

# `glossaries-extra.sty v1.21: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2017-11-03

## 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	32
1.3.1	Existence Checks	36
1.3.2	Document Definitions	43
1.3.3	Existing Glossary Style Modifications	48
1.3.4	Entry Formatting, Hyperlinks and Indexing	53
1.3.5	Entry Counting	88
1.3.6	Acronym Modifications	101
1.3.7	Indexing and Displaying Glossaries	103
1.3.8	Support for bib2gls	129
1.4	Integration with glossaries-accsupp	137
1.5	Categories	147
1.6	Abbreviations	171
1.6.1	Abbreviation Styles Setup	190
1.6.2	Predefined Styles (Default Font)	193
1.6.3	Predefined Styles (Small Capitals)	209
1.6.4	Predefined Styles (Fake Small Capitals)	223
1.6.5	Predefined Styles (Emphasized)	237
1.6.6	Predefined Styles (User Parentheses Hook)	259
1.6.7	Predefined Styles (Hyphen)	268
1.6.8	Predefined Styles (No Short on First Use)	282
1.7	Using Entries in Headings	284
1.8	Multi-Lingual Support	304
<b>2</b>	<b>Style Adjustments (glossaries-extra-stylemods.sty)</b>	<b>305</b>
2.1	Package Initialisation	305
2.2	List-Like Styles	306
2.3	Longtable Styles	309
2.4	Long Ragged Styles	311
2.5	Supertabular Styles	313
2.6	Super Ragged Styles	315
2.7	Inline Style	317
2.8	Tree Styles	317
2.9	Multicolumn Styles	333

<b>3 bookindex style (glossary-bookindex.sty)</b>	<b>339</b>
3.1 Package Initialisation and Options . . . . .	339
<b>Glossary</b>	<b>345</b>
<b>Change History</b>	<b>346</b>
<b>Index</b>	<b>361</b>

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

## 1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/11/03 v1.21 (NLCT)]

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

Requires etoolbox package.
4 \RequirePackage{etoolbox}

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

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

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

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

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

48 \newcommand*{\@glstr@do@redef@for@gl@sentries}{%
49   \renewcommand*{\for@gl@sentries}[3][\gl@defaulttype]{%
50     \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}{glsnumberformat}%

```

`\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\@gls@label{\glsdetoklabel{#1}}%
87 \let\glslabel\@gls@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 `\@gls@field@link` and commands like `\@gls@` (before the existence test). This means that it disregards the `wrgloss` key.

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

```

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 for easier integration
358%with \cs{glsxtrbookindexprelocation} provided by
359%\sty{glossary-bookindex}.
360%\changes{1.21}{2017-11-03}{new}
361% \begin{macrocode}
362\define@key{glossaries-extra.sty}{postpunc}{%
363 \glsxtr@doption{nopostdot=false}%
364 \ifstrequal{#1}{dot}%
365 {%
366 \renewcommand*{\@glsxtr@defpostpunc}{%
367 \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
368 }%
369 }%
370 {%
371 \ifstrequal{#1}{comma}%
372 {%
373 \renewcommand*{\@glsxtr@defpostpunc}{%
374 \renewcommand*{\glspostdescription}{,}%
375 }%
376 }%
377 {%
378 \ifstrequal{#1}{none}%
379 {%
380 \glsxtr@doption{nopostdot=true}%
381 \renewcommand*{\@glsxtr@defpostpunc}{%
382 \renewcommand*{\glspostdescription}{}%
383 }%
384 }%
385 {%
386 \renewcommand*{\@glsxtr@defpostpunc}{%
387 \renewcommand*{\glspostdescription}{#1}%
388 }%
389 }%
390 }%
391 }%
392}

```

`glsxtrabbrvtype` Glossary type for abbreviations.

```
393 \newcommand*{\glsxtrabbrvtype}{\glsdefaulttype}
```

`bbreviationsdef` Set by abbreviations option.

```
394 \newcommand*{\@glsxtr@abbreviationsdef}{}

```

`bbreviationsdef`

```

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

```



```

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

If the acronym option hasn't been used, change \acronymtype to \glxtrabbrvtype.
405 \ifglxsacronym
406 \else
407   \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
408 \fi
409 }%

```

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

```

410 \@glxtr@declareoption{abbreviations}{%
411   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
412 }

```

**AbbreviationShortcuts** Enable shortcut commands for the abbreviations. Unlike the analogous command provided by glossaries, this uses \newcommand instead of \let as a safety feature.

```

413 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
414   \newcommand*{\ab}{\cglS}%
415   \newcommand*{\abp}{\cglSpl}%
416   \newcommand*{\as}{\glxtrshort}%
417   \newcommand*{\asp}{\glxtrshortpl}%
418   \newcommand*{\al}{\glxtrlong}%
419   \newcommand*{\alp}{\glxtrlongpl}%
420   \newcommand*{\af}{\glxtrfull}%
421   \newcommand*{\afp}{\glxtrfullpl}%
422   \newcommand*{\Ab}{\cGLS}%
423   \newcommand*{\Abp}{\cGLSpl}%
424   \newcommand*{\As}{\Glsxtrshort}%
425   \newcommand*{\Asp}{\Glsxtrshortpl}%
426   \newcommand*{\Al}{\Glsxtrlong}%
427   \newcommand*{\Alp}{\Glsxtrlongpl}%
428   \newcommand*{\Af}{\Glsxtrfull}%
429   \newcommand*{\Afp}{\Glsxtrfullpl}%
430   \newcommand*{\AB}{\cGLS}%
431   \newcommand*{\ABP}{\cGLSpl}%
432   \newcommand*{\AS}{\GLSxtrshort}%
433   \newcommand*{\ASP}{\GLSxtrshortpl}%
434   \newcommand*{\AL}{\GLSxtrlong}%
435   \newcommand*{\ALP}{\GLSxtrlongpl}%
436   \newcommand*{\AF}{\GLSxtrfull}%
437   \newcommand*{\AFP}{\GLSxtrfullpl}%
438   \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

439 \let\GlsXtrDefineAbbreviationShortcuts\relax
440 }

```

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

```

441 \newcommand*{\GlsXtrDefineAcShortcuts}{%
442   \newcommand*{\ac}{\cglS}%
443   \newcommand*{\acp}{\cglSpl}%
444   \newcommand*{\acs}{\glSxtrshort}%
445   \newcommand*{\acsp}{\glSxtrshortpl}%
446   \newcommand*{\acl}{\glSxtrlong}%
447   \newcommand*{\aclp}{\glSxtrlongpl}%
448   \newcommand*{\acf}{\glSxtrfull}%
449   \newcommand*{\acfp}{\glSxtrfullpl}%
450   \newcommand*{\Ac}{\cGls}%
451   \newcommand*{\Acp}{\cGlspl}%
452   \newcommand*{\Acs}{\Glsxtrshort}%
453   \newcommand*{\Acsp}{\Glsxtrshortpl}%
454   \newcommand*{\Acl}{\Glsxtrlong}%
455   \newcommand*{\Aclp}{\Glsxtrlongpl}%
456   \newcommand*{\Acf}{\Glsxtrfull}%
457   \newcommand*{\Acfp}{\Glsxtrfullpl}%
458   \newcommand*{\AC}{\cGLS}%
459   \newcommand*{\ACP}{\cGLSpl}%
460   \newcommand*{\ACS}{\GLSxtrshort}%
461   \newcommand*{\ACSP}{\GLSxtrshortpl}%
462   \newcommand*{\ACL}{\GLSxtrlong}%
463   \newcommand*{\ACLP}{\GLSxtrlongpl}%
464   \newcommand*{\ACF}{\GLSxtrfull}%
465   \newcommand*{\ACFP}{\GLSxtrfullpl}%
466   \newcommand*{\newabbr}{\newabbreviation}%

```

Disable this command after it's been used.

```

467   \let\GlsXtrDefineAcShortcuts\relax
468 }

```

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

```

469 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
470   \newcommand*{\newentry}{\newglossaryentry}%
471   \ifdef\printsymbols
472     {%
473       \newcommand*{\newsym}{\glSxtrnewsymbol}%
474     }{}%
475   \ifdef\printnumbers
476     {%
477       \newcommand*{\newnum}{\glSxtrnewnumber}%
478     }{}%
479   \let\GlsXtrDefineOtherShortcuts\relax
480 }

```

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.

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

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

```
482 \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).

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

```

519      \GlsXtrDefineAbbreviationShortcuts
520      \GlsXtrDefineOtherShortcuts
521    }%

522  \or % ac
523    \renewcommand*{\@glxtr@setupshortcuts}{%
524      \glsacrshortcutstrue
525      \GlsXtrDefineAcShortcuts
526    }%

    Leave none and false as last option.
527  \else % none, false
528    \renewcommand*{\@glxtr@setupshortcuts}{}%
529  \fi
530 }

```

`lsxtr@doaccsupp`

```

531 \newcommand*{\@glxtr@doaccsupp}{}

```

`accsupp` If `accsupp`, load glossaries-accsupp package.

```

532 \@glxtr@declareoption{accsupp}{%
533 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

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

```

534 \newcommand{\glxtrNoGlossaryWarning}[1]{%
535 \@glxtr@defaultnoglossarywarning{#1}%
536 }

```

`nomissingglstext` If true, suppress the text produced if the external glossary file is missing.

```

537 \define@choicekey{glossaries-extra.sty}{nomissingglstext}[\val\nr]%
538 {true,false}[true]{%
539   \ifcase\nr\relax % true
540     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
541       \null
542     }%
543   \else % false
544     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
545       \@glxtr@defaultnoglossarywarning{#1}%
546     }%
547   \fi
548 }

```

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

`lsxtr@redefstyles`

```

549 \newcommand*{\@glxtr@redefstyles}{}

```

stylemods

```
550 \define@key{glossaries-extra.sty}{stylemods}[default]{%
551   \ifstrequal{#1}{default}%
552   {%
553     \renewcommand*{\@glxtr@redefstyles}{%
554       \RequirePackage{glossaries-extra-stylemods}}%
555   }%
556   {%
557     \ifstrequal{#1}{all}%
558     {%
559       \renewcommand*{\@glxtr@redefstyles}{%
560         \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
561         \RequirePackage{glossaries-extra-stylemods}%
562       }%
563     }%
564     {%
565       \renewcommand*{\@glxtr@redefstyles}{}%
566       \@for\@glxtr@tmp:=#1\do{%
567         \IfFileExists{glossary-\@glxtr@tmp.sty}%
568         {%
569           \eappto\@glxtr@redefstyles{%
570             \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
571         }%
572         {%
573           \PackageError{glossaries-extra}%
574             {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
575              doesn’t exist (did you mean to use the ‘style’ key?)}%
576             {The list of values (#1) in the ‘stylemods’ key should
577              match the glossary-xxx.sty files provided with
578              glossaries.sty}%
579         }%
580       }%
581       \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
582     }
583   }%
584 }
```

glxtr@do@style

```
585 \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.

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

Defer actual style change:

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

Set this as the default style:

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

Set this style:

```
589 \setglossarystyle{#1}%  
590 }%  
591 }
```

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

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

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

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

sxtrwrglossmark Does nothing by default.

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

debug Provide extra debug options.

```
596 \define@choicekey{glossaries-extra.sty}{debug}[\val\nr]%  
597 {true,false,showtargets,showwrgloss,all}[true]{%  
598 \ifcase\nr\relax % true  
599 \glxtr@doption{debug=true}%  
600 \renewcommand*{\@glxtrwrglossmark}{}%  
601 \or % false  
602 \glxtr@doption{debug=false}%  
603 \renewcommand*{\@glxtrwrglossmark}{}%  
604 \or % showtargets  
605 \glxtr@doption{debug=showtargets}%  
606 \or % showwrgloss  
607 \glxtr@doption{debug=true}%  
608 \renewcommand*{\@glxtrwrglossmark}{\glxtrwrglossmark}%  
609 \or % all  
610 \glxtr@doption{debug=showtargets}%  
611 \renewcommand*{\@glxtrwrglossmark}{\glxtrwrglossmark}%  
612 \fi  
613 }
```

Pass all other options to glossaries.

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

Process options.

```
616 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
617 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
618 \@glxtr@doaccsupp
```

Redefine \glspostdescription if required.

```
619 \@glstr@defpostpunc
```

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

```
620 \def\glsshowtarget#1{%
621   \glstrtitleorpdforheading
622   {%
623     \ifmmode
624       \texttt{\small [#1]}%
625     \else
626       \ifinner
627         \texttt{\small [#1]}%
628       \else
629         \marginpar{\texttt{\small #1}}%
630       \fi
631     \fi
632   }%
633   {[#1]}%
634   {\texttt{\small [#1]}}%
635 }
```

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

```
636 \let\@glstr@org@doseeglossary\@do@seeglossary
```

r@doseeglossary

```
637 \newcommand*{\@glstr@doseeglossary}[2]{%
638   \glsdofexists{#1}%
639   {%
640     \@glstrwrglossmark
641     \@glstr@org@doseeglossary{#1}{#2}%
642   }%
643 }
```

oindex@glossary

```
644 \newcommand*{\@glstr@dosee@alsoindex@glossary}[2]{%
645   \@glstr@recordsee{#1}{#2}%
646   \@glstr@doseeglossary{#1}{#2}%
647 }
```

@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.)

```
648 \let\@glstr@org@gloautosee\@glo@autosee
```

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

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

```

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

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the au-
toseeindex option.
658 \ifdef\@glo@autosee
659 {%
660 \renewcommand*{\@glo@autosee}{%
661 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
662 }%
663 {}

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.
664 \renewcommand*{\gls@checkseeallowed}{%
665 \if@glxtr@autoseeindex\@gls@see@noindex\fi
666 }

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

Setup shortcuts if required.
669 \@glxtr@setupshortcuts

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

Now define the user command:
672 \newcommand*{\glossariesextrasetup}[1]{%
673 \let\glxtr@setup@record\relax
674 \let\@glxtr@setupshortcuts\relax
675 \let\@glxtr@redef@for@gl@sentries\relax
676 \setkeys{glossaries-extra.sty}{#1}%
677 \@glxtr@abbreviationsdef
678 \let\@glxtr@abbreviationsdef\relax
679 \@glxtr@setupshortcuts
680 \glxtr@setup@record
681 \@glxtr@redef@for@gl@sentries
682 }

```



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

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

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

```
684 \newcommand*{\glstr@@do@wrglossary}[1]{%
685   \@@glstrwrglossmark
686   \glstr@org@@do@wrglossary{#1}%
687 }
```

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

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

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

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

Set up record option if required.

```
690 \glstr@setup@record
```

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

```
691 \AtBeginDocument{%
692   \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
693   \def\@glstrundeftag{\glstrundeftag}%
694 }
```

## 1.2 Extra Utilities

`trifemptyglossary` `\glstrtrifemptyglossary{<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.)

```
695 \newcommand{\glstrtrifemptyglossary}[3]{%
696   \ifcsdef{glolist@#1}%
697   {%
698     \ifcsstring{glolist@#1}{,}{#2}{#3}%
699   }%
700   {%
701     \glstrundefaction{Glossary type '#1' doesn't exist}{}%
702     #2%
703   }%
704 }
```

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

```
705 \newcommand*{\glxtrifkeydefined}[3]{%
706   \key@ifundefined{glossentry}{#1}{#3}{#2}%
707 }
```

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

```
708 \newcommand*{\glxtrprovidestoragekey}{%
709   \@ifstar\sglsxtr@provide@storagekey\glxtr@provide@storagekey
710 }
```

vide@storagekey Unstarred version.

```
711 \newcommand*{\@glxtr@provide@storagekey}[3]{%
712   \key@ifundefined{glossentry}{#1}%
713   {%
714     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
715     \appto\@gls@keymap{,{#1}{#1}}%
716     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
717     \appto\@newglossaryentryposthook{%
718       \letcs{\@glo@tmp}{@glo@#1}%
719       \gl s@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
720     }%

```

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

```
721   \ifblank{#3}
722   {}%
723   {%
724     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
725   }%
726 }%
727 {%

```

Provide the no-link command if not already defined.

```
728   \ifblank{#3}
729   {}%
730   {%
731     \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
732   }%
733 }%
734 }
```

vide@storagekey Starred version.

```
735 \newcommand*{\sglsxtr@provide@storagekey}[1]{%
736   \key@ifundefined{glossentry}{#1}%
737   {%
738     \expandafter\newcommand\expandafter*\expandafter
739     {\csname gls@assign@#1@field\endcsname}[2]{%
740       \@gls@expand@field{##1}{#1}{##2}%
741     }%

```

```

742 }%
743 {}%
744 \@glstr@provide@addstoragekey{#1}%
745 }

```

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`

```

746 \newcommand{\GlsXtrFmtField}{useri}

```

`\GlsXtrFmtDefaultOptions`

```

747 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

`\glstrfmt` The post-link hook isn't done.

```

748 \newrobustcmd*{\glstrfmt}[3][[]]{%
749   \glsdoifexistsordo{#2}%
750   {%
751     \ifglshasfield{\GlsXtrFmtField}{#2}%
752     {%
753       \let\do@glstr@link@checkfirsthyper\relax
754       \expandafter\@glstr@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
755       {\csuse{\glscurrentfieldvalue}{#3}}%
756     }%
757   }%
758 }%
759 {#3}%
760 }

```

`\glxtrententryfmt` No link or indexing.

```

761 \ifdef\texorpdfstring
762 {
763   \newcommand*{\glxtrententryfmt}[2]{%
764     \texorpdfstring{\@glxtrententryfmt{#1}{#2}}{#2}%
765   }
766 }
767 {
768   \newcommand*{\glxtrententryfmt}{\@glxtrententryfmt}
769 }

```

`@glxtrententryfmt`

```

770 \newrobustcmd*{\@glxtrententryfmt}[2]{%
771   \glsdoifexistsordo
772   {%
773     \ifglshasfield{\GlsXtrFmtField}{#1}%
774     {%

```

```

775     \csuse{\glscurrentfieldvalue}{#2}%
776   }%
777   {#2}%
778 }%
779 {#2}%
780 }

```

**xtrfieldlistadd** 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.

```

781 \newcommand*\glxtrfieldlistadd}[3]{%
782   \listcsadd{glo@\glsdetoklabel{#1}@#2}{#3}%
783 }

```

**trfieldlistgadd** Similarly but uses `\listcsgadd`.

```

784 \newcommand*\glxtrfieldlistgadd}[3]{%
785   \listcsgadd{glo@\glsdetoklabel{#1}@#2}{#3}%
786 }

```

**trfieldlistead** Similarly but uses `\listcseadd`.

```

787 \newcommand*\glxtrfieldlistead}[3]{%
788   \listcseadd{glo@\glsdetoklabel{#1}@#2}{#3}%
789 }

```

**trfieldlistxadd** Similarly but uses `\listcsxadd`.

```

790 \newcommand*\glxtrfieldlistxadd}[3]{%
791   \listcsxadd{glo@\glsdetoklabel{#1}@#2}{#3}%
792 }

```

Now provide commands to iterate over these lists.

**fieldddolistloop**

```

793 \newcommand*\glxtrfieldddolistloop}[2]{%
794   \dolistcsloop{glo@\glsdetoklabel{#1}@#2}%
795 }

```

**ieldforlistloop**

```

796 \newcommand*\glxtrfieldforlistloop}[3]{%
797   \forlistcsloop{glo@\glsdetoklabel{#1}@#2}{#3}%
798 }

```

List element tests:

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

```

799 \newcommand*\glxtrfieldifinlist}[5]{%
800   \ifinlistcs{#3}{glo@\glsdetoklabel{#1}@#2}{#4}{#5}%
801 }

```

`rfieldxifinlist` Expands item.

```

802 \newcommand*{\glstrfieldxifinlist}[5]{%
803   \xifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
804 }

```

`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.

```

805 \newrobustcmd{\glsxtrifhasfield}{%
806   \@ifstar{\s@glsxtrifhasfield}{\@glsxtrifhasfield}%
807 }

```

`lsxtrifhasfield` Unstarred version adds grouping.

```

808 \newcommand{\@glsxtrifhasfield}[4]{%
809   {\s@glsxtrifhasfield{#1}{#2}{#3}{#4}}%
810 }

```

`lsxtrifhasfield` Starred version omits grouping.

```

811 \newcommand{\s@glsxtrifhasfield}[4]{%
812   \letcs{\glscurrentfieldvalue}{glo@\glsetoklabel{#2}@#1}%
813   \ifundef\glscurrentfieldvalue
814     {#4}%
815     {%
816       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
817     }%
818 }

```

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

```

819 \newcommand*{\glsxtrusefield}[2]{%
820   \@gls@entry@field{#1}{#2}%
821 }

```

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

```

822 \newcommand*{\Glsxtrusefield}[2]{%
823   \@gls@entry@field{#1}{#2}%
824 }

```

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

```

825 \newcommand*{\glxtrdeffield}[2]{\csdef{glo@\glsetoklabel{#1}@#2}}

```

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

```

826 \newcommand*{\glxtredeffield}[2]{\csedef{glo@\glsetoklabel{#1}@#2}}

```

`etfieldifexists`

```

827 \newcommand*{\glxtrsetfieldifexists}[3]{\glsetfieldifexists{#1}{#3}}

```

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

```
828 \newrobustcmd*{\GlsXtrSetField}[3]{%
829   \glstrsetfieldifexists{#1}{#2}%
830   {\csdef{glo@glstoklabel{#1}@#2}{#3}}%
831 }
```

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

```
832 \newrobustcmd*{\GlsXtrLetField}[3]{%
833   \glstrsetfieldifexists{#1}{#2}%
834   {\cslet{glo@glstoklabel{#1}@#2}{#3}}%
835 }
```

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

```
836 \newrobustcmd*{\csGlsXtrLetField}[3]{%
837   \glstrsetfieldifexists{#1}{#2}%
838   {\csletcs{glo@glstoklabel{#1}@#2}{#3}}%
839 }
```

`\GlsXtrLetFieldToField` 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.

```
840 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
841   \glstrsetfieldifexists{#1}{#2}%
842   {\csletcs{glo@glstoklabel{#1}@#2}{glo@glstoklabel{#3}@#4}}%
843 }
```

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

```
844 \newrobustcmd*{\gGlsXtrSetField}[3]{%
845   \glstrsetfieldifexists{#1}{#2}%
846   {\csgdef{glo@glstoklabel{#1}@#2}{#3}}%
847 }
```

`\xGlsXtrSetField`

```
848 \newrobustcmd*{\xGlsXtrSetField}[3]{%
849   \glstrsetfieldifexists{#1}{#2}%
850   {\protected@csxdef{glo@glstoklabel{#1}@#2}{#3}}%
851 }
```

`\eGlsXtrSetField`

```
852 \newrobustcmd*{\eGlsXtrSetField}[3]{%
853   \glstrsetfieldifexists{#1}{#2}%
854   {\protected@csedef{glo@glstoklabel{#1}@#2}{#3}}%
855 }
```

`\XtrIfFieldEqStr`

```
856 \newrobustcmd*{\GlsXtrIfFieldEqStr}[5]{%
```

```

857 \glstrifhasfield{#1}{#2}%
858 {%
859 \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
860 }%
861 {%#5}%
862 }

```

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

```

863 \ifglssentrycounter
864 \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstoklabel{#1}}}
865 \else
866 \ifglssubentrycounter
867 \newcommand*{\glstrpageref}[1]{\pageref{glssentry-\glstoklabel{#1}}}
868 \else
869 \newcommand*{\glstrpageref}[1]{\glss{#1}}
870 \fi
871 \fi

```

`lossarypreamble`

```

872 \newcommand{\apptoglossarypreamble}[2][\glssdefaulttype]{%
873 \ifcsdef{glolist@#1}%
874 {%
875 \ifcsundef{@glossarypreamble@#1}%
876 {\csdef{@glossarypreamble@#1}{}}%
877 {}%
878 \csappto{@glossarypreamble@#1}{#2}%
879 }%
880 {%
881 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
882 }%
883 }

```

`lossarypreamble`

```

884 \newcommand{\preglossarypreamble}[2][\glssdefaulttype]{%
885 \ifcsdef{glolist@#1}%
886 {%
887 \ifcsundef{@glossarypreamble@#1}%
888 {\csdef{@glossarypreamble@#1}{}}%
889 {}%
890 \cspreto{@glossarypreamble@#1}{#2}%
891 }%
892 {%
893 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
894 }%
895 }

```

## 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`

```
896 \renewcommand*{\longnewglossaryentry}{%
897 \ifstar\@glstr@s@longnewglossaryentry\@glstr@longnewglossaryentry
898 }
```

`ewglossaryentry` Starred version.

```
899 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
900 \glsdoifnoexists{#1}%
901 {%
902 \bgroup
903 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
904 \long\def\@newglossaryentryprehook{%
905 \long\def\@glo@desc{#3}%
906 \@org@newglossaryentryprehook
907 }%
908 \renewcommand*{\gls@assign@desc}[1]{%
909 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
910 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
911 }
912 \gls@defglossaryentry{#1}{#2}%
913 \egroup
914 }%
915 }
```

`ewglossaryentry` Unstarred version.

```
916 \newcommand{\@glstr@longnewglossaryentry}[3]{%
917 \glsdoifnoexists{#1}%
918 {%
919 \bgroup
920 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
921 \long\def\@newglossaryentryprehook{%
922 \long\def\@glo@desc{#3\glstrpostlongdescription}%
923 \@org@newglossaryentryprehook
924 }%
925 \renewcommand*{\gls@assign@desc}[1]{%
926 \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
927 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
928 }
929 \gls@defglossaryentry{#1}{#2}%
930 }
```

The following is different from the base glossaries.sty:

```
927 \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
928 }
929 \gls@defglossaryentry{#1}{#2}%
930 }
```



```

930     \egroup
931 }%
932 }

```

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

```

933 \newcommand*{\glstrpostlongdescription}{\leavevmode\unskip\nopostdesc}

```

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

`\ignoredglossary` Redefine to check for star.

```

934 \renewcommand{\newignoredglossary}{%
935   \@ifstar\glstr@s@newignoredglossary\glstr@org@newignoredglossary
936 }

```

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

```

937 \newcommand*{\glstr@org@newignoredglossary}[1]{%
938   \ifcsdef{glolist@#1}
939   {%
940     \glstrundefaction{Glossary type ‘#1’ already exists}{}%
941   }%
942   {%
943     \ifdefempty\@ignored@glossaries
944     {%
945       \edef\@ignored@glossaries{#1}%
946     }%
947     {%
948       \eappto\@ignored@glossaries{,#1}%
949     }%
950     \csgdef{glolist@#1}{,}%
951     \ifcsundef{gls@#1@entryfmt}%
952     {%
953       \defglsentryfmt[#1]{\glsentryfmt}%
954     }%
955     {}%
956     \ifdefempty\@gls@nohyperlist
957     {%
958       \renewcommand*{\@gls@nohyperlist}{#1}%
959     }%
960     {%
961       \eappto\@gls@nohyperlist{,#1}%
962     }%
963   }%
964 }

```

`\ignoredglossary` Starred form.

```

965 \newcommand*{\glstr@s@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 }%
985 }

```

`\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.

```

986 \glsifusetranslator
987 {%
988 \renewcommand*{\glsettoctitle}[1]{%
989 \ifcsdef{gls@tr@set@#1@toctitle}%
990 {%
991 \csuse{gls@tr@set@#1@toctitle}%
992 }%
993 {%
994 \ifcsdef{@glotype@#1@title}%
995 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
996 {\def\glossarytoctitle{\glossarytitle}}%
997 }%
998 }%
999 }
1000 {
1001 \renewcommand*{\glsettoctitle}[1]{%
1002 \ifcsdef{@glotype@#1@title}%
1003 {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1004 {\def\glossarytoctitle{\glossarytitle}}%
1005 }
1006 }

```

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

```

1007 \newcommand{\provideignoredglossary}{%
1008 \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1009 }

```

`ignoredglossary` Unstarred version.

```

1010 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1011   \ifcsdef{glolist@#1}
1012   {}%
1013   {%
1014     \ifdefempty\@ignored@glossaries
1015     {%
1016       \edef\@ignored@glossaries{#1}%
1017     }%
1018     {%
1019       \eappto\@ignored@glossaries{,#1}%
1020     }%
1021     \csgdef{glolist@#1}{,}%
1022     \ifcsundef{gls@#1@entryfmt}%
1023     {%
1024       \defglsentryfmt[#1]{\glsentryfmt}%
1025     }%
1026     {}%
1027     \ifdefempty\@gls@nohyperlist
1028     {%
1029       \renewcommand*{\@gls@nohyperlist}{#1}%
1030     }%
1031     {%
1032       \eappto\@gls@nohyperlist{,#1}%
1033     }%
1034   }%
1035 }

```

`ignoredglossary` Starred form.

```

1036 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1037   \ifcsdef{glolist@#1}
1038   {}%
1039   {%
1040     \ifdefempty\@ignored@glossaries
1041     {%
1042       \edef\@ignored@glossaries{#1}%
1043     }%
1044     {%
1045       \eappto\@ignored@glossaries{,#1}%
1046     }%
1047     \csgdef{glolist@#1}{,}%
1048     \ifcsundef{gls@#1@entryfmt}%
1049     {%
1050       \defglsentryfmt[#1]{\glsentryfmt}%
1051     }%
1052     {}%
1053   }%
1054 }

```

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

is glossary label.

```
1055 \newcommand*{\glstrcopytoglossary}[2]{%
1056   \glsdoidexists{#1}%
1057   {%
1058     \ifcsdef{glolist@#2}
1059     {%
1060       \cseappto{glolist@#2}{#1,}%
1061     }%
1062     {%
1063       \glstrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1064     }%
1065   }%
1066 }
```

### 1.3.1 Existence Checks

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

```
1067 \renewcommand{\glsdoidexists}[2]{%
1068   \ifglstryexists{#1}{#2}%
1069   {%
```

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

```
1070   \edef\glslabel{\glsdetoklabel{#1}}%
1071   \glstrundefaction{Glossary entry ‘\glslabel’
1072     has not been defined}{You need to define a glossary entry before
1073     you can reference it.}%
1074   }%
1075 }
```

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

```
1076 \renewcommand{\glsdoidnoexists}[2]{%
1077   \ifglstryexists{#1}{%
1078     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1079       has already been defined}{}}{#2}%
1080 }
```

`\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.

```
1081 \ifdef\glsdoidexistsordo
1082 {%
1083   \renewcommand{\glsdoidexistsordo}[3]{%
1084     \ifglstryexists{#1}{#2}%
1085     {%
1086       \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1087         has not been defined}{You need to define a glossary entry
1088         before you can use it.}%
1089       #3%
1090     }%
1091   }
```

```

1091 }%
1092 }
1093 {%
1094   \glstr@warnonexistsordo\glsoifexistsordo
1095   \newcommand{\glsoifexistsordo}[3]{%
1096     \ifglstryexists{#1}{#2}%
1097     {%
1098       \glstrundefaction{Glossary entry ‘\glstoklabel{#1}’
1099       has not been defined}{You need to define a glossary entry
1100       before you can use it.}%
1101       #3%
1102     }%
1103   }%
1104 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1105 \ifdef\doifglossarynoexistsordo
1106 {%
1107   \renewcommand{\doifglossarynoexistsordo}[3]{%
1108     \ifglossaryexists{#1}%
1109     {%
1110       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1111       #3%
1112     }%
1113     {#2}%
1114   }%
1115 }
1116 {%
1117   \glstr@warnonexistsordo\doifglossarynoexistsordo
1118   \newcommand{\doifglossarynoexistsordo}[3]{%
1119     \ifglossaryexists{#1}%
1120     {%
1121       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1122       #3%
1123     }%
1124     {#2}%
1125   }%
1126 }
1127

```

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.

```

1128 \appto\newglossaryentryposthook{%
1129   \ifdefvoid\@glo@see
1130   {\csxdef{glo@\@glo@label @see}{}}%
1131   {%

```

```

1132 \csxdef{glo@\glo@label @see}{\@glo@see}%
1133 \if@glxtr@autoseeindex
1134 \@glxtr@autoindexcrossrefs
1135 \fi
1136 }%
1137 }
1138 \appto\@gls@keymap{,{see}{see}}

```

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

```

1139 \newcommand*{\glxtrusesee}[1]{%
1140 \glsdoifexists{#1}%
1141 {%
1142 \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1143 \ifdefempty\@glo@see
1144 {}%
1145 {%
1146 \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1147 }%
1148 }%
1149 }

```

**\glxtr@usesee**

```

1150 \newcommand*{\glxtr@usesee}[1][\seename]{%
1151 \@glxtr@usesee{#1}%
1152 }

```

**\@glxtr@usesee**

```

1153 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1154 \glxtruseseeformat{#1}{#2}%
1155 }

```

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

```

1156 \newcommand*{\glxtruseseeformat}[2]{%
1157 \glsseeformat{#1}{#2}{}%
1158 }

```

**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.

```

1159 \renewcommand*{\glsseeitemformat}[1]{%
1160 \glsifregular{#1}{\glsaccessname{#1}}{\glsaccesstext{#1}}%
1161 }

```

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

```

1162 \newcommand*{\glxtruseseealso}[1]{%

```

```

1163 \glsdoifexists{#1}%
1164 {%
1165   \letcs{\@glo@see}{glo\glsdetoklabel{#1}@seealso}%
1166   \ifdefempty\@glo@see
1167   {}%
1168   {%
1169     \expandafter\glsxtruseseealsoformat\expandafter{\@glo@see}%
1170   }%
1171 }%
1172 }

```

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

```

1173 \newcommand*{\glsxtruseseealsoformat}[1]{%
1174   \glsseeformat[\seealso]{#1}%
1175 }

```

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

```

1176 \newrobustcmd{\glsxtrseelist}[1]{%
1177   \edef\@glo@tmp{\noexpand\glsseelist{#1}}\@glo@tmp
1178 }

```

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

```

1179 \providecommand{\seealso}{see also}

```

`\glsxtrindexseealso` 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.

```

1180 \ifdef\@xdycrossrefhook
1181 {
1182   Add the cross-reference class definition to the hook.
1183   \appto\@xdycrossrefhook{%
1184     \write\glswrite{(define-crossref-class \string"seealso\string"
1185       :unverified )}%
1186     \write\glswrite{(markup-crossref-list
1187       :class \string"seealso\string"^^J\space\space\space
1188       :open \string"\string\glsxtruseseealsoformat\glsopenbrace\string"
1189       :close \string"\glsclosebrace\string")}%
1190   }

```

Append to class list.

```

1190 \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

This essentially works like `\@do@seeglossary` but uses the `seealso` class.

```

1191 \newrobustcmd*{\glsxtrindexseealso}[2]{%
1192   \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
1193     \@glsxtr@recordsee{#1}{#2}%

```

```

1194 \fi
1195 \glsdoifexists{#1}%
1196 {%
1197   \@@glsxtrwrglossmark
1198   \def\@gls@xref{#2}%
1199   \@onelevel@sanitize\@gls@xref
1200   \@gls@checkmkidxchars\@gls@xref
1201   \gls@glossary{\csname glo@#1@type\endcsname}{%
1202     (indexentry
1203       :tkey (\csname glo@#1@index\endcsname)
1204       :xref (\string"\@gls@xref\string")
1205       :attr \string"seealso\string"
1206     )
1207   }%
1208 }%
1209 }
1210 }
1211 {
  xindy not in use or glossaries version too old to support this.
1212 \newrobustcmd*{\glsxtrindexseealso}{\glssee[\seealsoname]}
1213 }

```

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

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

```

1214 \ifdef\gls@set@xr@key
1215 {

```

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

```

1216 \define@key{glossentry}{alias}{%
1217   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1218 }
1219 \define@key{glossentry}{seealso}{%
1220   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1221 }

```

Add to the key mappings.

```

1222 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1223 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1224 \appto\@newglossaryentryposthook{%
1225   \ifdefvoid\@glo@seealso
1226     {\csxdef\glo@\@glo@label @seealso{}}}%

```



```

1227    {%
1228        \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1229        \if@glstr@autoseeindex
1230            \@glstr@autoindexcrossrefs
1231        \fi
1232    }%

```

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

```

1233    \ifdefvoid\@glo@alias
1234        {\csxdef{glo@\@glo@label @alias}{}}%
1235        {%
1236            \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1237        }%
1238    }

```

Provide user-level commands to access the values.

`\glstralias`

```

1239    \newcommand*\glstralias[1]{\@gls@entry@field{#1}{alias}}

```

`trseealsolabels`

```

1240    \newcommand*\glstrseealsolabels[1]{\@gls@entry@field{#1}{seealso}}

```

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

```

1241    \appto\@glo@autoseehook{%
1242        \ifdefvoid\@glo@alias
1243            {%
1244                \ifdefvoid\@glo@seealso
1245                    {}%
1246                {%
1247                    \edef\@do@glsee{\noexpand\glstrindexseealso
1248                        {\@glo@label}{\@glo@seealso}}%
1249                    \@do@glsee
1250                }%
1251            }%
1252        {%

```

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

```

1253            \ifdefvoid\@glo@see
1254                {%
1255                    \edef\@do@glsee{\noexpand\glsee{\@glo@label}{\@glo@alias}}%
1256                    \@do@glsee
1257                }%
1258            {}%
1259        }%
1260    }%
1261 }
1262 {

```

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

```

\glxtralias
1263 \glsaddstoragekey*{alias}{\glxtralias}

trseealsolabels
1264 \glsaddstoragekey*{seealso}{\glxtrseealsolabels}

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

ryentryposthook Append to the hook to check for the alias and seealso keys.
1265 \appto\@newglossaryentryposthook{%
1266 \ifcsvoid{glo@\@glo@label @alias}%
1267 {%
1268 \ifcsvoid{glo@\@glo@label @seealso}%
1269 }%
1270 {%
1271 \edef\@do@glsee{\noexpand\glxtrindexseealso
1272 {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1273 \@do@glsee
1274 }%
1275 }%
1276 {%

Add cross-reference if see key hasn't been used.
1277 \ifdefvoid\@glo@see
1278 {%
1279 \edef\@do@glsee{\noexpand\glsee
1280 {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1281 \@do@glsee
1282 }%
1283 }%
1284 }%
1285 }

1286 }

Add all unused cross-references at the end of the document.
1287 \AtEndDocument{\if@glxtrindexcrossrefs\glxtraddallcrossrefs\fi}

addallcrossrefs Iterate through all used entries and if they have a cross-reference, make sure the cross-
reference has been added.
1288 \newcommand*\glxtraddallcrossrefs{%
1289 \forallglossaries{\@glo@type}%
1290 {%
1291 \for@gl@entries[\@glo@type]{\@glo@label}%
1292 {%
1293 \if@gl@used{\@glo@label}%
1294 {\expandafter\glxtr@addunusedxrefs\expandafter{\@glo@label}}}%
1295 }%

```

```

1296 }%
1297 }

```

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

```

1298 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
1299   \letcs{\@glo@see}{glo@glstdetoklabel{#1}@see}%
1300   \ifdefvoid\@glo@see
1301   {}%
1302   {%
1303     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1304   }%
1305   \letcs{\@glo@see}{glo@glstdetoklabel{#1}@seealso}%
1306   \ifdefvoid\@glo@see
1307   {}%
1308   {%
1309     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1310   }%
1311 }

```

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

```

1312 \newcommand*{\glxtr@addunused}[1][]{%
1313   \@glxtr@addunused
1314 }

```

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

```

1315 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1316   \@for\@glxtr@label:=#1\do
1317   {%
1318     \ifglxsused{\@glxtr@label}{}%
1319     {%
1320       \glxadd[format=glxtrunusedformat]{\@glxtr@label}%
1321       \glxunset{\@glxtr@label}%
1322       \expandafter\glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1323     }%
1324   }%
1325 }

```

**xtrunusedformat**

```

1326 \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.

```

1327 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
1328 \renewcommand{\makenoidxglossaries}{%
1329   \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off

```

```

1330 {%
1331   \glsxtr@orgmakenoidxglossaries
      Add marker to \@do@seeglossary
1332   \renewcommand{\@do@seeglossary}[2]{%
1333     \@glsxtrwrglossmark
1334     \edef\@gls@label{\glsdetoklabel{##1}}%
1335     \protected@write\@auxout{}\@do@seeglossary{##1}{%
1336       \string\@gls@reference
1337       {\csname glo@\@gls@label @type\endcsname}%
1338       {\@gls@label}%
1339       {%
1340         \string\glsseeformat##2}%
1341       }%
1342     }%
1343   }%

      Check for docdefs=restricted:
1344   \if@glsxtrdocdefrestricted
      If restricted document definitions allowed, adjust \@gls@reference so that it doesn't test for
      existence.
1345   \renewcommand*\@gls@reference[3]{%
1346     \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1347     \ifinlistcs{##2}{\glsref@##1}%
1348     {}%
1349     {\listcsgadd{\glsref@##1}{##2}}%
1350     \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1351     {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1352     {}%
1353     \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1354   }%
1355   \else
      Disable document definitions.
1356   \@glsxtrdocdeffalse
1357   \fi
1358   \disable@keys{glossaries-extra}{docdef}%
1359 }%
1360 {%
1361   \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1362     not permitted\MessageBreak
1363     with record=\@glsxtr@record@setting\space package option}%
1364   {You may only use \string\makenoidxglossaries\space with the
1365     record=off option}%
1366 }%
1367 }

ewglossaryentry Modify \gls@defdocnewglossaryentry so that it checks the docdef value.
1368 \renewcommand*\@gls@defdocnewglossaryentry{%
1369   \ifcase\@glsxtr@docdefval

```

docdef=false:

```
1370 \renewcommand*{\newglossaryentry}[2]{%
1371 \PackageError{glossaries-extra}{Glossary entries must
1372 be \MessageBreak defined in the preamble with \MessageBreak
1373 package option 'docdef=false'\MessageBreak(consider using
1374 'docdef=restricted')}{Move your glossary definitions to
1375 the preamble. You can also put them in a \MessageBreak separate file
1376 and load them with \string\loadglsentries.}%
1377 }%
1378 \or
```

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

```
1379 \let\gls@checkseeallowed\relax
1380 \let\newglossaryentry\new@glossaryentry
1381 \or
```

Restricted mode just needs to allow the see value.

```
1382 \let\gls@checkseeallowed\relax
1383 \fi
1384 }%
```

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

```
1385 \newcommand*{\GlsXtrEnableOnTheFly}{%
1386 \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1387 }
```

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.)

```
1388 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1389 \renewcommand*{\glsdetoklabel}[1]{%
1390 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1391 {%
1392 \expandafter\detokenize\expandafter{##1}%
1393 }%
1394 {\detokenize{##1}}%
1395 }%
1396 \@GlsXtrEnableOnTheFly
1397 }
1398 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1399 \expandafter\if\glsbackslash#1%
1400 #3%
1401 \else
```

```

1402     #4%
1403   \fi
1404 }

```

sxtrstarflywarn

```

1405 \newcommand*{\glxtrstarflywarn}{%
1406   \GlossariesExtraWarning{Experimental starred version of
1407   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1408   read the warnings in the glossaries-extra user manual)}}%
1409 }

```

rEnableOnTheFly

```

1410 \newcommand*{\@GlsXtrEnableOnTheFly}{%

```

Don't redefine \glsetoklabel 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

```

1411   \newcommand*{\glxtrcat}{general}

```

\glxtr

```

1412   \newcommand*{\glxtr}[1] [] {%
1413     \def\glxtr@keylist{##1}%
1414     \@glxtr
1415   }

```

\@glxtr

```

1416   \newcommand*{\@glxtr}[2] [] {%
1417     \ifglentryexists{##2}%
1418     {%
1419       \ifblank{##1}{ }\{\GlsXtrWarning{##1}{##2}}%
1420     }%
1421     {%
1422       \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1423       description={\nopostdesc},##1}%
1424     }%
1425     \expandafter\gls\expandafter[\glxtr@keylist]{##2}%
1426   }

```

\Glsxtr

```

1427   \newcommand*{\Glsxtr}[1] [] {%
1428     \def\glxtr@keylist{##1}%
1429     \@Glsxtr
1430   }

```

\@Glsxtr

```

1431   \newcommand*{\@Glsxtr}[2] [] {%

```

```

1432 \ifglstryexists{##2}%
1433 {%
1434   \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1435 }%
1436 {%
1437   \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1438     description={\nopostdesc},##1}%
1439 }%
1440 \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
1441 }

```

\glstrpl

```

1442 \newcommand*\glstrpl[1][]{%
1443   \def\glstr@keylist{##1}%
1444   \@glstrpl
1445 }

```

\@glstrpl

```

1446 \newcommand*\@glstrpl[2][]{%
1447   \ifglstryexists{##2}%
1448   {%
1449     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1450   }%
1451   {%
1452     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1453       description={\nopostdesc},##1}%
1454   }%
1455   \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
1456 }

```

\Glsxtrpl

```

1457 \newcommand*\Glsxtrpl[1][]{%
1458   \def\glstr@keylist{##1}%
1459   \@Glsxtrpl
1460 }

```

\@Glsxtrpl

```

1461 \newcommand*\@Glsxtrpl[2][]{%
1462   \ifglstryexists{##2}
1463   {%
1464     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1465   }%
1466   {%
1467     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1468       description={\nopostdesc},##1}%
1469   }%
1470   \expandafter\Glspl\expandafter[\glstr@keylist]{##2}%
1471 }

```

```

\GlsXtrWarning
1472 \newcommand*{\GlsXtrWarning}[2]{%
1473   \def\@glxtr@optlist{##1}%
1474   \@onelevel@sanitize\@glxtr@optlist
1475   \GlossariesExtraWarning{The options ‘\@glxtr@optlist’ have
1476     been ignored for entry ‘##2’ as it has already been defined}%
1477 }

```

Disable commands after the glossary:

```

1478 \renewcommand\@printglossary[2]{%
1479   \def\@glxtr@printglossopts{##1}%
1480   \@glxtr@orgprintglossary{##1}{##2}%
1481   \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
1482   \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1483   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1484   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1485 }

```

abledflycommand

```

1486 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1487   \PackageError{glossaries-extra}%
1488   {\string##1\space can't be used after any of the \MessageBreak
1489     glossaries have been displayed}%
1490   {The on-the-fly commands enabled by
1491     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1492     before the glossaries. If you want to use any entries \MessageBreak
1493     after any of the glossaries, you must use the standard \MessageBreak
1494     method of first defining the entry and then using the \MessageBreak
1495     entry with commands like \string\gls}%
1496   \@glxtr@disabledflycommand
1497 }%
1498 \newcommand*{\@glxtr@disabledflycommand}[2][{}]{##2}

```

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

```

1499 \let\GlsXtrEnableOnTheFly\relax
1500 }
1501 \@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.

\currentstyle Initialise the current style to the default style.

```

1502 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

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



etglossarystyle

```
1503 \renewcommand*{\setglossarystyle}[1]{%
1504   \ifcsundef{@glsstyle@#1}%
1505   {%
1506     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1507   }%
1508   {%
1509     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
1510   \protected@edef\@glsxtr@current@style{#1}%
1511 }%
1512 \ifx\@glossary@default@style\relax
1513   \protected@edef\@glossary@default@style{#1}%
1514 \fi
1515 }
```

In case we have an old version of glossaries:

```
1516 \ifdef\@glossary@default@style
1517 {}
1518 {%
1519   \let\@glossary@default@style\relax
1520 }
```

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

```
1521 \ifdef\glslistdottedwidth
1522 {%
1523   \ifdim\glslistdottedwidth=.5\hsize
1524     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1525     \AtBeginDocument{%
1526       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1527         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1528       \fi
1529     }%
1530   \fi
1531 }
1532 {}%
```

Similarly for \glsdescwidth:

\glsdescwidth

```
1533 \ifdef\glsdescwidth
1534 {%
1535   \ifdim\glsdescwidth=.6\hsize
1536     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1537     \AtBeginDocument{%
1538       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1539         \setlength{\glsdescwidth}{.6\columnwidth}%
1540       \fi
1541     }%
1542   \fi
1543 }
```

```

1540     \fi
1541   }%
1542 \fi
1543 }
1544 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

1545 \ifdef\glspagelistwidth
1546 {%
1547   \ifdim\glspagelistwidth=.1\hsize
1548     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1549     \AtBeginDocument{%
1550       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1551         \setlength{\glspagelistwidth}{.1\columnwidth}%
1552       \fi
1553     }%
1554   \fi
1555 }
1556 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```

1557 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1558 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1559   \glsnonumberlistfalse
1560   \renewcommand*{\glossaryentrynumbers}[1]{%
1561     \ifglentryexists{\glscurrententrylabel}%
1562     {%
1563       \@glxtrpreloctag
1564       \GlsXtrFormatLocationList{#1}%
1565       \@glxtrpostloctag
1566       \gls@save@numberlist{#1}%
1567     }{}%
1568   }%
1569 \else
1570   \glsnonumberlisttrue
1571   \renewcommand*{\glossaryentrynumbers}[1]{%
1572     \ifglentryexists{\glscurrententrylabel}%
1573     {%
1574       \gls@save@numberlist{#1}%
1575     }{}%
1576   }%
1577 \fi

```

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

```

1578 \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`

```

1579 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1580   \let\@glxstrpreloctag\@glxstrpreloctag
1581   \let\@glxstrpostloctag\@glxstrpostloctag
1582   \renewcommand*{\@glxstr@pagetag}{#1}%
1583   \renewcommand*{\@glxstr@pagetag}{#2}%
1584   \renewcommand*{\@glxstr@savepreloctag}[2]{%
1585     \csgdef{\@glxstr@preloctag##1}{##2}%
1586   }%
1587   \renewcommand*{\@glxstr@doloctag}{%
1588     \ifcsundef{\@glxstr@preloctag\@glscurrententrylabel}%
1589     {%
1590       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
1591         Rerun required}%
1592     }%
1593     {%
1594       \csuse{\@glxstr@preloctag\@glscurrententrylabel}%
1595     }%
1596   }%
1597 }
1598 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`glxstrpreloctag`

```

1599 \newcommand*{\@glxstrpreloctag}{%
1600   \let\@glxstr@org@delimN\delimN
1601   \let\@glxstr@org@delimR\delimR
1602   \let\@glxstr@org@glsgignore\glsgignore
1603   \gdef is required as the delimiters may occur inside a scope.
1604   \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1605   \renewcommand*{\delimN}{%
1606     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1607     \@glxstr@org@delimN}%
1608   \renewcommand*{\delimR}{%
1609     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1610     \@glxstr@org@delimR}%
1611   \renewcommand*{\glsgignore}[1]{%
1612     \gdef\@glxstr@thisloctag{\relax}%
1613     \@glxstr@org@glsgignore{##1}}%
1614   \@glxstr@doloctag

```

`glxstrpreloctag`

```

1615 \newcommand*{\@glxstrpreloctag}{}

```

@glsxtr@pagetag

```
1616 \newcommand*{\@glsxtr@pagetag}{}%
```

glsxtr@pagetag

```
1617 \newcommand*{\@glsxtr@pagetag}{}%
```

lsxtrpostloctag

```
1618 \newcommand*{\@glsxtrpostloctag}{%
1619   \let\delimN\@glsxtr@org@delimN
1620   \let\delimR\@glsxtr@org@delimR
1621   \let\glsignore\@glsxtr@org@glsignore
1622   \protected@write\@auxout{}{%
1623     {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
1624 }
```

lsxtrpostloctag

```
1625 \newcommand*{\@glsxtrpostloctag}{}
```

lsxtr@preloctag

```
1626 \newcommand*{\@glsxtr@savepreloctag}[2]{%
1627 \protected@write\@auxout{}{%
1628   \string\providecommand\string\@glsxtr@savepreloctag[2]{}}}
```

glsxtr@doloctag

```
1629 \newcommand*{\@glsxtr@doloctag}{}
```

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

```
1630 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1631   \XKV@plfalse
1632   \XKV@sttrue
1633   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1634   {%
1635     \csname glsnonumberlist\XKV@resa\endcsname
1636     \ifglsnonumberlist
1637       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1638     \else
1639       \def\glossaryentrynumbers##1{%
1640         \@glsxtrpreloctag
1641         \GlsXtrFormatLocationList{##1}%
1642         \@glsxtrpostloctag
1643         \gls@save@numberlist{##1}}%
1644     \fi
1645   }%
1646 }
```

### 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.

