> .....	80, 164, 289, 302

## I

<code>\if</code> .....	47
<code>\if@glxtr@autoseeindex</code> ....	23, 24, 39, 42
<code>\if@glxtr@format@override</code> .....	165
<code>\if@glxtrdocdefrestricted</code> .....	45
<code>\if@glxtrindexcrossrefs</code> .....	14, 43
<code>\ifblank</code> .....	26, 47–49, 105
<code>\ifcase</code> ..	7, 12, 19, 20, 22, 46, 57, 110, 323, 348

<code>\ifcsdef</code>	25, 32, 34–37, 58, 72, 73, 84–88, 96, 108, 114, 115, 126, 130, 133–135, 157, 158, 160–163, 173, 176, 192, 196, 314–321
<code>\ifcsstring</code>	25, 152, 195
<code>\ifcsundef</code>	30, 32, 34–36, 45, 50, 52, 80, 91, 96–100, 113, 114, 118, 130, 152, 195–197, 311, 328, 329, 336, 350
<code>\ifcsvoid</code>	43, 151
<code>\ifdef</code>	13, 18, 24, 28, 37, 38, 40, 41, 50, 51, 74, 78, 79, 85–87, 108, 112, 113, 124, 133, 154, 155, 168, 171, 264, 291, 302–308, 311–314, 322–326, 338–343, 346, 349, 350
<code>\ifdefempty</code>	7–9, 30, 34–36, 39, 40, 58, 59, 90, 102, 105, 108, 116, 128, 131, 135, 136, 170, 177, 192, 347
<code>\ifdefequal</code>	45, 121, 131, 162
<code>\ifdefstring</code>	6, 32, 165, 170, 346
<code>\ifdefvoid</code>	39, 42–44, 79, 96, 113, 117, 131, 132
<code>\ifdim</code>	50, 51, 104, 328–336
<code>\IfFileExists</code>	21, 118, 121, 122, 124, 310
<code>\ifglossaryexists</code>	38
<code>\ifglshasacronym</code>	17, 121
<code>\ifglshasrshortcuts</code>	19
<code>\ifglshasautomake</code>	108, 121, 124
<code>\ifglshasentrycounter</code>	32
<code>\ifglshasentryexists</code>	8, 37, 38, 47–49, 51, 60, 131, 152, 171, 172
<code>\ifglshasfield</code>	151
<code>\ifglshasfield</code>	27, 28, 264
<code>\ifglshaslong</code>	95, 96, 139
<code>\ifglshasparent</code>	126–128, 131, 329, 331, 332
<code>\ifglshasshort</code>	40, 54, 60
<code>\ifglshassymbol</code>	173, 323–325, 327
<code>\ifglshasindexonlyfirst</code>	76
<code>\ifglshasgroupskip</code>	312, 314–324, 326, 338, 346
<code>\ifglshasnonnumberlist</code>	53
<code>\ifglshasnopostdot</code>	15, 110
<code>\ifglssanitizesort</code>	107
<code>\ifglssubentrycounter</code>	32
<code>\ifglshasused</code>	44, 74, 76, 92, 101, 104, 192, 329–331, 333–335
<code>\ifglshasxindy</code>	118–120
<code>\ifglshasxtr@hyperoutside</code>	59
<code>\ifglshasxtrinittwrglossbefore</code>	57–59, 136
<code>\ifglshasxtrininsertinside</code>	185–191, 194, 199, 201, 203–212, 215, 217–229, 231–245, 247–263, 265–267, 269, 272, 273, 276, 279–282, 284, 286, 288
<code>\ifHy@hyperindex</code>	164
<code>\ifinlistcs</code>	29, 45
<code>\ifinner</code>	23
<code>\ifKV@glshlink@hyper</code>	55, 57, 59
<code>\ifKV@glshlink@local</code>	56, 136