```
1647 \renewcommand*{\glsentryfmt}{%
1648   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}{}%
1649   \glsifregular{\glslabel}%
1650   {\glsxtrregularfont{\glsentryfmt}}%
1651   {%
1652     \ifglshasshort{\glslabel}%
1653     {\glsxtrgenabbrvfmt}%
1654     {\glsxtrregularfont{\glsentryfmt}}%
1655   }%
1656 }
```

`sxtrregularfont` Font used for regular entries.

```
1657 \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.

```
1658 \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).

```
1659 \@glsxtr@record{#2}{#3}{glslink}%
1660 \glsdoifexists{#3}%
1661 {%
```

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).

```
1662 \let\glsxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1663 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1664 \def\glscustomtext{#4}%
1665 \@glsxtr@field@linkdefs
1666 #1%
1667 \@gls@link[#2]{#3}{#4}%
1668 \let\ifKV@glslink@hyper\glsxtrorg@ifKV@glslink@hyper
1669 }%
```

```

1670 \glspostlinkhook
1671 }

```

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

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

```

1672 \let\@glsxtr@org@gls@\@gls@
1673 \def\@gls@#1#2{%
1674   \@glsxtr@record{#1}{#2}{glslink}%
1675   \@glsxtr@org@gls@{#1}{#2}%
1676 }%

```

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

```

1677 \let\@glsxtr@org@glspl@\@glspl@
1678 \def\@glspl@#1#2{%
1679   \@glsxtr@record{#1}{#2}{glslink}%
1680   \@glsxtr@org@glspl@{#1}{#2}%
1681 }%

```

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

```

1682 \let\@glsxtr@org@Gls@\@Gls@
1683 \def\@Gls@#1#2{%
1684   \@glsxtr@record{#1}{#2}{glslink}%
1685   \@glsxtr@org@Gls@{#1}{#2}%
1686 }%

```

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

```

1687 \let\@glsxtr@org@Glspl@\@Glspl@
1688 \def\@Glspl@#1#2{%
1689   \@glsxtr@record{#1}{#2}{glslink}%
1690   \@glsxtr@org@Glspl@{#1}{#2}%
1691 }%

```

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

```

1692 \let\@glsxtr@org@GLS@\@GLS@
1693 \def\@GLS@#1#2{%
1694   \@glsxtr@record{#1}{#2}{glslink}%
1695   \@glsxtr@org@GLS@{#1}{#2}%
1696 }%

```

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

```

1697 \let\@glsxtr@org@GLSpl@\@GLSpl@
1698 \def\@GLSpl@#1#2{%
1699   \@glsxtr@record{#1}{#2}{glslink}%
1700   \@glsxtr@org@GLSpl@{#1}{#2}%
1701 }%

```

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

```

1702 \renewcommand*{\@glsdisp}[3][\]{%
1703   \@glsxtr@record{#1}{#2}{glslink}%
1704   \glsdoifexists{#2}{%
1705     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
1706     \let\glsifplural\@secondoftwo
1707     \let\glscapscase\@firstofthree
1708     \def\glscustomtext{#3}%
1709     \def\glsinsert{}%
1710     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1711     \@gls@link[#1]{#2}{\@glo@text}%
1712     \ifKV@glslink@local
1713       \glslocalunset{#2}%
1714     \else
1715       \glsunset{#2}%
1716     \fi
1717   }%
1718   \glspostlinkhook
1719 }
```

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

```

1720 \renewcommand*{\@gls@@link}[3][\]{%
1721   \@glsxtr@record{#1}{#2}{glslink}%
1722   \glsdoifexistsordo{#2}{%
1723     {%
1724       \let\do@gls@link@checkfirsthyper\relax
1725       \@gls@link[#1]{#2}{#3}%
1726     }%
1727     {%
1728       \glstextformat{#3}%
1729     }%
1730   \glspostlinkhook
1731 }
```

glsxtr@ninitwrgloss Set the default if the wrgloss is omitted.

```

1732 \newcommand*{\glsxtr@ninitwrgloss}{%
1733   \glsifattribute{\glslabel}{wrgloss}{after}%
1734   {%
1735     \glsxtr@ninitwrglossbeforefalse
1736   }%
1737   {%
1738     \glsxtr@ninitwrglossbeforetrue
1739   }%
1740 }
```

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

```

1741 \newif\ifglsxtr@ninitwrglossbefore
1742 \glsxtr@ninitwrglossbeforetrue
```

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

```

1743 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1744 {%
1745   \ifcase\nr\relax
1746     \glstrinitwrglossbeforetrue
1747   \or
1748     \glstrinitwrglossbeforefalse
1749   \fi
1750 }

1751 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}

1752 \define@key{glslink}{theHvalue}{\def\@glstr@theHvalue{#1}}

```

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

```

1753 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{}
1754 \glstr@hyperoutsidetrue

```

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

```

1755 \newcommand*{\glstrnithyperoutside}{%
1756   \glsifattribute{glslabel}{hyperoutside}{false}%
1757   {%
1758     \glstr@hyperoutsidefalse
1759   }%
1760   {%
1761     \glstr@hyperoutsidetrue
1762   }%
1763 }

```

`\@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`.

```

1764 \def\@gls@link[#1]#2#3{%
1765   \leavevmode
1766   \edef\glslabel{\glsdetoklabel{#2}}%
1767   \def\@gls@link@opts{#1}%
1768   \let\@gls@link@label\glslabel
1769   \let\@glsnumberformat\@glstr@defaultnumberformat
1770   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1771   \edef\glstype{\csname glo@\glslabel @type\endcsname}%
1772   \let\org@ifKV@glslink@hyper@ifKV@glslink@hyper

```

Initialise `thevalue` and `theHvalue` (v1.19).

```

1773 \def\@glstr@thevalue{%
1774 \def\@glstr@theHvalue{\@glstr@thevalue}%

```

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

```

1775 \glstrinitwrgloss

```



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

```
1776 \glxtrinithyperoutside
```

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

```
1777 \@gls@setdefault@glslink@opts
1778 \do@glssdisablehyperinlist
1779 \do@gls@link@checkfirsthyper
1780 \setkeys{glslink}{#1}%
1781 \glslinkpostsetkeys
```

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

```
1782 \ifdefempty{\@glxtr@thevalue}%
1783 {%
1784   \@gls@saveentrycounter
1785 }%
1786 {%
1787   \let\theglsentrycounter\@glxtr@thevalue
1788   \def\theHglentrycounter{\@glxtr@theHvalue}%
1789 }%
1790 \@gls@setsort{\glslabel}%
```

Check textformat attribute (new to v1.21).

```
1791 \glshasattribute{\glslabel}{textformat}%
1792 {%
1793   \edef\@glxtr@attrval{\glsggetattribute{\glslabel}{textformat}}%
1794   \ifcsdef{\@glxtr@attrval}%
1795   {%
1796     \letcs{\@glxtr@textformat}{\@glxtr@attrval}%
1797   }%
1798   {%
1799     \GlossariesExtraWarning{Unknown control sequence name
1800       '\@glxtr@attrval' supplied in textformat attribute
1801       for entry '\glslabel'. Reverting to default \string\glstextformat}%
1802     \let\@glxtr@textformat\glstextformat
1803   }%
1804 }%
1805 {%
1806   \let\@glxtr@textformat\glstextformat
1807 }%
```

Do write if it should occur before the link text:

```
1808 \ifglxtrinitwrglossbefore
1809   \@do@wrglossary{#2}%
1810 \fi
```

Do the link text:

```
1811 \ifKV@glslink@hyper
1812   \ifglxtr@hyperoutside
1813     \@glslink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1814   \else
1815     \@glxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
```

```

1816 \fi
1817 \else
1818 \ifglxtr@hyperoutside
1819 \glsdonohyperlink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1820 \else
1821 \@glxtr@textformat{\glsdonohyperlink{\glolinkprefix\glslabel}{#3}}%
1822 \fi
1823 \fi

```

Do write if it should occur after the link text:

```

1824 \ifglxtrinitwrglossbefore
1825 \else
1826 \@do@wrglossary{#2}%
1827 \fi

```

As the original definition:

```

1828 \let@ifKV@glslink@hyper\org@ifKV@glslink@hyper
1829 }

```

```

1830 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

```

```

1831 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

```

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

```

1832 \renewrobustcmd*{\glsadd}[2][\]{%
1833 \glxtrifinmark
1834 }%
1835 {%
1836 \@gls@adjustmode
1837 \@glxtr@record{#1}{#2}{glossadd}%
1838 \glsoifexists{#2}%
1839 {%
1840 \let\@glsnumberformat\@glxtr@defaultnumberformat
1841 \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
1842 \def\@glxtr@thevalue{}%
1843 \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1844 \setkeys{glossadd}{#1}%
1845 \ifdefempty{\@glxtr@thevalue}%
1846 {%
1847 \@gls@saveentrycounter
1848 }%
1849 {%
1850 \let\thegl@sentrycounter\@glxtr@thevalue
1851 \def\theHgl@sentrycounter{\@glxtr@theHvalue}%
1852 }%
1853 \@do@wrglossary{#2}%
1854 }%
1855 }%
1856 }

```

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

```
1857 \newcommand*{\@glsxtr@field@linkdefs}{%
1858   \let\glsxtrifwasfirstuse\@secondoftwo
1859   \let\glsifplural\@secondoftwo
1860   \let\glscapscase\@firstofthree
1861   \let\glsinsert\@empty
1862 }
```

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

assignfieldfont

```
1863 \newcommand*{\glsxtrassignfieldfont}[1]{%
1864   \ifglstryexists{#1}%
1865   {%
1866     \ifglshasshort{#1}%
1867     {%
1868       \glssetabbrvfmt{\glscategory{#1}}%
1869       \glsifregular{#1}%
1870       {\let\@gls@field@font\glsxtrregularfont}%
1871       {\let\@gls@field@font\@firstofone}%
1872     }%
1873     {%
1874       \glsifnotregular{#1}%
1875       {\let\@gls@field@font\@firstofone}%
1876       {\let\@gls@field@font\glsxtrregularfont}%
1877     }%
1878   }%
1879   {%
1880     \let\@gls@field@font\@gobble
1881   }%
1882 }
```

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

```
1883 \def\@glstext@#1#2[#3]{%
1884   \glsxtrassignfieldfont{#2}%
1885   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1886 }
```

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

```
1887 \def\@GLStext@#1#2[#3]{%
1888   \glsxtrassignfieldfont{#2}%
1889   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1890   {\@gls@field@font{\@GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1891 }
```

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

```
1892 \def\@Glstext@#1#2[#3]{%
1893   \glsxtrassignfieldfont{#2}%
1894 }
```

```

1894 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1895 {\@gls@field@font{\Glsaccessstext{#2}#3}}}%
1896 }

```

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`

```

1897 \newcommand*{\glstrchecknohyperfirst}[1]{%
1898 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1899 }

```

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

```

1900 \def\@glsfirst@#1#2[#3]{%
1901 \glstrassignfieldfont{#2}%
    Ensure that \glsfirst honours the nohyperfirst attribute.
1902 \@gls@field@link
1903 [\let\glstrifwasfirstuse\@firstoftwo
1904 \glstrchecknohyperfirst{#2}%
1905 ]{#1}{#2}%
1906 {\@gls@field@font{\glsaccessfirst{#2}#3}}}%
1907 }

```

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

```

1908 \def\@Glsfirst@#1#2[#3]{%
1909 \glstrassignfieldfont{#2}%
    Ensure that \Glsfirst honours the nohyperfirst attribute.
1910 \@gls@field@link
1911 [\let\glstrifwasfirstuse\@firstoftwo
1912 \let\glscapscase\@secondofthree
1913 \glstrchecknohyperfirst{#2}%
1914 ]%
1915 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}}%
1916 }

```

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

```

1917 \def\@GLSfirst@#1#2[#3]{%
1918 \glstrassignfieldfont{#2}%
    Ensure that \GLSfirst honours the nohyperfirst attribute.
1919 \@gls@field@link
1920 [\let\glstrifwasfirstuse\@firstoftwo
1921 \let\glscapscase\@thirdofthree
1922 \glstrchecknohyperfirst{#2}%
1923 ]%
1924 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1925 }

```

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

```
1926 \def\@glsplural@#1#2[#3]{%
1927   \glsxtrassignfieldfont{#2}%
1928   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1929   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1930 }
```

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

```
1931 \def\@Glsplural@#1#2[#3]{%
1932   \glsxtrassignfieldfont{#2}%
1933   \@gls@field@link
1934   [\let\glsifplural\@firstoftwo
1935    \let\glsupcase\@secondoftwo
1936   ]%
1937   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1938 }
```

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

```
1939 \def\@GLSplural@#1#2[#3]{%
1940   \glsxtrassignfieldfont{#2}%
1941   \@gls@field@link
1942   [\let\glsifplural\@firstoftwo
1943    \let\glsupcase\@thirdoftwo
1944   ]%
1945   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1946 }
```

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

```
1947 \def\glsfirstplural@#1#2[#3]{%
1948   \glsxtrassignfieldfont{#2}%
1949   \gls@field@link
1950   [\let\glsxtrifwasfirstuse\@firstoftwo
1951    \let\glsifplural\@firstoftwo
1952    \glsxtrchecknohyperfirst{#2}%
1953   ]%
1954   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1955 }
```

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

```
1956 \def\Glsfirstplural@#1#2[#3]{%
1957   \glsxtrassignfieldfont{#2}%
1958   \gls@field@link
1959   [\let\glsxtrifwasfirstuse\@firstoftwo
1960    \let\glsifplural\@firstoftwo
1961    \let\glsupcase\@secondoftwo
```

```

1962 \glstrchecknohyperfirst{#2}%
1963 ]%
1964 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1965 }

```

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

```

1966 \def\@GLSfirstplural@#1#2[#3]{%
1967 \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
1968 \@gls@field@link
1969 [\let\glstrifwasfirstuse\@firstoftwo
1970 \let\glsifplural\@firstoftwo
1971 \let\glscapscase\@thirdofthree
1972 \glstrchecknohyperfirst{#2}%
1973 ]%
1974 {#1}{#2}%
1975 {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1976 }

```

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

```

1977 \def\@glsname@#1#2[#3]{%
1978 \glstrassignfieldfont{#2}%
1979 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
1980 }

```

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

```

1981 \def\@GLSname@#1#2[#3]{%
1982 \glstrassignfieldfont{#2}%
1983 \@gls@field@link
1984 [\let\glscapscase\@secondoftwo]{#1}{#2}%
1985 {\@gls@field@font{\Glsaccessname{#2}#3}}%
1986 }

```

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

```

1987 \def\@GLSname@#1#2[#3]{%
1988 \glstrassignfieldfont{#2}%
1989 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1990 {#1}{#2}%
1991 {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1992 }

```

**\@glsdesc@**

```

1993 \def\@glsdesc@#1#2[#3]{%
1994 \glstrassignfieldfont{#2}%
1995 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1996 }

```

\@Glsdesc@ First letter uppercase version.

```

1997 \def\@Glsdesc@#1#2[#3]{%
1998   \glstrassignfieldfont{#2}%
1999   \@gls@field@link
2000   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2001   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2002 }

```

\@GLSdesc@ All uppercase version.

```

2003 \def\@GLSdesc@#1#2[#3]{%
2004   \glstrassignfieldfont{#2}%
2005   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2006   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2007 }

```

@glsdescplural@ No case-changing version.

```

2008 \def\@glsdescplural@#1#2[#3]{%
2009   \glstrassignfieldfont{#2}%
2010   \@gls@field@link
2011   [\let\glscapscase\@secondoftwo
2012   \let\glsifplural\@firstoftwo
2013   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2014 }

```

@Glsdescplural@ First letter uppercase version.

```

2015 \def\@Glsdescplural@#1#2[#3]{%
2016   \glstrassignfieldfont{#2}%
2017   \@gls@field@link
2018   [\let\glscapscase\@secondoftwo
2019   \let\glsifplural\@firstoftwo
2020   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2021 }

```

@GLSdescplural@ All uppercase version.

```

2022 \def\@GLSdesc@#1#2[#3]{%
2023   \glstrassignfieldfont{#2}%
2024   \@gls@field@link
2025   [\let\glscapscase\@thirdoftwo
2026   \let\glsifplural\@firstoftwo
2027   ]%
2028   {#1}{#2}%
2029   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2030 }

```

\@glssymbol@

```

2031 \def\@glssymbol@#1#2[#3]{%
2032   \glstrassignfieldfont{#2}%
2033   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2034 }

```

\@Glssymbol@ First letter uppercase version.

```
2035 \def\@Glssymbol@#1#2[#3]{%
2036   \glstrassignfieldfont{#2}%
2037   \@gls@field@link
2038   [\let\glscapscase\@secondoftwo]%
2039   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2040 }
```

\@GLSsymbol@ All uppercase version.

```
2041 \def\@GLSsymbol@#1#2[#3]{%
2042   \glstrassignfieldfont{#2}%
2043   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2044   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2045 }
```

lssymbolplural@ No case-changing version.

```
2046 \def\@lssymbolplural@#1#2[#3]{%
2047   \glstrassignfieldfont{#2}%
2048   \@gls@field@link
2049   [\let\glscapscase\@secondoftwo
2050   \let\glsifplural\@firstoftwo
2051   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2052 }
```

lssymbolplural@ First letter uppercase version.

```
2053 \def\@lssymbolplural@#1#2[#3]{%
2054   \glstrassignfieldfont{#2}%
2055   \@gls@field@link
2056   [\let\glscapscase\@secondoftwo
2057   \let\glsifplural\@firstoftwo
2058   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2059 }
```

LSsymbolplural@ All uppercase version.

```
2060 \def\@LSsymbol@#1#2[#3]{%
2061   \glstrassignfieldfont{#2}%
2062   \@gls@field@link
2063   [\let\glscapscase\@thirdoftwo
2064   \let\glsifplural\@firstoftwo
2065   ]%
2066   {#1}{#2}%
2067   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2068 }
```

\@Glsuseri@ First letter uppercase version.

```
2069 \def\@Glsuseri@#1#2[#3]{%
2070   \glstrassignfieldfont{#2}%
2071   \@gls@field@link
```



```

2072 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2073 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2074 }

```

\@GLSuseri@ All uppercase version.

```

2075 \def\@GLSuseri@#1#2[#3]{%
2076 \glstrassignfieldfont{#2}%
2077 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2078 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
2079 }

```

\@Glsuserii@ First letter uppercase version.

```

2080 \def\@Glsuserii@#1#2[#3]{%
2081 \glstrassignfieldfont{#2}%
2082 \@gls@field@link
2083 [\let\glscapscase\@secondoftwo]%
2084 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2085 }

```

\@GLSuserii@ All uppercase version.

```

2086 \def\@GLSuserii@#1#2[#3]{%
2087 \glstrassignfieldfont{#2}%
2088 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2089 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2090 }

```

\@Glsuseriii@ First letter uppercase version.

```

2091 \def\@Glsuseriii@#1#2[#3]{%
2092 \glstrassignfieldfont{#2}%
2093 \@gls@field@link
2094 [\let\glscapscase\@secondoftwo]%
2095 {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2096 }

```

\@GLSuseriii@ All uppercase version.

```

2097 \def\@GLSuseriii@#1#2[#3]{%
2098 \glstrassignfieldfont{#2}%
2099 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2100 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2101 }

```

\@Glsuseriv@ First letter uppercase version.

```

2102 \def\@Glsuseriv@#1#2[#3]{%
2103 \glstrassignfieldfont{#2}%
2104 \@gls@field@link
2105 [\let\glscapscase\@secondoftwo]%
2106 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2107 }

```

\@GLSuseriv@ All uppercase version.

```
2108 \def\@GLSuseriv@#1#2[#3]{%
2109   \glstrassignfieldfont{#2}%
2110   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2111   {#1}{#2}%
2112   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2113 }
```

\@Glsuserv@ First letter uppercase version.

```
2114 \def\@Glsuserv@#1#2[#3]{%
2115   \glstrassignfieldfont{#2}%
2116   \@gls@field@link
2117   [\let\glscapscase\@secondoftwo]%
2118   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}}%
2119 }
```

\@GLSuserv@ All uppercase version.

```
2120 \def\@GLSuserv@#1#2[#3]{%
2121   \glstrassignfieldfont{#2}%
2122   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2123   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2124 }
```

\@Glsuservi@ First letter uppercase version.

```
2125 \def\@Glsuservi@#1#2[#3]{%
2126   \glstrassignfieldfont{#2}%
2127   \@gls@field@link
2128   [\let\glscapscase\@secondoftwo]%
2129   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
2130 }
```

\@GLSuservi@ All uppercase version.

```
2131 \def\@GLSuservi@#1#2[#3]{%
2132   \glstrassignfieldfont{#2}%
2133   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2134   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2135 }
```

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

\@acrshort No case change.

```
2136 \def\@acrshort#1#2[#3]{%
2137   \glsdoifexists{#2}%
2138   {%
2139     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2140     \let\glxtrifwasfirstuse\@secondoftwo
2141     \let\glsifplural\@secondoftwo
```

```

2142 \let\glscapscase\@firstofthree
2143 \let\glsinsert\@empty
2144 \def\glscustomtext{%
2145     \acronymfont{\glsaccesssshort{#2}}#3%
2146 }%
2147 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2148 }%
2149 \glspostlinkhook
2150 }

```

\@Acrshort First letter uppercase.

```

2151 \def\@Acrshort#1#2[#3]{%
2152     \glsdoifexists{#2}%
2153     {%
2154         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2155         \let\glstrifwasfirstuse\@secondoftwo
2156         \let\glsifplural\@secondoftwo
2157         \let\glscapscase\@secondofthree
2158         \let\glsinsert\@empty
2159         \def\glscustomtext{%
2160             \acronymfont{\Glsaccesssshort{#2}}#3%
2161         }%
2162         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2163     }%
2164     \glspostlinkhook
2165 }

```

\@ACRshort All uppercase.

```

2166 \def\@ACRshort#1#2[#3]{%
2167     \glsdoifexists{#2}%
2168     {%
2169         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2170         \let\glstrifwasfirstuse\@secondoftwo
2171         \let\glsifplural\@secondoftwo
2172         \let\glscapscase\@thirdofthree
2173         \let\glsinsert\@empty
2174         \def\glscustomtext{%
2175             \mfirstucMakeUppercase{\acronymfont{\glsaccesssshort{#2}}#3}%
2176         }%
2177         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2178     }%
2179     \glspostlinkhook
2180 }

```

\@acrshortpl No case change.

```

2181 \def\@acrshortpl#1#2[#3]{%
2182     \glsdoifexists{#2}%
2183     {%
2184         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

2185 \let\glxtrifwasfirstuse\@secondoftwo
2186 \let\glsifplural\@firstoftwo
2187 \let\glscapscase\@firstofthree
2188 \let\glsinsert\@empty
2189 \def\glscustomtext{%
2190 \acronymfont{\glsaccessshortpl{#2}}#3%
2191 }%
2192 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2193 }%
2194 \glspostlinkhook
2195 }

```

\@Acrshortpl First letter uppercase.

```

2196 \def\@Acrshortpl#1#2[#3]{%
2197 \glsdoifexists{#2}%
2198 {%
2199 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2200 \let\glxtrifwasfirstuse\@secondoftwo
2201 \let\glsifplural\@firstoftwo
2202 \let\glscapscase\@secondofthree
2203 \let\glsinsert\@empty
2204 \def\glscustomtext{%
2205 \acronymfont{\Glsaccessshortpl{#2}}#3%
2206 }%
2207 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2208 }%
2209 \glspostlinkhook
2210 }

```

\@ACRshortpl All uppercase.

```

2211 \def\@ACRshortpl#1#2[#3]{%
2212 \glsdoifexists{#2}%
2213 {%
2214 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2215 \let\glxtrifwasfirstuse\@secondoftwo
2216 \let\glsifplural\@firstoftwo
2217 \let\glscapscase\@thirdofthree
2218 \let\glsinsert\@empty
2219 \def\glscustomtext{%
2220 \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2221 }%
2222 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2223 }%
2224 \glspostlinkhook
2225 }

```

\@acrlong No case change.

```

2226 \def\@acrlong#1#2[#3]{%
2227 \glsdoifexists{#2}%

```

```

2228 {%
2229   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2230   \let\glxtrifwasfirstuse\@secondoftwo
2231   \let\gl@sifplural\@secondoftwo
2232   \let\glscapscase\@firstofthree
2233   \let\gl$insert\@empty
2234   \def\glscustomtext{%
2235     \acronymfont{\gl@saccesslong{#2}}#3%
2236   }%
2237   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2238 }%
2239 \glspostlinkhook
2240 }

```

\@Acrlong First letter uppercase.

```

2241 \def\@Acrlong#1#2[#3]{%
2242   \gl@sdoifexists{#2}%
2243   {%
2244     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2245     \let\glxtrifwasfirstuse\@secondoftwo
2246     \let\gl@sifplural\@secondoftwo
2247     \let\glscapscase\@secondofthree
2248     \let\gl$insert\@empty
2249     \def\glscustomtext{%
2250       \acronymfont{\gl@saccesslong{#2}}#3%
2251     }%
2252     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2253   }%
2254   \glspostlinkhook
2255 }

```

\@ACRlong All uppercase.

```

2256 \def\@ACRlong#1#2[#3]{%
2257   \gl@sdoifexists{#2}%
2258   {%
2259     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2260     \let\glxtrifwasfirstuse\@secondoftwo
2261     \let\gl@sifplural\@secondoftwo
2262     \let\glscapscase\@thirdofthree
2263     \let\gl$insert\@empty
2264     \def\glscustomtext{%
2265       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2266     }%
2267     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2268   }%
2269   \glspostlinkhook
2270 }

```

\@acrlongpl No case change.

```

2271 \def\@acrlongpl#1#2[#3]{%
2272   \glsdoifexists{#2}%
2273   {%
2274     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2275     \let\glxtrifwasfirstuse\@secondoftwo
2276     \let\gl@sifplural\@firstoftwo
2277     \let\glscapscase\@firstofthree
2278     \let\gl$insert\@empty
2279     \def\glscustomtext{%
2280       \acronymfont{\gl@saccesslongpl{#2}}#3%
2281     }%
2282     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2283   }%
2284   \glspostlinkhook
2285 }

```

\@Acrlongpl First letter uppercase.

```

2286 \def\@Acrlongpl#1#2[#3]{%
2287   \glsdoifexists{#2}%
2288   {%
2289     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2290     \let\glxtrifwasfirstuse\@secondoftwo
2291     \let\gl@sifplural\@firstoftwo
2292     \let\glscapscase\@secondofthree
2293     \let\gl$insert\@empty
2294     \def\glscustomtext{%
2295       \acronymfont{\Glsaccesslongpl{#2}}#3%
2296     }%
2297     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2298   }%
2299   \glspostlinkhook
2300 }

```

\@ACRlongpl All uppercase.

```

2301 \def\@ACRlongpl#1#2[#3]{%
2302   \glsdoifexists{#2}%
2303   {%
2304     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2305     \let\glxtrifwasfirstuse\@secondoftwo
2306     \let\gl@sifplural\@firstoftwo
2307     \let\glscapscase\@thirdofthree
2308     \let\gl$insert\@empty
2309     \def\glscustomtext{%
2310       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%
2311     }%
2312     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2313   }%
2314   \glspostlinkhook
2315 }

```

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

\@glsaddkey

```

2316 \renewcommand*{\@glsaddkey}[7]{%
2317   \key@ifundefined{glossentry}{#1}%
2318   {%
2319     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2320     \appto\@gls@keymap{,{#1}{#1}}%
2321     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2322     \appto\@newglossaryentryposthook{%
2323       \letcs{\@glo@tmp}{@glo@#1}%
2324       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2325     }%
2326     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2327     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2328   \ifcsdef{@gls@user@#1@}%
2329   {%
2330     \PackageError{glossaries}%
2331     {Can't define '\string#5' as helper command
2332     '\expandafter\string\csname @gls@user@#1@\endcsname' already
2333     exists}%
2334     {}%
2335   }%
2336   {%
2337     \expandafter\newcommand\expandafter*\expandafter
2338     {\csname @gls@user@#1\endcsname}[2][ ]{%
2339       \new@ifnextchar[%
2340         {\csuse{@gls@user@#1@}{##1}{##2}}%
2341         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2342     \csdef{@gls@user@#1@}##1##2[##3]{%
2343       \@gls@field@link{##1}{##2}{#3{##2}##3}%
2344     }%
2345     \newrobustcmd*{#5}{%
2346       \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
2347   }%

```

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

```

2348   \ifcsdef{@Gls@user@#1@}%
2349   {%
2350     \PackageError{glossaries}%
2351     {Can't define '\string#6' as helper command
2352     '\expandafter\string\csname @Gls@user@#1@\endcsname' already
2353     exists}%
2354     {}%
2355   }%
2356   {%
2357     \expandafter\newcommand\expandafter*\expandafter

```

```

2358     {\csname @Gls@user@#1\endcsname}[2] [] {%
2359         \new@ifnextchar[%
2360             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2361             {\csuse{@Gls@user@#1@}{##1}{##2} [] }}}%
2362     \csdef{@Gls@user@#1@}##1##2[##3]{%
2363         \@gls@field@link[\let\glscapscase\secondofthree]%
2364         {##1}{##2}{#4{##2}##3}%
2365     }%
2366     \newrobustcmd*{#6}{%
2367         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2368     }%

```

Finally the all caps version (modified):

```

2369     \ifcsdef{@GLS@user@#1@}%
2370     {%
2371         \PackageError{glossaries}%
2372         {Can't define '\string#7' as helper command
2373         '\expandafter\string\csname @GLS@user@#1\endcsname' already
2374         exists}%
2375     }%
2376 }%
2377 {%
2378     \expandafter\newcommand\expandafter*\expandafter
2379     {\csname @GLS@user@#1\endcsname}[2] [] {%
2380         \new@ifnextchar[%
2381             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2382             {\csuse{@GLS@user@#1@}{##1}{##2} [] }}}%
2383     \csdef{@GLS@user@#1@}##1##2[##3]{%
2384         \@gls@field@link[\let\glscapscase\thirdofthree]%
2385         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2386     }%
2387     \newrobustcmd*{#7}{%
2388         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2389     }%
2390 }%
2391 {%
2392     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2393 }%
2394 }

```

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

```

2395 \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.

```

2396 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2397 \renewcommand*{\@gls@link@checkfirsthyper}{%

```

\ifglused isn't useful in the post link hook as it's already been unset by then, so define a command that can be used in the post link hook. Since \@gls@link@checkfirsthyper is



only used by commands like `\gls` but not by other commands, this seems the best place to put it.

```

2398 \ifglsused{\glslabel}%
2399 {\let\glstrifwasfirstuse\@secondoftwo}
2400 {\let\glstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.

2401 \edef\glscategorylabel{\glscategory{\glslabel}}%
2402 \ifglsused{\glslabel}%
2403 {%
2404   \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2405   {\KV@glslink@hyperfalse}{}}%
2406 }%
2407 {%
2408   \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2409   {\KV@glslink@hyperfalse}{}}%
2410 }%
2411 \glslinkcheckfirsthyperhook
2412 }

```

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.

```

2413 \ifdefined\do@glsglsdisablehyperinlist
2414 {%
2415   \let\@glstr@do@glsglsdisablehyperinlist\do@glsglsdisablehyperinlist
2416   \renewcommand*{\do@glsglsdisablehyperinlist}{%
2417     \@glstr@do@glsglsdisablehyperinlist
2418     \glsifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}}%
2419   }
2420 }
2421 {}

```

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

```

2422 \define@boolkey{glslink}{noindex}[true]{}
2423 \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

```

2424 \ifdefined\@gls@setdefault@glslink@opts
2425 {
2426   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2427     \KV@glslink@noindexfalse
2428     \@glstrsetaliasnoindex
2429   }
2430 }
2431 {

```

Not defined so prepend it to `\do@glsglslDisableHyperinlist` to achieve the same effect.

```

2432 \newcommand*{\@glsglslSetDefault@glslink@opts}{%
2433   \KV@glslink@noindexfalse
2434   \@glsxtrsetaliasnoindex
2435 }
2436 \preto\do@glsglslDisableHyperinlist{\@glsglslSetDefault@glslink@opts}
2437 }

```

`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.)

```

2438 \providecommand*{\glsxtrsetaliasnoindex}{%
2439   \KV@glslink@noindextrue
2440 }

```

`setaliasnoindex`

```

2441 \newcommand*{\@glsxtrsetaliasnoindex}{%
2442   \glsxtrifhasfield{alias}{\glslabel}%
2443   {%
2444     \let\glsxtrindexaliased\@glsxtrindexaliased
2445     \glsxtrsetaliasnoindex
2446     \let\glsxtrindexaliased\@no@glsxtrindexaliased
2447   }%
2448   }%
2449 }

```

`xtrindexaliased`

```

2450 \newcommand{\@glsxtrindexaliased}{%
2451   \ifKV@glslink@noindex
2452   \else
2453     \begingroup
2454     \let\@glslnumberformat\@glsxtr@defaultnumberformat
2455     \edef\@glsglsl@counter{\csname glo@glsglslDetoklabel{\glslabel}@counter\endcsname}%
2456     \glsxtr@saveentrycounter
2457     \do@wrglossary{\glsxtralias{\glslabel}}%
2458   \endgroup
2459   \fi
2460 }

```

`xtrindexaliased`

```

2461 \newcommand{\@no@glsxtrindexaliased}{%
2462   \PackageError{glossaries-extra}{\string\glsxtrindexaliased\space
2463     not permitted outside definition of \string\glsxtrsetaliasnoindex}%
2464   }%
2465 }

```

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

```

2466 \let\glsxtrindexaliased\@no@glsxtrindexaliased

```

DefaultGlsOpts Set the default options for \glslink etc.

```
2467 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2468   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2469     \setkeys{glslink}{#1}%
2470     \@glsxtrsetaliasnoindex
2471   }%
2472 }
```

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

```
2473 \newcommand*{\glsxtrifindexing}[2]{%
2474   \ifKV@glslink@noindex #2\else #1\fi
2475 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2476 \renewcommand*{\glswriteentry}[2]{%
2477   \glsxtrifindexing
2478   {%
2479     \ifglsindexonlyfirst
2480       \ifglsused{#1}
2481       {\glsxtrdoautoindexname{#1}{dualindex}}%
2482       {#2}%
2483     \else
2484       \glsifattribute{#1}{indexonlyfirst}{true}%
2485       {\ifglsused{#1}
2486        {\glsxtrdoautoindexname{#1}{dualindex}}%
2487        {#2}}%
2488       {#2}%
2489     \fi
2490   }%
2491   {}%
2492 }
```

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

```
2493 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
2494   \glsxtrdowrglossaryhook{\@gls@label}%
2495 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2496 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
2497   \glsxtrdowrglossaryhook{\@gls@label}%
2498 }
```

xtr@do@@wrindex

```
2499 \newcommand*{\@glsxtr@do@@wrindex}{%
```

```

2500 \glstrdoautoindexname{\@gls@label}{dualindex}%
2501 }

\newcommand*{\glstrdowrglossaryhook}[1]{%
2502 \newcommand*{\@gls@alt@hyp@opt}{%
2503 \let\glslinkvar\@firstofthree
2504 \let\@gls@hyp@opt@cs#1\relax
2505 \ifstar{\s@gls@hyp@opt}%
2506 {\@ifnextchar+%
2507 {\@firstoftwo{\p@gls@hyp@opt}}%
2508 {%
2509 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2510 {\@firstoftwo{\@alt@gls@hyp@opt}}%
2511 {\#1}%
2512 }%
2513 }%
2514 }%
2515 }

\newcommand*{\@alt@gls@hyp@opt}{%
2516 \let\glslinkvar\@firstofthree
2517 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}

\newcommand*{\@gls@alt@hyp@opt@char}{%
2519 \newcommand*{\@gls@alt@hyp@opt@keys}{%
2520 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2521 \let\@gls@hyp@opt\@gls@alt@hyp@opt
2522 \def\@gls@alt@hyp@opt@char{#1}%
2523 \def\@gls@alt@hyp@opt@keys{#2}%
2524 }%
2525 }

\let\glsdohyperlink\@gls@alt@hyp@opt@char

\newcommand*{\glsdohyperlink}{%
2526 \let\glsxtr@org@dohyperlink\glsdohyperlink

\newcommand*{\glsnavhyperlink}{%
2527 \let\glsxtr@org@dohyperlink\glsdohyperlink

```

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.)

gls@alt@hyp@opt Commands like \gls have a star or plus version. Provide a third symbol that the user can adapt for convenience.

alt@gls@hyp@opt User version

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

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

rSetAltModifier

org@dohyperlink

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.

```
2527 \ifdef\glsnavhyperlink
2528 {
2529   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
2530     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
```

Scope:

```
2531   {%
2532     \let\glsdohyperlink\glsxtr@org@dohyperlink
2533     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2534   }%
2535 }%
2536 }
2537 {}
```

`\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.

```
2538 \renewcommand*{\glsdohyperlink}[2]{%
2539   \glsasattribute{\glslabel}{targeturl}%
2540   {%
2541     \glsasattribute{\glslabel}{targetname}%
2542     {%
2543       \glsasattribute{\glslabel}{targetcategory}%
2544       {%
2545         \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2546           {\glsgetattribute{\glslabel}{targetcategory}}}%
2547         {\glsgetattribute{\glslabel}{targetname}}}%
2548         {\glsxtrprotectlinks#2}}}%
2549       }%
2550     }%
2551     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2552       }%
2553       {\glsgetattribute{\glslabel}{targetname}}}%
2554       {\glsxtrprotectlinks#2}}}%
2555     }%
2556   }%
2557   {%
2558     \href{\glsgetattribute{\glslabel}{targeturl}}{%
2559       {\glsxtrprotectlinks#2}}}%
2560   }%
2561 }%
2562 {}
```

Check for alias.

```

2563 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2564 \ifvoid\gloaliaslabel
2565 {%
2566   \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2567 }%
2568 {%

```

Redirect link to the alias target.

```

2569   \glsxtrhyperlink
2570   {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2571   {\glsxtrprotectlinks#2}}%
2572 }%
2573 }%
2574 }

```

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

```

2575 \ifdef\@glsshowtarget
2576 {
2577   \newcommand{\glsxtrhyperlink}[2]{%
2578     \@glsshowtarget{#1}%
2579     \hyperlink{#1}{#2}%
2580   }%
2581 }
2582 {
2583   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2584 }

```

`\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.

```

2585 \renewrobustcmd*{\glsdohyperlink}[2][\glssentrytext{\@glo@label}]{%
2586   \glsdoifexists{#2}%
2587   {%
2588     \def\@glo@label{#2}%
2589     {\edef\glslabel{#2}%
2590      \@glslink{\glolinkprefix\glslabel}{#1}}%
2591   }%
2592 }

```

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

```

2593 \renewcommand{\glsdisablehyper}{%
2594   \KV@glslink@hyperfalse
2595   \def\@glslink{\glsdohyperlink}%
2596   \let\@glstarget\@secondoftwo
2597 }

```

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

```
2598 \renewcommand{\glsenablehyper}{%
2599   \KV@glslink@hypertrue
2600   \def\@glslink{\glsdohyperlink}%
2601   \def\@glstarget{\glsdohypertarget}%
2602 }
```

`\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.

```
2603 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

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

```
2604 \ifcsundef{hyperlink}%
2605 {%
2606   \def\@glslink{\glsdonohyperlink}
2607 }%
2608 {%
2609   \def\@glslink{\glsdohyperlink}
2610 }
```

`\glstrprotectlinks` Make `\gls` (and variants) behave like the corresponding `\glstext` (and variants) with hyperlinking and indexing off.

```
2611 \newcommand*{\glstrprotectlinks}{%
2612   \KV@glslink@hyperfalse
2613   \KV@glslink@noindextrue
2614   \let\@gls@\@glstrp@text@
2615   \let\@Gls@\@Glsxtrp@text@
2616   \let\@GLS@\@GLSxtrp@text@
2617   \let\@glspl@\@glstrp@plural@
2618   \let\@Glspl@\@Glsxtrp@plural@
2619   \let\@GLSpl@\@GLSxtrp@plural@
2620   \let\@glstrshort@\@glstrp@short@
2621   \let\@Glsxtrshort@\@Glsxtrp@short@
2622   \let\@GLSxtrshort@\@GLSxtrp@short@
2623   \let\@glstrlong@\@glstrp@long@
2624   \let\@Glsxtrlong@\@Glsxtrp@long@
2625   \let\@GLSxtrlong@\@GLSxtrp@long@
2626   \let\@glstrshortpl@\@glstrp@shortpl@
2627   \let\@Glsxtrshortpl@\@Glsxtrp@shortpl@
2628   \let\@GLSxtrshortpl@\@GLSxtrp@shortpl@
2629   \let\@glstrlongpl@\@glstrp@longpl@
2630   \let\@Glsxtrlongpl@\@Glsxtrp@longpl@
2631   \let\@GLSxtrlongpl@\@GLSxtrp@longpl@
2632   \let\@acrshort@\@glstrp@acrshort@
2633   \let\@Acrshort@\@Glsxtrp@acrshort@
2634   \let\@ACRshort@\@GLSxtrp@acrshort@}
```

```

2635 \let\@acrshortpl\@glsxtrp@acrshortpl@
2636 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
2637 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
2638 \let\@acrlong\@glsxtrp@acrlong@
2639 \let\@Acrlong\@Glsxtrp@acrlong@
2640 \let\@ACRlong\@GLSxtrp@acrlong@
2641 \let\@acrlongpl\@glsxtrp@acrlongpl@
2642 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
2643 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
2644 }

```

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

@glsxtrp@text@

```
2645 \def\@glsxtrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2646 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2647 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2648 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lSxtrp@plural@

```
2649 \def\@lSxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
2650 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtrp@short@

```

2651 \def\@glsxtrp@short@#1#2[#3]{%
2652 {%
2653   \glssetabbrvfmt{\glscategory{#2}}%
2654   \glsabbrvfont{\glstryshort{#2}}#3%
2655 }%
2656 }

```

Glsxtrp@short@

```

2657 \def\@Glsxtrp@short@#1#2[#3]{%
2658 {%
2659   \glssetabbrvfmt{\glscategory{#2}}%
2660   \glsabbrvfont{\Glsentryshort{#2}}#3%
2661 }%
2662 }

```



GLSxtr@p@short@

```
2663 \def\@GLSxtr@p@short@#1#2[#3]{%
2664   {%
2665     \glsetabbrvfmt{\glscategory{#2}}%
2666     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
2667   }%
2668 }
```

sxtr@p@shortpl@

```
2669 \def\@glsxtr@p@shortpl@#1#2[#3]{%
2670   {%
2671     \glsetabbrvfmt{\glscategory{#2}}%
2672     \glsabbrvfont{\glsentryshortpl{#2}}#3%
2673   }%
2674 }
```

Sxtr@p@shortpl@

```
2675 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2676   {%
2677     \glsetabbrvfmt{\glscategory{#2}}%
2678     \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2679   }%
2680 }
```

Sxtr@p@shortpl@

```
2681 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2682   {%
2683     \glsetabbrvfmt{\glscategory{#2}}%
2684     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
2685   }%
2686 }
```

@glsxtr@p@long@

```
2687 \def\@glsxtr@p@long@#1#2[#3]{\{\glsentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2688 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2689 \def\@GLSxtr@p@long@#1#2[#3]{%
2690   {\mfirstucMakeUppercase{\glslongfont{\glsentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2691 \def\@glsxtr@p@longpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2692 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2693 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2694   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2695 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2696 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2697 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2698   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2699 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2700 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2701 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2702   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2703 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}#3}}

sxtr@p@acrlong@
2704 \def\@Glsxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}#3}}

Sxtr@p@acrlong@
2705 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2706   {\mfirstucMakeUppercase{\glsentrylong{#2}#3}}}

tr@p@acrlongpl@
2707 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
2708 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}#3}}

tr@p@acrlongpl@
2709 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2710   {\mfirstucMakeUppercase{\glsentrylongpl{#2}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2711 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
2712 \newcommand*{\glsxtrsetpopts}[1]{%
2713   \renewcommand*{\@glsxtrp@opt}{#1}%
2714 }
```

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

```
2715 \newcommand*{\glossxtrsetpopts}{%
2716   \glsxtrsetpopts{noindex}%
2717 }
```

`\@glsxtrp`

```
2718 \newrobustcmd*{\@glsxtrp}[2]{%
```

Add scope.

```
2719   {%
2720     \let\glspostlinkhook\relax
2721     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2722   }%
2723 }
```

`\@glsxtrp`

```
2724 \newrobustcmd*{\@glsxtrp}[2]{%
2725   \ifcsdef{gls#1}%
2726     {%
2727       \@glsxtrp{gls#1}{#2}%
2728     }%
2729     {%
2730       \ifcsdef{glsxtr#1}%
2731         {%
2732           \@glsxtrp{glsxtr#1}{#2}%
2733         }%
2734         {%
2735           \PackageError{glossaries-extra}{‘#1’ not recognised by
2736             \string\glsxtrp}{}%
2737         }%
2738       }%
2739 }
```

`\@Glsxtrp`

```
2740 \newrobustcmd*{\@Glsxtrp}[2]{%
2741   \ifcsdef{Gls#1}%
2742     {%
2743       \@glsxtrp{Gls#1}{#2}%
2744     }%
2745     {%
2746       \ifcsdef{Glsxtr#1}%
2747         {%
2748           \@glsxtrp{Glsxtr#1}{#2}%
2749         }%
2750     }
```

```

2750   {%
2751       \PackageError{glossaries-extra}{‘#1’ not recognised by
2752           \string\Glsxtrp}{}%
2753   }%
2754 }%
2755 }

```

\@GLSxtrp

```

2756 \newrobustcmd*{\@GLSxtrp}[2]{%
2757   \ifcsdef{GLS#1}%
2758   {%
2759       \@glsxtrp{GLS#1}{#2}%
2760   }%
2761   {%
2762       \ifcsdef{GLSxtr#1}%
2763       {%
2764           \@glsxtrp{GLSxtr#1}{#2}%
2765       }%
2766       {%
2767           \PackageError{glossaries-extra}{‘#1’ not recognised by
2768               \string\GLSxtrp}{}%
2769       }%
2770   }%
2771 }

```

\glsxtr@entry@p

```

2772 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
2773   \glsifattribute{#1}{headuc}{true}%
2774   {%
2775       \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
2776   }%
2777   {%
2778       \@gls@entry@field{#1}{#2}%
2779   }%
2780 }

```

\glsxtrp Not robust as it needs to expand somewhat.

```

2781 \ifdef\texorpdfstring
2782 {
2783   \newcommand{\glsxtrp}[2]{%
2784     \protect\NoCaseChange
2785     {%
2786         \protect\texorpdfstring
2787         {%
2788             \protect\glsxtrifinmark
2789             {%
2790                 \ifcsdef{glsxtrhead#1}%
2791                 {%
2792                     {\protect\csuse{glsxtrhead#1}{#2}}%

```

```

2793         }%
2794         {%
2795             \glstr@headentry@p{#2}{#1}%
2796         }%
2797     }%
2798     {%
2799         \@glstrp{#1}{#2}%
2800     }%
2801 }%
2802 {%
2803     \protect\@gls@entry@field{#2}{#1}%
2804 }%
2805 }%
2806 }
2807 }
2808 {
2809     \newcommand{\glstrp}[2]{%
2810         \protect\NoCaseChange
2811         {%
2812             \protect\glstrifinmark
2813             {%
2814                 \ifcsdef{glstrhead#1}%
2815                 {%
2816                     {\protect\csuse{glstrhead#1}}%
2817                 }%
2818                 {%
2819                     \glstr@headentry@p{#2}{#1}%
2820                 }%
2821             }%
2822             {%
2823                 \@glstrp{#1}{#2}%
2824             }%
2825         }%
2826     }
2827 }

```

Provide short synonyms for the most common option.

`\glsp`

```
2828 \newcommand*{\glsp}{\glstrp{short}}
```

`\glsp`

```
2829 \newcommand*{\glsp}{\glstrp{text}}
```

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

```
2830 \ifdef\teorpdfstring
```

```
2831 {
```

```
2832     \newcommand{\Glsxtrp}[2]{%
```

```

2833 \protect\NoCaseChange
2834 {%
2835 \protect\texorpdfstring
2836 {%
2837 \protect\glxtrifinmark
2838 {%
2839 \ifcsdef{Glsxtrhead#1}%
2840 {%
2841 {\protect\csuse{Glsxtrhead#1}{#2}}%
2842 }%
2843 {%
2844 \protect\@Gls@entry@field{#2}{#1}%
2845 }%
2846 }%
2847 {%
2848 \@Glsxtrp{#1}{#2}%
2849 }%
2850 }%
2851 {%
2852 \protect\@gls@entry@field{#2}{#1}%
2853 }%
2854 }%
2855 }
2856 }
2857 {
2858 \newcommand{\Glsxtrp}[2]{%
2859 \protect\NoCaseChange
2860 {%
2861 \protect\glxtrifinmark
2862 {%
2863 \ifcsdef{Glsxtrhead#1}%
2864 {%
2865 {\protect\csuse{Glsxtrhead#1}}%
2866 }%
2867 {%
2868 \protect\@Gls@entry@field{#2}{#1}%
2869 }%
2870 }%
2871 {%
2872 \@Glsxtrp{#1}{#2}%
2873 }%
2874 }%
2875 }
2876 }

```

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

```

2877 \ifdef\texorpdfstring
2878 {
2879 \newcommand{\GlsXtrp}[2]{%

```

```

2880 \protect\NoCaseChange
2881 {%
2882 \protect\texorpdfstring
2883 {%
2884 \protect\glstrifinmark
2885 {%
2886 \ifcsdef{GLSxtr#1}%
2887 {%
2888 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2889 }%
2890 {%
2891 \protect\mfirstucMakeUppercase
2892 {%
2893 \protect\@gls@entry@field{#2}{#1}%
2894 }%
2895 }%
2896 }%
2897 {%
2898 \@GLSxtrp{#1}{#2}%
2899 }%
2900 }%
2901 {%
2902 \protect\@gls@entry@field{#2}{#1}%
2903 }%
2904 }%
2905 }
2906 }
2907 {
2908 \newcommand{\GLSxtrp}[2]{%
2909 \protect\NoCaseChange
2910 {%
2911 \protect\glstrifinmark
2912 {%
2913 \ifcsdef{GLSxtr#1}%
2914 {%
2915 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2916 }%
2917 {%
2918 \protect\mfirstucMakeUppercase
2919 {%
2920 \protect\@gls@entry@field{#2}{#1}%
2921 }%
2922 }%
2923 }%
2924 {%
2925 \@GLSxtrp{#1}{#2}%
2926 }%
2927 }%
2928 }

```

2929 }

### 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.

```
2930 \renewcommand*{\@glsunset}[1]{%
2931   \@glsunset{#1}%
2932   \glsxtrpostunset{#1}%
2933 }%
```

glsxtrpostunset

```
2934 \newcommand*{\glsxtrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2935 \renewcommand*{\@glslocalunset}[1]{%
2936   \@glslocalunset{#1}%
2937   \glsxtrpostlocalunset{#1}%
2938 }%
```

rpostlocalunset

```
2939 \newcommand*{\glsxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2940 \renewcommand*{\@glsreset}[1]{%
2941   \@glsreset{#1}%
2942   \glsxtrpostreset{#1}%
2943 }%
```

glsxtrpostreset

```
2944 \newcommand*{\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2945 \renewcommand*{\@glslocalreset}[1]{%
2946   \@glslocalreset{#1}%
2947   \glsxtrpostlocalreset{#1}%
2948 }%
```

rpostlocalreset

```
2949 \newcommand*{\glsxtrpostlocalreset}[1]{}
```

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

```
2950 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```



Enable entry counting:

```
2951 \glsenableentrycount
```

Redefine \gls etc:

```
2952 \renewcommand*{\gls}{\cglsl}%
2953 \renewcommand*{\Gls}{\cGls}%
2954 \renewcommand*{\glspl}{\cglspl}%
2955 \renewcommand*{\Glspl}{\cGlspl}%
2956 \renewcommand*{\GLS}{\cGLS}%
2957 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2958 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2959 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2960 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2961   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2962     can't be used with \string\GlsXtrEnableEntryCounting}%
2963   {Use one or other but not both commands}}%
2964 }
```

ycountunsetattr

```
2965 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2966   \@for\@glsxtr@cat:=#1\do
2967   {%
2968     \ifdefempty{\@glsxtr@cat}{}%
2969     {%
2970       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2971     }%
2972   }%
2973 }
```

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

nableentrycount

```
2974 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2975 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
2976 \renewcommand*{\gls@defdocnewglossaryentry}{%
2977   \renewcommand*{\newglossaryentry}[2]{%
2978     \PackageError{glossaries}{\string\newglossaryentry\space
2979       may only be used in the preamble when entry counting has
2980       been activated}{If you use \string\glsenableentrycount\space
2981       you must place all entry definitions in the preamble not in
2982       the document environment}%
2983   }%
2984 }
```

New commands to access new fields:

```

2985 \newcommand*\glsentrycurrcount}[1]{%
2986   \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
2987   {0}{\@gls@entry@field{##1}{currcount}}%
2988 }%
2989 \newcommand*\glsentryprevcount}[1]{%
2990   \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
2991   {0}{\@gls@entry@field{##1}{prevcount}}%
2992 }%

```

Adjust post unset and reset:

```

2993 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
2994 \renewcommand*\@glsxtrpostunset}[1]{%
2995   \@glsxtr@entrycount@org@unset{##1}%
2996   \@gls@increment@currcount{##1}%
2997 }%
2998 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
2999 \renewcommand*\@glsxtrpostlocalunset}[1]{%
3000   \@glsxtr@entrycount@org@localunset{##1}%
3001   \@gls@local@increment@currcount{##1}%
3002 }%
3003 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3004 \renewcommand*\@glsxtrpostreset}[1]{%
3005   \@glsxtr@entrycount@org@reset{##1}%
3006   \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
3007 }%
3008 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3009 \renewcommand*\@glsxtrpostlocalreset}[1]{%
3010   \@glsxtr@entrycount@org@localreset{##1}%
3011   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
3012 }%

```

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

```

3013 \let\@cGls@\@cGls@
3014 \let\@cGlspl@\@cGlspl@

3015 \let\@cGls@\@cGls@
3016 \let\@cGlspl@\@cGlspl@
3017 \let\@cGLS@\@cGLS@
3018 \let\@cGLSpl@\@cGLSpl@

```

The rest is as the original definition.

```

3019 \AtEndDocument{\@gls@write@entrycounts}%
3020 \renewcommand*\@gls@entry@count}[2]{%
3021   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
3022 }%
3023 \let\glsenableentrycount\relax
3024 \renewcommand*\glsenableentryunitcount{%
3025   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space

```

```

3026      can't be used with \string\glsenableentrycount}%
3027      {Use one or other but not both commands}%
3028  }%
3029 }

```

ite@entrycounts Modify this command so that it only writes the information for entries with the entrycount attribute and issue warning if no entries have this attribute set.

```

3030 \renewcommand*{\@gls@write@entrycounts}{%
3031   \immediate\write\@auxout
3032   {\string\providecommand*\string\@gls@entry@count}[2]{}%
3033   \count@=0\relax
3034   \forallglsentries{\@glsentry}{%
3035     \glshasattribute{\@glsentry}{entrycount}%
3036     {%
3037       \ifglsused{\@glsentry}%
3038       {%
3039         \immediate\write\@auxout
3040         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3041       }%
3042     }%
3043     \advance\count@ by \@ne
3044   }%
3045 }%
3046 }%
3047 \ifnum\count@=0
3048   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3049     \MessageBreak with \string\glsenableentrycount\space but the
3050     \MessageBreak attribute 'entrycount' hasn't
3051     \MessageBreak been assigned to any of the defined
3052     \MessageBreak entries}%
3053 \fi
3054 }

```

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

```

3055 \newcommand*\glxtrifcounttrigger[3]{%
3056   \glshasattribute{#1}{entrycount}%
3057   {%
3058     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
3059     #3%
3060   \else
3061     #2%
3062   \fi
3063 }%
3064 {#3}%
3065 }

```

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

\@@cgl's@

```
3066 \def\@@cgl's@#1#2[#3]{%
3067   \gl'sxtrifcounttrigger{#2}%
3068   {%
3069     \cgl'sformat{#2}{#3}%
3070     \gl'sunset{#2}%
3071   }%
3072   {%
3073     \@gl's@{#1}{#2}[#3]%
3074   }%
3075 }%
```

\@@cgl'spl@

```
3076 \def\@@cgl'spl@#1#2[#3]{%
3077   \gl'sxtrifcounttrigger{#2}%
3078   {%
3079     \cgl'splformat{#2}{#3}%
3080     \gl'sunset{#2}%
3081   }%
3082   {%
3083     \@gl'spl@{#1}{#2}[#3]%
3084   }%
3085 }%
```

\@@cGl's@

```
3086 \def\@@cGl's@#1#2[#3]{%
3087   \gl'sxtrifcounttrigger{#2}%
3088   {%
3089     \cGl'sformat{#2}{#3}%
3090     \gl'sunset{#2}%
3091   }%
3092   {%
3093     \@Gl's@{#1}{#2}[#3]%
3094   }%
3095 }%
```

\@@cGl'spl@

```
3096 \def\@@cGl'spl@#1#2[#3]{%
3097   \gl'sxtrifcounttrigger{#2}%
3098   {%
3099     \cGl'splformat{#2}{#3}%
3100     \gl'sunset{#2}%
3101   }%
3102   {%
3103     \@Gl'spl@{#1}{#2}[#3]%
3104   }%
3105 }%
```

\@@cGLS@

```
3106 \def\@@cGLS@#1#2[#3]{%
3107   \glxtrifcounttrigger{#2}%
3108   {%
3109     \cGLSformat{#2}{#3}%
3110     \glset{#2}%
3111   }%
3112   {%
3113     \@GLS@{#1}{#2}[#3]%
3114   }%
3115 }%
```

\@@cGLSpl@

```
3116 \def\@@cGLSpl@#1#2[#3]{%
3117   \glxtrifcounttrigger{#2}%
3118   {%
3119     \cGLSplformat{#2}{#3}%
3120     \glset{#2}%
3121   }%
3122   {%
3123     \@GLSpl@{#1}{#2}[#3]%
3124   }%
3125 }%
```

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

\@cgl's@

```
3126 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
3127 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
3128 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```

\@cGL'spl@

```
3129 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
3130 \newrobustcmd*{\cGLS}{\@gl'shyp@opt\@cGLS}
```

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

```
3131 \newcommand*{\@cGLS}[2][ ]{%
3132   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}%
3133 }
```

```

\@cGLS@
3134 \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.
3135 \newcommand*\cGLSformat[2]{%
3136   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSformat{#1}{#2}}}%
3137 }

\cGLSpl
3138 \newrobustcmd*\cGLSpl{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
3139 \newcommand*\@cGLSpl[2][{}]{%
3140   \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[{}]}%
3141 }

\@cGLSpl@
3142 \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.
3143 \newcommand*\cGLSplformat[2]{%
3144   \expandafter\mfirstuc\MakeUppercase\expandafter{\cGLSplformat{#1}{#2}}}%
3145 }

      Modify the trigger formats to check for the regular attribute.

\cglformat
3146 \renewcommand*\cglformat[2]{%
3147   \glsifregular{#1}
3148   {\glsentryfirst{#1}}%
3149   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
3150 }

\cGlsformat
3151 \renewcommand*\cGlsformat[2]{%
3152   \glsifregular{#1}
3153   {\Glsentryfirst{#1}}%
3154   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
3155 }

\cglsplformat
3156 \renewcommand*\cglsplformat[2]{%
3157   \glsifregular{#1}
3158   {\glsentryfirstplural{#1}}%
3159   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
3160 }

```

\cGlsplformat

```
3161 \renewcommand*{\cGlsplformat}[2]{%
3162   \glsifregular{#1}
3163   {\Glsentryfirstplural{#1}}%
3164   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
3165 }
```

New code similar to above for unit counting.

defunitcounters

```
3166 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3167   \edef\@glo@countunit{\csuse{\glstr@categoryattr@@\@glo@category @unitcount}}%
3168   \ifvoid\@glo@countunit
3169   {}%
3170   {%
3171     \@glstr@ifunitcounter{\@glo@countunit}%
3172     {}%
3173     {\expandafter\@glstr@addunitcounter\expandafter{\@glo@countunit}}%
3174   }%
3175 }
```

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

```
3176 \newcommand*{\@glstr@unitcountlist}{}%
```

@addunitcounter

```
3177 \newcommand*{\@glstr@addunitcounter}[1]{%
3178   \listadd{\@glstr@unitcountlist}{#1}%
3179   \ifcsundef{glstr@theunit@#1}
3180   {%
3181     \ifcsdef{theH#1}%
3182     {\csdef{glstr@theunit@#1}{\csuse{theH#1}}}%
3183     {\csdef{glstr@theunit@#1}{\csuse{the#1}}}%
3184   }%
3185   {}%
3186 }
```

r@ifunitcounter

```
3187 \newcommand*{\@glstr@ifunitcounter}[3]{%
3188   \xifinlist{#1}{\@glstr@unitcountlist}{#2}{#3}%
3189 }
```

urrentunitcount

```
3190 \newcommand*{\@glstr@currentunitcount}[1]{%
3191   glo@\glstetoklabel{#1}@currunit@\glstetattribute{#1}{unitcount}.%
3192   \csuse{glstr@theunit@\glstetattribute{#1}{unitcount}}%
3193 }
```

previousunitcount

```
3194 \newcommand*\@glsxtr@previousunitcount[1]{%
3195   glo@\glsdetoklabel{#1}@prevunit@\glsgetattribute{#1}{unitcount}.%
3196   \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3197 }
```

t@currunitcount

```
3198 \newcommand*\@gls@increment@currunitcount[1]{%
3199   \gls@hasattribute{#1}{unitcount}%
3200   {%
3201     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3202     \ifcsundef{\@glsxtr@csname}%
3203     {%
3204       \csgdef{\@glsxtr@csname}{1}%
3205       \listcsxadd
3206         {glo@\glsdetoklabel{#1}@unitlist}%
3207         {\glsgetattribute{#1}{unitcount}.%
3208         \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3209     }%
3210   }%
3211   {%
3212     \csxdef{\@glsxtr@csname}%
3213     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3214   }%
3215   }%
3216   {}%
3217 }
```

t@currunitcount

```
3218 \newcommand*\@gls@local@increment@currunitcount[1]{%
3219   \gls@hasattribute{#1}{unitcount}%
3220   {%
3221     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3222     \ifcsundef{\@glsxtr@csname}%
3223     {%
3224       \csdef{\@glsxtr@csname}{1}%
3225       \listcseadd
3226         {glo@\glsdetoklabel{#1}@unitlist}%
3227         {\glsgetattribute{#1}{unitcount}.%
3228         \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3229     }%
3230   }%
3231   {%
3232     \csedef{\@glsxtr@csname}%
3233     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3234   }%
3235   }%
3236   {}%
3237 }
```



r@currunitcount

```
3238 \newcommand*{\@glxstr@currunitcount}[2]{%
3239   \ifcsundef
3240     {glo@\glstetoklabel{#1}@currunit@#2}%
3241     {0}%
3242     {\csuse{glo@\glstetoklabel{#1}@currunit@#2}}%
3243 }%
```

r@prevunitcount

```
3244 \newcommand*{\@glxstr@prevunitcount}[2]{%
3245   \ifcsundef
3246     {glo@\glstetoklabel{#1}@prevunit@#2}%
3247     {0}%
3248     {\csuse{glo@\glstetoklabel{#1}@prevunit@#2}}%
3249 }%
```

entryunitcount

```
3250 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3251   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3252   \renewcommand*{\glstetoknewglossaryentry}{%
3253     \renewcommand*\newglossaryentry[2]{%
3254       \PackageError{glossaries}{\string\newglossaryentry\space
3255         may only be used in the preamble when entry counting has
3256         been activated}{If you use \string\glsenableentryunitcount\space
3257         you must place all entry definitions in the preamble not in
3258         the document environment}%
3259     }%
3260   }%
```

New commands to access new fields:

```
3261 \newcommand*{\glstentrycurrcount}[1]{%
3262   \@glxstr@currunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3263   \csuse{\glxstr@theunit@\glstgetattribute{##1}{unitcount}}}%
3264 }%
3265 \newcommand*{\glstentryprevcount}[1]{%
3266   \@glxstr@prevunitcount{##1}{\glstgetattribute{##1}{unitcount}}.%
3267   \csuse{\glxstr@theunit@\glstgetattribute{##1}{unitcount}}}%
3268 }%
```

Access total count:

```
3269 \newcommand*{\glstentryprevtotalcount}[1]{%
3270   \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
3271     {0}%
3272     {%
3273       \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
3274     }%
3275   }%
```

Access max value:

```

3276 \newcommand*{\glstentryprevmaxcount}[1]{%
3277   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
3278     {0}%
3279     {%
3280       \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
3281     }%
3282   }%

```

Adjust post unset and reset:

```

3283 \let\@glstxtr@entryunitcount@org@unset\glstxtrpostunset
3284 \renewcommand*{\glstxtrpostunset}[1]{%
3285   \@glstxtr@entryunitcount@org@unset{##1}%
3286   \@glst@increment@currunitcount{##1}%
3287 }%
3288 \let\@glstxtr@entryunitcount@org@localunset\glstxtrpostlocalunset
3289 \renewcommand*{\glstxtrpostlocalunset}[1]{%
3290   \@glstxtr@entryunitcount@org@localunset{##1}%
3291   \@glst@local@increment@currunitcount{##1}%
3292 }%
3293 \let\@glstxtr@entryunitcount@org@reset\glstxtrpostreset
3294 \renewcommand*{\glstxtrpostreset}[1]{%
3295   \glshasattribute{##1}{unitcount}%
3296   {%
3297     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3298     \ifcsundef{\@glstxtr@csname}%
3299     {}%
3300     {\csgdef{\@glstxtr@csname}{0}}%
3301   }%
3302   {}%
3303 }%
3304 \let\@glstxtr@entryunitcount@org@localreset\glstxtrpostlocalreset
3305 \renewcommand*{\glstxtrpostlocalreset}[1]{%
3306   \@glstxtr@entryunitcount@org@localreset{##1}%
3307   \glshasattribute{##1}{unitcount}%
3308   {%
3309     \edef\@glstxtr@csname{\@glstxtr@currentunitcount{##1}}%
3310     \ifcsundef{\@glstxtr@csname}%
3311     {}%
3312     {\csdef{\@glstxtr@csname}{0}}%
3313   }%
3314   {}%
3315 }%

```

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

```

3316 \let\@cglst@\@cglst@
3317 \let\@cglstpl@\@cglstpl@
3318 \let\@cGls@\@cGls@

```

```

3319 \let\@cGlspl@\@cGlspl@
3320 \let\@cGLS@\@cGLS@
3321 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3322 \AtEndDocument{\@gls@write@entryunitcounts}%
3323 \renewcommand*{\@gls@entry@unitcount}[3]{%
3324   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
3325   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
3326   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
3327   {%
3328     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
3329       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
3330     }%
3331     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
3332     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
3333     {%
3334       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
3335       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
3336       \fi
3337     }%
3338   }%
3339 \let\glsenableentryunitcount\relax
3340 \renewcommand*{\glsenableentrycount}{%
3341   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3342     can't be used with \string\glsenableentryunitcount}%
3343   {Use one or other but not both commands}%
3344 }%
3345 }
3346 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3347 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3348 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3349   \immediate\write\@auxout
3350   {\string\@gls@entry@unitcount
3351     {\@gls@entry}%
3352     {\@glsxtr@currunitcount{\@gls@entry}{#1}%
3353     }%
3354     {#1}}%
3355 }

```

entryunitcounts

```

3356 \newcommand*{\@gls@write@entryunitcounts}{%
3357   \immediate\write\@auxout
3358   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3359   \count@=0\relax

```

```

3360 \foralllglsentries{\@glsentry}{%
3361   \glshasattribute{\@glsentry}{unitcount}%
3362   {%
3363     \ifglssused{\@glsentry}%
3364     {%
3365       \forlistcsloop
3366         {\@gls@write@entryunitcounts@do}%
3367         {glo@\glsdetoklabel{\@glsentry}@unitlist}%
3368     }%
3369   }%
3370   \advance\count@ by \@ne
3371 }%
3372 {}%
3373 }%
3374 \ifnum\count@=0
3375   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3376   \MessageBreak with \string\glsenableentryunitcount\space but the
3377   \MessageBreak attribute ‘unitcount’ hasn’t
3378   \MessageBreak been assigned to any of the defined
3379   \MessageBreak entries}%
3380 \fi
3381 }

```

`\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.

```

3382 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3383   \glsenableentryunitcount

```

Redefine \gls etc:

```

3384   \renewcommand*{\gls}{\cgl}%
3385   \renewcommand*{\Gls}{\cGls}%
3386   \renewcommand*{\glspl}{\cglspl}%
3387   \renewcommand*{\Glspl}{\cGlspl}%
3388   \renewcommand*{\GLS}{\cGLS}%
3389   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3390   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3391   \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
3392   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3393     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3394     can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3395     {Use one or other but not both commands}}%
3396 }

```

`\tcountunsetattr`

```

3397 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3398   \@for\@glsxtr@cat:=#1\do
3399   {%
3400     \ifdefempty{\@glsxtr@cat}{}%
3401     {%
3402       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3403       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3404     }%
3405   }%
3406 }

```

### 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`

```

3407 \renewcommand*{\SetGenericNewAcronym}{%
3408   \let\@Gls@entryname\@Gls@acrenryname
3409   \renewcommand{\newacronym}[4][]{%
3410     \ifdefempty{\@glsacronymlists}%
3411     {%
3412       \def\@glo@type{\acronymtype}%
3413       \setkeys{glossentry}{##1}%
3414       \DeclareAcronymList{\@glo@type}%
3415     }%
3416   }%
3417   \glskeylisttok{##1}%
3418   \glslabeltok{##2}%
3419   \glsshorttok{##3}%
3420   \glslongtok{##4}%
3421   \newacronymhook
3422   \protected@edef\@do@newglossaryentry{%
3423     \noexpand\newglossaryentry{\the\glslabeltok}%
3424     {%
3425       type=\acronymtype,%
3426       name={\expandonce{\acronymentry{##2}}},%
3427       sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
3428       text={\the\glsshorttok},%
3429       short={\the\glsshorttok},%
3430       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3431       long={\the\glslongtok},%
3432       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3433       category=acronym,

```

```

3434 \GenericAcronymFields,%
3435 \the\glskeylisttok
3436 }%
3437 }%
3438 \@do@newglossaryentry
3439 }%
3440 \renewcommand*{\acrfullfmt}[3]{%
3441 \glslink{##1}{##2}{\genacrfullformat{##2}{##3}}}%
3442 \renewcommand*{\Acrfullfmt}[3]{%
3443 \glslink{##1}{##2}{\Genacrfullformat{##2}{##3}}}%
3444 \renewcommand*{\ACRfullfmt}[3]{%
3445 \glslink{##1}{##2}{%
3446 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}}%
3447 \renewcommand*{\acrfullplfmt}[3]{%
3448 \glslink{##1}{##2}{\genplacrfullformat{##2}{##3}}}%
3449 \renewcommand*{\Acrfullplfmt}[3]{%
3450 \glslink{##1}{##2}{\Genplacrfullformat{##2}{##3}}}%
3451 \renewcommand*{\ACRfullplfmt}[3]{%
3452 \glslink{##1}{##2}{%
3453 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}}%
3454 \renewcommand*{\glstryfull}[1]{\genacrfullformat{##1}{}}%
3455 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
3456 \renewcommand*{\glstryfullpl}[1]{\genplacrfullformat{##1}{}}%
3457 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
3458 }

```

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:

```

3459 \let\@glstr@org@setacronymstyle\setacronymstyle
3460 \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`.

```

3461 \newcommand*{\MakeAcronymsAbbreviations}{%
3462 \renewcommand*{\newacronym}[4][ ]{%
3463 \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3464 }%
3465 \renewcommand*{\firstacronymfont}[1]{\glstr@firstabbrvfont{##1}}%
3466 \renewcommand*{\acronymfont}[1]{\glstr@abbrvfont{##1}}%
3467 \renewcommand*{\setacronymstyle}[1]{%
3468 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3469 unavailable.
3470 Use \string\setabbreviationstyle\space instead.
3471 The original acronym interface can be restored with
3472 \string\RestoreAcronyms}{}%
3473 }%
3474 \renewcommand*{\newacronymstyle}[1]{%

```

```

3475 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3476 available unless you restore the original acronym interface with
3477 \string\RestoreAcronyms}%
3478 \@glstr@org@newacronymstyle{##1}%
3479 }%
3480 }

```

Switch acronyms to abbreviations:

```

3481 \MakeAcronymsAbbreviations

```

**RestoreAcronyms** Restore acronyms to glossaries interface.

```

3482 \newcommand*{\RestoreAcronyms}{%
3483 \SetGenericNewAcronym
3484 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3485 \renewcommand{\acronymfont}[1]{##1}%
3486 \let\setacronymstyle\@glstr@org@setacronymstyle
3487 \let\newacronymstyle\@glstr@org@newacronymstyle

```

Need to restore the original definition of \@gls@link@checkfirsthyper but \glstrifwasfirstuse still needs setting for the benefit of the post-link hook.

```

3488 \renewcommand*\@gls@link@checkfirsthyper{%
3489 \ifglused{\glslabel}%
3490 {\let\glstrifwasfirstuse\@secondoftwo}
3491 {\let\glstrifwasfirstuse\@firstoftwo}%
3492 \@glstr@org@checkfirsthyper
3493 }
3494 \glssetcategoryattribute{acronym}{regular}{false}%
3495 \setacronymstyle{long-short}%
3496 }

```

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

```

3497 \renewcommand*{\glsacspace}[1]{%
3498 \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3499 \ifdim\dimen@<\glsacspacemax~\else\space\fi
3500 }

```

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

```

3501 \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**

```

3502 \newcommand*{\@glstr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```
3503 \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
```

```
3504 \renewcommand*{\makeglossaries}[1] [] {%
3505   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3506     \PackageError{glossaries-extra}{\string\makeglossaries\space
3507       not permitted\MessageBreak with record=only package option}%
3508     {You may only use \string\makeglossaries\space with
3509       record=off or record=alsoindex options}%
3510   \else
3511     \ifblank{#1}%
3512     {\@glxtr@org@makeglossaries}%
3513     {%
3514       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3515         \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3516           not permitted\MessageBreak with record=alsoindex package option}%
3517         {You may only use the hybrid \string\makeglossaries[...]\space with
3518           record=off option}%
3519       \else
3520         \edef\@glxtr@reg@glosslist{#1}%
3521         \ifundef{\glswrite}{\newwrite\glswrite}{}%
3522         \protected@write\@auxout{}{\string\providecommand
3523           \string\@glxtr@order[1]{}%
3524           \protected@write\@auxout{}{\string\providecommand
3525             \string\@istfilename[1]{}%
3526             \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3527             \protected@write\@auxout{}{\string\@glxtr@order{\glxtr@order}}%
3528             \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}%
3529             \write\@auxout{\string\providecommand\string\@glxtr@reference[3]{}%

```

Iterate through each supplied glossary type and activate it.

```
3530   \@for\@glo@type:=#1\do{%
3531     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3532   }%
```

New glossaries must be created before `\makeglossaries`:

```
3533   \renewcommand*\newglossary[4] [] {%
3534     \PackageError{glossaries}{New glossaries
3535       must be created before \string\makeglossaries}{You need
3536       to move \string\makeglossaries\space after all your
3537       \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
3538   \let\@makeglossary\relax
3539   \let\makeglossary\relax
```



```

3540      \renewcommand\makeglossaries[1][{}]{%
Disable all commands that have no effect after \makeglossaries
3541      \@disable@onlypremakeg
Allow see key:
3542      \let\gls@checkseeallowed\relax
Adjust \@do@seeglossary. This needs to check for the entries existence.
3543      \renewcommand*{\@do@seeglossary}[2]{%
3544      \glsdoifexists{##1}%
3545      {%
3546      \edef\gls@label{\glsdetoklabel{##1}}%
3547      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3548      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3549      {\@glsxtr@org@doseeglossary{##1}{##2}}%
3550      {%
3551      \@glsxtrwrglossmark
3552      \protected@write\@auxout{}{%
3553      \string\gls@reference
3554      {\gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
3555      }%
3556      }%
3557      }%
3558      }%
Adjust \@do@@wrglossary
3559      \let\@glsxtr@@do@@wrglossary\@do@@wrglossary
3560      \def\@do@@wrglossary{%
3561      \edef\@gls@type{\csname glo@\gls@label @type\endcsname}%
3562      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3563      {\@glsxtr@@do@@wrglossary}%
3564      {\gls@noidxglossary}%
3565      }%
Suppress warning about no \makeglossaries
3566      \let\warn@nomakeglossaries\relax
3567      \def\warn@noprintglossary{%
3568      \GlossariesWarningNoLine{No \string\printglossary\space
3569      or \string\printglossaries\space
3570      found.^^J(Remove \string\makeglossaries\space if you don't want
3571      any glossaries.)^^JThis document will not have a glossary}%
3572      }%
Only warn for glossaries not listed.
3573      \renewcommand{\@gls@noref@warn}[1]{%
3574      \edef\@gls@type{##1}%
3575      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3576      {%
3577      \GlossariesExtraWarning{Can't use
3578      \string\printnoidxglossary[type={\@gls@type}]
3579      when '\@gls@type' is listed in the optional argument of

```

```

3580         \string\makeglossaries}%
3581     }%
3582     {%
3583         \GlossariesWarning{Empty glossary for
3584         \string\printnoidxglossary[type={##1}].
3585         Rerun may be required (or you may have forgotten to use
3586         commands like \string\gls)}%
3587     }%
3588 }%

```

Adjust display number list to check for type:

```

3589     \renewcommand*{\glsdisplaynumberlist}[1]{%
3590         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3591         {\@glxtr@idx@displaynumberlist{##1}}%
3592         {\@glxtr@noidx@displaynumberlist{##1}}%
3593     }%

```

Adjust entry list:

```

3594     \renewcommand*{\glsentrynumberlist}[1]{%
3595         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3596         {\@glxtr@idx@entrynumberlist{##1}}%
3597         {\@glxtr@noidx@entrynumberlist{##1}}%
3598     }%

```

Adjust number list loop

```

3599     \renewcommand*{\glsnumberlistloop}[2]{%
3600         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3601         {%
3602             \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3603             not available for glossary ‘##1’}{}%
3604         }%
3605         {\@glxtr@noidx@numberlistloop{##1}{##2}}%
3606     }%

```

Only sanitize sort for normal indexing glossaries.

```

3607     \renewcommand*{\glsprestandardsort}[3]{%
3608         \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
3609         {%
3610             \glsdosanitizesort
3611         }%
3612         {%
3613             \ifglssanitizesort
3614                 \@gls@noidx@sanitizesort
3615             \else
3616                 \@gls@noidx@nosanitizesort
3617             \fi
3618         }%
3619     }%

```

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

```

3620 \renewcommand*\new@glossaryentry[2]{%
3621 \PackageError{glossaries-extra}{Glossary entries must be defined
3622 in the preamble\MessageBreak when you use the optional argument
3623 of \string\makeglossaries}{Either move your definitions to the
3624 preamble or don't use the optional argument of
3625 \string\makeglossaries}%
3626 }%

```

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).

```

3627 \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
3628 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3629 \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
3630 type=\glsdefaulttype,\@end@glxtr@gettype
3631 \def\@glo@sorttype{\@glo@default@sorttype}%
3632 }%

```

Check automake setting:

```

3633 \ifglsautomake
3634 \renewcommand*\@gls@doautomake{%
3635 \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
3636 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3637 }%
3638 }%
3639 \fi

```

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

```

3640 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3641 \fi
3642 }%
3643 \fi
3644 }

```

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` 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.)

```

3645 \newcommand{\@glxtr@orgprintglossary}[2]{%
3646 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3647 \def\glossarytitle{%
3648 \ifcsdef{\@glo@type@\@glo@type @title}%
3649 {\csuse{\@glo@type@\@glo@type @title}}%
3650 {\glossaryname}}%
3651 \def\glossarytoctitle{\glossarytitle}%

```

```

3652 \let\org@glossarytitle\glossarytitle
3653 \def\@glossarystyle{%
3654   \ifx\@glossary@default@style\relax
3655     \GlossariesWarning{No default glossary style provided \MessageBreak
3656       for the glossary '\@glo@type'. \MessageBreak
3657       Using deprecated fallback. \MessageBreak
3658       To fix this set the style with \MessageBreak
3659       \string\setglossarystyle\space or use the \MessageBreak
3660       style key=value option}%
3661   \fi
3662 }%
3663 \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
3664 \let\@org@glossaryentrynumbers\glossaryentrynumbers
3665 \bgroup
3666   \@printgloss@setsort
3667   \setkeys{printgloss}{#1}%
3668   \ifx\glossarytitle\org@glossarytitle
3669   \else
3670     \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3671   \fi
3672   \let\currentglossary\@glo@type
3673   \let\org@glossaryentrynumbers\glossaryentrynumbers
3674   \let\glsnonextpages\@glsnonextpages
3675   \let\glsnextpages\@glsnextpages
3676   \let\nopostdesc\@nopostdesc
3677   \gls@dotoc@title
3678   \@glossarystyle
3679   \let\gls@org@glossaryentryfield\glossentry
3680   \let\gls@org@glossarysubentryfield\subglossentry
3681   \renewcommand{\glossentry}[1]{%
3682     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3683     \gls@org@glossaryentryfield{##1}%
3684   }%
3685   \renewcommand{\subglossentry}[2]{%
3686     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3687     \gls@org@glossarysubentryfield{##1}{##2}%
3688   }%
3689   \@gls@preglossaryhook
3690   #2%
3691 \egroup
3692 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3693 \global\let\warn@noprintglossary\relax
3694 }

```

\@printglossary Redefine.

```

3695 \renewcommand{\@printglossary}[2]{%
3696   \def\@glsxtr@printglossopts{#1}%
3697   \@glsxtr@orgprintglossary{#1}{#2}%
3698 }

```

Add a key that switches off the entry targets:

```
3699 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3700   \ifcase\nr
3701     \let\@glstarget\glsdohypertarget
3702   \else
3703     \let\@glstarget\@secondoftwo
3704   \fi
3705 }
```

hypernameprefix

```
3706 \newcommand{\@glxtrhypernameprefix}{{}
```

New to v1.20:

```
3707 \define@key{printgloss}{targetnameprefix}{%
3708   \renewcommand{\@glxtrhypernameprefix}{#1}%
3709 }
```

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

```
3710 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3711 \renewcommand{\glsdohypertarget}[2]{%
3712   \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3713 }
```

@makeglossaries   For the benefit of makeglossaries

```
3714 \newcommand*{\glxtr@makeglossaries}[1]{{}
```

@glxtr@gettype   Get just the type.

```
3715 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3716   \def\@glo@type{#2}%
3717 }
```

@assign@sortkey   Assign the sort key.

```
3718 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3719   \edef\@glo@type{\@glo@type}%
3720   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3721   {%
3722     \@glo@no@assign@sortkey{#1}%
3723   }%
3724   {%
3725     \@glo@assign@sortkey{#1}%
3726   }%
3727 }%
```

Display number list for the regular version:

splaynumberlist

```
3728 \let\@glxtr@idx@displaynumberlist\glstdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```

3729 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
3730   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3731   \ifdef\@gls@loclist
3732   {%
3733     \def\@gls@noidxloclist@sep{%
3734       \def\@gls@noidxloclist@sep{%
3735         \def\@gls@noidxloclist@sep{%
3736           \glsnumlistsep
3737         }%
3738         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3739       }%
3740     }%
3741     \def\@gls@noidxloclist@finalsep{}%
3742     \def\@gls@noidxloclist@prev{}%
3743     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3744     \@gls@noidxloclist@finalsep
3745     \@gls@noidxloclist@prev
3746   }%
3747   {%

3748   \glsxtrundeftag
3749   \glsdoifexists{#1}%
3750   {%
3751     \GlossariesWarning{Missing location list for ‘#1’. Either
3752       a rerun is required or you haven’t referenced the entry.}%
3753   }%
3754 }%
3755 }%
3756

```

And for the number list loop:

@numberlistloop

```

3757 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3758   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3759   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3760   \let\@gls@org@glsseeformat\glsseeformat
3761   \let\glsnoidxdisplayloc#2\relax
3762   \let\glsseeformat#3\relax
3763   \ifdef\@gls@loclist
3764   {%
3765     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3766   }%
3767   {%

3768   \glsxtrundeftag
3769   \glsdoifexists{#1}%
3770   {%
3771     \GlossariesWarning{Missing location list for ‘##1’. Either

```

```

3772         a rerun is required or you haven't referenced the entry.}%
3773     }%
3774 }%
3775 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3776 \let\glsseeformat\@gls@org@glsseeformat
3777 }%

```

Same for entry number list.

entrynumberlist

```

3778 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3779   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3780   \ifdef\@gls@loclist
3781   {%
3782     \glsnoidxloclist{\@gls@loclist}%
3783   }%
3784   {%
3785     \glsxtrundeftag
3786     \glsdoifexists{#1}%
3787     {%
3788       \GlossariesWarning{Missing location list for '#1'. Either
3789         a rerun is required or you haven't referenced the entry.}%
3790     }%
3791   }%
3792 }%

```

entrynumberlist

```

3793 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3794 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3795   \protected@edef\@glsxtr@titlelabel{#1}%
3796   \ifdefvoid\@glsxtr@titlelabel
3797   {%
3798   {%
3799     \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
3800   }%
3801   \ifdefvoid{\@glsxtr@titlelabel}%
3802   {%
3803     \DTLifint{#1}%
3804     {%
3805       \ifnum#1<256\relax
3806       \edef#2{\char#1\relax}%
3807     \else
3808       \edef#2{#1}%
3809     \fi
3810   }%
3811   {%
3812     \ifcsundef{#1groupname}%

```

```

3813      {\def#2{#1}}%
3814      {\letcs#2{#1groupname}}%
3815    }%
3816  }%
3817  {%
3818    \let#2\@glstr@titlelabel
3819  }%
3820 }

```

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

```

3821 \let\glstr@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.

```

3822 \newrobustcmd{\glstrgetgrouptitle}[2]{%
3823   \protected@edef\@glstr@titlelabel{\glstr@grouptitle@#1}%
3824   \@onelevel@sanitize\@glstr@titlelabel
3825   \ifcsdef{\@glstr@titlelabel}
3826     {\letcs{#2}{\@glstr@titlelabel}}%
3827     {\glstr@org@getgrouptitle{#1}{#2}}%
3828 }
3829 \let\@gls@getgrouptitle\glstrgetgrouptitle

```

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

```

3830 \newcommand{\glstrsetgrouptitle}[2]{%
3831   \protected@edef\@glstr@titlelabel{\glstr@grouptitle@#1}%
3832   \@onelevel@sanitize\@glstr@titlelabel
3833   \csxdef{\@glstr@titlelabel}{#2}%
3834 }

```

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

```

3835 \renewcommand*{\glsnavigation}{%
3836   \def\@gls@between{}%
3837   \ifcsundef{\@gls@hypergroup@list@\@glo@type}%
3838     {%
3839       \def\@gls@list{}%
3840     }%
3841     {%
3842       \expandafter\let\expandafter\@gls@list
3843         \csname \@gls@hypergroup@list@\@glo@type\endcsname
3844     }%
3845     \@for\@gls@tmp:=\@gls@list\do{%
3846       \@gls@between
3847       \glstrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3848       \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3849       \let\@gls@between\glshypernavsep
3850     }%
3851 }

```



@noidx@glossary

```
3852 \renewcommand*{\@print@noidx@glossary}{%
3853   \ifcsdef{@glsref@{\@glo@type}%
3854   {%
3855     \ifcsdef{@glo@sortmacro@{\@glo@sorttype}%
3856     {%
3857       \csuse{@glo@sortmacro@{\@glo@sorttype}}{\@glo@type}%
3858     }%
3859   {%
3860     \PackageError{glossaries}{Unknown sort handler '@glo@sorttype'}{ }%
3861   }%
3862   \glossarysection[\glossarytoctitle]{\glossarytitle}%
3863   \glossarypreamble
```

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

```
3864   \def{@gls@currentlettergroup}{%
3865   \begin{theglossary}%
3866   \glossaryheader
3867   \glsresetentrylist
3868   \forlistcsloop{@gls@noidx@do}{@glsref@{\@glo@type}%
3869   \end{theglossary}%
3870   \glossarypostamble
3871 }%
3872 {%
```

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

```
3873   \glstrifemptyglossary{\@glo@type}%
3874   }%
3875   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3876   \@gls@noref@warn{\@glo@type}%
3877 }%
3878 }
```

noidxdisplayloc Patch to check for range formations.

```
3879 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3880   \setentrycounter[#1]{#2}%
3881   \@glstr@display@loc#3\empty\end@glstr@display@loc{#4}%
3882 }
```

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

```
3883 \def{@glstr@display@loc#1#2\end@glstr@display@loc#3{%
3884   \ifx#1(\relax
3885     \glstrdisplaystartloc{#2}{#3}%
3886   \else
3887     \ifx#1)\relax
3888       \glstrdisplayendloc{#2}{#3}%
3889     \else
```

```

3890      \glxtrdisplaysingleloc{#1#2}{#3}%
3891      \fi
3892      \fi
3893 }

```

`\glxtrdisplaysingleloc` Single location.

```

3894 \newcommand*{\glxtrdisplaysingleloc}[2]{%
3895   \csuse{#1}{#2}%
3896 }

```

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`.

`\glxtrdisplaystartloc` Start of a location range.

```

3897 \newcommand*{\glxtrdisplaystartloc}[2]{%
3898   \edef\glxtrlocrangefmt{#1}%
3899   \ifx\glxtrlocrangefmt\empty
3900     \def\glxtrlocrangefmt{\glsnnumberformat}%
3901   \fi
3902   \expandafter\glxtrdisplaysingleloc
3903   \expandafter{\glxtrlocrangefmt}{#2}%
3904 }

```

`\glxtrdisplayendloc` End of a location range.

```

3905 \newcommand*{\glxtrdisplayendloc}[2]{%
3906   \edef\@glxtr@tmp{#1}%
3907   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{\glsnnumberformat}}{}%
3908   \ifx\glxtrlocrangefmt\@glxtr@tmp
3909     \else
3910       \GlossariesExtraWarning{Mismatched end location range
3911         (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3912     \fi
3913     \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3914     \expandafter\glxtrdisplaysingleloc
3915     \expandafter{\glxtrlocrangefmt}{#2}%
3916     \def\glxtrlocrangefmt{}%
3917 }

```

`\glxtrdisplayendloohook` Allow the user to hook into the end of range command.

```

3918 \newcommand*{\glxtrdisplayendloohook}[2]{%

```

`\glxtrlocrangefmt` Current range format. Empty if not in a range.

```

3919 \newcommand*{\glxtrlocrangefmt}{}

```

`\glxtrls@removespaces` Redefine to allow adjustments to location hyperlink.

```

3920 \def\@glxtr@removespaces#1 #2\@nil{%
3921   \toks@=\expandafter{\the\toks@#1}%
3922   \ifx\#2\%

```

```

3923 \edef\x{\the\toks@}%
3924 \ifx\x\empty
3925 \else
3926 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3927 \fi
3928 \else
3929 \@gls@ReturnAfterFi{%
3930 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3931 }%
3932 \fi
3933 }

```

cationhyperlink

```

3934 \newcommand*{\glxtrlocationhyperlink}[3]{%
3935 \ifdefined\glxtrsupplocationurl
3936 {%
3937 \glxtrhyperlink{#1#2#3}{#3}%
3938 }%
3939 {%
3940 \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
3941 }%
3942 }

```

supphypernumber

```

3943 \newcommand*{\glxtrsupphypernumber}[1]{%
3944 {%
3945 \glshasattribute{\glscurrententrylabel}{externallocation}%
3946 {%
3947 \def\glxtrsupplocationurl{%
3948 \glsglshattribute{\glscurrententrylabel}{externallocation}}%
3949 }%
3950 {%
3951 \def\glxtrsupplocationurl{}%
3952 }%
3953 \glshypernumber{#1}%
3954 }%
3955 }

```

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

@print@glossary

```

3956 \renewcommand{\@print@glossary}{%
3957 \makeatletter
3958 \@input{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
3959 \ifFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
3960 {}%
3961 {\glxtrNoGlossaryWarning{\@glo@type}}%
3962 \ifglxindy

```

```

3963 \ifcsundef{@xdy@\@glo@type @language}%
3964 {%
3965 \edef\@do@auxoutstuff{%
3966 \noexpand\AtEndDocument{%
3967 \noexpand\immediate\noexpand\write\@auxout{%
3968 \string\providecommand\string\@xdy@language[2]{}}%
3969 \noexpand\immediate\noexpand\write\@auxout{%
3970 \string\@xdy@language{\@glo@type}{\@xdy@main@language}}%
3971 }%
3972 }%
3973 }%
3974 {%
3975 \edef\@do@auxoutstuff{%
3976 \noexpand\AtEndDocument{%
3977 \noexpand\immediate\noexpand\write\@auxout{%
3978 \string\providecommand\string\@xdy@language[2]{}}%
3979 \noexpand\immediate\noexpand\write\@auxout{%
3980 \string\@xdy@language{\@glo@type}{\csname @xdy@\@glo@type
3981 @language\endcsname}}%
3982 }%
3983 }%
3984 }%
3985 \@do@auxoutstuff
3986 \edef\@do@auxoutstuff{%
3987 \noexpand\AtEndDocument{%
3988 \noexpand\immediate\noexpand\write\@auxout{%
3989 \string\providecommand\string\@gls@codepage[2]{}}%
3990 \noexpand\immediate\noexpand\write\@auxout{%
3991 \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
3992 }%
3993 }%
3994 \@do@auxoutstuff
3995 \fi
3996 \renewcommand*{\@warn@nomakeglossaries}{%
3997 \GlossariesWarningNoLine{\string\makeglossaries\space
3998 hasn't been used,^^Jthe glossaries will not be updated}%
3999 }%
4000 }

```

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

`oGlsWarningHead` Header message.

```

4001 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4002 This document is incomplete. The external file associated with
4003 the glossary '#1' (which should be called \texttt{#2})
4004 hasn't been created.%
4005 }

```

`arningEmptyStart` No entries have been added to the glossary.

```

4006 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4007   This has probably happened because there are no entries defined
4008   in this glossary.%
4009 }

```

WarningEmptyMain The default “main” glossary is empty.

```

4010 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4011   If you don't want this glossary,
4012   add \texttt{nomain} to your package option list when you load
4013   \texttt{glossaries-extra.sty}. For example:%
4014 }

```

WarningEmptyNotMain A glossary that isn't the default “main” glossary is empty.

```

4015 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4016   Did you forget to use \texttt{type=#1} when you defined your
4017   entries? If you tried to load entries into this glossary with
4018   \texttt{\string\loadglsentries} did you remember to use
4019   \texttt{[#1]} as the optional argument? If you did, check that
4020   the definitions in the file you loaded all had the type set
4021   to \texttt{\string\glsdefaulttype}.%
4022 }

```

WarningCheckFile Advisory message to check the file contents.

```

4023 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4024   Check the contents of the file \texttt{#1}. If
4025   it's empty, that means you haven't indexed any of your entries in this
4026   glossary (using commands like \texttt{\string\gls} or
4027   \texttt{\string\glsadd}) so this list can't be generated.
4028   If the file isn't empty, the document build process hasn't been
4029   completed.%
4030 }

```

WarningAutoMake Message when automake option has been used.