<code>\ifKV@glshlink@noindex</code>	8, 9, 11, 28, 75, 76
<code>\ifmmode</code>	23
<code>\ifnum</code>	14, 92, 100, 101, 113, 123, 136, 324, 325, 337
<code>\ifstrempy</code>	126, 133
<code>\ifstrequal</code>	16, 21
<code>\ifthenelse</code>	121
<code>\IfTrackedLanguageFileExists</code>	309
<code>\ifundef</code>	30, 105, 170, 171
<code>\ifx</code>	8–10, 41, 50, 51, 105, 109, 116, 117, 124, 165, 166, 168, 175, 177, 179, 273, 344
<code>\immediate</code>	92, 100, 118, 124
<code>\index</code>	165
<code>\indexspace</code>	312, 323–326, 338–344, 346, 349
<code>\input</code>	309
<code>\inputencodingname</code>	124
<code>\istfilename</code>	105
<code>\item</code>	119, 120, 311–314, 322–324, 339, 340
<b>J</b>	
<code>\jobname</code>	118–124
<b>K</b>	
<code>\key@ifundefined</code>	11, 12, 26, 72, 128, 131
<code>\KV@glshlink@hyperfalse</code>	61, 74, 79, 80
<code>\KV@glshlink@hypertrue</code>	80
<code>\KV@glshlink@noindexfalse</code>	74, 75
<code>\KV@glshlink@noindextrue</code>	75, 80
<b>L</b>	
<code>\LaTeX</code>	119, 120
<code>\leaders</code>	311
<code>\leavevmode</code>	34, 57
<code>\let</code>	5, 7–13, 15, 17–19, 23–25, 27, 29, 33, 45, 46, 49, 50, 52, 53, 55–71, 73–78, 80, 81, 84, 90, 91, 99–106, 108–115, 122, 124, 128, 129, 131, 136, 157–166, 170, 171, 175–179, 182–191, 194, 196, 205, 227, 241, 262, 290–293, 322, 323, 327, 329, 340, 347–349
<code>\letabbreviationstyle</code>	204, 206, 207, 209, 212, 213, 219, 220, 233, 235, 251, 252
<code>\letcs</code>	26, 30, 39, 40, 44, 58, 72, 112–114, 130, 131, 157–164, 350
<code>\levelchar</code>	168
<code>\listadd</code>	96

<code>\listbreak</code> .....	170	<code>\newacronymhook</code> .....	102
<code>\listcsadd</code> .....	28	<code>\newacronymstyle</code> .....	103, 104
<code>\listcseadd</code> .....	29, 97	<code>\newcommand</code> .....	5–8, 10–12, 14–40,
<code>\listcsgadd</code> .....	29, 45		42, 44–50, 52–54, 56, 57, 60, 61, 72, 73,
<code>\listcsxadd</code> .....	29, 97		75–77, 79, 80, 83–92, 94–104, 108–114,
<code>\loadglsentries</code> .....	46, 119		116–127, 129–156, 162–165, 167–178,
<code>\long</code> .....	33		180–192, 194–197, 199, 201, 202, 208,
			214, 228, 242, 264, 265, 273, 274, 276,
			278, 279, 282, 284, 287, 289–310, 312,
			322, 326–329, 336, 337, 345, 346, 349, 350
<b>M</b>			
<code>\MakeAcronymsAbbreviations</code> .....	104	<code>\newcount</code> .....	14, 122, 132
<code>\makeatletter</code> .....	118, 122, 167	<code>\newentry</code> .....	18
<code>\makeatother</code> .....	167	<code>\newglossary</code> .....	16, 105
<code>\makebox</code> .....	311, 337, 338	<code>\newglossaryentry</code> .....	