```

4031 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4032   You may need to rerun \LaTeX. If you already have, it may be that
4033   \TeX's shell escape doesn't allow you to run
4034   \ifglxindy xindy\else makeindex\fi. Check the
4035   transcript file \texttt{\jobname.log}. If the shell escape is
4036   disabled, try one of the following:
4037
4038   \begin{itemize}
4039     \item Run the external (Lua) application:
4040
4041       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4042
4043     \item Run the external (Perl) application:
4044
4045       \texttt{makeglossaries \string"\jobname\string"}
4046   \end{itemize}

```

```

4047
4048 Then rerun \LaTeX\ on this document.
4049 \GlossariesExtraWarning{Rerun required to build the
4050 glossary ‘#1’ or check TeX’s shell escape allows
4051 you to run \ifglxsindy xindy\else makeindex\fi}%
4052 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4053 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
4054   You need to either replace \texttt{\string\makenoidxglossaries}
4055   with \texttt{\string\makeglossaries} or replace
4056   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4057   \texttt{\string\printnoidxglossary}
4058   (or \texttt{\string\printnoidxglossaries}) and then rebuild
4059   this document.%
4060 }

```

WarningBuildInfo Build advice.

```

4061 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4062   Try one of the following:
4063   \begin{itemize}
4064     \item Add \texttt{automake} to your package option list when you load
4065       \texttt{glossaries-extra.sty}. For example:
4066
4067       \texttt{\string\usepackage[automake]%
4068         \glsopenbrace glossaries-extra\glsclosebrace}
4069
4070     \item Run the external (Lua) application:
4071
4072     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4073
4074     \item Run the external (Perl) application:
4075
4076     \texttt{makeglossaries \string"\jobname\string"}
4077   \end{itemize}
4078
4079   Then rerun \LaTeX\ on this document.%
4080 }

```

oGlsWarningTail Final paragraph.

```

4081 \newcommand{\GlsXtrNoGlsWarningTail}{%
4082   This message will be removed once the problem has been fixed.%
4083 }

```

GlsWarningNoOut No out file created. Build advice.

```

4084 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4085   The file \texttt{#1} doesn’t exist. This most likely means you haven’t used
4086   \texttt{\string\makeglossaries} or you have used

```

```

4087 \texttt{\string\nofiles}. If this is just a draft version of the
4088 document, you can suppress this message using the
4089 \texttt{nomissingglstext} package option.%
4090 }

```

glossarywarning

```

4091 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
4092 \glossarysection[\glossarytoctitle]{\glossarytitle}
4093 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
4094 \par
4095 \glstrifemptyglossary{#1}%
4096 {%
4097 \GlsXtrNoGlsWarningEmptyStart\space
4098 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4099 \medskip
4100 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
4101 \glsoopenbrace glossaries-extra\glsclosebrace}
4102 \medskip
4103 }%
4104 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4105 }%
4106 {%
4107 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
4108 {%
4109 \GlsXtrNoGlsWarningCheckFile
4110 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
4111
4112 \ifglssautomake
4113
4114 \GlsXtrNoGlsWarningAutoMake{#1}
4115
4116 \else
4117
4118 \ifthenelse{\equal{#1}{main}}%
4119 {%
4120 \GlsXtrNoGlsWarningEmptyMain\par
4121 \medskip
4122 \noindent\texttt{\string\usepackage[nomain]%
4123 \glsoopenbrace glossaries-extra\glsclosebrace}
4124 \medskip
4125 }%
4126 {}%
4127
4128 \ifdefequal\makeglossaries\@no@makeglossaries
4129 {%
4130 \GlsXtrNoGlsWarningMisMatch
4131 }%
4132 {}%
4133 \GlsXtrNoGlsWarningBuildInfo

```

```

4134     }%
4135     \fi
4136   }%
4137   {%
4138     \GlsXtrNoGlsWarningNoOut
4139     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4140   }%
4141 }%
4142 \par
4143 \GlsXtrNoGlsWarningTail
4144 }

```

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

**xtresourcefile** 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.

```
4145 \newcommand*{\glxtrresourcefile}[2] [] {%
```

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

```

4146   \disable@keys{glossaries-extra.sty}{record}%
4147   \glxtr@writefields
4148   \protected@write\@auxout{\glxtrresourceinit}{\string\glxtr@resource{#1}{#2}}%
4149   \let\@glxtr@org@see@noindex\@gls@see@noindex
4150   \let\@gls@see@noindex\relax
4151   \IfFileExists{#2.glstex}%
4152   {%

```

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

```

4153     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4154     \makeatletter
4155     \@input{#2.glstex}%
4156     \@bibgls@restreat
4157   }%
4158   {%
4159     \GlossariesExtraWarning{No file '#2.glstex'}%
4160   }%
4161   \let\@gls@see@noindex\@glxtr@org@see@noindex
4162 }
4163 \@onlypreamble\glxtrresourcefile

```

**trresourcecount**

```
4164 \newcount\glxtrresourcecount
```

**trLoadResources** Short cut that uses \glxtrresourcefile with \jobname as the mandatory argument.

```

4165 \newcommand*{\GlsXtrLoadResources}[1] [] {%
4166   \ifnum\glxtrresourcecount=0\relax
4167     \glxtrresourcefile[#1]{\jobname}%
4168   \else
4169     \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%

```



```

4170 \fi
4171 \advance\glxstrresourcecount by 1\relax
4172 }

glxstr@resource
4173 \newcommand*{\glxstr@resource}[2]{

\glxstr@fields
4174 \newcommand*{\glxstr@fields}[1]{

xtr@texencoding
4175 \newcommand*{\glxstr@texencoding}[1]{

\glxstr@langtag
4176 \newcommand*{\glxstr@langtag}[1]{

@pluralsuffixes
4177 \newcommand*{\glxstr@pluralsuffixes}[4]{

tr@shortcutsval
4178 \newcommand*{\glxstr@shortcutsval}[1]{

sxtr@linkprefix
4179 \newcommand*{\glxstr@linkprefix}[1]{

xtr@writefields This information only needs to be written once, so disable it after it's been used.
4180 \newcommand*{\glxstr@writefields}{%

4181 \protected@write\@auxout{}%
4182 {\string\providecommand*{\string\glxstr@fields}[1]{}}%
4183 \protected@write\@auxout{}%
4184 {\string\providecommand*{\string\glxstr@resource}[2]{}}%
4185 \protected@write\@auxout{}%
4186 {\string\providecommand*{\string\glxstr@pluralsuffixes}[4]{}}%
4187 \protected@write\@auxout{}%
4188 {\string\providecommand*{\string\glxstr@shortcutsval}[1]{}}%
4189 \protected@write\@auxout{}%
4190 {\string\providecommand*{\string\glxstr@linkprefix}[1]{}}%
4191 \protected@write\@auxout{}{\string\glxstr@fields{\@gls@keymap}}%

If any languages have been loaded, the language tag will be available in \CurrentTrackedLanguageTag
(provided by tracklang). For multilingual documents, the required locale will have to be indi-
cated in the sort key when using \glxstrresourcefile.

4192 \ifdef\CurrentTrackedLanguageTag
4193 {%
4194 \protected@write\@auxout{}{%
4195 \string\glxstr@langtag{\CurrentTrackedLanguageTag}}%
4196 }%

```

```

4197 {}%
4198 \protected@write\@auxout{}\string\glstr@pluralsuffixes
4199   {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
4200   {\glstrabbrvpluralsuffix}}%
4201 \ifdef\inputencodingname
4202 {}%
4203   \protected@write\@auxout{}\string\glstr@texencoding{\inputencodingname}}%
4204 }%
4205 {}%

```

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.)

```

4206   \ifpackageloaded{fontspec}%
4207   {\protected@write\@auxout{}\string\glstr@texencoding{utf8}}}%
4208   {}%
4209 }%
4210 \protected@write\@auxout{}\string\glstr@shortcutsval{\glstr@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.

```

4211 \AtBeginDocument
4212   {\protected@write\@auxout{}\string\glstr@linkprefix{\glolinkprefix}}}%
4213 \let\glstr@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.

```

4214 \ifglautomake
4215   \IfFileExists{\jobname.aux}%
4216   {\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`.

```

4217   \ifx\@gl@doautomake\@gl@doautomake@err
4218     \let\@gl@doautomake\relax
4219   \fi
4220 \fi
4221 }

```

do@automake@err

```

4222 \newcommand*\@gl@doautomake@err{%
4223   \PackageError{glossaries}{You must use
4224     \string\makeglossaries\space with automake=true}
4225   {%
4226     Either remove the automake=true setting or
4227     add \string\makeglossaries\space to your document preamble.%
4228   }%
4229 }

```

Allow locations specific to a particular counter to be recorded.

`\glxtr@record`

```

4230 \newcommand*{\glxtr@record}[5]{%

```

`r@counterrecord` Aux file command.

```

4231 \newcommand*{\glxtr@counterrecord}[3]{%
4232   \glxtrfieldlistgadd{#1}{record.#2}{#3}%
4233 }

```

`unterrecordhook` Hook used by `\@glxtr@dorecord`.

```

4234 \newcommand*{\@glxtr@counterrecordhook}{%

```

`trRecordCounter` Activate recording for a particular counter (identified in the argument).

```

4235 \newcommand*{\GlsXtrRecordCounter}[1]{%
4236   \@glxtr@recordcounter{#1}%
4237 }
4238 \onlypreamble\GlsXtrRecordCounter

```

`docounterrecord`

```

4239 \newcommand*{\@glxtr@docounterrecord}[1]{%
4240   \protected@write\@auxout{}{\string\glxtr@counterrecord
4241     {\@glslabel}{#1}{\csuse{the#1}}}%
4242 }

```

`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.

```

4243 \newcommand*{\glxtrglossentry}[1]{%
4244   \glxtrtitleorpdforheading
4245   {\@glxtrglossentry{#1}}%
4246   {\glsentryname{#1}}%
4247   {\glxtrheadname{#1}}%
4248 }

```

`lsxtrglossentry` Another test is needed in case `\@glxtrglossentry` has been written to the table of contents.

```

4249 \newrobustcmd*{\@glxtrglossentry}[1]{%
4250   \glxtrtitleorpdforheading
4251   {%
4252     \glsdofexists{#1}%
4253     {%
4254       \begingroup
4255         \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4256         \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4257         \ifglshasparent{#1}%

```

```

4258      {\glssubentryitem{#1}}%
4259      {\glentryitem{#1}}%
4260      \glstarget{#1}{\glossentryname{#1}}%
4261      \endgroup
4262  }%
4263 }%
4264 {\glentryname{#1}}%
4265 {\glxtrheadname{#1}}%
4266 }

```

`\printunsrtglossary` Similar to `\printnoidxglossary` but it displays all entries defined for the given glossary without sorting.

```

4267 \newcommand*{\printunsrtglossary}{%
4268   \@ifstar\s@printunsrtglossary\@printunsrtglossary
4269 }

```

`\printunsrtglossary` Unstarred version.

```

4270 \newcommand*{\@printunsrtglossary}[1][]{%
4271   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4272 }

```

`\printunsrtglossary` Starred version.

```

4273 \newcommand*{\s@printunsrtglossary}[2][]{%
4274   \begingroup
4275     #2%
4276     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4277   \endgroup
4278 }

```

`\printglossaries` Similar to `\printnoidxglossaries` but it displays all entries defined for the given glossary without sorting.

```

4279 \newcommand*{\printunsrtglossaries}{%
4280   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4281 }

```

`\@unsrt@glossary`

```

4282 \newcommand*{\@print@unsrt@glossary}{%
4283   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4284   \glossarypreamble
4285   check for empty list
4286   \glxtrifemptyglossary{\@glo@type}%
4287   {\%
4288     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4289   }%
4290   {\key@ifundefined{glossentry}{group}%
4291     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%

```

```

4292 {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4293 \def\@gls@currentlettergroup{}}%

```

A loop within the tabular-like styles can cause problems, so move the loop outside.

```

4294 \def\@glsxtr@doglossary{%
4295   \begin{theglossary}%
4296   \glossaryheader
4297   \glsresetentrylist
4298   }%
4299   \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4300   : \expandafter=\csname glo@list@\@glo@type\endcsname\do{%
4301   \ifdefempty{\glscurrententrylabel}
4302   }{}%
4303   {%

```

Provide a hook (for example to measure width).

```

4304   \let\glsxtr@process\@firstofone
4305   \let\printunsrtglossaryskipentry
4306   \@glsxtr@printunsrtglossaryskipentry
4307   \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

4308   \glsxtr@process
4309   {%
4310     \ifglshasparent{\glscurrententrylabel}{}%
4311     {%
4312       \@glsxtr@checkgroup\glscurrententrylabel
4313       \expandafter\appto\expandafter\@glsxtr@doglossary\expandafter
4314       {\@glsxtr@groupheading}%
4315     }%
4316     \eappto\@glsxtr@doglossary{%
4317       \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4318   }%
4319 }%
4320 }%
4321 \appto\@glsxtr@doglossary{\end{theglossary}}%
4322 \printunsrtglossarypredoglossary
4323 \@glsxtr@doglossary
4324 }%
4325 \glossarypostamble
4326 }

```

entryprocesshook

```

4327 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}

```

entryprocesshook

```

4328 \newcommand*{\printunsrtglossaryskipentry}{%
4329   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4330 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4331 }

```

entryprocesshook

```
4332 \newcommand*{\@glstr@printunsrtglossaryskipentry}{%
4333   \let\glstr@process\@gobble
4334 }
```

rypredoglossary

```
4335 \newcommand*{\printunsrtglossarypredoglossary}{{}}
```

lossary@handler

```
4336 \newcommand{\@printunsrtglossary@handler}[1]{%
4337   \xdef\glscurrententrylabel{#1}%
4338   \printunsrtglossaryhandler\glscurrententrylabel
4339 }
```

glossaryhandler

```
4340 \newcommand{\printunsrtglossaryhandler}[1]{%
4341   \glstrunsrtdo{#1}%
4342 }
```

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 \@gls@ifinlist which ensures the label and list are fully expanded.

```
4343 \newrobustcmd*{\glstriflabelinlist}[4]{%
4344   \protected@edef\@glstr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
4345   \@glstr@doiflabelinlist{#3}{#4}%
4346 }
```

srtglossaryunit

```
4347 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
4348   \s@printunsrtglossary[type=\gldefaulttype,#1]{%
4349     \printunsrtglossaryunitsetup{#2}%
4350   }%
4351 }
```

ossaryunitsetup

```
4352 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4353   \renewcommand{\printunsrtglossaryhandler}[1]{%
4354     \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4355     {\glstrunsrtdo{##1}}%
4356   }%
4357   }%
```

Only the target names should have the prefixes adjusted as \gls etc need the original \glolinkprefix. The \@gobble part discards \glolinkprefix.

```
4358   \ifcsundef{theH#1}%
4359   {%
4360     \renewcommand*{\@glstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4361   }%
```

```

4362 {%
4363   \renewcommand*{\@glxtrhypernameprefix}{record.#1.\cuse{theH#1}.\@gobble}%
4364 }%
4365 \renewcommand*\glossarysection[2][{}]{%
4366   \appto\glossarypostamble{\glspare\medskip\glspare}%
4367 }

```

unsrtglossaryunit

```

4368 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
4369   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4370     requires the record=only or record=alsoindex package option}{}%
4371 }

```

glxtr@getgrouptitle

```

4372 \newrobustcmd*{\@glxtr@unsrt@getgrouptitle}[2]{%
4373   \protected@edef\@glxtr@titlelabel{\glxtr@grouptitle@#1}%
4374   \@onelevel@sanitize\@glxtr@titlelabel
4375   \ifcsdef{\@glxtr@titlelabel}
4376     {\letcs{#2}{\@glxtr@titlelabel}}%
4377     {\def#2{#1}}%
4378 }

```

\glxtrunsrtdo Provide a user-level call to \@glxtr@noidx@do to make it easier to define a new handler.

```

4379 \newcommand{\glxtrunsrtdo}{\@glxtr@noidx@do}

```

glxtrgroupfield bib2gls provides a supplementary field labelled secondarygroup for secondary glossaries, so provide a way of switching to that field. (The group key still needs checking. There's no associated key with the internal field).

```

4380 \newcommand*\glxtrgroupfield{group}

```

The tabular-like glossary styles cause quite a problem with the iterative approach. In particular for the group skip. To compensate for this, the groups are now determined while \@glxtr@doglossary is being constructed rather than in the handler.

glxtr@checkgroup The argument is the entry's label. (This block of code was formerly in \@glxtr@noidx@do.) Now that this is no longer within a tabular environment, the global definitions aren't needed. The result is now stored in \@glxtr@groupheading, which will be empty if no heading is required.

```

4381 \newcommand*\@glxtr@checkgroup[1]{%
4382   \def\@glxtr@groupheading{}%
4383   \key@ifundefined{glossentry}{group}%
4384   {%
4385     \letcs{\@gls@sort}{glo@glsetoklabel{#1}@sort}%
4386     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4387   }%
4388   {%

```

```

4389 \protected@edef\@glo@thislettergrp{%
4390 \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
4391 }%
4392 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4393 {}%
4394 {%
4395 \ifdefempty{\@gls@currentlettergroup}{}%
4396 {\def\@glsxtr@groupheading{\gls@groupskip}}%
4397 \eappto\@glsxtr@groupheading{%
4398 \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
4399 }%
4400 }%
4401 \let\@gls@currentlettergroup\@glo@thislettergrp
4402 }

```

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.

```

4403 \newcommand{\@glsxtr@noidx@do}[1]{%
4404 \ifglsentryexists{#1}%
4405 {%
4406 \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4407 \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
4408 \ifgls@hasparent{#1}%
4409 {%
4410 \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
4411 \ifdefvoid{\@gls@location}%
4412 {%
4413 \ifdefvoid{\@gls@loclist}%
4414 {%
4415 \subglossentry{\gls@level}{#1}{}%
4416 }%
4417 {%
4418 \subglossentry{\gls@level}{#1}%
4419 {%
4420 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4421 }%
4422 }%
4423 }%
4424 {%
4425 \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4426 }%
4427 }%
4428 {%
4429 \ifdefvoid{\@gls@location}%
4430 {%
4431 \ifdefvoid{\@gls@loclist}%
4432 {%
4433 \glossentry{#1}{}%
4434 }%

```



```

4435      {%
4436      \glossentry{#1}%
4437      {%
4438      \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4439      }%
4440      }%
4441      }%
4442      {%
4443      \glossentry{#1}%
4444      {%
4445      \glossaryentrynumbers{\@gls@location}%
4446      }%
4447      }%
4448      }%
4449      }%
4450      {}%
4451      }

```

### 1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4452 \newcommand*{\glshex}{\string\u}
```

`\glsxtrresourceinit` Code used during the protected write operation.

```
4453 \newcommand*{\glsxtrresourceinit}{}
```

Provide a way to conveniently define commands that behaves like `\gls` with a label prefix.

It's possible that the user might want minor variations with the same prefix but different default options, so use a counter to provide unique inner commands.

`\glsxtrnewgls`

```
4454 \newcount\@glsxtrnewgls@inner
```

(The default options supplied in *<options>* below could possibly be used to form the inner control sequence name to help make it unique, but it might feasibly contain the value where the value might contain commands.)

`\@glsxtrnewgls`

`\glsxtrnewgls[<options>]{<prefix>}{<cs>}{<inner cs name>}`

```

4455 \newcommand*{\@glsxtrnewgls}[4]{%
4456   \ifdef{#3}%
4457   {%
4458     \PackageError{glossaries-extra}{Command \string#3\space already
4459     defined}{}%

```

```

4460 }%
4461 {%
4462   \ifcsdef{@#4like@#2}%
4463   {%
4464     \advance\@glxstrnewgls@inner by \@ne
4465     \def\@glxstrnewgls@innercsname{@#4like\number\@glxstrnewgls@inner @#2}%
4466   }%
4467   {\def\@glxstrnewgls@innercsname{@#4like@#2}}%
4468   \expandafter\newrobustcmd\expandafter*\expandafter
4469   #3\expandafter{\expandafter\@glshyp@opt\csname\@glxstrnewgls@innercsname\endcsname}%
4470   \ifstrempy{#1}%
4471   {%
4472     \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2][ ]{%
4473       \new@ifnextchar[%
4474         {\csname @#4@\endcsname{##1}{#2##2}}%
4475         {\csname @#4@\endcsname{##1}{#2##2} [ ]}%
4476       }%
4477     }%
4478     {%
4479       \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2][ ]{%
4480         \new@ifnextchar[%
4481           {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4482           {\csname @#4@\endcsname{#1,##1}{#2##2} [ ]}%
4483         }%
4484       }%
4485     }%
4486 }

```

`\glxstrnewgls` `\glxstrnewgls[<options>]{<prefix>}{<cs>}`

The first argument prepends to the options and the second argument is the prefix.

```

4487 \newrobustcmd*{\glxstrnewgls}[3][ ]{%
4488   \@glxstrnewgls{#1}{#2}{#3}{gls}%
4489 }

```

`\glxstrnewglslike` Provide a way to conveniently define commands that behave like `\gls`, `\glsp1`, `\Gls` and `\Glsp1` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```

4490 \newrobustcmd*{\glxstrnewglslike}[6][ ]{%
4491   \@glxstrnewgls{#1}{#2}{#3}{gls}%
4492   \@glxstrnewgls{#1}{#2}{#4}{glsp1}%
4493   \@glxstrnewgls{#1}{#2}{#5}{Gls}%
4494   \@glxstrnewgls{#1}{#2}{#6}{Glsp1}%
4495 }

```

`\glxstrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSp1` with a label

prefix. The first argument prepends to the options and the second argument is the prefix.

```
4496 \newrobustcmd*{\glxtrnewGLSlike}[4] [] {%
4497   \@glxtrnewgls{#1}{#2}{#3}{GLS}%
4498   \@glxtrnewgls{#1}{#2}{#4}{GLSpl}%
4499 }
```

\glxtrnewrgls As \glxtrnewgls but for \rgls.

```
4500 \newrobustcmd*{\glxtrnewrgls}[3] [] {%
4501   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4502 }
```

sxtrnewrglslike As \glxtrnewglslike but for \rgls etc.

```
4503 \newrobustcmd*{\glxtrnewrglslike}[6] [] {%
4504   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4505   \@glxtrnewgls{#1}{#2}{#4}{rglspl}%
4506   \@glxtrnewgls{#1}{#2}{#5}{rGls}%
4507   \@glxtrnewgls{#1}{#2}{#6}{rGlspl}%
4508 }
```

sxtrnewrGLSlike As \glxtrnewGLSlike but for \rGLS etc.

```
4509 \newrobustcmd*{\glxtrnewrGLSlike}[4] [] {%
4510   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4511   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4512 }
```

Provide easy access to record count fields.

totalRecordCount Access total record count. This is designed to be expandable. The argument is the label.

```
4513 \newcommand*{\GlsXtrTotalRecordCount}[1] {%
4514   \ifcsdef{glo@\glsetoklabel{#1}@recordcount}%
4515   {\csname glo@\glsetoklabel{#1}@recordcount\endcsname}%
4516   {0}%
4517 }
```

sXtrRecordCount Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
4518 \newcommand*{\GlsXtrRecordCount}[2] {%
4519   \ifcsdef{glo@\glsetoklabel{#1}@recordcount.#2}%
4520   {\csname glo@\glsetoklabel{#1}@recordcount.#2\endcsname}%
4521   {0}%
4522 }
```

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.

```
4523 \newcommand*{\GlsXtrLocationRecordCount}[3] {%
4524   \ifcsdef{glo@\glsetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
```

```

4525 {\csname glo@glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
4526 {0}%
4527 }

```

trdetoklocation

```

4528 \newcommand*{\glxtrdetoklocation}[1]{#1}

```

ablerecordcount

```

4529 \newcommand*{\glxtrenablerecordcount}{%
4530   \renewcommand*{\glx}{\rglx}%
4531   \renewcommand*{\Glx}{\rGlx}%
4532   \renewcommand*{\glxpl}{\rglxpl}%
4533   \renewcommand*{\Glxpl}{\rGlxpl}%
4534   \renewcommand*{\GLS}{\rGLS}%
4535   \renewcommand*{\GLSpl}{\rGLSpl}%
4536 }

```

ordtriggervalue The value used by the record trigger test. The argument is the entry's label.

```

4537 \newcommand*{\glxtrrecordtriggervalue}[1]{%
4538   \GlsXtrTotalRecordCount{#1}%
4539 }

```

dCountAttribute

```

4540 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4541   \@for\@glxtr@cat:=#1\do
4542   {%
4543     \ifdefempty{\@glxtr@cat}{}%
4544     {%
4545       \glxsetcategoryattribute{\@glxtr@cat}{recordcount}{#2}%
4546     }%
4547   }%
4548 }

```

rifrecordtrigger

$\backslash\text{glxtrifrecordtrigger}\{\langle\text{label}\rangle\}\{\langle\text{trigger format}\rangle\}\{\langle\text{normal}\rangle\}$
---

```

4549 \newcommand*{\glxtrifrecordtrigger}[3]{%
4550   \glxhasattribute{#1}{recordcount}%
4551   {%
4552     \ifnum\glxtrrecordtriggervalue{#1}>\glxgetattribute{#1}{recordcount}\relax
4553     #3%
4554   \else
4555     #2%
4556   \fi
4557 }%
4558 {#3}%
4559 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```
4560 \newcommand*{\@glsxtr@rglstrigger@record}[3]{%
4561   \edef\glslabel{\glsdetoklabel{#2}}%
4562   \let\@gls@link@label\glslabel
4563   \def\@glsxtr@thevalue{%
4564     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
4565     \def\@glsnumberformat{glstriggerrecordformat}%
4566     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4567     \edef\@gls@type{\csname glo@\glslabel @type\endcsname}%
4568     \def\@glsxtr@thevalue{%
4569       \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
4570       \glsxtrinitwrgloss
4571       \setkeys{glslink}{#1}%
4572       \glslinkpostsetkeys
4573       \ifdefempty{\@glsxtr@thevalue}%
4574       {%
4575         \@gls@saveentrycounter
4576       }%
4577       {%
4578         \let\theglsentrycounter\@glsxtr@thevalue
4579         \def\theglsentrycounter{\@glsxtr@theHvalue}%
4580       }%
4581       \ifglsxtrinitwrglossbefore
4582       \@do@wrglossary{#2}%
4583       \fi
4584       #3%
4585       \ifglsxtrinitwrglossbefore
4586       \else
4587       \@do@wrglossary{#2}%
4588       \fi
4589       \ifKV@glslink@local
4590       \glslocalunset{#2}%
4591       \else
4592       \glsunset{#2}%
4593       \fi
4594 }
```

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```
4595 \newcommand*{\glstriggerrecordformat}[1]{}
```

\rgls

```
4596 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}
```

\@rgls

```
4597 \newcommand*{\@rgls}[2][ ]{%
4598   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4599 }
```

```

\@rgls@
4600 \def\@rgls@#1#2[#3]{%
4601   \glxtrifrecordtrigger{#2}%
4602   {%
4603     \@glxtr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4604   }%
4605   {%
4606     \@rgls@{#1}{#2}[#3]%
4607   }%
4608 }%

\rglspl
4609 \newrobustcmd*{\rglspl}{\@rgls@hyp@opt\@rglspl}

\@rglspl
4610 \newcommand*{\@rglspl}[2][{}]{%
4611   \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
4612 }

\@rglspl@
4613 \def\@rglspl@#1#2[#3]{%
4614   \glxtrifrecordtrigger{#2}%
4615   {%
4616     \@glxtr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
4617   }%
4618   {%
4619     \@rglspl@{#1}{#2}[#3]%
4620   }%
4621 }%

\rgls
4622 \newrobustcmd*{\rgls}{\@rgls@hyp@opt\@rgls}

\@rgls
4623 \newcommand*{\@rgls}[2][{}]{%
4624   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4625 }

\@rgls@
4626 \def\@rgls@#1#2[#3]{%
4627   \glxtrifrecordtrigger{#2}%
4628   {%
4629     \@glxtr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4630   }%
4631   {%
4632     \@rgls@{#1}{#2}[#3]%
4633   }%
4634 }%

```

```

\rGlspl
4635 \newrobustcmd*{\rGlspl}{\@gls@hyp@opt\rGlspl}

\@rGlspl
4636 \newcommand*{\@rGlspl}[2][{}]{%
4637   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4638 }

\@rGlspl@
4639 \def\@rGlspl@#1#2[#3]{%
4640   \glsxtrifrecordtrigger{#2}%
4641   {%
4642     \@glsxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4643   }%
4644   {%
4645     \@Glspl@{#1}{#2}[#3]%
4646   }%
4647 }%

\rGLS
4648 \newrobustcmd*{\rGLS}{\@gls@hyp@opt\rGLS}

\@rGLS
4649 \newcommand*{\@rGLS}[2][{}]{%
4650   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}%
4651 }

\@rGLS@
4652 \def\@rGLS@#1#2[#3]{%
4653   \glsxtrifrecordtrigger{#2}%
4654   {%
4655     \@glsxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
4656   }%
4657   {%
4658     \@GLS@{#1}{#2}[#3]%
4659   }%
4660 }%

\rGLSpl
4661 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}

\@rGLSpl
4662 \newcommand*{\@rGLSpl}[2][{}]{%
4663   \new@ifnextchar[{\@rGLSpl@{#1}{#2}}{\@rGLSpl@{#1}{#2}[]}%
4664 }

```

```

\@rGLSpl@
4665 \def \@rGLSpl@#1#2[#3]{%
4666   \glxtrifrecordtrigger{#2}%
4667   {%
4668     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
4669   }%
4670   {%
4671     \@GLSpl@{#1}{#2}[#3]%
4672   }%
4673 }%

\rglsformat
4674 \newcommand*{\rglsformat}[2]{%
4675   \glsifregular{#1}
4676   {\glsentryfirst{#1}}%
4677   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4678 }

\rglsplformat
4679 \newcommand*{\rglsplformat}[2]{%
4680   \glsifregular{#1}
4681   {\glsentryfirstplural{#1}}%
4682   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4683 }

\rGlsformat
4684 \newcommand*{\rGlsformat}[2]{%
4685   \glsifregular{#1}
4686   {\Glsentryfirst{#1}}%
4687   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4688 }

\rGlsplformat
4689 \newcommand*{\rGlsplformat}[2]{%
4690   \glsifregular{#1}
4691   {\Glsentryfirstplural{#1}}%
4692   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4693 }

\rGLSformat
4694 \newcommand*{\rGLSformat}[2]{%
4695   \expandafter\mfirstucMakeUppercase\expandafter{\rglsformat{#1}{#2}}%
4696 }

\rGLSplformat
4697 \newcommand*{\rGLSplformat}[2]{%
4698   \expandafter\mfirstucMakeUppercase\expandafter{\rglsplformat{#1}{#2}}%
4699 }

```



## 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.

```
4700 \@ifpackageloaded{glossaries-accsupp}
4701 {
```

Define (or redefine) commands to use the accessibility information.

`\glsaccessname` Display the name value (no link and no check for existence).

```
4702 \newcommand*{\glsaccessname}[1]{%
4703 \glsnameaccessdisplay
4704 {%
4705 \glstryname{#1}%
4706 }%
4707 {#1}%
4708 }
```

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.

```
4709 \newcommand*{\Glsaccessname}[1]{%
4710 \glsnameaccessdisplay
4711 {%
4712 \Glsstryname{#1}%
4713 }%
4714 {#1}%
4715 }
```

`\GLSaccessname` Display the name value (no link and no check for existence) converted to upper case.

```
4716 \newcommand*{\GLSaccessname}[1]{%
4717 \glsnameaccessdisplay
4718 {%
4719 \mfirstucMakeUppercase{\glstryname{#1}}%
4720 }%
4721 {#1}%
4722 }
```

`\glsaccesstext` Display the text value (no link and no check for existence).

```
4723 \newcommand*{\glsaccesstext}[1]{%
4724 \glstextaccessdisplay
4725 {%
4726 \glstrytext{#1}%
4727 }%
4728 {#1}%
4729 }
```

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.

```

4730 \newcommand*{\Glsaccesstext}[1]{%
4731 \glstextaccessdisplay
4732 {%
4733 \Glsentrytext{#1}%
4734 }%
4735 {#1}%
4736 }

```

`\GLSaccesstext` Display the text value (no link and no check for existence) converted to upper case.

```

4737 \newcommand*{\GLSaccesstext}[1]{%
4738 \glstextaccessdisplay
4739 {%
4740 \mfirstucMakeUppercase{\Glsentrytext{#1}}%
4741 }%
4742 {#1}%
4743 }

```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

4744 \newcommand*{\glsaccessplural}[1]{%
4745 \glspluralaccessdisplay
4746 {%
4747 \Glsentryplural{#1}%
4748 }%
4749 {#1}%
4750 }

```

`GLSaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

4751 \newcommand*{\GLSaccessplural}[1]{%
4752 \glspluralaccessdisplay
4753 {%
4754 \Glsentryplural{#1}%
4755 }%
4756 {#1}%
4757 }

```

`GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```

4758 \newcommand*{\GLSaccessplural}[1]{%
4759 \glspluralaccessdisplay
4760 {%
4761 \mfirstucMakeUppercase{\Glsentryplural{#1}}%
4762 }%
4763 {#1}%
4764 }

```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

4765 \newcommand*{\glsaccessfirst}[1]{%
4766   \glsfirstaccessdisplay
4767   {%
4768     \glstryfirst{#1}%
4769   }%
4770   {#1}%
4771 }

```

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

4772 \newcommand*{\Glsaccessfirst}[1]{%
4773   \glsfirstaccessdisplay
4774   {%
4775     \Glsstryfirst{#1}%
4776   }%
4777   {#1}%
4778 }

```

`\GLSaccessfirst` Display the first value (no link and no check for existence) converted to upper case.

```

4779 \newcommand*{\GLSaccessfirst}[1]{%
4780   \glsfirstaccessdisplay
4781   {%
4782     \mfirstucMakeUppercase{\glstryfirst{#1}}%
4783   }%
4784   {#1}%
4785 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

4786 \newcommand*{\glsaccessfirstplural}[1]{%
4787   \glsfirstpluralaccessdisplay
4788   {%
4789     \glstryfirstplural{#1}%
4790   }%
4791   {#1}%
4792 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

4793 \newcommand*{\Glsaccessfirstplural}[1]{%
4794   \glsfirstpluralaccessdisplay
4795   {%
4796     \Glsstryfirstplural{#1}%
4797   }%
4798   {#1}%
4799 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) converted to upper case.

```

4800 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

4801   \glsfirstpluralaccessdisplay
4802   {%
4803       \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4804   }%
4805   {#1}%
4806   }

```

**glsaccesssymbol** Display the symbol value (no link and no check for existence).

```

4807   \newcommand*{\glsaccesssymbol}[1]{%
4808       \glssymbolaccessdisplay
4809       {%
4810           \glsentrysymbol{#1}%
4811       }%
4812       {#1}%
4813   }

```

**Glsaccesssymbol** Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4814   \newcommand*{\Glsaccesssymbol}[1]{%
4815       \glssymbolaccessdisplay
4816       {%
4817           \Glsentrysymbol{#1}%
4818       }%
4819       {#1}%
4820   }

```

**GLSaccesssymbol** Display the symbol value (no link and no check for existence) converted to upper case.

```

4821   \newcommand*{\GLSaccesssymbol}[1]{%
4822       \glssymbolaccessdisplay
4823       {%
4824           \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
4825       }%
4826       {#1}%
4827   }

```

**esssymbolplural** Display the symbolplural value (no link and no check for existence).

```

4828   \newcommand*{\glsaccesssymbolplural}[1]{%
4829       \glssymbolpluralaccessdisplay
4830       {%
4831           \glsentrysymbolplural{#1}%
4832       }%
4833       {#1}%
4834   }

```

**esssymbolplural** Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4835   \newcommand*{\Glsaccesssymbolplural}[1]{%
4836       \glssymbolpluralaccessdisplay

```

```

4837   {%
4838       \Glsentrysymbolplural{#1}%
4839   }%
4840   {#1}%
4841 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence) converted to upper case.

```

4842 \newcommand*{\GLSaccesssymbolplural}[1]{%
4843     \glssymbolpluralaccessdisplay
4844     {%
4845         \mfirstucMakeUppercase{\glsentrysymbolplural{#1}}%
4846     }%
4847     {#1}%
4848 }

```

`\glsaccessdesc` Display the desc value (no link and no check for existence).

```

4849 \newcommand*{\glsaccessdesc}[1]{%
4850     \glsdescriptionaccessdisplay
4851     {%
4852         \glsentrydesc{#1}%
4853     }%
4854     {#1}%
4855 }

```

`\Glsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

4856 \newcommand*{\Glsaccessdesc}[1]{%
4857     \glsdescriptionaccessdisplay
4858     {%
4859         \Glsentrydesc{#1}%
4860     }%
4861     {#1}%
4862 }

```

`\GLSaccessdesc` Display the desc value (no link and no check for existence) converted to upper case.

```

4863 \newcommand*{\GLSaccessdesc}[1]{%
4864     \glsdescriptionaccessdisplay
4865     {%
4866         \mfirstucMakeUppercase{\glsentrydesc{#1}}%
4867     }%
4868     {#1}%
4869 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence).

```

4870 \newcommand*{\glsaccessdescplural}[1]{%
4871     \glsdescriptionpluralaccessdisplay
4872     {%
4873         \glsentrydescplural{#1}%

```

```

4874 }%
4875 {#1}%
4876 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

```

4877 \newcommand*{\Glsaccessdescplural}[1]{%
4878   \glsdescriptionpluralaccessdisplay
4879   {%
4880     \Glsentrydescplural{#1}%
4881   }%
4882   {#1}%
4883 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

4884 \newcommand*{\GLSaccessdescplural}[1]{%
4885   \glsdescriptionpluralaccessdisplay
4886   {%
4887     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4888   }%
4889   {#1}%
4890 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

4891 \newcommand*{\glsaccessshort}[1]{%
4892   \glsshortaccessdisplay
4893   {%
4894     \glsentryshort{#1}%
4895   }%
4896   {#1}%
4897 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

4898 \newcommand*{\Glsaccessshort}[1]{%
4899   \glsshortaccessdisplay
4900   {%
4901     \Glsentryshort{#1}%
4902   }%
4903   {#1}%
4904 }

```

`\GLSAccessshort` Display the short value (no link and no check for existence) converted to upper case.

```

4905 \newcommand*{\GLSAccessshort}[1]{%
4906   \glsshortaccessdisplay
4907   {%
4908     \mfirstucMakeUppercase{\glsentryshort{#1}}%
4909   }%

```

```

4910     {#1}%
4911 }

```

**lsaccessshortpl** Display the short plural form (no link and no check for existence).

```

4912 \newcommand*{\lsaccessshortpl}[1]{%
4913   \glshortpluralaccessdisplay
4914   {%
4915     \glentryshortpl{#1}%
4916   }%
4917   {#1}%
4918 }

```

**Lsaccessshortpl** Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

4919 \newcommand*{\Lsaccessshortpl}[1]{%
4920   \glshortpluralaccessdisplay
4921   {%
4922     \Glentryshortpl{#1}%
4923   }%
4924   {#1}%
4925 }

```

**LSaccessshortpl** Display the shortplural value (no link and no check for existence) converted to upper case.

```

4926 \newcommand*{\LSaccessshortpl}[1]{%
4927   \glshortpluralaccessdisplay
4928   {%
4929     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
4930   }%
4931   {#1}%
4932 }

```

**\glsaccesslong** Display the long form (no link and no check for existence).

```

4933 \newcommand*{\glsaccesslong}[1]{%
4934   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
4935 }

```

**\Glsaccesslong** Display the long form (no link and no check for existence).

```

4936
4937 \newcommand*{\Glsaccesslong}[1]{%
4938   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
4939 }

```

**\GLSaccesslong** Display the long value (no link and no check for existence) converted to upper case.

```

4940 \newcommand*{\GLSaccesslong}[1]{%
4941   \glslongaccessdisplay
4942   {%
4943     \mfirstucMakeUppercase{\glentrylong{#1}}%
4944   }%

```

```

4945     {#1}%
4946 }

```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

4947 \newcommand*{\glsaccesslongpl}[1]{%
4948     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
4949 }

```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

4950
4951 \newcommand*{\Glsaccesslongpl}[1]{%
4952     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
4953 }

```

`\GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```

4954 \newcommand*{\GLSaccesslongpl}[1]{%
4955     \glslongpluralaccessdisplay
4956     {%
4957         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
4958     }%
4959     {#1}%
4960 }

```

End of if part

```

4961 }
4962 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

`\glsaccessname` Display the name value (no link and no check for existence).

```

4963 \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.

```

4964 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

`\GLSaccessname` Display the name value (no link and no check for existence). converted to upper case.

```

4965 \newcommand*{\GLSaccessname}[1]{%
4966     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

`\glsaccesstext` Display the text value (no link and no check for existence).

```

4967 \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.

```

4968 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```



`\GLSaccessstext` Display the text value (no link and no check for existence). converted to upper case.

```

4969 \newcommand*{\GLSaccessstext}[1]{%
4970 \protect\mfirstucMakeUppercase{\glstentrytext{#1}}}
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

4971 \newcommand*{\glsaccessplural}[1]{\glstentryplural{#1}}
```

`Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

4972 \newcommand*{\Glsaccessplural}[1]{\Glstentryplural{#1}}
```

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.

```

4973 \newcommand*{\GLSaccessplural}[1]{%
4974 \protect\mfirstucMakeUppercase{\glstentryplural{#1}}}
```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

4975 \newcommand*{\glsaccessfirst}[1]{\glstentryfirst{#1}}
```

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

4976 \newcommand*{\Glsaccessfirst}[1]{\Glstentryfirst{#1}}
```

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.

```

4977 \newcommand*{\GLSaccessfirst}[1]{%
4978 \protect\mfirstucMakeUppercase{\glstentryfirst{#1}}}
```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

4979 \newcommand*{\glsaccessfirstplural}[1]{\glstentryfirstplural{#1}}
```

`Glsaccessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

4980 \newcommand*{\Glsaccessfirstplural}[1]{\Glstentryfirstplural{#1}}
```

`GLSaccessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.

```

4981 \newcommand*{\GLSaccessfirstplural}[1]{%
4982 \protect\mfirstucMakeUppercase{\glstentryfirstplural{#1}}}
```

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

```

4983 \newcommand*{\glsaccesssymbol}[1]{\glstentrysymbol{#1}}
```

`Glsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4984 \newcommand*{\Glsaccesssymbol}[1]{\Glstentrysymbol{#1}}
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

```

4985 \newcommand*{\GLSaccesssymbol}[1]{%
4986 \protect\mfirstucMakeUppercase{\glstentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).  
4987 \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.  
4988 \newcommand\*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.  
4989 \newcommand\*{\GLSaccesssymbolplural}[1]{%  
4990 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).  
4991 \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.  
4992 \newcommand\*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.  
4993 \newcommand\*{\GLSaccessdesc}[1]{%  
4994 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).  
4995 \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.  
4996 \newcommand\*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.  
4997 \newcommand\*{\GLSaccessdescplural}[1]{%  
4998 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).  
4999 \newcommand\*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).  
5000 \newcommand\*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.  
5001 \newcommand\*{\GLSaccessshort}[1]{%  
5002 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).  
5003 \newcommand\*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

`\lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5004 \newcommand*{\GLsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

`\LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.

```
5005 \newcommand*{\GLSaccessshortpl}[1]{%
5006 \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5007 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5008 \newcommand*{\GLsaccesslong}[1]{\Glsentrylong{#1}}
```

`\GLSaccesslong` Display the long value (no link and no check for existence). converted to upper case.

```
5009 \newcommand*{\GLSaccesslong}[1]{%
5010 \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}
```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5011 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5012 \newcommand*{\GLsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`\GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.

```
5013 \newcommand*{\GLSaccesslongpl}[1]{%
5014 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

End of else part

```
5015 }
```

## 1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5016 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5017 \newcommand{\glsifcategory}[4]{%
5018 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5019 }
```

Categories can have attributes.

categoryattribute

```
\glissetcategoryattribute{<category>}{<attribute-label>}{<value>}
```

Set (or override if already set) an attribute for the given category.

```
5020 \newcommand*{\glissetcategoryattribute}[3]{%
5021   \csdef{@glsxtr@categoryattr@@#1@#2}{#3}%
5022 }
```

categoryattribute

```
\glsggetcategoryattribute{<category>}{<attribute-label>}
```

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5023 \newcommand*{\glsggetcategoryattribute}[2]{%
5024   \csuse{@glsxtr@categoryattr@@#1@#2}%
5025 }
```

categoryattribute

```
\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}
```

Tests if the category has the given attribute set.

```
5026 \newcommand*{\glshascategoryattribute}[4]{%
5027   \ifcsvoid{@glsxtr@categoryattr@@#1@#2}{#4}{#3}%
5028 }
```

\glissetattribute

```
\glissetattribute{<entry label>}{<attribute-label>}{<value>}
```

Short cut where the category label is obtained from the entry information.

```
5029 \newcommand*{\glissetattribute}[3]{%
5030   \glissetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5031 }
```

\glsggetattribute

```
\glsggetattribute{<entry label>}{<attribute-label>}
```

Short cut where the category label is obtained from the entry information.

```
5032 \newcommand*{\glsggetattribute}[2]{%
5033   \glsggetcategoryattribute{\glscategory{#1}}{#2}%
5034 }
```

\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.

```
5035 \newcommand*{\glshasattribute}[4]{%
5036   \ifglentryexists{#1}%
5037   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5038   {#4}%
5039 }
```

categoryattribute

```
\glcifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true
part>}{<false part>}
```

True if category has the attribute with the given value.

```
5040 \newcommand{\glcifcategoryattribute}[5]{%
5041   \ifcsundef{@glxtr@categoryattr@#1@#2}%
5042   {#5}%
5043   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5044 }
```

\glcifattribute

```
\glcifattribute{<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.

```
5045 \newcommand{\glcifattribute}[5]{%
5046   \ifglentryexists{#1}%
5047   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5048   {#5}%
5049 }
```

Set attributes for the default general category:

```
5050 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5051 \glsetcategoryattribute{acronym}{regular}{true}
```

regularcategory

Convenient shortcut to create add the regular attribute.

```
5052 \newcommand*{\glsetregularcategory}[1]{%
5053   \glsetcategoryattribute{#1}{regular}{true}%
5054 }
```

fregularcategory

```
\glsifregularcategory{<category>}{<true part>}{<false part>}
```

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5055 \newcommand{\glsifregularcategory}[3]{%  
5056   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%  
5057 }
```

tregularcategory

```
\glsifnotregularcategory{<category>}{<true part>}{<false part>}
```

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5058 \newcommand{\glsifnotregularcategory}[3]{%  
5059   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%  
5060 }
```

\glsifregular

```
\glsifregular{<entry label>}{<true part>}{<false part>}
```

Short cut to determine if an entry has a regular attribute set to true.

```
5061 \newcommand{\glsifregular}[3]{%  
5062   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%  
5063 }
```

\glsifnotregular

```
\glsifnotregular{<entry label>}{<true part>}{<false part>}
```

Short cut to determine if an entry has a regular attribute set to false.

```
5064 \newcommand{\glsifnotregular}[3]{%  
5065   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%  
5066 }
```

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.

```
5067 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```

```

5068 \forallglossaries[#1]{#3}%
5069 {%
5070     \forallsentries[#3]{#4}%
5071     {%
5072         \glsifcategory{#4}{#2}{#5}{}%
5073     }%
5074 }%
5075 }

```

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.

```

5076 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5077     \forallglossaries[#1]{#4}%
5078     {%
5079         \forallsentries[#4]{#5}%
5080         {%
5081             \glsifattribute{#5}{#2}{#3}{#6}{}%
5082         }%
5083     }%
5084 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```

5085 \ifdef\newterm
5086 {%

```

`\newterm`

```

5087     \renewcommand*{\newterm}[2][ ]{%
5088         \newglossaryentry{#2}%
5089         {type=index,category=index,name={#2},%
5090          description={\glstrpostdescription\nopostdesc},#1}%
5091     }

```

Indexed terms are regular by default.

```

5092     \glssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5093     \newcommand*{\glstrpostdescindex}{}

5094 }
5095 {}

```

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.

```
5096 \ifdef\printsymbols
5097 {%
```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```
5098 \newcommand*{\glsxtrnewsymbol}[3] [] {%
5099 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5100 }
```

Symbols are regular by default.

```
5101 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
5102 \newcommand*{\glsxtrpostdescsymbol}{}

5103 }
5104 {}
```

Similar for the numbers option.

```
5105 \ifdef\printnumbers
5106 {%
```

`glsxtrnewnumber`

```
5107 \ifdef\printnumbers
5108 \newcommand*{\glsxtrnewnumber}[3] [] {%
5109 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5110 }
```

Numbers are regular by default.

```
5111 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
5112 \newcommand*{\glsxtrpostdescnumber}{}

5113 }
5114 {}
```

`sxtrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
5115 \newcommand*{\glsxtrsetcategory}[2] {%
5116 \@for\@glsxtr@label:=#1\do
5117 {%
5118 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
5119 }%
5120 }
```



`\categoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```

5121 \newcommand*{\glxtrsetcategoryforall}[2]{%
5122   \forallglossaries[#1]{\@glxtr@type}{%
5123     \forallglsentries[\@glxtr@type]{\@glxtr@label}%
5124     {%
5125       \glsfieldxdef{\@glxtr@label}{category}{#2}%
5126     }%
5127   }%
5128 }

```

`\trfieldtitlecase` `\glxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```

5129 \newcommand*{\glxtrfieldtitlecase}[2]{%
5130   \expandafter\glxtrfieldtitlecasesecs\expandafter
5131   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
5132 }

```

`\fieldtitlecasesecs` The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

5133 \newcommand*{\glxtrfieldtitlecasesecs}[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.

```

5134 \@ifpackageloaded{glossaries-accsupp}
5135 {
5136   \renewcommand*{\glossentrydesc}[1]{%
5137     \glsdoifexistsorwarn{#1}%
5138     {%
5139       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

5140     \glshasattribute{#1}{glossdescfont}%
5141     {%
5142       \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5143       \ifcsdef{\@glxtr@attrval}%
5144       {%
5145         \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5146       }%
5147       {%
5148         \GlossariesExtraWarning{Unknown control sequence name
5149           ‘\@glxtr@attrval’ supplied in glossdescfont attribute

```

```

5150         for entry '#1'. Ignoring}%
5151         \let\@glxtr@glossdescfont\@firstofone
5152     }%
5153 }%
5154 {\let\@glxtr@glossdescfont\@firstofone}%
5155 \glsifattribute{#1}{glossdesc}{firstuc}%
5156 {%
5157     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5158 }%
5159 {%
5160     \glsifattribute{#1}{glossdesc}{title}%
5161     {%
5162         \@glxtr@do@titlecaps@warn
5163         \glsdescriptionaccessdisplay
5164         {%
5165             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5166         }%
5167         {#1}%
5168     }%
5169     {%
5170         \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5171     }%
5172 }%
5173 }%
5174 }
5175 }
5176 {
5177 \renewcommand*{\glossentrydesc}[1]{%
5178     \glsdoifexistsorwarn{#1}%
5179     {%
5180         \glssetabbrvfmt{\glscategory{#1}}%
5181         \glsattribute{#1}{glossdescfont}%
5182         {%
5183             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5184             \ifcsdef{\@glxtr@attrval}%
5185             {%
5186                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5187             }%
5188             {%
5189                 \GlossariesExtraWarning{Unknown control sequence name
5190                     '\@glxtr@attrval' supplied in glossdescfont attribute
5191                     for entry '#1'. Ignoring}%
5192                 \let\@glxtr@glossdescfont\@firstofone
5193             }%
5194         }%
5195         {\let\@glxtr@glossdescfont\@firstofone}%
5196         \glsifattribute{#1}{glossdesc}{firstuc}%
5197         {%
5198             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```

```

5199     }%
5200     {%
5201         \glsifattribute{#1}{glossdesc}{title}%
5202         {%
5203             \@glsxtr@do@titlecaps@warn
5204             \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5205         }%
5206         {%
5207             \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
5208         }%
5209     }%
5210 }%
5211 }
5212 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5213 \ifpackageloaded{glossaries-accsupp}
5214 {
5215     \renewcommand*{\glossentryname}[1]{%
5216         \@glsdoifexistsorwarn{#1}%
5217         {%
5218             \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

5219     \glsifattribute{#1}{glossnamefont}%
5220     {%
5221         \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5222         \ifcsdef{\@glsxtr@attrval}%
5223         {%
5224             \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5225         }%
5226         {%
5227             \GlossariesExtraWarning{Unknown control sequence name
5228                 ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
5229                 for entry ‘#1’. Reverting to default \string\glsnamefont}%
5230             \let\@glsxtr@glossnamefont\glsnamefont
5231         }%
5232     }%
5233     {\let\@glsxtr@glossnamefont\glsnamefont}%
5234     \glsifattribute{#1}{glossname}{firstuc}%
5235     {%
5236         \glsnameaccessdisplay
5237         {%
5238             \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5239         }%
5240         {#1}%
5241     }%
5242     {%
5243         \glsifattribute{#1}{glossname}{title}%

```

```

5244      {%
5245      \@glstr@do@titlecaps@warn
5246      \glsnameaccessdisplay
5247      {%
5248      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
5249      }%
5250      {#1}%
5251      }%
5252      {%
5253      \glsifattribute{#1}{glossname}{uc}%
5254      {%
5255      \glsnameaccessdisplay
5256      {%

```

Hide the label from the upper-casing command.

```

5257      \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
5258      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5259      }%
5260      {#1}%
5261      }%
5262      {%
5263      \letcs{\glo@name}{glo@\glsetoklabel{#1}@name}%
5264      \glsnameaccessdisplay
5265      {%
5266      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5267      }%
5268      {#1}%
5269      }%
5270      }%
5271      }%

```

Do post-name hook:

```

5272      \glstrpostnamehook{#1}%
5273      }%
5274  }
5275 }
5276 {
5277 \renewcommand*{\glossentryname}[1]{%
5278   \@glsdofexistsorwarn{#1}%
5279   {%
5280     \glsetabbrvfmt{\glscategory{#1}}%
5281     \glshasattribute{#1}{glossnamefont}%
5282     {%
5283       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5284       \ifcsdef{\@glstr@attrval}%
5285       {%
5286         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5287       }%
5288       {%
5289         \GlossariesExtraWarning{Unknown control sequence name

```

```

5290         ‘\@glstr@attrval’ supplied in glossnamefont attribute
5291         for entry ‘#1’. Reverting to default \string\glstrfont}%
5292         \let\@glstr@glossnamefont\glstrfont
5293     }%
5294 }%
5295 {\let\@glstr@glossnamefont\glstrfont}%
5296 \glsifattribute{#1}{glossname}{firstuc}%
5297 {%
5298     \@glstr@glossnamefont{\Glsentryname{#1}}%
5299 }%
5300 {%
5301     \glsifattribute{#1}{glossname}{title}%
5302     {%
5303         \@glstr@do@titlecaps@warn
5304         \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
5305     }%
5306     {%
5307         \glsifattribute{#1}{glossname}{uc}%
5308         {%

```

Hide the label from the upper-casing command.

```

5309         \letcs{\glo@name}{glo@glstrdetoklabel{#1}@name}%
5310         \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5311     }%
5312     {%

```

This little trick is used by glossaries to allow the user to redefine \glstrfont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

5313         \letcs{\glo@name}{glo@glstrdetoklabel{#1}@name}%
5314         \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5315     }%
5316 }%
5317 }%

```

Do post-name hook.

```

5318     \glstrpostnamehook{#1}%
5319 }%
5320 }
5321 }

```

**\Glossentryname** Redefine to set the abbreviation format and accessibility support.

```

5322 \ifpackageloaded{glossaries-accsupp}
5323 {
5324     \renewcommand*{\Glossentryname}[1]{%
5325         \@glstrdoifexistsorwarn{#1}%
5326         {%
5327             \glstrsetabbrvfmt{\glstrcategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5328         \glshasattribute{#1}{glossnamefont}%
5329         {%

```

```

5330     \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5331     \ifcsdef{\@glxtr@attrval}%
5332     {%
5333         \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5334     }%
5335     {%
5336         \GlossariesExtraWarning{Unknown control sequence name
5337             '\@glxtr@attrval' supplied in glossnamefont attribute
5338             for entry '#1'. Reverting to default \string\glsnamefont}%
5339         \let\@glxtr@glossnamefont\glsnamefont
5340     }%
5341 }%
5342 {\let\@glxtr@glossnamefont\glsnamefont}%
5343 \glsnameaccessdisplay
5344 {%
5345     \@glxtr@glossnamefont{\Glsentryname{#1}}%
5346 }%
5347 {#1}%

```

Do post-name hook:

```

5348     \glxtrpostnamehook{#1}%
5349 }%
5350 }
5351 }
5352 {
5353     \renewcommand*{\Glossentryname}[1]{%
5354         \@glsdoifexistsorwarn{#1}%
5355     {%
5356         \glsssetabbrvfmt{\glscategory{#1}}%
5357         \glshasattribute{#1}{glossnamefont}%
5358     {%
5359         \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5360         \ifcsdef{\@glxtr@attrval}%
5361         {%
5362             \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5363         }%
5364         {%
5365             \GlossariesExtraWarning{Unknown control sequence name
5366                 '\@glxtr@attrval' supplied in glossnamefont attribute
5367                 for entry '#1'. Reverting to default \string\glsnamefont}%
5368             \let\@glxtr@glossnamefont\glsnamefont
5369         }%
5370     }%
5371     {\let\@glxtr@glossnamefont\glsnamefont}%
5372     \@glxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5373     \glxtrpostnamehook{#1}%
5374 }%
5375 }

```

5376 }

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`glstrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```
5377 \newcommand*{\glstrpostnamehook}[1]{%
5378   \let\@glstrnumberformat\@glstr@defaultnumberformat
5379   \glstrdoautoindexname{#1}{indexname}%
```

Allow categories to hook in here.

```
5380   \csuse{glstrpostname\glscategory{#1}}%
5381 }
```

`glstrformat@override` Determines if the format key should override the indexing attribute value.

```
5382 \newif\if@glstr@format@override
5383 \@glstr@format@overridefalse
```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`glstrFormatOverride`

```
5384 \@ifpackageloaded{hyperref}
5385 {
```

If `hyperref`'s `hyperindex` option is on, then `hyperref` will automatically add `\hyperpage`, so don't add it.

```
5386   \ifHy@hyperindex
5387     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5388       \@glstr@format@override true
5389       \appto\theindex{\let\glshypernumber\@firstofone}%
5390     }
5391   \else
5392     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5393       \@glstr@format@override true
5394       \appto\theindex{\let\glshypernumber\hyperpage}%
5395     }
5396   \fi
5397 }
5398 {
5399   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5400     \@glstr@format@override true
5401   }
5402 }
5403 \@onlypreamble\GlsXtrEnableIndexFormatOverride
```

`glstrdoautoindexname`

```
5404 \newcommand*{\glstrdoautoindexname}[2]{%
5405   \glshasattribute{#1}{#2}%
5406   {%
```

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.

```
5407 \@glxtr@autoindex@setname{#1}%
```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```
5408 \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{#2}}%
5409 \if@glxtr@format@override

5410 \ifx\@glsnumberformat\@glxtr@defaultnumberformat
5411 \else
5412 \let\@glxtr@attrval\@glsnumberformat
5413 \fi
5414 \fi
5415 \ifdefstring{\@glxtr@attrval}{true}%
5416 {}%
5417 {\eappto\@glo@name{\@glxtr@autoindex@encap\@glxtr@attrval}}%
5418 \expandafter\glxtrautoindex\expandafter{\@glo@name}%
5419 }%
5420 {}%
5421 }
```

glxtrautoindex

```
5422 \newcommand*{\glxtrautoindex}{\index}
```

toindex@setname Assign \@glo@name for use with indexname attribute.

```
5423 \newcommand*{\@glxtr@autoindex@setname}[1]{%
5424 \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
5425 \glxtrautoindexassignsort{\@glo@sort}{#1}%
5426 \@gls@checkmkidxchars\@glo@sort
5427 \@glxtr@autoindex@doextra@esc\@glo@sort
5428 \epreto\@glo@name{\@glo@sort\@glxtr@autoindex@at}%
5429 }
```

rautoindexentry Command used for the actual part when auto-indexing.

```
5430 \newcommand*{\glxtrautoindexentry}[1]{\string\glstryname{#1}}
```

trautoindexsort Used to assign the sort value when auto-indexing.

```
5431 \newcommand*{\glxtrautoindexassignsort}[2]{%
5432 \glsletentryfield{#1}{#2}{sort}%
5433 }
```

dex@doextra@esc

```
5434 \newcommand*{\@glxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5435 \ifx\@glxtr@autoindex@esc\@gls@quotechar
5436 \else
5437 \def\@gls@checkedmkidx{}%
5438 \edef\@@glxtr@checkspch{%
```



```

5439 \noexpand\@glstr@autoindex@escquote\expandonce{#1}%
5440 \noexpand\@empty\@glstr@autoindex@esc\noexpand\@nnil
5441 \@glstr@autoindex@esc\noexpand\@empty\noexpand\@glstr@endescspch}%
5442 \@@glstr@checkspch
5443 \let#1\@glstr@checkedmkidx\relax
5444 \fi

```

Escape actual character unless it has already been escaped.

```

5445 \ifx\@glstr@autoindex@at\@glstr@actualchar
5446 \else
5447 \def\@glstr@checkedmkidx{}%
5448 \edef\@@glstr@checkspch{%
5449 \noexpand\@glstr@autoindex@escat\expandonce{#1}%
5450 \noexpand\@empty\@glstr@autoindex@at\noexpand\@nnil
5451 \@glstr@autoindex@at\noexpand\@empty\noexpand\@glstr@endescspch}%
5452 \@@glstr@checkspch
5453 \let#1\@glstr@checkedmkidx\relax
5454 \fi

```

Escape level character unless it has already been escaped.

```

5455 \ifx\@glstr@autoindex@level\@glstr@levelchar
5456 \else
5457 \def\@glstr@checkedmkidx{}%
5458 \edef\@@glstr@checkspch{%
5459 \noexpand\@glstr@autoindex@esclevel\expandonce{#1}%
5460 \noexpand\@empty\@glstr@autoindex@level\noexpand\@nnil
5461 \@glstr@autoindex@level\noexpand\@empty\noexpand\@glstr@endescspch}%
5462 \@@glstr@checkspch
5463 \let#1\@glstr@checkedmkidx\relax
5464 \fi

```

Escape encap character unless it has already been escaped.

```

5465 \ifx\@glstr@autoindex@encap\@glstr@encapchar
5466 \else
5467 \def\@glstr@checkedmkidx{}%
5468 \edef\@@glstr@checkspch{%
5469 \noexpand\@glstr@autoindex@escencap\expandonce{#1}%
5470 \noexpand\@empty\@glstr@autoindex@encap\noexpand\@nnil
5471 \@glstr@autoindex@encap\noexpand\@empty\noexpand\@glstr@endescspch}%
5472 \@@glstr@checkspch
5473 \let#1\@glstr@checkedmkidx\relax
5474 \fi
5475 }

```

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.

```

5476 \newcommand*{\@glstr@autoindex@at}{ }

```

trSetActualChar Set the actual character.

```
5477 \newcommand*{\GlsXtrSetActualChar}[1]{%
5478   \gdef\@glstr@autoindex@at{#1}%
5479   \def\@glstr@autoindex@escat##1##2##3\@glstr@endescspch{%
5480     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escat}{##1}{##2}{##3}%
5481   }%
5482 }
5483 \@onlypreamble\GlsXtrSetActualChar
5484 \makeatother
5485 \GlsXtrSetActualChar{@}
5486 \makeatletter
```

autoindex@encap Encap character for use with \index.

```
5487 \newcommand*{\@glstr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
5488 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5489   \gdef\@glstr@autoindex@encap{#1}%
5490   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
5491     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
5492   }%
5493 }
5494 \GlsXtrSetEncapChar{||}
5495 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
5496 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
5497 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5498   \gdef\@glstr@autoindex@level{#1}%
5499   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
5500     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
5501   }%
5502 }
5503 \GlsXtrSetLevelChar{!}
5504 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
5505 \newcommand*{\@glstr@autoindex@esc}{""}
```

lsXtrSetEscChar Set the escape character.

```
5506 \newcommand*{\GlsXtrSetEscChar}[1]{%
5507   \gdef\@glstr@autoindex@esc{#1}%
5508   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
5509     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
5510   }%
5511 }
```

```

5512 \GlsXtrSetEscChar{}
5513 \@onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```

5514 \ifdef\actualchar
5515 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5516 {}

```

Quote character \quotechar:

```

5517 \ifdef\quotechar
5518 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5519 {}

```

Level character \levelchar:

```

5520 \ifdef\levelchar
5521 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5522 {}

```

Encap character \encapchar:

```

5523 \ifdef\encapchar
5524 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5525 {}

```

leto@endescspch

```

5526 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}

```

toindex@esc@spch

```

\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}

```

```

5527 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
5528   \@glstmpb=\expandafter{\@glstmpb\the\@glstmpb}%
5529   \toks@={#3}%
5530   \ifx\@nnil#3\relax
5531     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
5532   \else
5533     \ifx\@nnil#4\relax
5534       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}%
5535       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5536         #4#5\@glxtr@endescspch}%
5537     \else
5538       \edef\@glstmpb{\the\@glstmpb\the\@glstmpb}
5539       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5540         #4#5\@glxtr@endescspch}%
5541     \fi
5542   \fi
5543   \@glxtr@checkspch
5544 }

```

`\Glossentrydesc` Redefine to set the abbreviation format and accessibility support.

```
5545 \renewcommand*{\Glossentrydesc}[1]{%
5546   \glsdoifexistsorwarn{#1}%
5547   {%
5548     \glssetabbrvfmt{\glscategory{#1}}%
5549     \Glsaccessdesc{#1}%
5550   }%
5551 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5552 \renewcommand*{\glossentrysymbol}[1]{%
5553   \glsdoifexistsorwarn{#1}%
5554   {%
5555     \glssetabbrvfmt{\glscategory{#1}}%
5556     \glsaccesssymbol{#1}%
5557   }%
5558 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5559 \renewcommand*{\Glossentrysymbol}[1]{%
5560   \glsdoifexistsorwarn{#1}%
5561   {%
5562     \glssetabbrvfmt{\glscategory{#1}}%
5563     \Glsaccesssymbol{#1}%
5564   }%
5565 }
```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\enableInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
5566 \newcommand*{\GlsXtrEnableInitialTagging}{%
5567   \@ifstar\s@glsxtr@enabletagging\@glsxtr@enabletagging
5568 }
5569 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
5570 \newcommand*{\s@glsxtr@enabletagging}[2]{%
5571   \undef#2%
5572   \@glsxtr@enabletagging{#1}{#2}%
5573 }
```

`\r@enabletagging` Internal command.

```
5574 \newcommand*{\@glsxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
5575   \@for\@glsxtr@cat:=#1\do
5576   {%
```

```

5577 \ifdefempty\@glsxtr@cat
5578 {}%
5579 {\glssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
5580 }%
5581 \newrobustcmd*#2[1]{##1}%
5582 \def\@glsxtr@taggingcs{#2}%
5583 \renewcommand*\@glsxtr@activate@initialtagging{%
5584 \let#2\@glsxtr@tag
5585 }%
5586 \ifundef\@gls@preglossaryhook
5587 {\GlossariesExtraWarning{Initial tagging requires at least
5588 glossaries.sty v4.19 to work correctly}}%
5589 {}%
5590 }

```

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

```

5591 \ifundef\mfu@checkword@do
5592 {
5593 \newcommand*{\mfu@checkword@do}[1]{%
5594 \ifdefstring{\mfu@checkword@arg}{#1}%
5595 {%
5596 \let\@mfu@domakefirstuc\@firstofone
5597 \listbreak
5598 }%
5599 }%
5600 }

```

\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.

```

5601 \ifundef\mfu@checkword
5602 {
5603 \newcommand{\@glsxtr@do@titlecaps@warn}{%
5604 \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5605 support not available}%

```

One warning should suffice.

```

5606 \let\@glsxtr@do@titlecaps@warn\relax
5607 }
5608 }
5609 {
5610 \renewcommand*{\mfu@checkword}[1]{%
5611 \def\mfu@checkword@arg{#1}%
5612 \let\@mfu@domakefirstuc\makefirstuc
5613 \forlistloop\mfu@checkword@do\@mfu@nocaplist
5614 }
5615 }
5616 }

```

```

5617 {}% no patch required

@titlecaps@warn Do warning if title case not supported.
5618 \newcommand*{\@glxtr@do@titlecaps@warn}{}

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.
5619 \newcommand*{\@glxtr@activate@initialtagging}{}

\@glxtr@tag Definition of tagging command when used in glossary.
5620 \newrobustcmd*{\@glxtr@tag}[1]{%
5621   \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%
5622   {\glxtrtagfont{#1}}{#1}%
5623 }

\glxtrtagfont Used in the glossary.
5624 \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.
5625 \ifdef\@glspreglossaryhook
5626 {
5627   \renewcommand*{\@glspreglossaryhook}{%
5628     \@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.

5629     \ifundef\@glxtr@org@postdescription
5630     {%
5631       \let\@glxtr@org@postdescription\glspostdescription
5632       \renewcommand*{\glspostdescription}{%
5633         \ifglentryexists{\glscurrententrylabel}%
5634         {%
5635           \glxtrpostdescription
5636           \@glxtr@org@postdescription
5637         }%
5638       }%
5639     }%
5640   }%
5641   {}%

    Enable the options used by \@glxtrp:
5642     \glossxtrsetopts
5643   }%
5644 }
5645 {}

```

postdescription This command will only be used if `\@gls@preglossaryhook` is available *and* the glossary style uses `\glspostdescription` without modifying it. (`\nopostdesc` will suppress this.) The glossaries-extra-stylemods package will add the post description hook to all the predefined styles that don't include it.

```
5646 \newcommand*{\glsxtrpostdescription}{%
5647   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5648 }
```

postdescgeneral

```
5649 \newcommand*{\glsxtrpostdescgeneral}{}%
```

xtrpostdescterm

```
5650 \newcommand*{\glsxtrpostdescterm}{}%
```

postdescacronym

```
5651 \newcommand*{\glsxtrpostdescacronym}{}%
```

descabbreviation

```
5652 \newcommand*{\glsxtrpostdescabbreviation}{}%
```

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.

```
5653 \renewcommand*{\glspostlinkhook}{%
5654   \ifglentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
5655 }
```

xtrpostlinkhook The entry label should already be stored in `\glslabel` by `\@gls@link`.

```
5656 \newcommand*{\glsxtrpostlinkhook}{%
5657   \glsxtrdiscardperiod{\glslabel}%
5658   {\glsxtrpostlinkendsentence}%
5659   {\glsxtrpostlink}%
5660 }
```

\glsxtrpostlink

```
5661 \newcommand*{\glsxtrpostlink}{%
5662   \csuse{glsxtrpostlink\glscategory{\glslabel}}%
5663 }
```

linkendsentence Done by `\glsxtrpostlinkhook` if a full stop is discarded.

```
5664 \newcommand*{\glsxtrpostlinkendsentence}{%
5665   \ifcsdef{glsxtrpostlink\glscategory{\glslabel}}%
5666   {%
5667     \csuse{glsxtrpostlink\glscategory{\glslabel}}%
5668   }%
```

Put the full stop back.

```
5668 .\spacefactor\sffcode'\. \relax
5669 }%
5670 {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

```
5671 \spacefactor\sffcode'\. \relax
5672 }%
5673 }
```

**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.

```
5674 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
5675   \glxtrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
5676 }
```

**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.

```
5677 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5678   \glxtrifwasfirstuse
5679   {%
5680     \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
5681   }%
5682   {}%
5683 }
```

**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).

```
5684 \newcommand*{\glxtrdiscardperiod}[3]{%
5685   \glxtrifwasfirstuse
5686   {%
5687     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5688     {#3}%
5689     {%
5690       \glusifattribute{#1}{discardperiod}{true}%
5691       {%
5692         \glusifplural
5693         {%
5694           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5695           {\glxtrifperiod{#2}{#3}}%
5696           {#3}%
5697         }%
5698         {%
5699           \glxtrifperiod{#2}{#3}%
5700         }%
5701       }%
5702     }%
5703   }
```



```

5701     }%
5702     {#3}%
5703     }%
5704 }%
5705 {%
5706   \glsifattribute{#1}{discardperiod}{true}%
5707   {%
5708     \glsifplural
5709     {%
5710       \glsifattribute{#1}{pluraldiscardperiod}{true}%
5711       {\glsxtrifperiod{#2}{#3}}%
5712       {#3}%
5713     }%
5714     {%
5715       \glsxtrifperiod{#2}{#3}%
5716     }%
5717   }%
5718   {#3}%
5719 }%
5720 }

```

`\glsxtrifperiod` 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`

```
5721 \newcommand*\glsxtrifperiod[1]{\new@ifnextchar.\@firstoftwo{#1}}
```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glsxtr@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 ' ').

```
5722 \newcommand*\glsxtr@punclist{.,:;?!}
```

`punctuationmark` Add character to punctuation list.

```
5723 \newcommand*\glsxtraddpunctuationmark[1]{\appto\glsxtr@punclist{#1}}
```

`punctuationmarks` Reset the punctuation list.

```
5724 \newcommand*\glsxtrsetpunctuationmarks[1]{\def\glsxtr@punclist{#1}}
```

`\glsxtrifpunc` `\glsxtrifnextpunc{<true part>}{<false part>}`

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```

5725 \newcommand*\glsxtrifnextpunc[2]{%
5726   \def\reserved@a{#1}%
5727   \def\reserved@b{#2}%
5728   \futurelet\@glspunc@token\glsxtr@ifnextpunc
5729 }

```

sxtr@ifnextpunc

```
5730 \newcommand*{\glsxtr@ifnextpunc}{%
5731   \glsxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5732   \reserved@b
5733 }
```

xtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
5734 \newcommand*{\glsxtr@ifpunctoken}[1]{%
5735   \expandafter\@glsxtr@ifpunctoken\expandafter#1\glsxtr@punclist\@nnil
5736 }
```

xtr@ifpunctoken

```
5737 \def\@glsxtr@ifpunctoken#1#2{%
5738   \let\reserved@d=#2%
5739   \ifx\reserved@d\@nnil
5740     \let\glsxtr@next\@glsxtr@notfoundinlist
5741   \else
5742     \ifx#1\reserved@d
5743       \let\glsxtr@next\@glsxtr@foundinlist
5744     \else
5745       \let\glsxtr@next\@glsxtr@ifpunctoken
5746     \fi
5747   \fi
5748   \glsxtr@next#1%
5749 }
```

xtr@foundinlist

```
5750 \def\@glsxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5751 \def\@glsxtr@notfoundinlist#1{\@secondoftwo}
```

glsxtrdopostpunc

```
\glsxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do <code> after the character otherwise do <code> before whatever comes next.

```
5752 \newcommand{\glsxtrdopostpunc}[1]{%
5753   \glsxtr@ifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
5754 }
```

@glsxtr@swaptwo

```
5755 \newcommand{\@glsxtr@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.

```
5756 \define@key{glstrabbrv}{category}{%
5757   \edef\glscategorylabel{#1}%
5758   \ifcsdef{@glssabbrv@current@#1}%
5759   {%
```

Warning should already have been issued.

```
5760   \let\@glstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5761   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5762   \glstr@applyabbrvstyle{\csname @glssabbrv@current@#1\endcsname}%
5763   \let\GlsXtrWarnDeprecatedAbbrStyle\@glstr@orgwarndep
5764   }%
5765   {%}%
5766 }
```

Save the short plural form. This may be needed before the entry is defined.

```
5767 \define@key{glstrabbrv}{shortplural}{%
5768   \def\@glss@shortpl{#1}%
5769 }
```

Similarly for the long plural form.

```
5770 \define@key{glstrabbrv}{longplural}{%
5771   \def\@glss@longpl{#1}%
5772 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

`\glsshortpltok`

```
5773 \newtoks\glsshortpltok
```

`\glslongpltok`

```
5774 \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.

```
5775 \newcommand*{\@glstr@insertdots}[2]{%
5776   \def#1{%
5777     \@glstr@insert@dots#1#2\@nnil
5778   }
```

xtr@insert@dots

```
5779 \newcommand*{\@glsxtr@insert@dots}[2]{%
5780   \ifx\@nnil#2\relax
5781   \let\@glsxtr@insert@dots@next\@gobble
5782   \else
5783   \ifx\relax#2\relax
5784   \else
5785     \appto#1{#2.}%
5786   \fi
5787   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
5788   \fi
5789   \@glsxtr@insert@dots@next#1%
5790 }
```

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep

```
5791 \newcommand*{\glsxtrwordsep}{\space}
```

Each word is marked with

\glsxtrword

```
5792 \newcommand*{\glsxtrword}[1]{#1}
```

tr@markwordseps

```
5793 \newcommand*{\@glsxtr@markwordseps}[2]{%
5794   \def#1{}%
5795   \@glsxtr@mark@wordseps#1#2 \@nnil
5796 }
```

r@mark@wordseps

```
5797 \def\@glsxtr@mark@wordseps#1#2 #3{%
5798   \ifdefempty{#1}%
5799   {\def#1{\protect\glsxtrword{#2}}}%
5800   {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
5801   \ifx\@nnil#3\relax
5802   \let\@glsxtr@mark@wordseps@next\relax
5803   \else
5804     \def\@glsxtr@mark@wordseps@next{%
5805       \@glsxtr@mark@wordseps#1#3}%
5806   \fi
5807   \@glsxtr@mark@wordseps@next
5808 }
```

newabbreviation Define a new generic abbreviation.

```
5809 \newcommand*{\newabbreviation}[4][ ]{%
5810   \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
5811 }
```

`\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.)

```
5812 \newcommand*{\glxtr@newabbreviation}[4]{%
5813   \glskeylisttok{#1}%
5814   \glslabeltok{#2}%
5815   \glsshorttok{#3}%
5816   \glslongtok{#4}%
```

Save the original short and long values (before attribute settings modify them).

```
5817   \def\glxtrorgshort{#3}%
5818   \def\glxtrorglong{#4}%
```

Get the category.

```
5819   \def\glscategorylabel{abbreviation}%
5820   \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
```

Ignore the shortplural and longplural keys.

```
5821   \setkeys*{\glxtrabbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
5822   \def\@gls@longpl{#4\glspluralsuffix}%
5823   \let\@gls@default@longpl\@gls@longpl
```

Has the markwords attribute been set?

```
5824   \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
5825   {%
5826     \@glxtr@markwordseps\@gls@long{#4}%
5827     \expandafter\def\expandafter\@gls@longpl\expandafter
5828       {\@gls@long\glspluralsuffix}%
5829     \let\@gls@default@longpl\@gls@longpl
```

Update `\glslongtok`.

```
5830     \expandafter\glslongtok\expandafter{\@gls@long}%
5831   }%
5832   {}%
```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
5833   \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
5834   {%
5835     \@glxtr@markwordseps\@gls@short{#3}%
5836   }%
5837   {}%
```

Has the insertdots attribute been set?

```
5838   \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
5839   {%
5840     \@glxtr@insertdots\@gls@short{#3}%
5841     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
5842   }%
5843   {\def\@gls@short{#3}}%
5844   }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
5845 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%  
5846 {%  
5847   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5848     '\abbrvpluralsuffix}%  
5849 }%  
5850 {%
```

Has the noshortplural attribute been set?

```
5851 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%  
5852 {%  
5853   \let\@gls@shortpl\@gls@short  
5854 }%  
5855 {%  
5856   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5857     \abbrvpluralsuffix}%  
5858 }%  
5859 }%
```

Update \glsshorttok:

```
5860 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
5861 \glxtrnewabbrevpresetkeyhook{#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).

```
5862 \setkeys*{\glxtrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
5863 \ifx\@gls@default@longpl\@gls@longpl  
5864 \else
```

Has the markwords attribute been set?

```
5865 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%  
5866 {%  
5867   \expandafter\@glxtr@keywordseps\expandafter\@gls@longpl\expandafter  
5868     {\@gls@longpl}%  
5869 }%  
5870 {}%  
5871 \fi
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
5872 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%  
5873 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```
5874 \newabbreviationhook
```

Define this entry:

```
5875 \protected@edef\@do@newglossaryentry{%  
5876   \noexpand\newglossaryentry{\the\glslabeltok}%  
5877   {%
```

```

5878     type=\glstrabbrvtype,%
5879     category=abbreviation,%
5880     short={\the\glsshorttok},%
5881     shortplural={\the\glsshortpltok},%
5882     long={\the\glslongtok},%
5883     longplural={\the\glslongpltok},%
5884     name={\the\glsshorttok},%
5885     \CustomAbbreviationFields,%
5886     \the\glskeylisttok
5887   }%
5888 }%
5889 \do@newglossaryentry
5890 \GlsXtrPostNewAbbreviation
5891 }

```

`\evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```
5892 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```
5893 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`\bbreviationhook` Hook for use with `\newabbreviation`.

```
5894 \newcommand*{\newabbreviationhook}{}

```

`\reivationFields`

```
5895 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrparen` For the parenthetical styles.

```
5896 \newcommand*{\glstrparen}[1]{(#1)}

```

`\lsxtrfullformat` Full format without case change.

```

5897 \newcommand*{\glxtrfullformat}[2]{%
5898   \glsfirstlongfont{\glsaccesslong{#1}}#2\glxtrfullsep{#1}%
5899   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
5900 }

```

`\lsxtrfullformat` Full format with case change.

```

5901 \newcommand*{\Glsxtrfullformat}[2]{%
5902   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glxtrfullsep{#1}%
5903   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
5904 }

```

`\xtrfullplformat` Plural full format without case change.

```

5905 \newcommand*{\glxtrfullplformat}[2]{%
5906   \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
5907   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
5908 }

```

`\glstrfullplformat` Plural full format with case change.

```
5909 \newcommand*{\Glsxtrfullplformat}[2]{%
5910   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
5911   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
5912 }
```

`\glstrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
5913 \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.

```
5914 \newcommand*{\Glsxtrinlinefullformat}{\glstrfullformat}
```

`\glstrinlinefullformat` Full format with case change.

```
5915 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glstrfullplformat` Plural full format without case change.

```
5916 \newcommand*{\Glsxtrinlinefullplformat}{\glstrfullplformat}
```

`\glstrinlinefullplformat` Plural full format with case change.

```
5917 \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`

```
5918 \renewcommand*{\glentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
5919 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\glentryfullpl`

```
5920 \renewcommand*{\glentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
5921 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
5922 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
5923 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
5924 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```



abbrvdefaultfont

```
5925 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
5926 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
5927 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
5928 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
5929 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
5930 \newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
5931 \newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
5932 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
5933 \newcommand*{\ns@glsxtrfull}[2][\%
```

```
5934 \new@ifnextchar[\@glsxtr@full{#1}{#2}}%
```

```
5935 \@glsxtr@full{#1}{#2}[]}%
```

```
5936 }
```

`\@glsxtr@full` Low-level macro:

```
5937 \def\@glsxtr@full#1#2[#3]{%
```

```
5938 \glsdoifexists{#2}}%
```

```
5939 {%
```

```
5940 \glssetabbrvfmt{\glscategory{#2}}%
```

```
5941 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
5942 \let\glsifplural\@secondoftwo
```

```
5943 \let\glsupcase\@firstofthree
```

```
5944 \let\glsinsert\@empty
```

```
5945 \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.

```
5946 \glsxtrsetupfulldefs
```

```
5947 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}}%
```

```
5948 }%
```

```
5949 \glspostlinkhook
```

```
5950 }
```

trsetupfulldefs

```
5951 \newcommand*{\glxtrsetupfulldefs}{%
5952   \let\glxtrifwasfirstuse\@firstoftwo
5953 }
```

\Glsxtrfull Full form (first letter uppercase).

```
5954 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
5955 \newcommand*\ns@Glsxtrfull[2][{}]{%
5956   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}]{%
5957     {\@Glsxtr@full{#1}{#2}[]}%
5958 }
```

\@Glsxtr@full Low-level macro:

```
5959 \def\@Glsxtr@full#1#2[#3]{%
5960   \glsdoifexists{#2}%
5961   {%
5962     \glssetabbrvfmt{\glscategory{#2}}%
5963     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5964     \let\glsifplural\@secondoftwo
5965     \let\glscapscase\@secondofthree
5966     \let\glsinsert\@empty
5967     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
5968     \glxtrsetupfulldefs
5969     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5970   }%
5971   \glspostlinkhook
5972 }
```

\GLSxtrfull Full form (all uppercase).

```
5973 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
5974 \newcommand*\ns@GLSxtrfull[2][{}]{%
5975   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}]{%
5976     {\@GLSxtr@full{#1}{#2}[]}%
5977 }
```

\@GLSxtr@full Low-level macro:

```
5978 \def\@GLSxtr@full#1#2[#3]{%
5979   \glsdoifexists{#2}%
5980   {%
5981     \glssetabbrvfmt{\glscategory{#2}}%
5982     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
5983     \let\glsifplural\@secondoftwo
5984     \let\glscapscase\@thirdofthree
5985     \let\glsinsert\@empty
5986     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
5987     \glxtrsetupfulldefs
5988     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
5989   }%
5990   \glspostlinkhook
```

5991 }

`\glstrfullpl` Plural full form (no case-change).

```
5992 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\@ns@glstrfullpl}
5993 \newcommand*\ns@glstrfullpl[2][]{%
5994   \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}%
5995   {\@glstr@fullpl{#1}{#2}[]}%
5996 }
```

`\@glstr@fullpl` Low-level macro:

```
5997 \def\@glstr@fullpl#1#2[#3]{%
5998   \glsdoifexists{#2}%
5999   {%
6000     \glssetabbrvfmt{\glscategory{#2}}%
6001     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6002     \let\glsifplural\@firstoftwo
6003     \let\glscapscase\@firstofthree
6004     \let\glsinsert\@empty
6005     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
6006     \glsxtrsetupfulldefs
6007     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6008   }%
6009   \glspostlinkhook
6010 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
6011 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6012 \newcommand*\ns@Glsxtrfullpl[2][]{%
6013   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6014   {\@Glsxtr@fullpl{#1}{#2}[]}%
6015 }
```

`\@Glsxtr@fullpl` Low-level macro:

```
6016 \def\@Glsxtr@fullpl#1#2[#3]{%
6017   \glsdoifexists{#2}%
6018   {%
6019     \glssetabbrvfmt{\glscategory{#2}}%
6020     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6021     \let\glsifplural\@firstoftwo
6022     \let\glsapsase\@secondofthree
6023     \let\glsinsert\@empty
6024     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6025     \glsxtrsetupfulldefs
6026     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6027   }%
6028   \glspostlinkhook
6029 }
```

`\GLSxtrfullpl` Plural full form (all upper case).

```

6030 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
6031 \newcommand*\ns@GLSxtrfullpl[2] [] {%
6032   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
6033     {\@GLSxtr@fullpl{#1}{#2} []}%
6034 }

```

\@GLSxtr@fullpl Low-level macro:

```

6035 \def\@GLSxtr@fullpl#1#2[#3] {%
6036   \glsdoifexists{#2}%
6037   {%
6038     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6039     \let\glsifplural\@firstoftwo
6040     \let\glscapscase\@thirdofthree
6041     \let\glsinsert\@empty
6042     \def\glscustomtext{%
6043       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6044     \glsxtrsetupfulldefs
6045     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6046   }%
6047   \glspostlinkhook
6048 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

6049 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6050 \newcommand*{\ns@glsxtrshort}[2] [] {%
6051   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}]{\@glsxtrshort{#1}{#2} []}%
6052 }

```

Read in the final optional argument:

```

6053 \def\@glsxtrshort#1#2[#3] {%
6054   \glsdoifexists{#2}%
6055   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

6056   \glssetabbrvfmt{\glscategory{#2}}%
6057   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6058   \let\glsxtrifwasfirstuse\@secondoftwo
6059   \let\glsifplural\@secondoftwo
6060   \let\glscapscase\@firstofthree
6061   \let\glsinsert\@empty
6062   \def\glscustomtext{%
6063     \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
6064     \ifglsxtrininsertinside\else#3\fi
6065   }%
6066   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6067   }%
6068   \glspostlinkhook

```

6069 }

\Glsxtrshort

6070 \newrobustcmd\*{\Glsxtrshort}{\@gls@hyp@opt\@ns@Glsxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

6071 \newcommand\*{\ns@Glsxtrshort}[2] [] {%

6072 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} [] }%

6073 }

Read in the final optional argument:

6074 \def\@Glsxtrshort#1#2[#3] {%

6075 \glsdoifexists{#2}%

6076 {%

6077 \glssetabbrvfmt{\glscategory{#2}}%

6078 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

6079 \let\glsxtrifwasfirstuse\@secondoftwo

6080 \let\glsifplural\@secondoftwo

6081 \let\glsapscase\@thirdofthree

6082 \let\glsinsert\@empty

6083 \def\glscustomtext{%

6084 \glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

6085 \ifglsxtrininsertinside\else#3\fi

6086 }%

6087 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

6088 }%

6089 \glspostlinkhook

6090 }

\GLSxtrshort

6091 \newrobustcmd\*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}

Define the un-starred form. Need to determine if there is a final optional argument

6092 \newcommand\*{\ns@GLSxtrshort}[2] [] {%

6093 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} [] }%

6094 }

Read in the final optional argument:

6095 \def\@GLSxtrshort#1#2[#3] {%

6096 \glsdoifexists{#2}%

6097 {%

6098 \glssetabbrvfmt{\glscategory{#2}}%

6099 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

6100 \let\glsxtrifwasfirstuse\@secondoftwo

6101 \let\glsifplural\@secondoftwo

6102 \let\glsapscase\@thirdofthree

6103 \let\glsinsert\@empty

6104 \def\glscustomtext{%

6105 \mfirstucMakeUppercase

6106 {\glsabbrvfont{\Glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%

6107 \ifglsxtrininsertinside\else#3\fi

```

6108     }%
6109     }%
6110     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6111     }%
6112     \glspostlinkhook
6113 }

```

`\glsxtrlong`

```

6114 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6115 \newcommand*{\ns@glsxtrlong}[2][{}]{%
6116   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
6117 }

    Read in the final optional argument:
6118 \def\@glsxtrlong#1#2[#3]{%
6119   \glsdoifexists{#2}%
6120   {%
6121     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6122     \let\glsxtrifwasfirstuse\@secondoftwo
6123     \let\glsifplural\@secondoftwo
6124     \let\glscapscase\@firstofthree
6125     \let\glsinsert\@empty
6126     \def\glscustomtext{%
6127       \glslongfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
6128       \ifglsxtrininsertinside\else#3\fi
6129     }%
6130     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6131   }%
6132   \glspostlinkhook
6133 }

```

`\Glsxtrlong`

```

6134 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6135 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
6136   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
6137 }

    Read in the final optional argument:
6138 \def\@Glsxtrlong#1#2[#3]{%
6139   \glsdoifexists{#2}%
6140   {%
6141     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6142     \let\glsxtrifwasfirstuse\@secondoftwo
6143     \let\glsifplural\@secondoftwo
6144     \let\glscapscase\@secondofthree
6145     \let\glsinsert\@empty
6146     \def\glscustomtext{%

```

```

6147      \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6148      \ifglxtrinsertinside\else#3\fi
6149  }%
6150  \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6151  }%
6152  \glspostlinkhook
6153 }

```

\GLSxtrlong

```

6154 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6155 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6156   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6157 }

    Read in the final optional argument:
6158 \def\@GLSxtrlong#1#2[#3] {%
6159   \glsdoifexists{#2}%
6160   {%
6161     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6162     \let\glxtrifwasfirstuse\@secondoftwo
6163     \let\glsifplural\@secondoftwo
6164     \let\glscapscase\@thirdofthree
6165     \let\glsinsert\@empty
6166     \def\glscustomtext{%
6167       \mfirstucMakeUppercase
6168       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6169       \ifglxtrinsertinside\else#3\fi
6170     }%
6171   }%
6172   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6173   }%
6174   \glspostlinkhook
6175 }

```

Plural short forms:

\glxtrshortpl

```

6176 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6177 \newcommand*{\ns@glxtrshortpl}[2] [] {%
6178   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
6179 }

    Read in the final optional argument:
6180 \def\@glxtrshortpl#1#2[#3] {%
6181   \glsdoifexists{#2}%
6182   {%
6183     \glsssetabbrvfmt{\glscategory{#2}}%

```

```

6184 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6185 \let\glxtrifwasfirstuse\@secondoftwo
6186 \let\gl@sifplural\@firstoftwo
6187 \let\glscapscase\@firstofthree
6188 \let\gl$insert\@empty
6189 \def\glscustomtext{%
6190 \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6191 \ifglxtrininsertinside\else#3\fi
6192 }%
6193 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6194 }%
6195 \glspostlinkhook
6196 }

```

\Glsxtrshortpl

```

6197 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6198 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
6199 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
6200 }

    Read in the final optional argument:
6201 \def\@Glsxtrshortpl#1#2[#3] {%
6202 \glsoifexists{#2}%
6203 {%
6204 \glsetabbrvfmt{\glscategory{#2}}%
6205 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6206 \let\glxtrifwasfirstuse\@secondoftwo
6207 \let\gl@sifplural\@firstoftwo
6208 \let\glscapscase\@secondofthree
6209 \let\gl$insert\@empty
6210 \def\glscustomtext{%
6211 \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6212 \ifglxtrininsertinside\else#3\fi
6213 }%
6214 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6215 }%
6216 \glspostlinkhook
6217 }

```

\GLSxtrshortpl

```

6218 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6219 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6220 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6221 }

    Read in the final optional argument:
6222 \def\@GLSxtrshortpl#1#2[#3] {%

```



```

6223 \glsdoifexists{#2}%
6224 {%
6225   \glssetabbrvfmt{\glscategory{#2}}%
6226   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6227   \let\glsxtrifwasfirstuse\@secondoftwo
6228   \let\glsifplural\@firstoftwo
6229   \let\gls caps case\@thirdofthree
6230   \let\glsinsert\@empty
6231   \def\gls custom text{%
6232     \mfirstucMakeUppercase
6233     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6234     \ifglsxtrininsertinside\else#3\fi
6235   }%
6236 }%
6237 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6238 }%
6239 \gls post link hook
6240 }

```

Plural long forms:

\glsxtrlongpl

```

6241 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6242 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
6243   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
6244 }

```

Read in the final optional argument:

```

6245 \def\@glsxtrlongpl#1#2[#3]{%
6246   \glsdoifexists{#2}%
6247   {%
6248     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6249     \let\glsxtrifwasfirstuse\@secondoftwo
6250     \let\glsifplural\@firstoftwo
6251     \let\gls caps case\@firstofthree
6252     \let\glsinsert\@empty
6253     \def\gls custom text{%
6254       \gls long font{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
6255       \ifglsxtrininsertinside\else#3\fi
6256     }%
6257     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6258   }%
6259   \gls post link hook
6260 }

```

\Glsxtrlongpl

```

6261 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns\Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
6262 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
6263   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
6264 }
```

Read in the final optional argument:

```
6265 \def\@Glsxtrlongpl#1#2[#3]{%
6266   \glsdoifexists{#2}%
6267   {%
6268     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6269     \let\glxtrifwasfirstuse\@secondoftwo
6270     \let\gl@ifplural\@firstoftwo
6271     \let\glscapscase\@secondofthree
6272     \let\glinsert\@empty
6273     \def\glscustomtext{%
6274       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6275       \ifglxtrininsertinside\else#3\fi
6276     }%
6277     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6278   }%
6279   \glspostlinkhook
6280 }
```

\GLSxtrlongpl

```
6281 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6282 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
6283   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
6284 }
```

Read in the final optional argument:

```
6285 \def\@GLSxtrlongpl#1#2[#3]{%
6286   \glsdoifexists{#2}%
6287   {%
6288     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6289     \let\glxtrifwasfirstuse\@secondoftwo
6290     \let\gl@ifplural\@firstoftwo
6291     \let\glscapscase\@thirdofthree
6292     \let\glinsert\@empty
6293     \def\glscustomtext{%
6294       \mfirstucMakeUppercase
6295       {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6296       \ifglxtrininsertinside\else#3\fi
6297     }%
6298   }%
6299   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6300 }%
6301 \glspostlinkhook
6302 }
```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6303 \newcommand*{\glssetabbrvfmt}[1]{%
6304   \ifcsdef{@glsabbrv@current@#1}%
6305     {\glstr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6306     {\glstr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6307 }

```

`\glsuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6308 \newrobustcmd*{\glsuseabbrvfont}[2]{\glssetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\glsuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6309 \newrobustcmd*{\glsuselongfont}[2]{\glssetabbrvfmt{#2}\glsfont{#1}}

```

`\glstrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6310 \newcommand*{\glstrgenabbrvfmt}{%
6311   \ifdefempty\glscustomtext
6312     {%
6313       \ifglsused\glslabel
6314         {%

```

Subsequent use:

```

6315         \glsifplural
6316         {%

```

Subsequent plural form:

```

6317         \glscapscase
6318         {%

```

Subsequent plural form, don't adjust case:

```

6319         \glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6320         }%
6321         {%

```

Subsequent plural form, make first letter upper case:

```

6322         \Glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6323         }%
6324         {%

```

Subsequent plural form, all caps:

```

6325         \mfirstucMakeUppercase
6326         {\glstrsubsequentplfmt{\glslabel}{\glsinsert}}%
6327         }%
6328         }%
6329         {%

```

Subsequent singular form

```

6330         \glscapscase
6331         {%

```

Subsequent singular form, don't adjust case:

```
6332      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6333      }%  
6334      {%
```

Subsequent singular form, make first letter upper case:

```
6335      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6336      }%  
6337      {%
```

Subsequent singular form, all caps:

```
6338      \mfirstucMakeUppercase  
6339      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6340      }%  
6341      }%  
6342      }%  
6343      {%
```

First use:

```
6344      \glsifplural  
6345      {%
```

First use plural form:

```
6346      \glscapscase  
6347      {%
```

First use plural form, don't adjust case:

```
6348      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6349      }%  
6350      {%
```

First use plural form, make first letter upper case:

```
6351      \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
6352      }%  
6353      {%
```

First use plural form, all caps:

```
6354      \mfirstucMakeUppercase  
6355      {\glxtrfullplformat{\glslabel}{\glsinsert}}%  
6356      }%  
6357      }%  
6358      {%
```

First use singular form

```
6359      \glscapscase  
6360      {%
```

First use singular form, don't adjust case:

```
6361      \glxtrfullformat{\glslabel}{\glsinsert}%  
6362      }%  
6363      {%
```

First use singular form, make first letter upper case:

```
6364      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6365      }%
6366      {%
```

First use singular form, all caps:

```
6367      \mfirstucMakeUppercase
6368      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6369      }%
6370      }%
6371      }%
6372      }%
6373      {%
```

User supplied text.

```
6374      \glscustomtext
6375      }%
6376 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6377 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6378   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6379   \ifglsxtrininsertinside \else#2\fi
6380 }
6381 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6382 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6383   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6384   \ifglsxtrininsertinside \else#2\fi
6385 }
6386 \let\glsxtrdefaultsubsequentplfmt\glsxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
6387 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6388   \glsabbrvfont{\Glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6389   \ifglsxtrininsertinside \else#2\fi
6390 }
6391 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```
6392 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6393   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6394   \ifglsxtrininsertinside \else#2\fi
6395 }
6396 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt
```