<code>\makefirstuc</code> .....	170		18, 46, 90, 98, 102, 154, 155, 179
<code>makeglossaries</code> .....	111	<code>\newglossaryentry options</code>	
<code>\makeglossaries</code> ....	105, 118, 120, 121, 125	alias .....	15, 38, 41–44
<code>\makeglossary</code> .....	105	desc .....	144, 149
<code>makeindex</code> .....	351	descplural .....	145, 149
<code>makeindex</code> .....	104	first ....	78, 142, 148, 198, 297–299, 305, 351
<code>\makenoidxglossaries</code> .....	120	firstplural ..	142, 143, 148, 198, 298, 306, 351
<code>\MakeTextUppercase</code> .....	292	group .....	130, 131
<code>\MakeUppercase</code> .....	291, 292	loclist .....	28
<code>\marginpar</code> .....	23	long .....	147, 150, 306
<code>\markboth</code> .....	291	longplural .....	147, 150, 307
<code>\markright</code> .....	290	name .....	39, 40, 140, 147, 165, 295, 303
<code>\maxdimen</code> .....	50, 51	plural .....	141, 148, 198, 296, 297, 304
<code>\mbox</code> .....	313, 314, 337	see .....	15, 24, 38, 39, 41, 44, 46, 106
<code>\medskip</code> .....	121, 130	seealso .....	15, 38, 40, 41, 43, 44, 362
<code>\MessageBreak</code> 46, 49, 92, 101, 105, 108, 109, 195		short .....	146, 150, 176
<code>mfistuc package</code> .....	170	shortplural .....	146, 150, 176
<code>\mfistucMakeUppercase</code> .....	61–	symbol .....	143, 148, 149
	71, 73, 82, 83, 85, 88, 95, 103, 140–150,	symbolplural .....	143, 144, 149
	159, 160, 164, 183, 185, 186, 188, 190–194	text ....	78, 141, 148, 198, 200, 295, 296, 304
<code>\mfu@checkword@arg</code> .....	170	<code>\newglossarystyle</code> .....	346
<code>\mfu@checkword@do</code> .....	171	<code>\newif</code> .....	57, 164, 198
<b>N</b>			
<code>\NeedsTeXFormat</code> .....	5, 310, 345	<code>\newlength</code> .....	327
<code>\new@glossaryentry</code> .....	46, 108	<code>\newnum</code> .....	18
<code>\new@ifnextchar</code> .....	27,	<code>\newrobustcmd</code> ....	27, 28, 30–32, 40, 41,
	72, 73, 94, 95, 133, 137–139, 174, 182–191		72, 73, 84, 85, 94, 95, 110, 114, 125, 126,
<code>\newabbr</code> .....	17, 18		129, 130, 133, 134, 137–139, 163, 170,
<code>\newabbreviation</code> .....	17, 18		171, 182–192, 273, 291, 293–302, 329–335
<code>\newabbreviationhook</code> .....	179	<code>\newsym</code> .....	18
<code>\newabbreviationstyle</code> ..	198–202, 204,	<code>\newterm</code> .....	154
	206–221, 223, 224, 226, 228, 230–233,	<code>\newtoks</code> .....	176
	235, 237, 239, 240, 243–249, 251–253,	<code>\newwrite</code> .....	105
	255–258, 260, 262, 265–268, 270–272,	<code>\nobreak</code> .....	313, 314, 323–326, 339–342
	274–276, 278, 279, 281–284, 286, 287, 289	<code>\NoCaseChange</code> .....	85–88, 126, 293–302
<code>\newacronym</code> .....	102, 103		

<code>\noexpand</code> ..	10, 11, 21, 40, 42, 43, 102, 118, 122, 128, 129, 131, 166, 179, 310, 347, 348
<code>\nofiles</code> .....	121
<code>\noindent</code> .....	121, 325, 326, 340–343
<code>\nopagebreak</code> .....	323–326, 338–345, 349
<code>\nopostdesc</code> .....	34, 47–49, 109, 155
<code>\nr</code> .....	7, 12, 14, 19, 20, 22, 57, 110
<code>\ns@GLSxtrfull</code> .....	183
<code>\ns@Glsxtrfull</code> .....	183
<code>\ns@glxtrfull</code> .....	182
<code>\ns@GLSxtrfullpl</code> .....	185
<code>\ns@Glsxtrfullpl</code> .....	184
<code>\ns@glxtrfullpl</code> .....	184
<code>\ns@GLSxtrlong</code> .....	188
<code>\ns@Glsxtrlong</code> .....	187
<code>\ns@glxtrlong</code> .....	187
<code>\ns@GLSxtrlongpl</code> .....	191
<code>\ns@Glsxtrlongpl</code> .....	190, 191
<code>\ns@glxtrlongpl</code> .....	190
<code>\ns@GLSxtrshort</code> .....	186
<code>\ns@Glsxtrshort</code> .....	186
<code>\ns@glxtrshort</code> .....	185
<code>\ns@GLSxtrshortpl</code> .....	189