## 1.6.1 Abbreviation Styles Setup

abbreviationstyle

```
6397 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6398   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6399   {%
6400     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6401   }%
6402   {%
```

Have abbreviations already been defined for this category?

```
6403   \ifcsstring{@glsabbrv@current@#1}{#2}%
6404   {%
```

Style already set.

```
6405   }%
6406   {%
6407     \def\@glsxtr@dostylewarn{%
6408       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6409       {%
6410         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
6411           style has been switched \MessageBreak
6412           for category ‘#1’, \MessageBreak
6413           but there have already been entries \MessageBreak
6414           defined for this category. Unwanted \MessageBreak
6415           side-effects may result}}%
6416         \@endfortrue
6417       }%
6418     \@glsxtr@dostylewarn
```

Set up the style for the given category.

```
6419     \csdef{@glsabbrv@current@#1}{#2}%
6420     \glsxtr@applyabbrvstyle{#2}%
6421   }%
6422 }%
6423 }
```

applyabbrvstyle Apply the abbreviation style without existence check.

```
6424 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
6425   \csuse{@glsabbrv@dispstyle@setup@#1}%
6426   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6427 }
```

r@applyabbrvfmt Only apply the style formats.

```
6428 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
6429   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6430 }
```

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.

```

6431 \newcommand*{\newabbreviationstyle}[3]{%
6432   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6433   {%
6434     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6435       defined}{}%
6436   }%
6437   {%
6438     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6439       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6440       #2}%
6441     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6442       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6443       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6444       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6445       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
        Reset \glsxtrsubsequentfmt etc in case a style changes this.
6446       \let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt
6447       \let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt
6448       \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6449       \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6450       #3}%
6451     }%
6452 }

```

breivationstyle

```

6453 \newcommand*{\renewabbreviationstyle}[3]{%
6454   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
6455   {%
6456     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6457   }%
6458   {%
6459     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6460       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6461       #2}%
6462     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6463       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6464       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6465       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6466       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6467       #3}%
6468     }%
6469 }

```

`abbreviationstyle` Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```
6470 \newcommand*{\letabbreviationstyle}[2]{%
6471   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6472   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6473 }
```

`deprecated@abbrstyle` `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```
6474 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6475   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6476     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6477     \csuse{@glsabbrv@dispstyle@setup@#2}%
6478   }%
6479   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6480 }
```

`deprecatedAbbrStyle` Generate warning for deprecated style use.

```
6481 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6482   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6483     use ‘#2’ instead}%
6484 }
```

`useAbbrStyleSetup`

```
6485 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6486   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
6487   {%
6488     \PackageError{glossaries-extra}%
6489     {Unknown abbreviation style definitions ‘#1’}{}%
6490   }%
6491   {%
6492     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6493   }%
6494 }
```

`useAbbrStyleFmt`

```
6495 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
6496   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6497   {%
6498     \PackageError{glossaries-extra}%
6499     {Unknown abbreviation style formats ‘#1’}{}%
6500   }%
6501   {%
6502     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6503   }%
6504 }
```



## 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.

```
6505 \newif\ifglstrinsertinside
6506 \glstrinsertinsidefalse
```

`long-short`

```
6507 \newabbreviationstyle{long-short}%
6508 {%
6509   \renewcommand*{\CustomAbbreviationFields}{%
6510     name={\protect\glsabbrvfont{\the\glsshorttok}},
6511     sort={\the\glsshorttok},
6512     first={\protect\glsfirstlongfont{\the\glslongtok}%
6513       \protect\glstrfullsep{\the\glslabeltok}%
6514       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6515     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6516       \protect\glstrfullsep{\the\glslabeltok}%
6517       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
6518     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6519     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
6520 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6521   \glshasattribute{\the\glslabeltok}{regular}%
6522   {%
6523     \glissetattribute{\the\glslabeltok}{regular}{false}%
6524     }%
6525   }%
6526 }%
6527 }%
6528 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6529 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6530 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6531 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6532 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6533 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

6534 \renewcommand*{\glxtrfullformat}[2]{%
6535   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6536   \ifglxtrinsertinside\else##2\fi
6537   \glxtrfullsep{##1}%
6538   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6539 }%
6540 \renewcommand*{\glxtrfullplformat}[2]{%
6541   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6542   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6543   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6544 }%
6545 \renewcommand*{\Glsxtrfullformat}[2]{%
6546   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6547   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6548   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6549 }%
6550 \renewcommand*{\Glsxtrfullplformat}[2]{%
6551   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6552   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6553   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6554 }%
6555 }

```

Set this as the default style for general abbreviations:

```

6556 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

6557 \newcommand*{\glxtrlongshortdescsort}{%
6558   \expandonce\glxtrorglongspace (\expandonce\glxtrorgshort)%
6559 }

```

ngshortdescname

```

6560 \newcommand*{\glxtrlongshortdescname}{%
6561   \protect\glslongfont{\the\glslongtok}
6562   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6563 }

```

long-short-desc User supplies description. The long form is included in the name.

```

6564 \newabbreviationstyle{long-short-desc}%
6565 {%
6566   \renewcommand*{\CustomAbbreviationFields}{%
6567     name={\glxtrlongshortdescname},
6568     sort={\glxtrlongshortdescsort},%
6569     first={\protect\glsfirstlongfont{\the\glslongtok}%
6570       \protect\glxtrfullsep{\the\glslabeltok}%
6571       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6572     firstplural={\protect\glsfirstlongfont{\the\glslongtok}%
6573       \protect\glxtrfullsep{\the\glslabeltok}%

```

```
6574 \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
6575 text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```
6576 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6577 }%
```

Unset the regular attribute if it has been set.

```
6578 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6579 \glshasattribute{\the\glslabeltok}{regular}%
6580 {%
6581 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6582 }%
6583 {}%
6584 }%
6585 }%
6586 {%
6587 \GlsXtrUseAbbrStyleFmts{long-short}%
6588 }
```

**short-long** Short form followed by long form in parenthesis on first use.

```
6589 \newabbreviationstyle{short-long}%
6590 {%
6591 \renewcommand*{\CustomAbbreviationFields}{%
6592 name={\protect\glsabbrvfont{\the\glsshorttok}},
6593 sort={\the\glsshorttok},
6594 description={\the\glslongtok},%
6595 first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6596 \protect\glstrfullsep{\the\glslabeltok}%
6597 \glstrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6598 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6599 \protect\glstrfullsep{\the\glslabeltok}%
6600 \glstrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6601 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```
6602 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6603 \glshasattribute{\the\glslabeltok}{regular}%
6604 {%
6605 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6606 }%
6607 {}%
6608 }%
6609 }%
6610 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
6611 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6612 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%
```

```

6613 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6614 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6615 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6616 \renewcommand*{\glsxtrfullformat}[2]{%
6617   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6618   \ifglsxtrininsertinside\else##2\fi
6619   \glsxtrfullsep{##1}%
6620   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6621 }%
6622 \renewcommand*{\glsxtrfullplformat}[2]{%
6623   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6624   \ifglsxtrininsertinside\else##2\fi
6625   \glsxtrfullsep{##1}%
6626   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6627 }%
6628 \renewcommand*{\Glsxtrfullformat}[2]{%
6629   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6630   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6631   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6632 }%
6633 \renewcommand*{\Glsxtrfullplformat}[2]{%
6634   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6635   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6636   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6637 }%
6638 }

```

ortlongdescsort

```

6639 \newcommand*{\glsxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

6640 \newcommand*{\glsxtrshortlongdescname}{%
6641   \protect\glsabbrvfont{\the\glsshorttok}
6642   \glsxtrparen{\protect\glslongfont{\the\glslongtok}}}%
6643 }

```

short-long-desc User supplies description. The long form is included in the name.

```

6644 \newabbreviationstyle{short-long-desc}%
6645 {%
6646   \renewcommand*{\CustomAbbreviationFields}{%
6647     name={\glsxtrshortlongdescname},
6648     sort={\glsxtrshortlongdescsort},
6649     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6650     \protect\glsxtrfullsep{\the\glslabeltok}}%
6651     \glsxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6652     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6653     \protect\glsxtrfullsep{\the\glslabeltok}}%
6654     \glsxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%

```

```

6655     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6656     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6657 }%

Unset the regular attribute if it has been set.
6658 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6659   \glshasattribute{\the\glslabeltok}{regular}%
6660   {%
6661     \glissetattribute{\the\glslabeltok}{regular}{false}%
6662   }%
6663   {}}%
6664 }%
6665 }%
6666 {%
6667   \GlsXtrUseAbbrStyleFmts{short-long}%
6668 }

```

ongfootnotefont Only used by the “footnote” styles.

```
6669 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6670 \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`).

```
6671 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```

6672 \newabbreviationstyle{footnote}%
6673 {%
6674   \renewcommand*{\CustomAbbreviationFields}{%
6675     name={\protect\glsabbrvfont{\the\glsshorttok}},
6676     sort={\the\glsshorttok},
6677     description={\the\glslongtok},%

6678     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6679     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6680     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6681     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6682     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6683     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

```

6684 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.

```
6685 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6686   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6687   \glsattribute{\the\glslabeltok}{regular}%
6688   {%
6689     \glssetattribute{\the\glslabeltok}{regular}{false}%
6690   }%
6691   {}}%
6692 }%
6693 }%
6694 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6695 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6696 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6697 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6698 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6699 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
6700 \renewcommand*{\glsxtrfullformat}[2]{%
6701   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6702   \ifglsxtrininsertinside\else##2\fi
6703   \protect\glsxtrabbrvfootnote{##1}%
6704   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6705 }%
6706 \renewcommand*{\glsxtrfullplformat}[2]{%
6707   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6708   \ifglsxtrininsertinside\else##2\fi
6709   \protect\glsxtrabbrvfootnote{##1}%
6710   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6711 }%
6712 \renewcommand*{\Glsxtrfullformat}[2]{%
6713   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6714   \ifglsxtrininsertinside\else##2\fi
6715   \protect\glsxtrabbrvfootnote{##1}%
6716   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6717 }%
6718 \renewcommand*{\Glsxtrfullplformat}[2]{%
6719   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6720   \ifglsxtrininsertinside\else##2\fi
6721   \protect\glsxtrabbrvfootnote{##1}%
6722   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6723 }%
```

The first use full form and the inline full form use the short (long) style.

```
6724 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6725   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6726   \ifglsxtrininsertinside\else##2\fi
6727   \protect\glsxtrabbrvfootnote{##1}%
6728   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6729 }%
```

```

6726     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6727     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
6728 }%
6729 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6730     \glsfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6731     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6732     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
6733 }%
6734 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6735     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6736     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6737     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
6738 }%
6739 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6740     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6741     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6742     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
6743 }%
6744 }

```

#### short-footnote

```

6745 \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.

```

6746 \newabbreviationstyle{postfootnote}%
6747 {%
6748     \renewcommand*{\CustomAbbreviationFields}{%
6749         name={\protect\glsabbrvfont{\the\glsshorttok}},
6750         sort={\the\glsshorttok},
6751         description={\the\glslongtok},%
6752         first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6753         firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6754         plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

6755 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6756     \csdef{glxtrpostlink\glscategorylabel}{%
6757         \glxtrifwasfirstuse
6758         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6759         \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
6760             {\glsfirstlongfootnotefont{\glentrylong{\glslabel}}}%
6761         }%

```

```

6762     {}%
6763   }%
6764   \glshasattribute{\the\glslabeltok}{regular}%
6765   {%
6766     \glsetattribute{\the\glslabeltok}{regular}{false}%
6767   }%
6768   {}%
6769 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

6770 \renewcommand*{\glxtrsetupfulldefs}{%
6771   \let\glxtrifwasfirstuse\@secondoftwo
6772 }%
6773 }%
6774 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6775 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6776 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
6777 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvdefaultfont{##1}}%
6778 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
6779 \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6780 \renewcommand*{\glxtrfullformat}[2]{%
6781   \glssfirstabrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6782   \ifglxtrininsertinside\else##2\fi
6783 }%
6784 \renewcommand*{\glxtrfullplformat}[2]{%
6785   \glssfirstabrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6786   \ifglxtrininsertinside\else##2\fi
6787 }%
6788 \renewcommand*{\Glsxtrfullformat}[2]{%
6789   \glssfirstabrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6790   \ifglxtrininsertinside\else##2\fi
6791 }%
6792 \renewcommand*{\Glsxtrfullplformat}[2]{%
6793   \glssfirstabrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6794   \ifglxtrininsertinside\else##2\fi
6795 }%

```

The first use full form and the inline full form use the short (long) style.

```

6796 \renewcommand*{\glxtrinlinefullformat}[2]{%
6797   \glssfirstabrvfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6798   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6799   \glxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
6800 }%
6801 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6802   \glssfirstabrvfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6803   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

6804 \glstrparen{\glfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6805 }%
6806 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6807 \glfirstabbrvfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
6808 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
6809 \glstrparen{\glfirstlongfootnotefont{\glsaccesslong{##1}}}%
6810 }%
6811 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6812 \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
6813 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
6814 \glstrparen{\glfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6815 }%
6816 }

```

rt-postfootnote

```
6817 \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.

```

6818 \newabbreviationstyle{short}%
6819 {%
6820 \renewcommand*{\CustomAbbreviationFields}{%
6821 name={\protect\glssabbrvfont{\the\glssshorttok}},
6822 sort={\the\glssshorttok},
6823 first={\protect\glssfirstabbrvfont{\the\glssshorttok}},
6824 firstplural={\protect\glssfirstabbrvfont{\the\glssshortpltok}},
6825 text={\protect\glssabbrvfont{\the\glssshorttok}},
6826 plural={\protect\glssabbrvfont{\the\glssshortpltok}},
6827 description={\the\glslongtok}}%
6828 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6829 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
6830 }%
6831 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6832 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6833 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
6834 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
6835 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
6836 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

6837 \renewcommand*{\glstrinlinefullformat}[2]{%
6838 \protect\glssfirstabbrvfont{\glsaccessshort{##1}}%
6839 \ifglstrinsertinside##2\fi}%
6840 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
6841 \glstrparen{\glssfirstlongfont{\glsaccesslong{##1}}}%
6842 }%

```

```

6843 \renewcommand*{\glxstrinlinefullplformat}[2]{%
6844   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
6845     \ifglxstrinsertinside##2\fi}%
6846   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
6847   \glxstrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6848 }%
6849 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6850   \protect\glsfirstabbrvfont{\glsaccessshort{##1}%
6851     \ifglxstrinsertinside##2\fi}%
6852   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
6853   \glxstrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
6854 }%
6855 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6856   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
6857     \ifglxstrinsertinside##2\fi}%
6858   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
6859   \glxstrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
6860 }%

```

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.

```

6861 \renewcommand*{\glxstrfullformat}[2]{%
6862   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
6863   \ifglxstrinsertinside\else##2\fi
6864 }%
6865 \renewcommand*{\glxstrfullplformat}[2]{%
6866   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
6867   \ifglxstrinsertinside\else##2\fi
6868 }%
6869 \renewcommand*{\Glsxtrfullformat}[2]{%
6870   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
6871   \ifglxstrinsertinside\else##2\fi
6872 }%
6873 \renewcommand*{\Glsxtrfullplformat}[2]{%
6874   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
6875   \ifglxstrinsertinside\else##2\fi
6876 }%
6877 }

```

Set this as the default style for acronyms:

```

6878 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

6879 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

6880 \newabbreviationstyle{short-nolong-noreg}%
6881 {%
6882   \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

6883 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6884   \glshasattribute{\the\glslabelltok}{regular}%
6885   {%
6886     \glsssetattribute{\the\glslabelltok}{regular}{false}%
6887   }%
6888   {}%
6889 }%
6890 }%
6891 {%
6892   \GlsXtrUseAbbrStyleFmts{short-nolong}%
6893 }

```

trshortdescname

```

6894 \newcommand*{\glxtrshortdescname}{%
6895   \protect\glsabbrvfont{\the\glssshorttok}%
6896 }

```

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.

```

6897 \newabbreviationstyle{short-desc}%
6898 {%
6899   \renewcommand*{\CustomAbbreviationFields}{%
6900     name={\glxtrshortdescname},
6901     sort={\the\glssshorttok},
6902     first={\protect\glsfirstabbrvfont{\the\glssshorttok}},
6903     firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},
6904     text={\protect\glsabbrvfont{\the\glssshorttok}},
6905     plural={\protect\glsabbrvfont{\the\glssshortpltok}},
6906     description={\the\glslongtok}}%
6907   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6908     \glsssetattribute{\the\glslabelltok}{regular}{true}}%
6909 }%
6910 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6911 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6912 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6913 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6914 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6915 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

6916 \renewcommand*{\glxtrinlinefullformat}[2]{%
6917   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6918   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6919   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6920 }%
6921 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6922   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

6923 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6924 \glxtrparen{\glsfirslongfont{\glssaccesslongpl{##1}}}%
6925 }%
6926 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6927 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6928 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6929 \glxtrparen{\glsfirslongfont{\glssaccesslong{##1}}}%
6930 }%
6931 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6932 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6933 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6934 \glxtrparen{\glsfirslongfont{\glssaccesslongpl{##1}}}%
6935 }%

```

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.

```

6936 \renewcommand*{\glxtrfullformat}[2]{%
6937 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6938 \ifglxtrinsertinside\else##2\fi
6939 }%
6940 \renewcommand*{\glxtrfullplformat}[2]{%
6941 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6942 \ifglxtrinsertinside\else##2\fi
6943 }%
6944 \renewcommand*{\Glsxtrfullformat}[2]{%
6945 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6946 \ifglxtrinsertinside\else##2\fi
6947 }%
6948 \renewcommand*{\Glsxtrfullplformat}[2]{%
6949 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6950 \ifglxtrinsertinside\else##2\fi
6951 }%
6952 }

```

ort-nolong-desc

```
6953 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

6954 \newabbreviationstyle{short-nolong-desc-noreg}%
6955 {%
6956 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

6957 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6958 \glshasattribute{\the\glslabeltok}{regular}%
6959 {%
6960 \glissetattribute{\the\glslabeltok}{regular}{false}%
6961 }%
6962 {}%
6963 }%

```

```

6964 }%
6965 {%
6966   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
6967 }

```

**nolong-short** Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

6968 \newabbreviationstyle{nolong-short}%
6969 {%
6970   \GlsXtrUseAbbrStyleSetup{short-nolong}%
6971 }%
6972 {%
6973   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

6974 \renewcommand*{\glxtrinlinefullformat}[2]{%
6975   \protect\glstlongfont{\glsaccesslong{##1}%
6976     \ifglxtrinsertinside##2\fi}%
6977   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6978   \glxtrparen{\glstfirstabbrvfont{\glsaccessshort{##1}}}%
6979 }%
6980 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6981   \protect\glstlongfont{\glsaccesslongpl{##1}%
6982     \ifglxtrinsertinside##2\fi}%
6983   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6984   \glxtrparen{\glstfirstabbrvfont{\glsaccessshortpl{##1}}}%
6985 }%
6986 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6987   \protect\glstlongfont{\glsaccesslong{##1}%
6988     \ifglxtrinsertinside##2\fi}%
6989   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6990   \glxtrparen{\glstfirstabbrvfont{\Glsaccessshort{##1}}}%
6991 }%
6992 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6993   \protect\glstlongfont{\glsaccesslongpl{##1}%
6994     \ifglxtrinsertinside##2\fi}%
6995   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6996   \glxtrparen{\glstfirstabbrvfont{\Glsaccessshortpl{##1}}}%
6997 }%
6998 }

```

**nolong-short-noreg** Like nolong-short but doesn't set the regular attribute.

```

6999 \newabbreviationstyle{nolong-short-noreg}%
7000 {%
7001   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7002 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7003   \glshasattribute{\the\glstlabeltok}{regular}%
7004   {%

```

```

7005     \glssetattribute{\the\glslabeltok}{regular}{false}%
7006   }%
7007   {}%
7008 }%
7009}%
7010{%
7011  \GlsXtrUseAbbrStyleFmts{nolong-short}%
7012}

```

`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.

```

7013 \newabbreviationstyle{long-desc}%
7014 {%
7015   \renewcommand*{\CustomAbbreviationFields}{%
7016     name={\protect\protect\glslongfont{\the\glslongtok}},
7017     sort={\the\glslongtok},
7018     first={\protect\glsfirstlongfont{\the\glslongtok}},
7019     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7020     text={\glslongfont{\the\glslongtok}},
7021     plural={\glslongfont{\the\glslongpltok}}}%
7022   }%
7023   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7024     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7025 }%
7026 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7027 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
7028 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7029 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7030 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7031 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7032 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
7033   \glslongfont{\glsaccesslong{##1}}\ifglsxtrininsertinside ##2\fi}%
7034 \ifglsxtrininsertinside \else##2\fi
7035 }%
7036 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
7037   \glslongfont{\glsaccesslongpl{##1}}\ifglsxtrininsertinside ##2\fi}%
7038 \ifglsxtrininsertinside \else##2\fi
7039 }%
7040 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7041   \glslongfont{\Glsaccesslong{##1}}\ifglsxtrininsertinside ##2\fi}%
7042 \ifglsxtrininsertinside \else##2\fi
7043 }%
7044 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7045   \glslongfont{\Glsaccesslongpl{##1}}\ifglsxtrininsertinside ##2\fi}%
7046 \ifglsxtrininsertinside \else##2\fi

```

7047 }%

The inline full form displays the long format followed by the short form in parentheses.

```
7048 \renewcommand*{\glxtrinlinefullformat}[2]{%
7049   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7050   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7051   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7052 }%
7053 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7054   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7055   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7056   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7057 }%
7058 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7059   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7060   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7061   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7062 }%
7063 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7064   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7065   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7066   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7067 }%
```

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.

```
7068 \renewcommand*{\glxtrfullformat}[2]{%
7069   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7070   \ifglxtrinsertinside\else##2\fi
7071 }%
7072 \renewcommand*{\glxtrfullplformat}[2]{%
7073   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7074   \ifglxtrinsertinside\else##2\fi
7075 }%
7076 \renewcommand*{\Glsxtrfullformat}[2]{%
7077   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7078   \ifglxtrinsertinside\else##2\fi
7079 }%
7080 \renewcommand*{\Glsxtrfullplformat}[2]{%
7081   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7082   \ifglxtrinsertinside\else##2\fi
7083 }%
7084 }
```

ng-noshort-desc Provide a synonym that matches similar styles.

```
7085 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

hort-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```
7086 \newabbreviationstyle{long-noshort-desc-noreg}%
7087 {%
```

```

7088 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
    Unset the regular attribute if it has been set.
7089 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7090   \glshasattribute{\the\glslabeltok}{regular}%
7091   {%
7092     \glissetattribute{\the\glslabeltok}{regular}{false}%
7093   }%
7094   {}%
7095 }%
7096}%
7097{%
7098 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7099 }

```

`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.

```

7100 \newabbreviationstyle{long}%
7101 {%
7102   \renewcommand*{\CustomAbbreviationFields}{%
7103     name={\protect\glsabbrvfont{\the\glsshorttok}},
7104     sort={\the\glsshorttok},
7105     first={\protect\glsfirstlongfont{\the\glslongtok}},
7106     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7107     text={\glslongfont{\the\glslongtok}},
7108     plural={\glslongfont{\the\glslongpltok}},%
7109     description={\the\glslongtok}%
7110   }%
7111   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7112     \glissetattribute{\the\glslabeltok}{regular}{true}}%
7113 }%
7114 {%
7115   \GlsXtrUseAbbrStyleFmts{long-desc}%
7116 }

```

`long-noshort` Provide a synonym that matches similar styles.

```

7117 \letabbreviationstyle{long-noshort}{long}

```

`g-noshort-noreg` Like `long-noshort` but doesn't set the regular attribute.

```

7118 \newabbreviationstyle{long-noshort-noreg}%
7119 {%
7120   \GlsXtrUseAbbrStyleSetup{long-noshort}%
    Unset the regular attribute if it has been set.
7121   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7122     \glshasattribute{\the\glslabeltok}{regular}%
7123     {%
7124       \glissetattribute{\the\glslabeltok}{regular}{false}%

```



```

7125 }%
7126 {}%
7127 }%
7128 }%
7129 {%
7130 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7131 }

```

### 1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

```

\glxtrsrcfont    Maintained for backward-compatibility.
7132 \newcommand*{\glxtrsrcfont}[1]{\textsc{#1}}

\glsabrvsrcfont  Added for consistent naming.
7133 \newcommand*{\glsabrvsrcfont}{\glxtrsrcfont}

\glxtrfirstsrcfont  Maintained for backward-compatibility.
7134 \newcommand*{\glxtrfirstsrcfont}[1]{\glsabrvsrcfont{#1}}

\glxtrfirstabrvsrcfont  Added for consistent naming.
7135 \newcommand*{\glxtrfirstabrvsrcfont}{\glxtrfirstsrcfont}

```

and for the default short form suffix:

```

\glxtrscsuffix
7136 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabrvpluralsuffix}}

long-short-sc
7137 \newabbreviationstyle{long-short-sc}%
7138 {%
7139   \renewcommand*{\CustomAbbreviationFields}{%
7140     name={\protect\glsabrvsrcfont{\the\glsshorttok}},
7141     sort={\the\glsshorttok},
7142     first={\protect\glxtrfirstlongdefaultfont{\the\glslongtok}%
7143       \protect\glxtrfullsep{\the\glslabeltok}}%
7144     \glxtrparen{\protect\glxtrfirstabrvsrcfont{\the\glsshorttok}}},%
7145     firstplural={\protect\glxtrfirstlongdefaultfont{\the\glslongtok}%
7146       \protect\glxtrfullsep{\the\glslabeltok}}%
7147     \glxtrparen{\protect\glxtrfirstabrvsrcfont{\the\glsshorttok}}},%
7148     plural={\protect\glsabrvsrcfont{\the\glsshorttok}}},%
7149     description={\the\glslongtok}}%
7150   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7151     \glshasattribute{\the\glslabeltok}{regular}%
7152     {%
7153       \glsselattribute{\the\glslabeltok}{regular}{false}%
7154     }%
7155   }%

```

```

7156 }%
7157 }%
7158 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7159 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7160 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7161 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

7162 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7163 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7164 \renewcommand*{\glxtrfullformat}[2]{%
7165   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7166   \ifglxtrinsertinside\else##2\fi
7167   \glxtrfullsep{##1}%
7168   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7169 }%
7170 \renewcommand*{\glxtrfullplformat}[2]{%
7171   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7172   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7173   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7174 }%
7175 \renewcommand*{\Glsxtrfullformat}[2]{%
7176   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7177   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7178   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7179 }%
7180 \renewcommand*{\Glsxtrfullplformat}[2]{%
7181   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7182   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7183   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7184 }%
7185 }

```

g-short-sc-desc

```

7186 \newabbreviationstyle{long-short-sc-desc}%
7187 {%
7188   \renewcommand*{\CustomAbbreviationFields}{%
7189     name={\glxtrlongshortdescname},
7190     sort={\glxtrlongshortdescsort},%
7191     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7192       \protect\glxtrfullsep{\the\glslabeltok}%
7193       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7194     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7195       \protect\glxtrfullsep{\the\glslabeltok}%
7196       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7197     text={\protect\glsabbrvscfont{\the\glsshorttok}},%

```

```

7198 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7199 }%

```

Unset the regular attribute if it has been set.

```

7200 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7201   \glshasattribute{\the\glslabeltok}{regular}%
7202   {%
7203     \glssetattribute{\the\glslabeltok}{regular}{false}%
7204   }%
7205   {}%
7206 }%
7207 }%
7208 {%

```

As long-short-sc style:

```

7209 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7210 }

```

Now the short (long) version

```

7211 \newabbreviationstyle{short-sc-long}%
7212 {%
7213   \renewcommand*{\CustomAbbreviationFields}{%
7214     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7215     sort={\the\glsshorttok},
7216     description={\the\glslongtok},%
7217     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7218     \protect\glsxtrfullsep{\the\glslabeltok}%
7219     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7220     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7221     \protect\glsxtrfullsep{\the\glslabeltok}%
7222     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7223     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7224 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7225   \glshasattribute{\the\glslabeltok}{regular}%
7226   {%
7227     \glssetattribute{\the\glslabeltok}{regular}{false}%
7228   }%
7229   {}%
7230 }%
7231 }%
7232 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7233 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7234 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7235 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7236 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7237 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7238 \renewcommand*{\glxtrfullformat}[2]{%
7239   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7240   \ifglxtrinsertinside\else##2\fi
7241   \glxtrfullsep{##1}%
7242   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7243 }%
7244 \renewcommand*{\glxtrfullplformat}[2]{%
7245   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7246   \ifglxtrinsertinside\else##2\fi
7247   \glxtrfullsep{##1}%
7248   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7249 }%
7250 \renewcommand*{\Glsxtrfullformat}[2]{%
7251   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7252   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7253   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7254 }%
7255 \renewcommand*{\Glsxtrfullplformat}[2]{%
7256   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7257   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7258   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7259 }%
7260 }

```

As before but user provides description

```

7261 \newabbreviationstyle{short-sc-long-desc}%
7262 {%
7263   \renewcommand*{\CustomAbbreviationFields}{%
7264     name={\glxtrshortlongdescname},
7265     sort={\glxtrshortlongdescsort},
7266     first={\protect\glsfirstabbrvscfont{\the\glssshorttok}%
7267       \protect\glxtrfullsep{\the\glslabeltok}%
7268       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7269     firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}%
7270       \protect\glxtrfullsep{\the\glslabeltok}%
7271       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7272     text={\protect\glssabbrvscfont{\the\glssshorttok}},%
7273     plural={\protect\glssabbrvscfont{\the\glssshortpltok}}%
7274   }%

```

Unset the regular attribute if it has been set.

```

7275 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7276   \glshasattribute{\the\glslabeltok}{regular}%
7277   {%
7278     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7279   }%
7280   {}%
7281 }%
7282 }%
7283 {%

```

As short-sc-long style:

```
7284 \GlsXtrUseAbbrStyleFmts{short-sc-long}%  
7285 }
```

short-sc

```
7286 \newabbreviationstyle{short-sc}%  
7287 {%  
7288 \renewcommand*{\CustomAbbreviationFields}{%  
7289 name={\protect\glsabbrvscfont{\the\glsshorttok}},  
7290 sort={\the\glsshorttok},  
7291 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},  
7292 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},  
7293 text={\protect\glsabbrvscfont{\the\glsshorttok}},  
7294 plural={\protect\glsabbrvscfont{\the\glsshortpltok}},  
7295 description={\the\glslongtok}}%  
7296 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7297 \glssetattribute{\the\glslabeltok}{regular}{true}}%  
7298 }%  
7299 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7300 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
7301 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%  
7302 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%  
7303 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%  
7304 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7305 \renewcommand*{\glxtrinlinefullformat}[2]{%  
7306 \protect\glsfirstabbrvscfont{\glsaccesssshort{##1}}%  
7307 \ifglxtrininsertinside##2\fi}%  
7308 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7309 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%  
7310 }%  
7311 \renewcommand*{\glxtrinlinefullplformat}[2]{%  
7312 \protect\glsfirstabbrvscfont{\glsaccesssshortpl{##1}}%  
7313 \ifglxtrininsertinside##2\fi}%  
7314 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7315 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%  
7316 }%  
  
7317 \renewcommand*{\Glsxtrinlinefullformat}[2]{%  
7318 \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}}%  
7319 \ifglxtrininsertinside##2\fi}%  
7320 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7321 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%  
7322 }%  
7323 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%  
7324 \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}}%  
7325 \ifglxtrininsertinside##2\fi}%
```

```

7326 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7327 \glxtrparen{\glsfirslongdefaultfont{\glssaccesslongpl{##1}}}%
7328 }%

```

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.

```

7329 \renewcommand*{\glxtrfullformat}[2]{%
7330 \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7331 \ifglxtrinsertinside\else##2\fi
7332 }%
7333 \renewcommand*{\glxtrfullplformat}[2]{%
7334 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7335 \ifglxtrinsertinside\else##2\fi
7336 }%
7337 \renewcommand*{\Glsxtrfullformat}[2]{%
7338 \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7339 \ifglxtrinsertinside\else##2\fi
7340 }%
7341 \renewcommand*{\Glsxtrfullplformat}[2]{%
7342 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7343 \ifglxtrinsertinside\else##2\fi
7344 }%
7345 }

```

short-sc-nolong

```

7346 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7347 \newabbreviationstyle{short-sc-desc}{%
7348 {%
7349 \renewcommand*{\CustomAbbreviationFields}{%
7350 name={\glxtrshortdescname},
7351 sort={\the\glssshorttok},
7352 first={\protect\glsfirstabbrvscfont{\the\glssshorttok}},
7353 firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}},
7354 text={\protect\glsabbrvscfont{\the\glssshorttok}},
7355 plural={\protect\glsabbrvscfont{\the\glssshortpltok}},
7356 description={\the\glslongtok}}%
7357 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7358 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7359 }%
7360 }%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7361 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7362 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7363 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7364 \renewcommand*{\glsfirslongfont}[1]{\glsfirslongdefaultfont{##1}}%
7365 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7366 \renewcommand*{\glxtrinlinefullformat}[2]{%
7367   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7368   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7369   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7370 }%
7371 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7372   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7373   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7374   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7375 }%
7376 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7377   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7378   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7379   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7380 }%
7381 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7382   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7383   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7384   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7385 }%

```

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.

```

7386 \renewcommand*{\glxtrfullformat}[2]{%
7387   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7388   \ifglxtrinsertinside\else##2\fi
7389 }%
7390 \renewcommand*{\glxtrfullplformat}[2]{%
7391   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7392   \ifglxtrinsertinside\else##2\fi
7393 }%
7394 \renewcommand*{\Glsxtrfullformat}[2]{%
7395   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7396   \ifglxtrinsertinside\else##2\fi
7397 }%
7398 \renewcommand*{\Glsxtrfullplformat}[2]{%
7399   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7400   \ifglxtrinsertinside\else##2\fi
7401 }%
7402 }

```

-sc-nolong-desc

```

7403 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7404 \newabbreviationstyle{nolong-short-sc}%
7405 {%
7406   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%

```

```

7407 }%
7408 {%
7409   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7410   \renewcommand*{\glsxtrinlinefullformat}[2]{%
7411     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
7412       \ifglsxtrinsertinside##2\fi}%
7413     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7414     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7415   }%
7416   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7417     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
7418       \ifglsxtrinsertinside##2\fi}%
7419     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7420     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7421   }%
7422   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7423     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
7424       \ifglsxtrinsertinside##2\fi}%
7425     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7426     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7427   }%
7428   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7429     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
7430       \ifglsxtrinsertinside##2\fi}%
7431     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7432     \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7433   }%
7434 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsxtrshort`.

```

7435 \newabbreviationstyle{long-noshort-sc}%
7436 {%
7437   \renewcommand*{\CustomAbbreviationFields}{%
7438     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7439     sort={\the\glsshorttok},
7440     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7441     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7442     text={\protect\glslongdefaultfont{\the\glslongtok}},
7443     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7444     description={\the\glslongtok}%
7445   }%
7446   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7447     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7448 }%
7449 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.



```

7450 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
7451 \renewcommand*\glssabbrvfont[1]{\glssabbrvscfont{##1}}%
7452 \renewcommand*\glssfirstabbrvfont[1]{\glssfirstabbrvscfont{##1}}%
7453 \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
7454 \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7455 \renewcommand*\glxtrsubsequentfmt}[2]{%
7456   \glsslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7457   \ifglxtrininsertinside \else##2\fi
7458 }%
7459 \renewcommand*\glxtrsubsequentplfmt}[2]{%
7460   \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7461   \ifglxtrininsertinside \else##2\fi
7462 }%
7463 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
7464   \glsslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7465   \ifglxtrininsertinside \else##2\fi
7466 }%
7467 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
7468   \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7469   \ifglxtrininsertinside \else##2\fi
7470 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7471 \renewcommand*\glxtrininlinefullformat}[2]{%
7472   \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7473   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7474   \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshort{##1}}}%
7475 }%
7476 \renewcommand*\glxtrininlinefullplformat}[2]{%
7477   \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7478   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7479   \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshortpl{##1}}}%
7480 }%
7481 \renewcommand*\Glsxtrininlinefullformat}[2]{%
7482   \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7483   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7484   \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshort{##1}}}%
7485 }%
7486 \renewcommand*\Glsxtrininlinefullplformat}[2]{%
7487   \glssfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7488   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7489   \glxtrparen{\protect\glssfirstabbrvscfont{\glssaccessshortpl{##1}}}%
7490 }%

```

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.

```

7491 \renewcommand*\glxtrfullformat}[2]{%
7492   \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7493   \ifglxtrininsertinside\else##2\fi

```

```

7494 }%
7495 \renewcommand*{\glxtrfullplformat}[2]{%
7496   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7497   \ifglxtrininsertinside\else##2\fi
7498 }%
7499 \renewcommand*{\Glsxtrfullformat}[2]{%
7500   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7501   \ifglxtrininsertinside\else##2\fi
7502 }%
7503 \renewcommand*{\Glsxtrfullplformat}[2]{%
7504   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7505   \ifglxtrininsertinside\else##2\fi
7506 }%
7507 }

```

long-sc Backward compatibility:

```

7508 \@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.

```

7509 \newabbreviationstyle{long-noshort-sc-desc}%
7510 {%
7511   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7512 }%
7513 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7514 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7515 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
7516 \renewcommand*{\glslfirstabbrvfont}[1]{\glslfirstabbrvscfont{##1}}%
7517 \renewcommand*{\glslfirstlongfont}[1]{\glslfirstlongdefaultfont{##1}}%
7518 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7519 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7520   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7521   \ifglxtrininsertinside \else##2\fi
7522 }%
7523 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7524   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7525   \ifglxtrininsertinside \else##2\fi
7526 }%
7527 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7528   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7529   \ifglxtrininsertinside \else##2\fi
7530 }%
7531 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7532   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7533   \ifglxtrininsertinside \else##2\fi
7534 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7535 \renewcommand*{\glxtrinlinefullformat}[2]{%
7536   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7537   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7538   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7539 }%
7540 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7541   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7542   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7543   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7544 }%
7545 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7546   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7547   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7548   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7549 }%
7550 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7551   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7552   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7553   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7554 }%

```

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.

```

7555 \renewcommand*{\glxtrfullformat}[2]{%
7556   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7557   \ifglxtrininsertinside\else##2\fi
7558 }%
7559 \renewcommand*{\glxtrfullplformat}[2]{%
7560   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7561   \ifglxtrininsertinside\else##2\fi
7562 }%
7563 \renewcommand*{\Glsxtrfullformat}[2]{%
7564   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7565   \ifglxtrininsertinside\else##2\fi
7566 }%
7567 \renewcommand*{\Glsxtrfullplformat}[2]{%
7568   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7569   \ifglxtrininsertinside\else##2\fi
7570 }%
7571 }

```

long-desc-sc Backward compatibility:

```

7572 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7573 \newabbreviationstyle{short-sc-footnote}%
7574 {%
7575   \renewcommand*{\CustomAbbreviationFields}{%

```

```

7576 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7577 sort={\the\glsshorttok},
7578 description={\the\glslongtok},%
7579 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7580 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7581 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7582 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7583 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7584 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7585 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.

```

7586 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7587 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7588 \glsattribute{\the\glslabeltok}{regular}%
7589 {%
7590 \glssetattribute{\the\glslabeltok}{regular}{false}%
7591 }%
7592 {}%
7593 }%
7594 }%
7595 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7596 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7597 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7598 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7599 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7600 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7601 \renewcommand*{\glsxtrfullformat}[2]{%
7602 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7603 \ifglsxtrininsertinside\else##2\fi
7604 \protect\glsxtrabbrvfootnote{##1}%
7605 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7606 }%
7607 \renewcommand*{\glsxtrfullplformat}[2]{%
7608 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7609 \ifglsxtrininsertinside\else##2\fi
7610 \protect\glsxtrabbrvfootnote{##1}%
7611 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7612 }%
7613 \renewcommand*{\Glsxtrfullformat}[2]{%
7614 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7615 \ifglsxtrininsertinside\else##2\fi
7616 \protect\glsxtrabbrvfootnote{##1}%
7617 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7618 }%
7619 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

7620 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7621 \ifglxtrinsertinside\else##2\fi
7622 \protect\glxtrabbrvfootnote{##1}%
7623 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7624 }%

```

The first use full form and the inline full form use the short (long) style.

```

7625 \renewcommand*{\glxtrinlinefullformat}[2]{%
7626 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7627 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7628 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7629 }%
7630 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7631 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7632 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7633 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7634 }%
7635 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7636 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7637 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7638 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7639 }%
7640 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7641 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7642 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7643 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7644 }%
7645 }

```

footnote-sc Backward compatibility:

```

7646 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

7647 \newabbreviationstyle{short-sc-postfootnote}%
7648 {%
7649 \renewcommand*{\CustomAbbreviationFields}{%
7650 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7651 sort={\the\glsshorttok},
7652 description={\the\glslongtok},%
7653 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7654 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7655 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.

```

7656 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7657 \csdef{glxtrpostlink\glscategorylabel}{%
7658 \glxtrifwasfirstuse
7659 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7660      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}}%
7661      {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
7662      }%
7663      {}%
7664      }%
7665      \glshasattribute{\the\glslabeltok}{regular}%
7666      {%
7667      \glissetattribute{\the\glslabeltok}{regular}{false}%
7668      }%
7669      {}%
7670      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7671 \renewcommand*{\glxtrsetupfulldefs}{%
7672   \let\glxtrifwasfirstuse\@secondoftwo
7673 }%
7674 }%
7675 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7676 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7677 \renewcommand*{\glxtrabbrvfont}[1]{\glxtrabbrvscfont{##1}}%
7678 \renewcommand*{\glxtrfirstabbrvfont}[1]{\glxtrfirstabbrvscfont{##1}}%
7679 \renewcommand*{\glxtrfirstlongfont}[1]{\glxtrfirstlongfootnotefont{##1}}%
7680 \renewcommand*{\glxtrlongfont}[1]{\glxtrlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7681 \renewcommand*{\glxtrfullformat}[2]{%
7682   \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7683   \ifglxtrininsertinside\else##2\fi
7684 }%
7685 \renewcommand*{\glxtrfullplformat}[2]{%
7686   \glxtrfirstabbrvscfont{\glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7687   \ifglxtrininsertinside\else##2\fi
7688 }%
7689 \renewcommand*{\Glsxtrfullformat}[2]{%
7690   \glxtrfirstabbrvscfont{\Glsxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7691   \ifglxtrininsertinside\else##2\fi
7692 }%
7693 \renewcommand*{\Glsxtrfullplformat}[2]{%
7694   \glxtrfirstabbrvscfont{\Glsxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7695   \ifglxtrininsertinside\else##2\fi
7696 }%

```

The first use full form and the inline full form use the short (long) style.

```

7697 \renewcommand*{\glxtrininlinefullformat}[2]{%
7698   \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7699   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7700 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7701 }%
7702 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7703 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7704 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7705 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7706 }%
7707 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7708 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7709 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7710 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7711 }%
7712 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7713 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7714 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7715 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7716 }%
7717 }

```

postfootnote-sc Backward compatibility:

```

7718 \@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.

```

7719 \newcommand*{\glsxtrsmfont}[1]{\textsmaller{#1}}

```

\glsabbrvsmfont Added for consistent naming.

```

7720 \newcommand*{\glsabbrvsmfont}{\glsxtrsmfont}

```

sxtrfirstsmfont Maintained for backward compatibility.

```

7721 \newcommand*{\glsxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

```

irstabbrvsmfont Added for consistent naming.

```

7722 \newcommand*{\glsfirstabbrvsmfont}{\glsxtrfirstsmfont}

```

and for the default short form suffix:

\glsxtrsmsuffix

```

7723 \newcommand*{\glsxtrsmsuffix}{\glsxtrabbrvpluralsuffix}

```

long-short-sm

```

7724 \newabbreviationstyle{long-short-sm}%

```

```

7725 {%

```

```

7726 \renewcommand*{\CustomAbbreviationFields}{%

```

```

7727   name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7728   sort={\the\glsshorttok},
7729   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7730     \protect\glsxtrfullsep{\the\glslabeltok}%
7731     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7732   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7733     \protect\glsxtrfullsep{\the\glslabeltok}%
7734     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7735   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
7736   description={\the\glslongtok}}%
7737 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7738   \glsattribute{\the\glslabeltok}{regular}%
7739   {%
7740     \glssetattribute{\the\glslabeltok}{regular}{false}%
7741   }}%
7742 {}%
7743 }%
7744 }%
7745 {%
7746 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7747 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7748 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%

```

Use the default long fonts.

```

7749 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7750 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7751 \renewcommand*\glsxtrfullformat[2]{%
7752   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7753   \ifglsxtrininsertinside\else##2\fi
7754   \glsxtrfullsep{##1}%
7755   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7756 }%
7757 \renewcommand*\glsxtrfullplformat[2]{%
7758   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7759   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7760   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7761 }%
7762 \renewcommand*\Glsxtrfullformat[2]{%
7763   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7764   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7765   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7766 }%
7767 \renewcommand*\Glsxtrfullplformat[2]{%
7768   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7769   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7770   \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7771 }%
7772 }

```



g-short-sm-desc

```
7773 \newabbreviationstyle{long-short-sm-desc}%
7774 {%
7775   \renewcommand*{\CustomAbbreviationFields}{%
7776     name={\glxtrlongshortdescname},
7777     sort={\glxtrlongshortdescsort},%
7778     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7779       \protect\glxtrfullsep{\the\glslabeltok}%
7780       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7781     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7782       \protect\glxtrfullsep{\the\glslabeltok}%
7783       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7784     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7785     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7786   }%
```

Unset the regular attribute if it has been set.

```
7787 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7788   \glshasattribute{\the\glslabeltok}{regular}%
7789   {%
7790     \glissetattribute{\the\glslabeltok}{regular}{false}%
7791   }%
7792   {}%
7793 }%
7794 }%
7795 {%
```

As long-short-sm style:

```
7796 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
7797 }
```

short-sm-long Now the short (long) version

```
7798 \newabbreviationstyle{short-sm-long}%
7799 {%
7800   \renewcommand*{\CustomAbbreviationFields}{%
7801     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7802     sort={\the\glsshorttok},
7803     description={\the\glslongtok},%
7804     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7805       \protect\glxtrfullsep{\the\glslabeltok}%
7806       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7807     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7808       \protect\glxtrfullsep{\the\glslabeltok}%
7809       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7810     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7811   }%
```

Unset the regular attribute if it has been set.

```
7811 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7812   \glshasattribute{\the\glslabeltok}{regular}%
7813   {%
```

```

7814 \glsssetattribute{\the\glslabeltok}{regular}{false}%
7815 }%
7816 {}%
7817 }%
7818 }%
7819 {%
7820 \renewcommand*\glssabbrvfont[1]{\glssabbrvsmfont{##1}}%
7821 \renewcommand*\glssfirstabbrvfont[1]{\glssfirstabbrvsmfont{##1}}%
7822 \renewcommand*\glssabbrvpluralsuffix{\protect\glssxtrsmsuffix}%
7823 \renewcommand*\glssfirstlongfont[1]{\glssfirstlongdefaultfont{##1}}%
7824 \renewcommand*\glsslongfont[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7825 \renewcommand*\glssxtrfullformat[2]{%
7826 \glssfirstabbrvsmfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
7827 \ifglssxtrininsertinside\else##2\fi
7828 \glssxtrfullsep{##1}%
7829 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
7830 }%
7831 \renewcommand*\glssxtrfullplformat[2]{%
7832 \glssfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
7833 \ifglssxtrininsertinside\else##2\fi
7834 \glssxtrfullsep{##1}%
7835 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7836 }%
7837 \renewcommand*\glssxtrfullformat[2]{%
7838 \glssfirstabbrvsmfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
7839 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
7840 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
7841 }%
7842 \renewcommand*\glssxtrfullplformat[2]{%
7843 \glssfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
7844 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
7845 \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7846 }%
7847 }

```

rt-sm-long-desc As before but user provides description

```

7848 \newabbreviationstyle{short-sm-long-desc}%
7849 {%
7850 \renewcommand*\CustomAbbreviationFields{%
7851 name={\glssxtrshortlongdescname},
7852 sort={\glssxtrshortlongdescsort},
7853 first={\protect\glssfirstabbrvsmfont{\the\glssshorttok}%
7854 \protect\glssxtrfullsep{\the\glslabeltok}%
7855 \glssxtrparen{\protect\glssfirstlongdefaultfont{\the\glsslongtok}}},%
7856 firstplural={\protect\glssfirstabbrvsmfont{\the\glssshortpltok}%
7857 \protect\glssxtrfullsep{\the\glslabeltok}%
7858 \glssxtrparen{\protect\glssfirstlongdefaultfont{\the\glsslongpltok}}},%
7859 text={\protect\glssabbrvsmfont{\the\glssshorttok}},%

```

```

7860 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
7861 }%

```

Unset the regular attribute if it has been set.

```

7862 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7863   \glshasattribute{\the\glslabeltok}{regular}%
7864   {%
7865     \glissetattribute{\the\glslabeltok}{regular}{false}%
7866   }%
7867   {}%
7868 }%
7869 }%
7870 {%

```

As short-sm-long style:

```

7871 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
7872 }

```

short-sm

```

7873 \newabbreviationstyle{short-sm}%
7874 {%
7875   \renewcommand*{\CustomAbbreviationFields}{%
7876     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7877     sort={\the\glsshorttok},
7878     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
7879     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
7880     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
7881     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
7882     description={\the\glslongtok}}%
7883   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7884     \glissetattribute{\the\glslabeltok}{regular}{true}}%
7885 }%
7886 {%
7887   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7888   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7889   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
7890   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7891   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7892 \renewcommand*{\glxtrinlinefullformat}[2]{%
7893   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
7894   \ifglxtrininsertinside##2\fi}%
7895   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7896   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7897 }%
7898 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7899   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
7900   \ifglxtrininsertinside##2\fi}%
7901   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7902 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
7903 }%

7904 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7905 \protect\glfirstabbrvsmfont{\Glaccessshort{##1}%
7906 \ifglxtrinsertinside##2\fi}%
7907 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7908 \glstrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
7909 }%

7910 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7911 \protect\glfirstabbrvsmfont{\Glaccessshortpl{##1}%
7912 \ifglxtrinsertinside##2\fi}%
7913 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7914 \glstrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
7915 }%

```

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.

```

7916 \renewcommand*{\glxtrfullformat}[2]{%
7917 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7918 \ifglxtrinsertinside\else##2\fi
7919 }%

7920 \renewcommand*{\glxtrfullplformat}[2]{%
7921 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7922 \ifglxtrinsertinside\else##2\fi
7923 }%

7924 \renewcommand*{\Glsxtrfullformat}[2]{%
7925 \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7926 \ifglxtrinsertinside\else##2\fi
7927 }%

7928 \renewcommand*{\Glsxtrfullplformat}[2]{%
7929 \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7930 \ifglxtrinsertinside\else##2\fi
7931 }%
7932 }

```

short-sm-nolong

```

7933 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

7934 \newabbreviationstyle{short-sm-desc}%
7935 {%
7936 \renewcommand*{\CustomAbbreviationFields}{%
7937 name={\glxtrshortdescname},
7938 sort={\the\glsshorttok},
7939 first={\protect\glfirstabbrvsmfont{\the\glsshorttok}},
7940 firstplural={\protect\glfirstabbrvsmfont{\the\glsshortpltok}},
7941 text={\protect\glabbrvsmfont{\the\glsshorttok}},
7942 plural={\protect\glabbrvsmfont{\the\glsshortpltok}},

```

```

7943   description={\the\glslongtok}}%
7944   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7945     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7946 }%
7947 {%
7948   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7949   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7950   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
7951   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7952   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7953   \renewcommand*{\glsxtrinlinefullformat}[2]{%
7954     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7955     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7956     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7957 }%
7958   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7959     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7960     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7961     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7962 }%
7963   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7964     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7965     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7966     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7967 }%
7968   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7969     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7970     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7971     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7972 }%

```

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.

```

7973   \renewcommand*{\glsxtrfullformat}[2]{%
7974     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7975     \ifglsxtrininsertinside\else##2\fi
7976 }%
7977   \renewcommand*{\glsxtrfullplformat}[2]{%
7978     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7979     \ifglsxtrininsertinside\else##2\fi
7980 }%
7981   \renewcommand*{\Glsxtrfullformat}[2]{%
7982     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7983     \ifglsxtrininsertinside\else##2\fi
7984 }%
7985   \renewcommand*{\Glsxtrfullplformat}[2]{%
7986     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7987     \ifglsxtrininsertinside\else##2\fi

```

```
7988 }%
7989 }
```

-sm-nolong-desc

```
7990 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}
```

nolong-short-sm

```
7991 \newabbreviationstyle{nolong-short-sm}%
7992 {%
7993   \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
7994 }%
7995 {%
7996   \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%
```

The inline full form displays the long form followed by the short form in parentheses.

```
7997 \renewcommand*{\glxtrinlinefullformat}[2]{%
7998   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
7999     \ifglxtrinsertinside##2\fi}%
8000   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8001   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8002 }%
8003 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8004   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8005     \ifglxtrinsertinside##2\fi}%
8006   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8007   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8008 }%
8009 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8010   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8011     \ifglxtrinsertinside##2\fi}%
8012   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8013   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8014 }%
8015 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8016   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8017     \ifglxtrinsertinside##2\fi}%
8018   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8019   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8020 }%
8021 }
```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```
8022 \newabbreviationstyle{long-noshort-sm}%
8023 {%
8024   \renewcommand*{\CustomAbbreviationFields}{%
8025     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8026     sort={\the\glsshorttok},
8027     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
```

```

8028   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8029   text={\protect\glslongdefaultfont{\the\glslongtok}},
8030   plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8031   description={\the\glslongtok}%
8032 }%
8033 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8034   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8035 }%
8036 {%
8037   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8038   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8039   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8040   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8041   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8042   \renewcommand*\glsxtrsubsequentfmt}[2]{%
8043     \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8044     \ifglsxtrininsertinside \else##2\fi
8045   }%
8046   \renewcommand*\glsxtrsubsequentplfmt}[2]{%
8047     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8048     \ifglsxtrininsertinside \else##2\fi
8049   }%
8050   \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8051     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8052     \ifglsxtrininsertinside \else##2\fi
8053   }%
8054   \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8055     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8056     \ifglsxtrininsertinside \else##2\fi
8057   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8058   \renewcommand*\glsxtrinlinefullformat}[2]{%
8059     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8060     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8061     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8062   }%
8063   \renewcommand*\glsxtrinlinefullplformat}[2]{%
8064     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8065     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8066     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8067   }%
8068   \renewcommand*\Glsxtrinlinefullformat}[2]{%
8069     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8070     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8071     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8072   }%
8073   \renewcommand*\Glsxtrinlinefullplformat}[2]{%

```

```

8074 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8075 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8076 \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8077 }%

```

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.

```

8078 \renewcommand*{\glxtrfullformat}[2]{%
8079 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8080 \ifglxtrinsertinside\else##2\fi
8081 }%
8082 \renewcommand*{\glxtrfullplformat}[2]{%
8083 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8084 \ifglxtrinsertinside\else##2\fi
8085 }%
8086 \renewcommand*{\Glsxtrfullformat}[2]{%
8087 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8088 \ifglxtrinsertinside\else##2\fi
8089 }%
8090 \renewcommand*{\Glsxtrfullplformat}[2]{%
8091 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8092 \ifglxtrinsertinside\else##2\fi
8093 }%
8094 }

```

long-sm Backward compatibility:

```

8095 \@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`.

```

8096 \newabbreviationstyle{long-noshort-sm-desc}%
8097 {%
8098 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8099 }%
8100 {%
8101 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8102 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8103 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8104 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8105 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8106 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8107 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8108 \ifglxtrinsertinside \else##2\fi
8109 }%
8110 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8111 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8112 \ifglxtrinsertinside \else##2\fi
8113 }%

```



```

8114 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8115   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8116   \ifglxtrinsertinside \else##2\fi
8117 }%
8118 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8119   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8120   \ifglxtrinsertinside \else##2\fi
8121 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8122 \renewcommand*{\glxtrinlinefullformat}[2]{%
8123   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8124   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8125   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8126 }%
8127 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8128   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8129   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8130   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8131 }%
8132 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8133   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8134   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8135   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8136 }%
8137 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8138   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8139   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8140   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8141 }%

```

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.

```

8142 \renewcommand*{\glxtrfullformat}[2]{%
8143   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8144   \ifglxtrinsertinside\else##2\fi
8145 }%
8146 \renewcommand*{\glxtrfullplformat}[2]{%
8147   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8148   \ifglxtrinsertinside\else##2\fi
8149 }%
8150 \renewcommand*{\Glsxtrfullformat}[2]{%
8151   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8152   \ifglxtrinsertinside\else##2\fi
8153 }%
8154 \renewcommand*{\Glsxtrfullplformat}[2]{%
8155   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8156   \ifglxtrinsertinside\else##2\fi
8157 }%
8158 }

```

long-desc-sm    Backward compatibility:

```
8159 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

ort-sm-footnote

```
8160 \newabbreviationstyle{short-sm-footnote}%
8161 {%
8162   \renewcommand*{\CustomAbbreviationFields}{%
8163     name={\protect\glssabbrvsmfont{\the\glssshorttok}},
8164     sort={\the\glssshorttok},
8165     description={\the\glslongtok},%
8166     first={\protect\glsfirstabbrvsmfont{\the\glssshorttok}%
8167       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8168       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8169     firstplural={\protect\glsfirstabbrvsmfont{\the\glssshortpltok}%
8170       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8171       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8172     plural={\protect\glssabbrvsmfont{\the\glssshortpltok}}}%
8173   }
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
8173   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8174     \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8175     \glshasattribute{\the\glslabeltok}{regular}%
8176     {%
8177       \glsssetattribute{\the\glslabeltok}{regular}{false}%
8178     }%
8179   }%
8180 }%
8181 }%
8182 {%
8183   \renewcommand*\glssabbrvfont[1]{\glssabbrvsmfont{##1}}%
8184   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8185   \renewcommand*\glssabbrvplfont[1]{\glssabbrvsmfont{##1}}%
8186   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8187   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
8188 }
```

The full format displays the short form followed by the long form as a footnote.

```
8188 \renewcommand*\glxtrfullformat[2]{%
8189   \glsfirstabbrvsmfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8190   \ifglxtrininsertinside\else##2\fi
8191   \protect\glxtrabbrvfootnote{##1}%
8192   {\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
8193 }%
8194 \renewcommand*\glxtrfullplformat[2]{%
8195   \glsfirstabbrvsmfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8196   \ifglxtrininsertinside\else##2\fi
8197   \protect\glxtrabbrvfootnote{##1}%
8198   {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
8199 }%
8200 \renewcommand*\GlsXtrFullFormat[2]{%
8201   \glxtrfullformat{##1}{##2}%
8202 }
```

```

8201 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8202 \ifglxtrinsertinside\else##2\fi
8203 \protect\glxtrabbrvfootnote{##1}%
8204 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8205 }%
8206 \renewcommand*{\Glsxtrfullplformat}[2]{%
8207 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8208 \ifglxtrinsertinside\else##2\fi
8209 \protect\glxtrabbrvfootnote{##1}%
8210 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8211 }%

```

The first use full form and the inline full form use the short (long) style.

```

8212 \renewcommand*{\glxtrinlinefullformat}[2]{%
8213 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8214 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8215 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8216 }%
8217 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8218 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8219 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8220 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8221 }%
8222 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8223 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8224 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8225 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8226 }%
8227 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8228 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8229 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8230 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8231 }%
8232 }

```

footnote-sm Backward compatibility:

```

8233 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8234 \newabbreviationstyle{short-sm-postfootnote}%
8235 {%
8236 \renewcommand*{\CustomAbbreviationFields}{%
8237 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8238 sort={\the\glsshorttok},
8239 description={\the\glslongtok},%
8240 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8241 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8242 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.

```
8243 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8244 \csdef{glxstrpostlink\glscategorylabel}{%
8245 \glxstrifwasfirstuse
8246 }
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
8247 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8248 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
8249 }%
8250 {}%
8251 }%
8252 \glshasattribute{\the\glslabeltok}{regular}%
8253 {%
8254 \glissetattribute{\the\glslabeltok}{regular}{false}%
8255 }%
8256 {}%
8257 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
8258 \renewcommand*{\glxtrsetupfulldefs}{%
8259 \let\glxstrifwasfirstuse\@secondoftwo
8260 }%
8261 }%
8262 {%
8263 \renewcommand*{\glxtrabbrvfont}[1]{\glxtrabbrvsmfont{##1}}%
8264 \renewcommand*{\glxtrfirstabbrvfont}[1]{\glxtrfirstabbrvsmfont{##1}}%
8265 \renewcommand*{\glxtrabbrvpluralsuffix}{\protect\glxtrrmsuffix}%
8266 \renewcommand*{\glxtrfirstlongfont}[1]{\glxtrfirstlongfootnotefont{##1}}%
8267 \renewcommand*{\glxtrlongfont}[1]{\glxtrlongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
8268 \renewcommand*{\glxtrfullformat}[2]{%
8269 \glxtrfirstabbrvsmfont{\glxtraccessshort{##1}\ifglxtrinsertinside##2\fi}%
8270 \ifglxtrinsertinside\else##2\fi
8271 }%
8272 \renewcommand*{\glxtrfullplformat}[2]{%
8273 \glxtrfirstabbrvsmfont{\glxtraccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8274 \ifglxtrinsertinside\else##2\fi
8275 }%
8276 \renewcommand*{\Glsxtrfullformat}[2]{%
8277 \glxtrfirstabbrvsmfont{\Glsxtraccessshort{##1}\ifglxtrinsertinside##2\fi}%
8278 \ifglxtrinsertinside\else##2\fi
8279 }%
8280 \renewcommand*{\Glsxtrfullplformat}[2]{%
8281 \glxtrfirstabbrvsmfont{\Glsxtraccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8282 \ifglxtrinsertinside\else##2\fi
```

8283 }%

The first use full form and the inline full form use the short (long) style.

```
8284 \renewcommand*{\glxstrinlinefullformat}[2]{%
8285   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
8286   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
8287   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8288 }%
8289 \renewcommand*{\glxstrinlinefullplformat}[2]{%
8290   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
8291   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
8292   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8293 }%
8294 \renewcommand*{\Glsxstrinlinefullformat}[2]{%
8295   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
8296   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
8297   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8298 }%
8299 \renewcommand*{\Glsxstrinlinefullplformat}[2]{%
8300   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
8301   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
8302   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8303 }%
8304 }
```

postfootnote-sm Backward compatibility:

```
8305 \@glxstr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}
```

## 1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```
8306 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%
```

`\glfirstabbrvemfont`

```
8307 \newcommand*{\glfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%
```

The default short form suffix:

`\glxtremsuffix`

```
8308 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}
```

`\glfirstlongemfont` Only used by the “long-em” styles.

```
8309 \newcommand*{\glfirstlongemfont}[1]{\glslongemfont{##1}}%
```

`\glslongemfont` Only used by the “long-em” styles.

```
8310 \newcommand*{\glslongemfont}[1]{\emph{##1}}%
```

long-short-em The long form is just set in the default long font.

```

8311 \newabbreviationstyle{long-short-em}%
8312 {%
8313   \renewcommand*{\CustomAbbreviationFields}{%
8314     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8315     sort={\the\glsshorttok},
8316     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8317       \protect\glsxtrfullsep{\the\glslabeltok}%
8318       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8319     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8320       \protect\glsxtrfullsep{\the\glslabeltok}%
8321       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8322     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8323     description={\the\glslongtok}}%
8324   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8325     \glsattribute{\the\glslabeltok}{regular}%
8326     {%
8327       \glssetattribute{\the\glslabeltok}{regular}{false}%
8328     }%
8329   }%
8330 }%
8331 }%
8332 {%
8333   \renewcommand*{\glsabbrvfont[1]{\glsabbrvemfont{##1}}}%
8334   \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}}%
8335   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```

8336 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8337 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8338 \renewcommand*{\glsxtrfullformat}[2]{%
8339   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8340   \ifglsxtrininsertinside\else##2\fi
8341   \glsxtrfullsep{##1}%
8342   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8343 }%
8344 \renewcommand*{\glsxtrfullplformat}[2]{%
8345   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8346   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8347   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8348 }%
8349 \renewcommand*{\Glsxtrfullformat}[2]{%
8350   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8351   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8352   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8353 }%
8354 \renewcommand*{\Glsxtrfullplformat}[2]{%
8355   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

8356 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8357 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8358 }%
8359 }

```

g-short-em-desc

```

8360 \newabbreviationstyle{long-short-em-desc}%
8361 {%
8362 \renewcommand*{\CustomAbbreviationFields}{%
8363 name={\glxtrlongshortdescname},
8364 sort={\glxtrlongshortdescsort},%
8365 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8366 \protect\glxtrfullsep{\the\glslabeltok}%
8367 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8368 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8369 \protect\glxtrfullsep{\the\glslabeltok}%
8370 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8371 text={\protect\glssabbrvemfont{\the\glsshorttok}},%
8372 plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
8373 }%

```

Unset the regular attribute if it has been set.

```

8374 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8375 \glshasattribute{\the\glslabeltok}{regular}%
8376 {%
8377 \glissetattribute{\the\glslabeltok}{regular}{false}%
8378 }%
8379 {}%
8380 }%
8381 }%
8382 {%

```

As long-short-em style:

```

8383 \GlsXtrUseAbbrStyleFmts{long-short-em}%
8384 }

```

ong-em-short-em

```

8385 \newabbreviationstyle{long-em-short-em}%
8386 {%
8387 \renewcommand*{\CustomAbbreviationFields}{%
8388 name={\protect\glssabbrvemfont{\the\glsshorttok}},
8389 sort={\the\glsshorttok},
8390 first={\protect\glsfirstlongemfont{\the\glslongtok}%
8391 \protect\glxtrfullsep{\the\glslabeltok}%
8392 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8393 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8394 \protect\glxtrfullsep{\the\glslabeltok}%
8395 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

```

```

8396 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8397 description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8398 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8399   \glsattribute{\the\glslabeltok}{regular}%
8400   {%
8401     \glssetattribute{\the\glslabeltok}{regular}{false}%
8402   }%
8403   {}}%
8404 }%
8405 }%
8406 {%
8407   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8408   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8409   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8410   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8411   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8412 \renewcommand*{\glsxtrfullformat}[2]{%
8413   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8414   \ifglsxtrininsertinside\else##2\fi
8415   \glsxtrfullsep{##1}%
8416   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8417 }%
8418 \renewcommand*{\glsxtrfullplformat}[2]{%
8419   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8420   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8421   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8422 }%
8423 \renewcommand*{\Glsxtrfullformat}[2]{%
8424   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8425   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8426   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8427 }%
8428 \renewcommand*{\Glsxtrfullplformat}[2]{%
8429   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8430   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8431   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8432 }%
8433 }

```

m-short-em-desc

```

8434 \newabbreviationstyle{long-em-short-em-desc}%
8435 {%
8436   \renewcommand*{\CustomAbbreviationFields}{%
8437     name={\glsxtrlongshortdescname},
8438     sort={\glsxtrlongshortdescsort},%
8439     first={\protect\glsfirstlongemfont{\the\glslongtok}}%

```



```

8440 \protect\glxtrfullsep{\the\glslabeltok}%
8441 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8442 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8443 \protect\glxtrfullsep{\the\glslabeltok}%
8444 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8445 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
8446 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8447 }%

```

Unset the regular attribute if it has been set.

```

8448 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8449 \glshasattribute{\the\glslabeltok}{regular}%
8450 {%
8451 \glissetattribute{\the\glslabeltok}{regular}{false}%
8452 }%
8453 {}%
8454 }%
8455 }%
8456 {%
8457 \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8458 }

```

short-em-long Now the short (long) version

```

8459 \newabbreviationstyle{short-em-long}%
8460 {%
8461 \renewcommand*{\CustomAbbreviationFields}{%
8462 name={\protect\glsabbrvemfont{\the\glsshorttok}},
8463 sort={\the\glsshorttok},
8464 description={\the\glslongtok},%
8465 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8466 \protect\glxtrfullsep{\the\glslabeltok}%
8467 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8468 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8469 \protect\glxtrfullsep{\the\glslabeltok}%
8470 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8471 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8472 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8473 \glshasattribute{\the\glslabeltok}{regular}%
8474 {%
8475 \glissetattribute{\the\glslabeltok}{regular}{false}%
8476 }%
8477 {}%
8478 }%
8479 }%
8480 {%

```

Mostly as short-long style:

```

8481 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%

```

```

8482 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8483 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8484 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8485 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8486 \renewcommand*\glsxtrfullformat[2]{%
8487   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8488   \ifglsxtrininsertinside\else##2\fi
8489   \glsxtrfullsep{##1}%
8490   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8491 }%
8492 \renewcommand*\glsxtrfullplformat[2]{%
8493   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8494   \ifglsxtrininsertinside\else##2\fi
8495   \glsxtrfullsep{##1}%
8496   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8497 }%
8498 \renewcommand*\Glsxtrfullformat[2]{%
8499   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8500   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8501   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8502 }%
8503 \renewcommand*\Glsxtrfullplformat[2]{%
8504   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8505   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8506   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8507 }%
8508 }

```

rt-em-long-desc As before but user provides description

```

8509 \newabbreviationstyle{short-em-long-desc}%
8510 {%
8511   \renewcommand*\CustomAbbreviationFields{%
8512     name={\glsxtrshortlongdescname},
8513     sort={\glsxtrshortlongdescsort},
8514     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8515       \protect\glsxtrfullsep{\the\glslabeltok}%
8516       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8517     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8518       \protect\glsxtrfullsep{\the\glslabeltok}%
8519       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8520     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8521     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8522   }%

```

Unset the regular attribute if it has been set.

```

8523 \renewcommand*\GlsXtrPostNewAbbreviation{%
8524   \glshasattribute{\the\glslabeltok}{regular}%
8525   {%

```

```

8526     \glsetattribute{\the\glslabeltok}{regular}{false}%
8527   }%
8528   {}%
8529 }%
8530}%
8531{%
8532  \GlsXtrUseAbbrStyleFmts{short-em-long}%
8533 }

```

hort-em-long-em

```

8534 \newabbreviationstyle{short-em-long-em}%
8535 {%
    \glslongemfont is used in the description since \glsgdesc doesn't set the style.
8536  \renewcommand*{\CustomAbbreviationFields}{%
8537    name={\protect\glsabbrvemfont{\the\glsshorttok}},
8538    sort={\the\glsshorttok},
8539    description={\protect\glslongemfont{\the\glslongtok}},%
8540    first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8541    \protect\glsxtrfullsep{\the\glslabeltok}%
8542    \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8543    firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8544    \protect\glsxtrfullsep{\the\glslabeltok}%
8545    \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8546    plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

    Unset the regular attribute if it has been set.
8547  \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8548    \glshasattribute{\the\glslabeltok}{regular}%
8549    {%
8550      \glsetattribute{\the\glslabeltok}{regular}{false}%
8551    }%
8552    {}%
8553  }%
8554}%
8555{%
8556  \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8557  \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8558  \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8559  \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8560  \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

    The first use full form and the inline full form are the same for this style.
8561  \renewcommand*{\glsxtrfullformat}[2]{%
8562    \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrinsetinside##2\fi}%
8563    \ifglsxtrinsetinside\else##2\fi
8564    \glsxtrfullsep{##1}%
8565    \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8566  }%
8567  \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

8568 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8569 \ifglxtrinsertinside\else##2\fi
8570 \glxtrfullsep{##1}%
8571 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8572 }%
8573 \renewcommand*{\Glsxtrfullformat}[2]{%
8574 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8575 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8576 \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8577 }%
8578 \renewcommand*{\Glsxtrfullplformat}[2]{%
8579 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8580 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8581 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8582 }%
8583 }

```

em-long-em-desc

```

8584 \newabbreviationstyle{short-em-long-em-desc}%
8585 {%
8586 \renewcommand*{\CustomAbbreviationFields}{%
8587 name={\glxtrshortlongdescname},%
8588 sort={\glxtrshortlongdescsort},%
8589 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8590 \protect\glxtrfullsep{\the\glslabeltok}%
8591 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8592 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8593 \protect\glxtrfullsep{\the\glslabeltok}%
8594 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8595 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8596 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8597 }%

```

Unset the regular attribute if it has been set.

```

8598 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8599 \glshasattribute{\the\glslabeltok}{regular}%
8600 {%
8601 \glissetattribute{\the\glslabeltok}{regular}{false}%
8602 }%
8603 {}%
8604 }%
8605 }%
8606 {%
8607 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8608 }

```

short-em

```

8609 \newabbreviationstyle{short-em}%
8610 {%
8611 \renewcommand*{\CustomAbbreviationFields}{%

```

```

8612   name={\protect\glsabbrvemfont{\the\glsshorttok}},
8613   sort={\the\glsshorttok},
8614   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8615   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8616   text={\protect\glsabbrvemfont{\the\glsshorttok}},
8617   plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8618   description={\the\glslongtok}}%
8619 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8620   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8621 }%
8622 {%
8623 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8624 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8625 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8626 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8627 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8628 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8629   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
8630   \ifglsxtrininsertinside##2\fi}%
8631 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8632 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8633 }%
8634 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8635   \protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}%
8636   \ifglsxtrininsertinside##2\fi}%
8637 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8638 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8639 }%
8640 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8641   \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}%
8642   \ifglsxtrininsertinside##2\fi}%
8643 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8644 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8645 }%
8646 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8647   \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}%
8648   \ifglsxtrininsertinside##2\fi}%
8649 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8650 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8651 }%

```

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.

```

8652 \renewcommand*{\glsxtrfullformat}[2]{%
8653   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8654   \ifglsxtrininsertinside\else##2\fi
8655 }%

```

```

8656 \renewcommand*{\glxtrfullplformat}[2]{%
8657   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8658   \ifglxtrininsertinside\else##2\fi
8659 }%
8660 \renewcommand*{\Glsxtrfullformat}[2]{%
8661   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8662   \ifglxtrininsertinside\else##2\fi
8663 }%
8664 \renewcommand*{\Glsxtrfullplformat}[2]{%
8665   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8666   \ifglxtrininsertinside\else##2\fi
8667 }%
8668 }

```

short-em-nolong

```

8669 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8670 \newabbreviationstyle{short-em-desc}%
8671 {%
8672   \renewcommand*{\CustomAbbreviationFields}{%
8673     name={\glxtrshortdescname},
8674     sort={\the\glssshorttok},
8675     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},
8676     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},
8677     text={\protect\glsabbrvemfont{\the\glssshorttok}},
8678     plural={\protect\glsabbrvemfont{\the\glssshortpltok}},
8679     description={\the\glslongtok}}%
8680 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8681   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8682 }%
8683 {%
8684   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8685   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8686   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8687   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8688   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8689 \renewcommand*{\glxtrininlinefullformat}[2]{%
8690   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8691   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8692   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
8693 }%
8694 \renewcommand*{\glxtrininlinefullplformat}[2]{%
8695   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8696   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8697   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8698 }%
8699 \renewcommand*{\Glsxtrininlinefullformat}[2]{%

```

```

8700 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8701 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8702 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8703 }%
8704 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8705 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8706 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8707 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8708 }%

```

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.

```

8709 \renewcommand*{\glxtrfullformat}[2]{%
8710 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8711 \ifglxtrinsertinside\else##2\fi
8712 }%
8713 \renewcommand*{\glxtrfullplformat}[2]{%
8714 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8715 \ifglxtrinsertinside\else##2\fi
8716 }%
8717 \renewcommand*{\Glsxtrfullformat}[2]{%
8718 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8719 \ifglxtrinsertinside\else##2\fi
8720 }%
8721 \renewcommand*{\Glsxtrfullplformat}[2]{%
8722 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8723 \ifglxtrinsertinside\else##2\fi
8724 }%
8725 }

```

-em-nolong-desc

```

8726 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8727 \newabbreviationstyle{nolong-short-em}%
8728 {%
8729 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8730 }%
8731 {%
8732 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8733 \renewcommand*{\glxtrinlinefullformat}[2]{%
8734 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8735 \ifglxtrinsertinside##2\fi}%
8736 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8737 \glxtrparen{\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8738 }%
8739 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8740 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%

```

```

8741     \ifglxtrinsertinside##2\fi}%
8742     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8743     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8744 }%
8745 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8746     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8747         \ifglxtrinsertinside##2\fi}%
8748     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8749     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8750 }%
8751 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8752     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8753         \ifglxtrinsertinside##2\fi}%
8754     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8755     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8756 }%
8757 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8758 \newabbreviationstyle{long-noshort-em}%
8759 {%
8760     \renewcommand*{\CustomAbbreviationFields}{%
8761         name={\protect\glsabbrvemfont{\the\glsshorttok}},
8762         sort={\the\glsshorttok},
8763         first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8764         firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8765         text={\protect\glslongdefaultfont{\the\glslongtok}},
8766         plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8767         description={\the\glslongtok}%
8768     }%
8769     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8770         \glssetattribute{\the\glslabeltok}{regular}{true}}%
8771 }%
8772 {%
8773     \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8774     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8775     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8776     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8777     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8778     \renewcommand*{\glxtrsubsequentfmt}[2]{%
8779         \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8780         \ifglxtrinsertinside \else##2\fi
8781     }%
8782     \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8783         \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8784         \ifglxtrinsertinside \else##2\fi
8785     }%
8786     \renewcommand*{\Glsxtrsubsequentfmt}[2]{%

```



```

8787 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8788 \ifglxtrinsertinside \else##2\fi
8789 }%
8790 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8791 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8792 \ifglxtrinsertinside \else##2\fi
8793 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8794 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8795 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8796 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8797 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8798 }%
8799 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8800 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8801 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8802 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8803 }%
8804 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8805 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8806 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8807 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
8808 }%
8809 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8810 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8811 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8812 \glsxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8813 }%

```

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.

```

8814 \renewcommand*{\glsxtrfullformat}[2]{%
8815 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8816 \ifglxtrinsertinside\else##2\fi
8817 }%
8818 \renewcommand*{\glsxtrfullplformat}[2]{%
8819 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8820 \ifglxtrinsertinside\else##2\fi
8821 }%
8822 \renewcommand*{\Glsxtrfullformat}[2]{%
8823 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8824 \ifglxtrinsertinside\else##2\fi
8825 }%
8826 \renewcommand*{\Glsxtrfullplformat}[2]{%
8827 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8828 \ifglxtrinsertinside\else##2\fi
8829 }%
8830 }

```

long-em    Backward compatibility:

```
8831 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em    The short form is explicitly invoked through commands like `\glsshort`.

```
8832 \newabbreviationstyle{long-em-noshort-em}%
8833 {%
8834   \renewcommand*{\CustomAbbreviationFields}{%
8835     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8836     sort={\the\glsshorttok},
8837     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8838     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8839     text={\protect\glslongemfont{\the\glslongtok}},
8840     plural={\protect\glslongemfont{\the\glslongpltok}},%
8841     description={\protect\glslongemfont{\the\glslongtok}}%
8842   }%
8843   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8844     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8845 }%
8846 {%
8847   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8848   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8849   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8850   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8851   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
8852 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8853   \glslongemfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
8854   \ifglxtrininsertinside \else##2\fi
8855 }%
8856 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8857   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
8858   \ifglxtrininsertinside \else##2\fi
8859 }%
8860 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8861   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
8862   \ifglxtrininsertinside \else##2\fi
8863 }%
8864 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8865   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
8866   \ifglxtrininsertinside \else##2\fi
8867 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
8868 \renewcommand*{\glxtrinlinefullformat}[2]{%
8869   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8870   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8871   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8872 }%
8873 \renewcommand*{\glxtrinlinefullplformat}[2]{%
```

```

8874 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8875 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8876 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8877 }%
8878 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8879 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8880 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8881 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8882 }%
8883 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8884 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8885 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8886 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8887 }%

```

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.

```

8888 \renewcommand*{\glxtrfullformat}[2]{%
8889 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8890 \ifglxtrinsertinside\else##2\fi
8891 }%
8892 \renewcommand*{\glxtrfullplformat}[2]{%
8893 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8894 \ifglxtrinsertinside\else##2\fi
8895 }%
8896 \renewcommand*{\Glsxtrfullformat}[2]{%
8897 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8898 \ifglxtrinsertinside\else##2\fi
8899 }%
8900 \renewcommand*{\Glsxtrfullplformat}[2]{%
8901 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8902 \ifglxtrinsertinside\else##2\fi
8903 }%
8904 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

8905 \newabbreviationstyle{long-em-noshort-em-noreg}%
8906 {%
8907 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

8908 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8909 \glshasattribute{\the\glslabeltok}{regular}%
8910 {%
8911 \glissetattribute{\the\glslabeltok}{regular}{false}%
8912 }%
8913 {}%
8914 }%
8915 }%
8916 {%

```

```

8917 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
8918 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

8919 \newabbreviationstyle{long-noshort-em-desc}%
8920 {%
8921 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8922 }%
8923 {%
8924 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
8925 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
8926 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvemfont{##1}}%
8927 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
8928 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8929 \renewcommand*{\glssxtrsubsequentfmt}[2]{%
8930 \glsslongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
8931 \ifglssxtrininsertinside \else##2\fi
8932 }%
8933 \renewcommand*{\glssxtrsubsequentplfmt}[2]{%
8934 \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
8935 \ifglssxtrininsertinside \else##2\fi
8936 }%
8937 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8938 \glsslongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
8939 \ifglssxtrininsertinside \else##2\fi
8940 }%
8941 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8942 \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
8943 \ifglssxtrininsertinside \else##2\fi
8944 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8945 \renewcommand*{\glssxtrinlinefullformat}[2]{%
8946 \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8947 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8948 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
8949 }%
8950 \renewcommand*{\glssxtrinlinefullplformat}[2]{%
8951 \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
8952 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8953 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshortpl{##1}}}%
8954 }%
8955 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8956 \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8957 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8958 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
8959 }%

```

```

8960 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8961   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8962   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8963   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8964 }%

```

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.

```

8965 \renewcommand*{\glxtrfullformat}[2]{%
8966   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8967   \ifglxtrinsertinside\else##2\fi
8968 }%
8969 \renewcommand*{\glxtrfullplformat}[2]{%
8970   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8971   \ifglxtrinsertinside\else##2\fi
8972 }%
8973 \renewcommand*{\Glsxtrfullformat}[2]{%
8974   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8975   \ifglxtrinsertinside\else##2\fi
8976 }%
8977 \renewcommand*{\Glsxtrfullplformat}[2]{%
8978   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8979   \ifglxtrinsertinside\else##2\fi
8980 }%
8981 }

```

long-desc-em Backward compatibility:

```

8982 \@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.

```

8983 \newabbreviationstyle{long-em-noshort-em-desc}%
8984 {%
8985   \renewcommand*{\CustomAbbreviationFields}{%
8986     name={\protect\protect\glslongemfont{\the\glslongtok}},
8987     sort={\the\glslongtok},
8988     first={\protect\glsfirstlongemfont{\the\glslongtok}},
8989     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8990     text={\glslongemfont{\the\glslongtok}},
8991     plural={\glslongemfont{\the\glslongpltok}}}%
8992 }%
8993 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8994   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8995 }%
8996 {%
8997   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8998   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8999   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9000   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
9001   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9002 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9003   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9004   \ifglxtrinsertinside \else##2\fi
9005 }%
9006 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9007   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9008   \ifglxtrinsertinside \else##2\fi
9009 }%
9010 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9011   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9012   \ifglxtrinsertinside \else##2\fi
9013 }%
9014 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9015   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9016   \ifglxtrinsertinside \else##2\fi
9017 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9018 \renewcommand*{\glxtrinlinefullformat}[2]{%
9019   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9020   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9021   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9022 }%
9023 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9024   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9025   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9026   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9027 }%
9028 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9029   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9030   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9031   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9032 }%
9033 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9034   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9035   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9036   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9037 }%

```

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.

```

9038 \renewcommand*{\glxtrfullformat}[2]{%
9039   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9040   \ifglxtrinsertinside\else##2\fi
9041 }%
9042 \renewcommand*{\glxtrfullplformat}[2]{%
9043   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9044   \ifglxtrinsertinside\else##2\fi
9045 }%

```

```

9046 \renewcommand*{\GlsXtrfullformat}[2]{%
9047   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9048   \ifglxtrinsertinside\else##2\fi
9049 }%
9050 \renewcommand*{\GlsXtrfullplformat}[2]{%
9051   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9052   \ifglxtrinsertinside\else##2\fi
9053 }%
9054 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9055 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9056 {%
9057   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%

```

Unset the regular attribute if it has been set.

```

9058 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9059   \glshasattribute{\the\glslabeltok}{regular}%
9060   {%
9061     \glsselattribute{\the\glslabeltok}{regular}{false}%
9062   }%
9063   {}%
9064 }%
9065 }%
9066 {%
9067   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9068 }

```

ort-em-footnote

```

9069 \newabbreviationstyle{short-em-footnote}%
9070 {%
9071   \renewcommand*{\CustomAbbreviationFields}{%
9072     name={\protect\glsabbrvemfont{\the\glsshorttok}},
9073     sort={\the\glsshorttok},
9074     description={\the\glslongtok},%
9075     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
9076       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9077       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9078     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
9079       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9080       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9081     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

9082 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9083   \glsselattribute{\the\glslabeltok}{nohyperfirst}{true}%
9084   \glshasattribute{\the\glslabeltok}{regular}%
9085   {%
9086     \glsselattribute{\the\glslabeltok}{regular}{false}%

```

```

9087 }%
9088 {}%
9089 }%
9090 }%
9091 {%
9092 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9093 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9094 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9095 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9096 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9097 \renewcommand*{\glsxtrfullformat}[2]{%
9098   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9099   \ifglsxtrininsertinside\else##2\fi
9100   \protect\glsxtrabbrvfootnote{##1}%
9101   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9102 }%
9103 \renewcommand*{\glsxtrfullplformat}[2]{%
9104   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9105   \ifglsxtrininsertinside\else##2\fi
9106   \protect\glsxtrabbrvfootnote{##1}%
9107   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9108 }%
9109 \renewcommand*{\Glsxtrfullformat}[2]{%
9110   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9111   \ifglsxtrininsertinside\else##2\fi
9112   \protect\glsxtrabbrvfootnote{##1}%
9113   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9114 }%
9115 \renewcommand*{\Glsxtrfullplformat}[2]{%
9116   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9117   \ifglsxtrininsertinside\else##2\fi
9118   \protect\glsxtrabbrvfootnote{##1}%
9119   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9120 }%

```

The first use full form and the inline full form use the short (long) style.

```

9121 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9122   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9123   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9124   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9125 }%
9126 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9127   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9128   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9129   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9130 }%
9131 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9132   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```



```

9133     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9134     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9135 }%
9136 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9137     \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9138     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9139     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9140 }%
9141 }

```

footnote-em Backward compatibility:

```

9142 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9143 \newabbreviationstyle{short-em-postfootnote}%
9144 {%
9145     \renewcommand*{\CustomAbbreviationFields}{%
9146         name={\protect\glssabbrvemfont{\the\glssshorttok}},
9147         sort={\the\glssshorttok},
9148         description={\the\glslongtok},%
9149         first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},%
9150         firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},%
9151         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.

```

9152 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9153     \csdef{glxtrpostlink\glscategorylabel}{%
9154         \glxtrifwasfirstuse
9155         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9156         \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9157             {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
9158         }%
9159     }%
9160 }%
9161 \glshasattribute{\the\glslabeltok}{regular}%
9162 {%
9163     \glissetattribute{\the\glslabeltok}{regular}{false}%
9164 }%
9165 {}%
9166 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9167 \renewcommand*{\glxtrsetupfulldefs}{%
9168     \let\glxtrifwasfirstuse\@secondoftwo
9169 }%

```

```

9170 }%
9171 {%
9172   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9173   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9174   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
9175   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
9176   \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9177   \renewcommand*{\glsxtrfullformat}[2]{%
9178     \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9179     \ifglsxtrininsertinside\else##2\fi
9180   }%
9181   \renewcommand*{\glsxtrfullplformat}[2]{%
9182     \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9183     \ifglsxtrininsertinside\else##2\fi
9184   }%
9185   \renewcommand*{\Glsxtrfullformat}[2]{%
9186     \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9187     \ifglsxtrininsertinside\else##2\fi
9188   }%
9189   \renewcommand*{\Glsxtrfullplformat}[2]{%
9190     \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9191     \ifglsxtrininsertinside\else##2\fi
9192   }%

```

The first use full form and the inline full form use the short (long) style.

```

9193   \renewcommand*{\glsxtrininlinefullformat}[2]{%
9194     \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9195     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9196     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
9197   }%
9198   \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9199     \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9200     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9201     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9202   }%
9203   \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9204     \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9205     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9206     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
9207   }%
9208   \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
9209     \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9210     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9211     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9212   }%
9213 }

```

postfootnote-em Backward compatibility:

9214 \@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.

9215 \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.

```

9216 \ifdef\glscurrentfieldvalue
9217 {
9218   \newcommand*{\glxtruserparen}[2]{%
9219     \glxtrfullsep{#2}%
9220     \glxtrparen
9221     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9222   }
9223 }
9224 {
9225   \newcommand*{\glxtruserparen}[2]{%
9226     \glxtrfullsep{#2}%
9227     \glxtrparen
9228     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9229   }
9230 }
```

Font used for short form:

lsabbrvuserfont

9231 \newcommand\*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

Font used for short form on first use:

stabbrvuserfont

9232 \newcommand\*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

Font used for long form:

glslonguserfont

9233 \newcommand\*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

Font used for long form on first use:

rstlonguserfont

9234 \newcommand\*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}

The default short form suffix:

lsxtrusersuffix

```
9235 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

long-short-user

```
9236 \newabbreviationstyle{long-short-user}%
9237 {%
9238   \renewcommand*{\CustomAbbreviationFields}{%
9239     name={\protect\glssabbrvuserfont{\the\glssshorttok}},
9240     sort={\the\glssshorttok},
9241     first={\protect\glslfirstlonguserfont{\the\glslongtok}}%
9242     \protect\glsxtruserparen{\protect\glslfirstabbrvuserfont{\the\glssshorttok}}}%
9243     {\the\glslabeltok}},%
9244     firstplural={\protect\glslfirstlonguserfont{\the\glslongpltok}}%
9245     \protect\glsxtruserparen
9246     {\protect\glslfirstabbrvuserfont{\the\glssshortpltok}}{\the\glslabeltok}},%

9247     plural={\protect\glssabbrvuserfont{\the\glssshortpltok}},%
9248     description={\protect\glslonguserfont{\the\glslongtok}}}%

  Unset the regular attribute if it has been set.
```

```
9249   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9250     \glshasattribute{\the\glslabeltok}{regular}}%
9251   {%
9252     \glissetattribute{\the\glslabeltok}{regular}{false}%
9253   }%
9254   {}%
9255 }%
9256 }%
9257 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9258 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9259 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvuserfont{##1}}%
9260 \renewcommand*{\glslfirstabbrvfont}[1]{\glslfirstabbrvuserfont{##1}}%
9261 \renewcommand*{\glslfirstlongfont}[1]{\glslfirstlonguserfont{##1}}%
9262 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9263 \renewcommand*{\glxtrfullformat}[2]{%
9264   \glslfirstlonguserfont{\glssaccesslong{##1}}\ifglxtrinsertinside##2\fi}%
9265   \ifglxtrinsertinside\else##2\fi
9266   \glsxtruserparen{\glslfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9267 }%
9268 \renewcommand*{\glxtrfullplformat}[2]{%
9269   \glslfirstlonguserfont{\glssaccesslongpl{##1}}\ifglxtrinsertinside##2\fi}%
9270   \ifglxtrinsertinside\else##2\fi
9271   \glsxtruserparen{\glslfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9272 }%
9273 \renewcommand*{\Glsxtrfullformat}[2]{%
9274   \glslfirstlonguserfont{\glssaccesslong{##1}}\ifglxtrinsertinside##2\fi}%

```

```

9275 \ifglxtrinsertinside\else##2\fi
9276 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9277 }%
9278 \renewcommand*{\Glsxtrfullplformat}[2]{%
9279 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9280 \ifglxtrinsertinside\else##2\fi
9281 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9282 }%
9283 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9284 \newabbreviationstyle{long-postshort-user}%
9285 {%
9286 \renewcommand*{\CustomAbbreviationFields}{%
9287 name={\protect\glssabbrvuserfont{\the\glssshorttok}},
9288 sort={\the\glssshorttok},
9289 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9290 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9291 plural={\protect\glssabbrvuserfont{\the\glssshortpltok}},%
9292 description={\protect\glslonguserfont{\the\glslongtok}}}%
9293 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9294 \csdef{glxtrpostlink\glscategorylabel}{%
9295 \glxtrifwasfirstuse
9296 {%
9297 \glxtruserparen
9298 {\glsfirstabbrvuserfont{\glssentryshort{\glslabel}}}%
9299 {\glslabel}%
9300 }%
9301 }%
9302 }%
9303 \glshasattribute{\the\glslabeltok}{regular}%
9304 {%
9305 \glissetattribute{\the\glslabeltok}{regular}{false}%
9306 }%
9307 }%
9308 }%
9309 }%
9310 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9311 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9312 \renewcommand*{\glssabbrvfnt}[1]{\glssabbrvuserfont{##1}}%
9313 \renewcommand*{\glsfirstabbrvfnt}[1]{\glsfirstabbrvuserfont{##1}}%
9314 \renewcommand*{\glsfirstlongfnt}[1]{\glsfirstlonguserfont{##1}}%
9315 \renewcommand*{\glslongfnt}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9316 \renewcommand*{\glxtrfullformat}[2]{%
9317 \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%

```

```

9318   \ifglxtrinsertinside\else##2\fi
9319 }%
9320 \renewcommand*{\glxtrfullplformat}[2]{%
9321   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9322   \ifglxtrinsertinside\else##2\fi
9323 }%
9324 \renewcommand*{\Glsxtrfullformat}[2]{%
9325   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9326   \ifglxtrinsertinside\else##2\fi
9327 }%
9328 \renewcommand*{\Glsxtrfullplformat}[2]{%
9329   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9330   \ifglxtrinsertinside\else##2\fi
9331 }%

```

In-line format:

```

9332 \renewcommand*{\glxtrinlinefullformat}[2]{%
9333   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9334   \ifglxtrinsertinside\else##2\fi
9335   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9336 }%
9337 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9338   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9339   \ifglxtrinsertinside\else##2\fi
9340   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9341 }%
9342 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9343   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9344   \ifglxtrinsertinside\else##2\fi
9345   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9346 }%
9347 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9348   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9349   \ifglxtrinsertinside\else##2\fi
9350   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9351 }%
9352 }

```

**short-user-desc** Like long-postshort-user but the user supplies the description.

```

9353 \newabbreviationstyle{long-postshort-user-desc}%
9354 {%
9355   \renewcommand*{\CustomAbbreviationFields}{%
9356     name={\protect\glslonguserfont{\the\glslongtok}}%
9357     \protect\glxtruserparen
9358     {\protect\glssabbrvuserfont{\the\glssshorttok}}{\the\glslabeltok}},
9359     sort={\the\glslongtok},
9360     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9361     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9362     text={\protect\glssabbrvuserfont{\the\glssshorttok}},%

```

```

9363 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9364 }%
9365 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9366 \csdef{glxtrpostlink\glscategorylabel}{%
9367 \glxtrifwasfirstuse
9368 {%
9369 \glxtruserparen
9370 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9371 {\glslabel}%
9372 }%
9373 {}%
9374 }%
9375 \glshasattribute{\the\glslabeltok}{regular}%
9376 {%
9377 \glssetattribute{\the\glslabeltok}{regular}{false}%
9378 }%
9379 {}%
9380 }%
9381 }%
9382 {%
9383 \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9384 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9385 \newabbreviationstyle{short-postlong-user}%
9386 {%
9387 \renewcommand*{\CustomAbbreviationFields}{%
9388 name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9389 sort={\the\glsshorttok},
9390 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9391 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9392 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9393 description={\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9394 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9395 \csdef{glxtrpostlink\glscategorylabel}{%
9396 \glxtrifwasfirstuse
9397 {%
9398 \glxtruserparen
9399 {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9400 {\glslabel}%
9401 }%
9402 {}%
9403 }%
9404 \glshasattribute{\the\glslabeltok}{regular}%
9405 {%
9406 \glssetattribute{\the\glslabeltok}{regular}{false}%
9407 }%
9408 {}%
9409 }%

```

```
9410 }%
9411 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9412 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9413 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9414 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9415 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9416 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

First use full form:

```
9417 \renewcommand*{\glxtrfullformat}[2]{%
9418   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9419   \ifglxtrininsertinside\else##2\fi
9420 }%
9421 \renewcommand*{\glxtrfullplformat}[2]{%
9422   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9423   \ifglxtrininsertinside\else##2\fi
9424 }%
9425 \renewcommand*{\Glsxtrfullformat}[2]{%
9426   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9427   \ifglxtrininsertinside\else##2\fi
9428 }%
9429 \renewcommand*{\Glsxtrfullplformat}[2]{%
9430   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9431   \ifglxtrininsertinside\else##2\fi
9432 }%
```

In-line format:

```
9433 \renewcommand*{\glxtrinlinefullformat}[2]{%
9434   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9435   \ifglxtrininsertinside\else##2\fi
9436   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9437 }%
9438 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9439   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9440   \ifglxtrininsertinside\else##2\fi
9441   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9442 }%
9443 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9444   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9445   \ifglxtrininsertinside\else##2\fi
9446   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9447 }%
9448 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9449   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9450   \ifglxtrininsertinside\else##2\fi
9451   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9452 }%
9453 }
```



long-user-desc Like short-postlong-user but leaves the user to specify the description.

```
9454 \newabbreviationstyle{short-postlong-user-desc}%
9455 {%
9456   \renewcommand*{\CustomAbbreviationFields}{%
9457     name={\protect\glsabbrvuserfont{\the\glsshorttok}%
9458       \protect\glsxtruserparen
9459         {\protect\glslonguserfont{\the\glslongpltok}}}%
9460       {\the\glslabeltok}},
9461     sort={\the\glsshorttok},
9462     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9463     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9464     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9465     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9466   }%
9467   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9468     \csdef{glsxtrpostlink\glscategorylabel}{%
9469       \glsxtrifwasfirstuse
9470       {%
9471         \glsxtruserparen
9472         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9473         {\glslabel}}%
9474       }%
9475     }%
9476   }%
9477   \glsattribute{\the\glslabeltok}{regular}%
9478   {%
9479     \glssetattribute{\the\glslabeltok}{regular}{false}%
9480   }%
9481   {}%
9482 }%
9483 }%
9484 {%
9485   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9486 }
```

short-user-desc

```
9487 \newabbreviationstyle{long-short-user-desc}%
9488 {%
9489   \renewcommand*{\CustomAbbreviationFields}{%
9490     name={\glsxtrlongshortdescname},
9491     sort={\glsxtrlongshortdescsort},%

9492     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9493       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9494     {\the\glslabeltok}},%
9495     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9496       \protect\glsxtruserparen
9497       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9498     text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```

9499 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
9500 }%

```

Unset the regular attribute if it has been set.