<code>\ns@Glsxtrshortpl</code> .....	189
<code>\ns@glxtrshortpl</code> .....	188
<code>\null</code> .....	20
<code>\number</code> .....	97–100, 122, 133
<code>\numexpr</code> .....	97, 100
<b>O</b>	
<code>\or</code> .....	7, 13, 19, 20, 22, 46, 57, 323, 348
<code>\org@glossaryentrynumbers</code> .....	51, 109
<code>\org@glossarytitle</code> .....	109
<code>\org@ifKV@glslink@hyper</code> .....	57, 59
<b>P</b>	
<code>\p@gl@s@hyp@opt</code> .....	77
package options:	
abbreviations .....	16, 17
accsupp .....	20, 140
acronym .....	17
automake .....	108, 119, 124
true .....	124
autoseeindex .....	24
false .....	23
debug	
showtargets .....	79
docdef .....	14, 45, 46, 90, 98
false .....	46
restricted .....	14
true .....	46
docdefs	
restricted .....	45
nonumberlist .....	51
nopostdot .....	15
false .....	15
numbers .....	18
postdot .....	15
record .....	7, 12, 45, 54, 105, 122, 360
alsoindex .....	8, 10
only .....	7, 8
shortcuts .....	19
ac .....	19
all .....	19
false .....	19
none .....	19
true .....	19
sort	
use .....	60
style .....	21
stylemods .....	21
symbols .....	18, 155
undefaction .....	37
error .....	6
warn .....	6
xindy .....	40, 41
<code>\PackageError</code> .....	6, 11, 21, 24, 46, 49, 50, 72, 73, 75, 84, 85, 90, 91, 98, 100, 101, 103, 105, 107, 108, 115, 124, 129, 130, 133, 195–197, 310, 311
<code>\PackageWarning</code> .....	15
<code>\PackageWarningNoLine</code> .....	15
<code>\pageref</code> .....	32
<code>\par</code> ..	121, 122, 313, 314, 323–327, 337–343, 346
<code>\parindent</code> .....	322, 324, 325, 327, 337, 338, 340–344, 347
<code>\parskip</code> .....	322, 324, 325, 340–342, 347
<code>\PassOptionsToPackage</code> .....	5, 21
<code>\pdfbookmark</code> .....	346
<code>\preglossarypreamble</code> .....	32
<code>\preto</code> .....	75
<code>\print@noop@unsrtglossaryunit</code> ...	11, 13
<code>\print@op@unsrtglossaryunit</code> .....	13
<code>\printabbreviations</code> .....	17
<code>\printglossaries</code> .....	106, 120
<code>\printglossary</code> .....	17, 106, 120
<code>\printglossary options</code>	
nonumberlist .....	53
type .....	108



90–92, 98, 100, 101, 103–109, 118–125, 129, 130, 132, 133, 158, 160–163, 165, 349	tracklang package . . . . . 124
<b>U</b>	
\strut . . . . . 311–321, 325	\u . . . . . 132
\subglossentry . . . . .	\undef . . . . . 13, 169
. . . . 109, 131, 311–321, 323–325, 337, 348	\underline . . . . . 171
\subitem . . . . . 322, 323	\unskip . . . . . 34, 45, 311
\subsubitem . . . . . 323	\usepackage . . . . . 120, 121
<b>T</b>	
\tablehead . . . . . 318–321	<b>V</b>
\tabletail . . . . . 318–321	\val . . . . . 7, 12, 14, 19, 20, 22, 57, 110
\tabularnewline . . . . . 314–322	<b>W</b>
\TeX . . . . . 119	\warn@nomakeglossaries . . . . . 106
\texorpdfstring . . . . 28, 85–88, 291, 302–308	\warn@noprintglossary . . . . . 106, 109
textcase package . . . . . 290	\write . . . . . 40, 92, 100, 105, 118, 124
\textsc . . . . . 214	<b>X</b>
\textsmaller . . . . . 228	\x . . . . . 117
\texttt . . . . . 23, 119–121	\xcapitalisewords . . . . . 156
\the . . . . . 102, 103, 117, 123, 168, 179, 180, 198–206, 208–211, 213–219, 221, 225–227, 229–236, 239–241, 243–251, 253, 255, 256, 258, 260, 262, 265–268, 270–272, 274–276, 278, 279, 281–287, 289	\xdef . . . . . 109, 129
\theHglentrycounter . . . . 8–10, 58, 59, 136	\xifinlist . . . . . 96
\theHglentrycounter . . . . 8–10, 58, 59, 136	\xifinlistcs . . . . . 29
\theindex . . . . . 164	xindy . . . . . 351
\this@dialect . . . . . 309	xindy . . . . . 104
\thisgrptitle . . . . . 349	xkeyval package . . . . . 5
\toks@ . . . . . 117, 168	\XKV@checkchoice . . . . . 53
	\XKV@plfalse . . . . . 53
	\XKV@resa . . . . . 53
	\XKV@sttrue . . . . . 53