```

9501 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9502   \glshasattribute{\the\glslabeltok}{regular}%
9503   {%
9504     \glissetattribute{\the\glslabeltok}{regular}{false}%
9505   }%
9506   {}%
9507 }%
9508 }%
9509 {%
9510   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9511 }

```

short-long-user

```

9512 \newabbreviationstyle{short-long-user}%
9513 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

9514 \renewcommand*{\CustomAbbreviationFields}{%
9515   name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9516   sort={\the\glsshorttok},
9517   description={\protect\glslonguserfont{\the\glslongtok}},%
9518   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9519   \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9520   {\the\glslabeltok}},%
9521   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9522   \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
9523   {\the\glslabeltok}},%
9524   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9525 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9526   \glshasattribute{\the\glslabeltok}{regular}%
9527   {%
9528     \glissetattribute{\the\glslabeltok}{regular}{false}%
9529   }%
9530   {}%
9531 }%
9532 }%
9533 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9534 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9535 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9536 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9537 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9538 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline form are the same for this style.

```

9539 \renewcommand*{\glxtrfullformat}[2]{%
9540   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9541   \ifglxtrinsertinside\else##2\fi
9542   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9543 }%
9544 \renewcommand*{\glxtrfullplformat}[2]{%
9545   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9546   \ifglxtrinsertinside\else##2\fi
9547   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9548 }%
9549 \renewcommand*{\Glsxtrfullformat}[2]{%
9550   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9551   \ifglxtrinsertinside\else##2\fi
9552   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9553 }%
9554 \renewcommand*{\Glsxtrfullplformat}[2]{%
9555   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9556   \ifglxtrinsertinside\else##2\fi
9557   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9558 }%
9559 }

```

-long-user-desc

```

9560 \newabbreviationstyle{short-long-user-desc}%
9561 {%
9562   \renewcommand*{\CustomAbbreviationFields}{%
9563     name={\glxtrshortlongdescname},
9564     sort={\glxtrshortlongdescsort},%
9565     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9566     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9567     {\the\glslabeltok}},%
9568     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9569     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9570     {\the\glslabeltok}},%
9571     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9572     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9573   }%

```

Unset the regular attribute if it has been set.

```

9574 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9575   \glshasattribute{\the\glslabeltok}{regular}%
9576   {%
9577     \glissetattribute{\the\glslabeltok}{regular}{false}%
9578   }%
9579   {}%
9580 }%
9581 }%
9582 {%

```

```

9583 \GlsXtrUseAbbrStyleFmts{short-long-user}%
9584 }

```

### 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.

```

9585 \newrobustcmd*{\glstrifhyphenstart}[3]{%
9586   \ifx\glsinsert#1\relax
9587   \expandafter\@glstrifhyphenstart#1\relax\relax
9588   \@end@glstrifhyphenstart{#2}{#3}%
9589   \else
9590   \@glstrifhyphenstart#1\relax\relax\@end@glstrifhyphenstart{#2}{#3}%
9591   \fi
9592 }

```

`\trifhyphenstart`

```

9593 \def\@glstrifhyphenstart#1#2\@end@glstrifhyphenstart#3#4{%
9594   \ifx-#1\relax#3\else #4\fi
9595 }

```

`\glstrlonghyphenshort`

```
\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.

```

9596 \newcommand*{\glstrlonghyphenshort}[4]{%

```

Grouping is needed to localise the redefinitions.

```

9597 {%

```

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.

```

9598   \glstrifhyphenstart{#4}{\def\glstrwordsep{-}}{}%
9599   \glsfirstlonghyphenfont{#2\ifglstrinsertinside{#4}\fi}%
9600   \ifglstrinsertinside\else{#4}\fi
9601   \glstrfullsep{#1}%
9602   \glstrparen{\glsfirstabbrvhyphenfont{#3\ifglstrinsertinside{#4}\fi}%
9603   \ifglstrinsertinside\else{#4}\fi}%
9604 }%
9605 }

```

abbrvhyphenfont

```
9606 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
9607 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
9608 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%
```

tlonghyphenfont

```
9609 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
9610 \newcommand*{\glsxtrhyphensuffix}{\glsxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the markwords attribute.

```
9611 \newabbreviationstyle{long-hyphen-short-hyphen}%
9612 {%
9613   \renewcommand*{\CustomAbbreviationFields}{%
9614     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9615     sort={\the\glsshorttok},
9616     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
9617       \protect\glsxtrfullsep{\the\glslabeltok}%
9618       \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9619     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
9620       \protect\glsxtrfullsep{\the\glslabeltok}%
9621       \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9622     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9623     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
9624   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9625     \glshasattribute{\the\glslabeltok}{regular}%
9626     {%
9627       \glssetattribute{\the\glslabeltok}{regular}{false}%
9628     }%
9629   }%
9630 }%
9631 }%
9632 {%
9633   \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9634   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9635   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9636   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9637   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

9638 \renewcommand*{\glxstrfullformat}[2]{%
9639   \glxstrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9640 }%
9641 \renewcommand*{\glxstrfullplformat}[2]{%
9642   \glxstrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
9643   {\glsaccessshortpl{##1}}{##2}%
9644 }%
9645 \renewcommand*{\Glsxstrfullformat}[2]{%
9646   \glxstrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9647 }%
9648 \renewcommand*{\Glsxstrfullplformat}[2]{%
9649   \glxstrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9650   {\glsaccessshortpl{##1}}{##2}%
9651 }%
9652 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9653 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9654 {%
9655   \renewcommand*{\CustomAbbreviationFields}{%
9656     name={\glxstrlongshortdescname},
9657     sort={\glxstrlongshortdescsort},
9658     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
9659     \protect\glxstrfullsep{\the\glslabeltok}}%
9660     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9661     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
9662     \protect\glxstrfullsep{\the\glslabeltok}}%
9663     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9664     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}}},%
9665     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9666   }%

```

Unset the regular attribute if it has been set.

```

9667 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9668   \glshasattribute{\the\glslabeltok}{regular}%
9669   {%
9670     \glissetattribute{\the\glslabeltok}{regular}{false}%
9671   }%
9672   {}%
9673 }%
9674 }%
9675 {%
9676   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9677 }

```

onghyphennoshort `\glxstrlonghyphennoshort{<label>}{<long>}{<insert>}`

```
9678 \newcommand*{\glxstrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
9679 {%
```

If *<insert>* starts with a hyphen, redefine `\glxstrwordsep` 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 `\glxstrwordsep` if *<insert>* doesn't start with a hyphen.

```
9680 \glxstrifhyphenstart{#3}{\def\glxstrwordsep{-}}{%
9681 \glsfirstlonghyphenfont{#2\ifglxstrinsertinside{#3}\fi}%
9682 \ifglxstrinsertinsideelse{#3}\fi
9683 }%
9684 }
```

`short-desc-noreg` This version doesn't show the short form (except explicitly with `\glxstrshort`). Since `\glxstrshort` 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.

```
9685 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
9686 {%
9687 \renewcommand*{\CustomAbbreviationFields}{%
9688 name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
9689 sort={\expandonce\glxtrorglong},
9690 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9691 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9692 plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
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-hyphen-short-hyphen}%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9704 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
9705 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
9706 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
9707 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9708 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9709 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9710 \glxstrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
9711 }%
```

```

9712 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9713   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9714 }%
9715 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9716   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9717 }%
9718 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9719   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9720 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9721 \renewcommand*{\glxtrinlinefullformat}[2]{%
9722   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9723   \glxtrfullsep{##1}%
9724   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9725 }%
9726 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9727   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9728   \glxtrfullsep{##1}%
9729   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9730 }%
9731 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9732   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9733   \glxtrfullsep{##1}%
9734   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
9735 }%
9736 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9737   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9738   \glxtrfullsep{##1}%
9739   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
9740 }%

```

The first use full form only displays the long form.

```

9741 \renewcommand*{\glxtrfullformat}[2]{%
9742   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9743 }%
9744 \renewcommand*{\glxtrfullplformat}[2]{%
9745   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9746 }%
9747 \renewcommand*{\Glsxtrfullformat}[2]{%
9748   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9749 }%
9750 \renewcommand*{\Glsxtrfullplformat}[2]{%
9751   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9752 }%
9753 }

```

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.



```

9754 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9755 {%
9756   \renewcommand*{\CustomAbbreviationFields}{%
9757     name={\protect\glsabbrvfont{\the\glsshorttok}},
9758     sort={\the\glsshorttok},
9759     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9760     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9761     text={\protect\glslonghyphenfont{\the\glslongtok}},%
9762     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
9763     description={\the\glslongtok}%
9764   }%

```

Unset the regular attribute if it has been set.

```

9765   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9766     \glshasattribute{\the\glslabeltok}{regular}%
9767     {%
9768       \glssetattribute{\the\glslabeltok}{regular}{false}%
9769     }%
9770   }%
9771 }%
9772 }%
9773 {%
9774   \GlsXtrUseAbbrStyleFmts{long-desc}%
9775 }

```

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.

```

9776 \newcommand*{\glsxtrlonghyphen}[3]{%
  Grouping is needed to localise the redefinitions.
9777 {%
9778   \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9779   \glsfirstlonghyphenfont{#1}%
9780 }%
9781 }

```

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.

```

9782 \newcommand*{\glsxtrposthyphenshort}[2]{%
9783 {%

```

```

9784 \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
9785 \ifglxtrininsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
9786 \glxtrfullsep{#1}%
9787 \glxtrparen
9788 {\glsfirstabbrvhyphenfont{\glsentryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
9789 \ifglxtrininsertinside\else{#2}\fi
9790 }%
9791 }%
9792 }

```

hyphensubsequent

```
\glxtrposthyphensubsequent{<label>}{<insert>}
```

Format in the post-link hook for subsequent use. The label is ignored by default.

```

9793 \newcommand*{\glxtrposthyphensubsequent}[2]{%
9794 \glsabbrvfont{\ifglxtrininsertinside {#2}\fi}%
9795 \ifglxtrininsertinside \else{#2}\fi
9796 }

```

postshort-hyphen

Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9797 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
9798 {%
9799 \renewcommand*{\CustomAbbreviationFields}{%
9800 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9801 sort={\the\glsshorttok},
9802 first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
9803 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
9804 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9805 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9806 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9807 \csdef{glxtrpostlink\glscategorylabel}{%
9808 \glxtrifwasfirstuse
9809 {%
9810 \glxtrposthyphenshort{\glslabel}{\glsininsert}%
9811 }%
9812 }%

```

Put the insertion into the post-link:

```

9813 \glxtrposthyphensubsequent{\glslabel}{\glsininsert}%
9814 }%
9815 }%
9816 \glshasattribute{\the\glslabeltok}{regular}%
9817 {%
9818 \glissetattribute{\the\glslabeltok}{regular}{false}%
9819 }%
9820 {}%
9821 }%

```

9822 }%

9823 {%

In case the user wants to mix and match font styles, these are redefined here.

```
9824 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
9825 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9826 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9827 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9828 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

Subsequent use needs to omit the insertion:

```
9829 \renewcommand*{\glxtrs subsequentfmt}[2]{%
9830   \glsabbrvfont{\glsaccessshort{##1}}%
9831 }%
9832 \renewcommand*{\glxtrs subsequentplfmt}[2]{%
9833   \glsabbrvfont{\glsaccessshortpl{##1}}%
9834 }%
9835 \renewcommand*{\Glsxtrs subsequentfmt}[2]{%
9836   \glsabbrvfont{\Glsaccessshort{##1}}%
9837 }%
9838 \renewcommand*{\Glsxtrs subsequentplfmt}[2]{%
9839   \glsabbrvfont{\Glsaccessshortpl{##1}}%
9840 }%
```

First use full form:

```
9841 \renewcommand*{\glxtrfullformat}[2]{%
9842   \glxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
9843 }%
9844 \renewcommand*{\glxtrfullplformat}[2]{%
9845   \glxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
9846 }%
9847 \renewcommand*{\Glsxtrfullformat}[2]{%
9848   \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
9849 }%
9850 \renewcommand*{\Glsxtrfullplformat}[2]{%
9851   \glxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
9852 }%
```

In-line format.

```
9853 \renewcommand*{\glxtrinlinefullformat}[2]{%
9854   \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
9855   \ifglxtrininsertinside{##2}\fi}%
9856   \ifglxtrininsertinside \else{##2}\fi
9857 }%
9858 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9859   \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%
9860   \ifglxtrininsertinside{##2}\fi}%
9861   \ifglxtrininsertinside \else{##2}\fi
9862 }%
9863 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9864   \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
```

```

9865     \ifglxtrinsertinside{##2}\fi}%
9866     \ifglxtrinsertinside \else{##2}\fi
9867 }%
9868 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9869     \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
9870     \ifglxtrinsertinside{##2}\fi}%
9871     \ifglxtrinsertinside \else{##2}\fi
9872 }%
9873 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

9874 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
9875 {%
9876   \renewcommand*{\CustomAbbreviationFields}{%
9877     name={\glxtrlongshortdescname},
9878     sort={\glxtrlongshortdescsort},%
9879     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9880     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9881     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9882     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9883 }%
9884 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9885   \csdef{glxtrpostlink\glscategorylabel}{%
9886     \glxtrifwasfirstuse
9887     {%
9888       \glxtrposthyphenshort{\glslabel}{\glsinsert}%
9889     }%
9890     {%

```

Put the insertion into the post-link:

```

9891       \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9892     }%
9893   }%
9894   \glshasattribute{\the\glslabeltok}{regular}%
9895   {%
9896     \glissetattribute{\the\glslabeltok}{regular}{false}%
9897   }%
9898   {}%
9899 }%
9900 }%
9901 {%
9902   \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
9903 }

```

`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.

```
9904 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9905 {%
```

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.)

```
9906 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
```

```
9907 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
```

```
9908 \ifglxtrininsertinside\else{#4}\fi
```

```
9909 \glxtrfullsep{#1}%
```

```
9910 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
```

```
9911 \ifglxtrininsertinside\else{#4}\fi}%
```

```
9912 }%
```

```
9913 }
```

hen-long-hyphen Designed for use with the markwords attribute.

```
9914 \newabbreviationstyle{short-hyphen-long-hyphen}%
```

```
9915 {%
```

```
9916 \renewcommand*{\CustomAbbreviationFields}{%
```

```
9917 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
```

```
9918 sort={\the\glsshorttok},
```

```
9919 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
```

```
9920 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
9921 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
```

```
9922 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
```

```
9923 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
9924 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
```

```
9925 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%
```

```
9926 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
9927 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
9928 \glshasattribute{\the\glslabeltok}{regular}}%
```

```
9929 {%
```

```
9930 \glissetattribute{\the\glslabeltok}{regular}{false}}%
```

```
9931 }%
```

```
9932 {}%
```

```
9933 }%
```

```
9934 }%
```

```
9935 {%
```

```
9936 \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
```

```
9937 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
```

```
9938 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
```

```
9939 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
```

```
9940 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

9941 \renewcommand*{\glxtrfullformat}[2]{%
9942   \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
9943 }%
9944 \renewcommand*{\glxtrfullplformat}[2]{%
9945   \glxtrshorthyphenlong{##1}%
9946   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
9947 }%
9948 \renewcommand*{\Glsxtrfullformat}[2]{%
9949   \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
9950 }%
9951 \renewcommand*{\Glsxtrfullplformat}[2]{%
9952   \glxtrshorthyphenlong{##1}%
9953   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
9954 }%
9955 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

9956 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
9957 {%
9958   \renewcommand*{\CustomAbbreviationFields}{%
9959     name={\glxtrshortlongdescname},
9960     sort={\glxtrshortlongdescsort},
9961     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
9962       \protect\glxtrfullsep{\the\glslabeltok}%
9963       \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
9964     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
9965       \protect\glxtrfullsep{\the\glslabeltok}%
9966       \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
9967     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
9968     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9969 }%

```

Unset the regular attribute if it has been set.

```

9970 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9971   \glshasattribute{\the\glslabeltok}{regular}%
9972   {%
9973     \glissetattribute{\the\glslabeltok}{regular}{false}%
9974   }%
9975   {}%
9976 }%
9977 }%
9978 {%
9979   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
9980 }

```

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.

```
9981 \newcommand*{\glxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
9982 {%
9983   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
9984   \glsfirstabbrvhyphenfont{#1}%
9985 }%
9986 }
```

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.

```
9987 \newcommand*{\glxtrposthyphenlong}[2]{%
9988 {%
9989   \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
9990   \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
9991   \glxtrfullsep{#1}%
9992   \glxtrparen
9993   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
9994   \ifglxtrininsertinside\else{#2}\fi
9995 }%
9996 }%
9997 }
```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
9998 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
9999 {%
10000   \renewcommand*{\CustomAbbreviationFields}{%
10001     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
10002     sort={\the\glsshorttok},
10003     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10004     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10005     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10006     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10007   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10008     \csdef{glxtrpostlink\glscategorylabel}{%
10009       \glxtrifwasfirstuse
10010       {%
10011         \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10012       }%
10013     }%
```

Put the insertion into the post-link:

```

10014      \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10015      }%
10016      }%
10017      \glshasattribute{\the\glslabeltok}{regular}%
10018      {%
10019      \glsetattribute{\the\glslabeltok}{regular}{false}%
10020      }%
10021      {}%
10022  }%
10023 }%
10024 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10025 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10026 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10027 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10028 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10029 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10030 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10031   \glsabbrvfont{\glsaccessshort{##1}}%
10032   }%
10033 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10034   \glsabbrvfont{\glsaccessshortpl{##1}}%
10035   }%
10036 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10037   \glsabbrvfont{\Glsaccessshort{##1}}%
10038   }%
10039 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10040   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10041   }%

```

First use full form:

```

10042 \renewcommand*{\glxtrfullformat}[2]{%
10043   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10044   }%
10045 \renewcommand*{\glxtrfullplformat}[2]{%
10046   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10047   }%
10048 \renewcommand*{\Glsxtrfullformat}[2]{%
10049   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10050   }%
10051 \renewcommand*{\Glsxtrfullplformat}[2]{%
10052   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10053   }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10054 \renewcommand*{\glxtrinlinefullformat}[2]{%

```



```

10055 \glsfirstabbrvhyphenfont{\glsaccessshort{##1}%
10056 \ifglxtrinsertinside{##2}\fi}%
10057 \ifglxtrinsertinside \else{##2}\fi
10058 }%
10059 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10060 \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}%
10061 \ifglxtrinsertinside{##2}\fi}%
10062 \ifglxtrinsertinside \else{##2}\fi
10063 }%
10064 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10065 \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}%
10066 \ifglxtrinsertinside{##2}\fi}%
10067 \ifglxtrinsertinside \else{##2}\fi
10068 }%
10069 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10070 \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
10071 \ifglxtrinsertinside{##2}\fi}%
10072 \ifglxtrinsertinside \else{##2}\fi
10073 }%
10074 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

10075 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10076 {%
10077 \renewcommand*{\CustomAbbreviationFields}{%
10078 name={\glxtrshortlongdescname},
10079 sort={\glxtrshortlongdescsort},%
10080 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10081 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10082 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10083 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10084 }%
10085 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10086 \csdef{glxtrpostlink\glscategorylabel}{%
10087 \glxtrifwasfirstuse
10088 {%
10089 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10090 }%
10091 }%

```

Put the insertion into the post-link:

```

10092 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10093 }%
10094 }%
10095 \glshasattribute{\the\glslabeltok}{regular}%
10096 {%
10097 \glissetattribute{\the\glslabeltok}{regular}{false}%
10098 }%
10099 {}%
10100 }%

```

```

10101 }%
10102 {%
10103   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10104 }

```

## 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

```

10105 \newcommand*{\glssabbrvonlyfont}{\glssabbrvdefaultfont}%

```

stabbrvonlyfont

```

10106 \newcommand*{\glssfistabbrvonlyfont}{\glssabbrvonlyfont}%

```

glslongonlyfont

```

10107 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

rstlongonlyfont

```

10108 \newcommand*{\glssfistlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

lsxtronlysuffix

```

10109 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}

```

only-short-only

```

10110 \newabbreviationstyle{long-only-short-only}%
10111 {%
10112   \renewcommand*{\CustomAbbreviationFields}{%
10113     name={\protect\glssabbrvonlyfont{\the\glssshorttok}},
10114     sort={\the\glssshorttok},
10115     first={\protect\glssfistlongonlyfont{\the\glslongtok}},%
10116     firstplural={\protect\glssfistlongonlyfont{\the\glslongpltok}},%
10117     plural={\protect\glssabbrvonlyfont{\the\glssshortpltok}},%
10118     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10119   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10120     \glshasattribute{\the\glslabeltok}{regular}%
10121     {%
10122       \glissetattribute{\the\glslabeltok}{regular}{false}%
10123     }%
10124   }%
10125 }%
10126 }%
10127 {%
10128   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtronlysuffix}%
10129   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvonlyfont{##1}}%

```

```

10130 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10131 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10132 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

10133 \renewcommand*{\glsxtrfullformat}[2]{%
10134   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10135   \ifglsxtrininsertinside\else##2\fi
10136 }%
10137 \renewcommand*{\glsxtrfullplformat}[2]{%
10138   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10139   \ifglsxtrininsertinside\else##2\fi
10140 }%
10141 \renewcommand*{\Glsxtrfullformat}[2]{%
10142   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10143   \ifglsxtrininsertinside\else##2\fi
10144 }%
10145 \renewcommand*{\Glsxtrfullplformat}[2]{%
10146   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10147   \ifglsxtrininsertinside\else##2\fi
10148 }%

```

The inline full form does show the short form.

```

10149 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10150   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10151   \ifglsxtrininsertinside\else##2\fi
10152   \glsxtrfullsep{##1}%
10153   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10154 }%
10155 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10156   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10157   \ifglsxtrininsertinside\else##2\fi
10158   \glsxtrfullsep{##1}%
10159   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10160 }%
10161 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10162   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10163   \ifglsxtrininsertinside\else##2\fi
10164   \glsxtrfullsep{##1}%
10165   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10166 }%
10167 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10168   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10169   \ifglsxtrininsertinside\else##2\fi
10170   \glsxtrfullsep{##1}%
10171   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10172 }%
10173 }

```

xtronlydescsort

```

10174 \newcommand*{\glxstronlydescsort}{\the\glslongtok}

xtronlydescname
10175 \newcommand*{\glxtronlydescname}{%
10176   \protect\glslongfont{\the\glslongtok}%
10177 }

short-only-desc
10178 \newabbreviationstyle{long-only-short-only-desc}%
10179 {%
10180   \renewcommand*{\CustomAbbreviationFields}{%
10181     name={\glxtronlydescname},
10182     sort={\glxtronlydescsort},%
10183     first={\protect\glslongonlyfont{\the\glslongtok}},%
10184     firstplural={\protect\glslongonlyfont{\the\glslongpltok}},%
10185     text={\protect\glssabbrvonlyfont{\the\glssshorttok}},%
10186     plural={\protect\glssabbrvonlyfont{\the\glssshortpltok}}}%
10187   }%

   Unset the regular attribute if it has been set.
10188   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10189     \glshasattribute{\the\glslabeltok}{regular}%
10190     {%
10191       \glissetattribute{\the\glslabeltok}{regular}{false}%
10192     }%
10193   }%
10194 }%
10195 }%
10196 {%
10197   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10198 }

```

## 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 `\glsentryshort`, 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 `\glsxtrshort` 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:

```
10199 \let\@glxstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
10200 \renewcommand*{\markright}[1]{%
10201   \glxstrmarkhook
10202   \@glxstr@org@markright{\@glxstrinmark#1\@glxstrnotinmark}%
10203   \glxstrrestoremarkhook
10204 }
```

`\markboth` Save original definition:

```
10205 \let\@glxstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
10206 \renewcommand*{\markboth}[2]{%
10207   \glxstrmarkhook
10208   \@glxstr@org@markboth
10209   {\@glxstrinmark#1\@glxstrnotinmark}%
10210   {\@glxstrinmark#2\@glxstrnotinmark}%
10211   \glxstrrestoremarkhook
10212 }
```

Also do this for `\@starttoc`

`\@starttoc` Save original definition:

```
10213 \let\@glxstr@org@@starttoc\@starttoc
```

Redefine:

```
10214 \renewcommand*{\@starttoc}[1]{%
10215   \glxstrmarkhook
10216   \@glxstrinmark
10217   \@glxstr@org@@starttoc{#1}%
10218   \@glxstrnotinmark
10219   \glxstrrestoremarkhook
10220 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`sxtrRevertMarks`

```
10221 \newcommand*{\glxstrRevertMarks}{%
10222   \let\markright\@glxstr@org@markright
```

```

10223 \let\markboth\@glsxtr@org@markboth
10224 \let\@starttoc\@glsxtr@org@starttoc
10225 }

```

\glsxtrifinmark

```

10226 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

10227 \newrobustcmd*{\@glsxtrinmark}{%
10228   \let\glsxtrifinmark\@firstoftwo
10229 }

```

glsxtrnotinmark

```

10230 \newrobustcmd*{\glsxtrnotinmark}{%
10231   \let\glsxtrifinmark\@secondoftwo
10232 }

```

eorpdforheading

```

10233 \ifdef\texorpdfstring
10234 {
10235   \newcommand*{\glsxtrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10236 }
10237 {
10238   \newcommand*{\glsxtrtitleorpdforheading}[3]{#1}
10239 }

```

\glsxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

10240 \newcommand*{\glsxtrmarkhook}{%

```

Save current definitions:

```

10241 \let\@glsxtr@org@MakeUppercase\MakeUppercase
10242 \let\@glsxtr@org@glsxtrtitleorpdforheading\glsxtrtitleorpdforheading
10243 \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
10244 \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
10245 \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10246 \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
10247 \let\@glsxtr@org@glsxtrtitlename\glsxtrtitlename
10248 \let\@glsxtr@org@Glsxtrtitlename\Glsxtrtitlename
10249 \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
10250 \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
10251 \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
10252 \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10253 \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
10254 \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10255 \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
10256 \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10257 \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
10258 \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl

```

```

10259 \let\@glxstr@org@Glsxtrtitlelong\Glsxtrtitlelong
10260 \let\@glxstr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10261 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
10262 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
10263 \let\@glxstr@org@Glsxtrtitlefull\Glsxtrtitlefull
10264 \let\@glxstr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

#### New definitions

```

10265 \let\glxstrifinmark\@firstoftwo
10266 \let\MakeUppercase\MakeTextUppercase
10267 \let\glxstrtitleorpdforheading\@thirdofthree
10268 \let\glxstrtitleshort\glxstrheadshort
10269 \let\glxstrtitleshortpl\glxstrheadshortpl
10270 \let\Glsxtrtitleshort\Glsxtrheadshort
10271 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
10272 \let\glxstrtitlename\glxstrheadname
10273 \let\Glsxtrtitlename\Glsxtrheadname
10274 \let\glxstrtitletext\glxstrheadtext
10275 \let\Glsxtrtitletext\Glsxtrheadtext
10276 \let\glxstrtitleplural\glxstrheadplural
10277 \let\Glsxtrtitleplural\Glsxtrheadplural
10278 \let\glxstrtitlefirst\glxstrheadfirst
10279 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10280 \let\glxstrtitlefirstplural\glxstrheadfirstplural
10281 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10282 \let\glxstrtitlelong\glxstrheadlong
10283 \let\glxstrtitlelongpl\glxstrheadlongpl
10284 \let\Glsxtrtitlelong\Glsxtrheadlong
10285 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10286 \let\glxstrtitlefull\glxstrheadfull
10287 \let\glxstrtitlefullpl\glxstrheadfullpl
10288 \let\Glsxtrtitlefull\Glsxtrheadfull
10289 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10290 }

```

`\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.)

```

10291 \newcommand*{\glxstrrestoremarkhook}{%
10292   \let\glxstrifinmark\@secondoftwo
10293   \let\MakeUppercase\@glxstr@org@MakeUppercase
10294   \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
10295   \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
10296   \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
10297   \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
10298   \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
10299   \let\glxstrtitlename\@glxstr@org@glxstrtitlename
10300   \let\Glsxtrtitlename\@glxstr@org@Glsxtrtitlename

```

```

10301 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
10302 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
10303 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
10304 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
10305 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
10306 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
10307 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
10308 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
10309 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
10310 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
10311 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
10312 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
10313 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
10314 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
10315 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
10316 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
10317 }

```

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.

```

10318 \newcommand*{\glsxtrheadshort}[1]{%
10319 \protect\NoCaseChange
10320 {%
10321 \glsifattribute{#1}{headuc}{true}%
10322 {%
10323 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10324 }%
10325 {%
10326 \glsxtrshort[noindex,hyper=false]{#1}[]%
10327 }%
10328 }%
10329 }

```

**lgsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents.

```

10330 \newrobustcmd*{\lgsxtrtitleshort}[1]{%
10331 \glsxtrshort[noindex,hyper=false]{#1}[]%
10332 }

```

**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.

```

10333 \newcommand*{\glsxtrheadshortpl}[1]{%
10334 \protect\NoCaseChange
10335 {%
10336 \glsifattribute{#1}{headuc}{true}%
10337 {%
10338 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%

```



```

10339 }%
10340 {%
10341     \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10342 }%
10343 }%
10344 }

```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents.

```

10345 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10346     \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10347 }

```

**Glsxtrheadshort** Command used to display short form in the page header with the first letter converted to upper case.

```

10348 \newcommand*{\Glsxtrheadshort}[1]{%
10349     \protect\NoCaseChange
10350 {%
10351     \glsifattribute{#1}{headuc}{true}%
10352     {%
10353         \GLSxtrshort[noindex,hyper=false]{#1}[]%
10354     }%
10355     {%
10356         \Glsxtrshort[noindex,hyper=false]{#1}[]%
10357     }%
10358 }%
10359 }

```

**lsextrtitleshort** Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10360 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10361     \Glsxtrshort[noindex,hyper=false]{#1}[]%
10362 }

```

**sxtrheadshortpl** Command used to display plural short form in the page header with the first letter converted to upper case.

```

10363 \newcommand*{\Glsxtrheadshortpl}[1]{%
10364     \protect\NoCaseChange
10365 {%
10366     \glsifattribute{#1}{headuc}{true}%
10367     {%
10368         \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10369     }%
10370     {%
10371         \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10372     }%
10373 }%
10374 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10375 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10376   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10377 }
```

`\glxtrheadname` As above but for the name value.

```
10378 \newcommand*{\glxtrheadname}[1]{%
10379   \protect\NoCaseChange
10380   {%
10381     \glsifattribute{#1}{headuc}{true}%
10382     {%
10383       \GLSname[noindex,hyper=false]{#1}[]%
10384     }%
10385     {%
10386       \glsname[noindex,hyper=false]{#1}[]%
10387     }%
10388   }%
10389 }
```

`glxtrtitlename` Command to display name value in section title and table of contents.

```
10390 \newrobustcmd*{\glxtrtitlename}[1]{%
10391   \glsname[noindex,hyper=false]{#1}[]%
10392 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10393 \newcommand*{\Glsxtrheadname}[1]{%
10394   \protect\NoCaseChange
10395   {%
10396     \glsifattribute{#1}{headuc}{true}%
10397     {%
10398       \GLSname[noindex,hyper=false]{#1}[]%
10399     }%
10400     {%
10401       \Glsname[noindex,hyper=false]{#1}[]%
10402     }%
10403   }%
10404 }
```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
10405 %\changes{1.21}{2017-11-03}{new}
10406 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10407   \Glsname[noindex,hyper=false]{#1}[]%
10408 }
```

`\glxtrheadtext` As above but for the text value.

```
10409 \newcommand*{\glxtrheadtext}[1]{%
```

```

10410 \protect\NoCaseChange
10411 {%
10412   \glsifattribute{#1}{headuc}{true}%
10413   {%
10414     \GLStext[noindex,hyper=false]{#1}[]%
10415   }%
10416   {%
10417     \glstext[noindex,hyper=false]{#1}[]%
10418   }%
10419 }%
10420 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

10421 \newrobustcmd*{\glsxtrtitletext}[1]{%
10422   \glstext[noindex,hyper=false]{#1}[]%
10423 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

10424 \newcommand*{\Glsxtrheadtext}[1]{%
10425   \protect\NoCaseChange
10426   {%
10427     \glsifattribute{#1}{headuc}{true}%
10428     {%
10429       \GLStext[noindex,hyper=false]{#1}[]%
10430     }%
10431     {%
10432       \Glstext[noindex,hyper=false]{#1}[]%
10433     }%
10434   }%
10435 }

```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

10436 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10437   \Glstext[noindex,hyper=false]{#1}[]%
10438 }

```

`lslxtrheadplural` As above but for the plural value.

```

10439 \newcommand*{\glsxtrheadplural}[1]{%
10440   \protect\NoCaseChange
10441   {%
10442     \glsifattribute{#1}{headuc}{true}%
10443     {%
10444       \GLSplural[noindex,hyper=false]{#1}[]%
10445     }%
10446     {%
10447       \glsplural[noindex,hyper=false]{#1}[]%
10448     }%
10449   }%

```

10450 }

**sxtrtitleplural** Command to display plural value in section title and table of contents.

```
10451 \newrobustcmd*{\glsxtrtitleplural}[1]{%
10452   \glsplural[noindex,hyper=false]{#1}[]%
10453 }
```

**lsxtrheadplural** Convert first letter to upper case.

```
10454 \newcommand*{\Glsxtrheadplural}[1]{%
10455   \protect\NoCaseChange
10456   {%
10457     \glusifattribute{#1}{headuc}{true}%
10458     {%
10459       \GLSplural[noindex,hyper=false]{#1}[]%
10460     }%
10461     {%
10462       \Glsplural[noindex,hyper=false]{#1}[]%
10463     }%
10464   }%
10465 }
```

**sxtrtitleplural** Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
10466 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10467   \Glsplural[noindex,hyper=false]{#1}[]%
10468 }
```

**glsxtrheadfirst** As above but for the first value.

```
10469 \newcommand*{\glsxtrheadfirst}[1]{%
10470   \protect\NoCaseChange
10471   {%
10472     \glusifattribute{#1}{headuc}{true}%
10473     {%
10474       \GLSfirst[noindex,hyper=false]{#1}[]%
10475     }%
10476     {%
10477       \glsfirst[noindex,hyper=false]{#1}[]%
10478     }%
10479   }%
10480 }
```

**lsxtrtitlefirst** Command to display first value in section title and table of contents.

```
10481 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10482   \glsfirst[noindex,hyper=false]{#1}[]%
10483 }
```

**Glsxtrheadfirst** First letter converted to upper case

```
10484 \newcommand*{\Glsxtrheadfirst}[1]{%
```

```

10485 \protect\NoCaseChange
10486 {%
10487   \glsifattribute{#1}{headuc}{true}%
10488   {%
10489     \GLSfirst[noindex,hyper=false]{#1}[]%
10490   }%
10491   {%
10492     \Glsfirst[noindex,hyper=false]{#1}[]%
10493   }%
10494 }%
10495 }

```

**lsxtrtitlefirst** Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10496 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10497   \GLSfirst[noindex,hyper=false]{#1}[]%
10498 }

```

**headfirstplural** As above but for the firstplural value.

```

10499 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10500   \protect\NoCaseChange
10501   {%
10502     \glsifattribute{#1}{headuc}{true}%
10503     {%
10504       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10505     }%
10506     {%
10507       \glsfirstplural[noindex,hyper=false]{#1}[]%
10508     }%
10509   }%
10510 }

```

**itlefirstplural** Command to display firstplural value in section title and table of contents.

```

10511 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
10512   \glsfirstplural[noindex,hyper=false]{#1}[]%
10513 }

```

**headfirstplural** First letter converted to upper case

```

10514 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10515   \protect\NoCaseChange
10516   {%
10517     \glsifattribute{#1}{headuc}{true}%
10518     {%
10519       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10520     }%
10521     {%
10522       \Glsfirstplural[noindex,hyper=false]{#1}[]%
10523     }%
10524   }%

```

10525 }

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10526 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10527   \Glsfirstplural[noindex,hyper=false]{#1}[]%
10528 }
```

`\glxstrheadlong` Command used to display long form in the page header.

```
10529 \newcommand*{\glxstrheadlong}[1]{%
10530   \protect\NoCaseChange
10531   {%
10532     \glsifattribute{#1}{headuc}{true}%
10533     {%
10534       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10535     }%
10536     {%
10537       \glxstrlong[noindex,hyper=false]{#1}[]%
10538     }%
10539   }%
10540 }
```

`\glxstrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
10541 \newrobustcmd*{\glxstrtitlelong}[1]{%
10542   \glxstrlong[noindex,hyper=false]{#1}[]%
10543 }
```

`\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.

```
10544 \newcommand*{\glxstrheadlongpl}[1]{%
10545   \protect\NoCaseChange
10546   {%
10547     \glsifattribute{#1}{headuc}{true}%
10548     {%
10549       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10550     }%
10551     {%
10552       \glxstrlongpl[noindex,hyper=false]{#1}[]%
10553     }%
10554   }%
10555 }
```

`\glxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
10556 \newrobustcmd*{\glxstrtitlelongpl}[1]{%
10557   \glxstrlongpl[noindex,hyper=false]{#1}[]%
10558 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
10559 \newcommand*{\Glsxtrheadlong}[1]{%
10560   \protect\NoCaseChange
10561   {%
10562     \glsifattribute{#1}{headuc}{true}%
10563     {%
10564       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10565     }%
10566     {%
10567       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10568     }%
10569   }%
10570 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10571 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10572   \GLSxtrlong[noindex,hyper=false]{#1}[]%
10573 }
```

`lGlsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```
10574 \newcommand*{\Glsxtrheadlongpl}[1]{%
10575   \protect\NoCaseChange
10576   {%
10577     \glsifattribute{#1}{headuc}{true}%
10578     {%
10579       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10580     }%
10581     {%
10582       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10583     }%
10584   }%
10585 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10586 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10587   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10588 }
```

`\glsxtrheadfull` Command used to display full form in the page header.

```
10589 \newcommand*{\glsxtrheadfull}[1]{%
10590   \protect\NoCaseChange
10591   {%
10592     \glsifattribute{#1}{headuc}{true}%
10593     {%
```

```

10594     \GLSxtrfull[noindex,hyper=false]{#1}[]%
10595 }%
10596 {%
10597     \glsxtrfull[noindex,hyper=false]{#1}[]%
10598 }%
10599 }%
10600 }

```

**glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents.

```

10601 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10602   \glsxtrfull[noindex,hyper=false]{#1}[]%
10603 }

```

**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.

```

10604 \newcommand*{\glsxtrheadfullpl}[1]{%
10605   \protect\NoCaseChange
10606   {%
10607     \glsifattribute{#1}{headuc}{true}%
10608     {%
10609       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10610     }%
10611     {%
10612       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10613     }%
10614   }%
10615 }

```

**glsxtrtitlefullpl** Command to display plural full form of abbreviation in section title and table of contents.

```

10616 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
10617   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10618 }

```

**\Glsxtrheadfull** Command used to display full form in the page header with the first letter converted to upper case.

```

10619 \newcommand*{\Glsxtrheadfull}[1]{%
10620   \protect\NoCaseChange
10621   {%
10622     \glsifattribute{#1}{headuc}{true}%
10623     {%
10624       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10625     }%
10626     {%
10627       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10628     }%
10629   }%
10630 }

```



`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10631 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10632   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10633 }
```

`\lsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```
10634 \newcommand*{\Glsxtrheadfullpl}[1]{%
10635   \protect\NoCaseChange
10636   {%
10637     \glsifattribute{#1}{headuc}{true}%
10638     {%
10639       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10640     }%
10641     {%
10642       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10643     }%
10644   }%
10645 }
```

`\sxttrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10646 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10647   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10648 }
```

`\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.

```
10649 \ifdef\texorpdfstring
10650 {
10651   \newcommand*{\glsfmtshort}[1]{%
10652     \texorpdfstring
10653     {\glsxtrtitleshort{#1}}%
10654     {\glsentryshort{#1}}%
10655   }
10656 }
10657 {
10658   \newcommand*{\glsfmtshort}[1]{%
10659     \glsxtrtitleshort{#1}}
10660 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
10661 \ifdef\texorpdfstring
10662 {
10663   \newcommand*{\glsfmtshortpl}[1]{%
```

```

10664 \texorpdfstring
10665     {\glstrtitleshortpl{#1}}}%
10666     {\glstryshortpl{#1}}}%
10667 }
10668 }
10669 {
10670 \newcommand*{\glfmtshortpl}[1]{%
10671     \glstrtitleshortpl{#1}}
10672 }

```

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).

```

10673 \ifdef\texorpdfstring
10674 {
10675     \newcommand*{\Glsfmtshort}[1]{%
10676         \texorpdfstring
10677             {\Glsxtrtitleshort{#1}}}%
10678             {\glstryshort{#1}}}%
10679     }
10680 }
10681 {
10682     \newcommand*{\Glsfmtshort}[1]{%
10683         \Glsxtrtitleshort{#1}}
10684 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10685 \ifdef\texorpdfstring
10686 {
10687     \newcommand*{\Glsfmtshortpl}[1]{%
10688         \texorpdfstring
10689             {\Glsxtrtitleshortpl{#1}}}%
10690             {\glstryshortpl{#1}}}%
10691     }
10692 }
10693 {
10694     \newcommand*{\Glsfmtshortpl}[1]{%
10695         \Glsxtrtitleshortpl{#1}}
10696 }

```

`\glfmtname` As above but for the name value.

```

10697 \ifdef\texorpdfstring
10698 {
10699     \newcommand*{\glfmtname}[1]{%
10700         \texorpdfstring
10701             {\glxtrtitlename{#1}}}%
10702             {\glstryname{#1}}}%
10703     }

```

```

10704 }
10705 {
10706   \newcommand*{\Glsfmtname}[1]{%
10707     \Glsxtrtitlename{#1}}
10708 }

```

`\Glsfmtname` First letter converted to upper case.

```

10709 \ifdef\texorpdfstring
10710 {
10711   \newcommand*{\Glsfmtname}[1]{%
10712     \texorpdfstring
10713     {\Glsxtrtitlename{#1}}%
10714     {\Glsentryname{#1}}%
10715   }
10716 }
10717 {
10718   \newcommand*{\Glsfmtname}[1]{%
10719     \Glsxtrtitlename{#1}}
10720 }

```

`\glsfmttext` As above but for the text value.

```

10721 \ifdef\texorpdfstring
10722 {
10723   \newcommand*{\glsfmttext}[1]{%
10724     \texorpdfstring
10725     {\Glsxtrtitletext{#1}}%
10726     {\Glsentrytext{#1}}%
10727   }
10728 }
10729 {
10730   \newcommand*{\glsfmttext}[1]{%
10731     \Glsxtrtitletext{#1}}
10732 }

```

`\Glsfmttext` First letter converted to upper case.

```

10733 \ifdef\texorpdfstring
10734 {
10735   \newcommand*{\Glsfmttext}[1]{%
10736     \texorpdfstring
10737     {\Glsxtrtitletext{#1}}%
10738     {\Glsentrytext{#1}}%
10739   }
10740 }
10741 {
10742   \newcommand*{\Glsfmttext}[1]{%
10743     \Glsxtrtitletext{#1}}
10744 }

```

`\glsfmtplural` As above but for the plural value.

```

10745 \ifdef\teorpdfstring
10746 {
10747   \newcommand*\glsfmtplural}[1]{%
10748     \teorpdfstring
10749     {\glsxtrtitleplural{#1}}%
10750     {\glsentryplural{#1}}%
10751   }
10752 }
10753 {
10754   \newcommand*\glsfmtplural}[1]{%
10755     \glsxtrtitleplural{#1}}
10756 }

```

`\Glsfmtplural` First letter converted to upper case.

```

10757 \ifdef\teorpdfstring
10758 {
10759   \newcommand*\Glsfmtplural}[1]{%
10760     \teorpdfstring
10761     {\Glsxtrtitleplural{#1}}%
10762     {\glsentryplural{#1}}%
10763   }
10764 }
10765 {
10766   \newcommand*\Glsfmtplural}[1]{%
10767     \Glsxtrtitleplural{#1}}
10768 }

```

`\glsfmtfirst` As above but for the first value.

```

10769 \ifdef\teorpdfstring
10770 {
10771   \newcommand*\glsfmtfirst}[1]{%
10772     \teorpdfstring
10773     {\glsxtrtitlefirst{#1}}%
10774     {\glsentryfirst{#1}}%
10775   }
10776 }
10777 {
10778   \newcommand*\glsfmtfirst}[1]{%
10779     \glsxtrtitlefirst{#1}}
10780 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

10781 \ifdef\teorpdfstring
10782 {
10783   \newcommand*\Glsfmtfirst}[1]{%
10784     \teorpdfstring
10785     {\Glsxtrtitlefirst{#1}}%
10786     {\glsentryfirst{#1}}%
10787   }

```

```

10788 }
10789 {
10790   \newcommand*{\Glsfmtfirst}[1]{%
10791     \Glsxtrtitlefirst{#1}}
10792 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

10793 \ifdef\texorpdfstring
10794 {
10795   \newcommand*{\glsfmtfirstpl}[1]{%
10796     \texorpdfstring
10797     {\Glsxtrtitlefirstplural{#1}}%
10798     {\glsentryfirstplural{#1}}%
10799   }
10800 }
10801 {
10802   \newcommand*{\glsfmtfirstpl}[1]{%
10803     \Glsxtrtitlefirstplural{#1}}
10804 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

10805 \ifdef\texorpdfstring
10806 {
10807   \newcommand*{\Glsfmtfirstpl}[1]{%
10808     \texorpdfstring
10809     {\Glsxtrtitlefirstplural{#1}}%
10810     {\glsentryfirstplural{#1}}%
10811   }
10812 }
10813 {
10814   \newcommand*{\Glsfmtfirstpl}[1]{%
10815     \Glsxtrtitlefirstplural{#1}}
10816 }

```

`\glsfmtlong` As above but for the long value.

```

10817 \ifdef\texorpdfstring
10818 {
10819   \newcommand*{\glsfmtlong}[1]{%
10820     \texorpdfstring
10821     {\Glsxtrtitlelong{#1}}%
10822     {\glsentrylong{#1}}%
10823   }
10824 }
10825 {
10826   \newcommand*{\glsfmtlong}[1]{%
10827     \Glsxtrtitlelong{#1}}
10828 }

```

`\Glsfmtlong` First letter converted to upper case.

```

10829 \ifdef\texorpdfstring
10830 {
10831   \newcommand*\Glsfmtlong}[1]{%
10832     \texorpdfstring
10833     {\Glsxtrtitlelong{#1}}%
10834     {\glsentrylong{#1}}%
10835   }
10836 }
10837 {
10838   \newcommand*\Glsfmtlong}[1]{%
10839     \Glsxtrtitlelong{#1}}
10840 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

10841 \ifdef\texorpdfstring
10842 {
10843   \newcommand*\glsfmtlongpl}[1]{%
10844     \texorpdfstring
10845     {\glsxtrtitlelongpl{#1}}%
10846     {\glsentrylongpl{#1}}%
10847   }
10848 }
10849 {
10850   \newcommand*\glsfmtlongpl}[1]{%
10851     \glsxtrtitlelongpl{#1}}
10852 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

10853 \ifdef\texorpdfstring
10854 {
10855   \newcommand*\Glsfmtlongpl}[1]{%
10856     \texorpdfstring
10857     {\Glsxtrtitlelongpl{#1}}%
10858     {\glsentrylongpl{#1}}%
10859   }
10860 }
10861 {
10862   \newcommand*\Glsfmtlongpl}[1]{%
10863     \Glsxtrtitlelongpl{#1}}
10864 }

```

`\glsfmtfull` In-line full format.

```

10865 \ifdef\texorpdfstring
10866 {
10867   \newcommand*\glsfmtfull}[1]{%
10868     \texorpdfstring
10869     {\glsxtrtitlefull{#1}}%
10870     {\glsxtrinlinefullformat{#1}{}}%
10871   }

```

```

10872 }
10873 {
10874   \newcommand*{\glsfmtfull}[1]{%
10875     \glsxtrtitlefull{#1}}
10876 }

```

`\Glsfmtfull` First letter converted to upper case.

```

10877 \ifdef\teorpdfstring
10878 {
10879   \newcommand*{\Glsfmtfull}[1]{%
10880     \teorpdfstring
10881     {\Glsxtrtitlefull{#1}}%
10882     {\Glsxtrinlinefullformat{#1}{}}%
10883   }
10884 }
10885 {
10886   \newcommand*{\Glsfmtfull}[1]{%
10887     \Glsxtrtitlefull{#1}}
10888 }

```

`\glsfmtfullpl` In-line full plural format.

```

10889 \ifdef\teorpdfstring
10890 {
10891   \newcommand*{\glsfmtfullpl}[1]{%
10892     \teorpdfstring
10893     {\glsxtrtitlefullpl{#1}}%
10894     {\glsxtrinlinefullplformat{#1}{}}%
10895   }
10896 }
10897 {
10898   \newcommand*{\glsfmtfullpl}[1]{%
10899     \glsxtrtitlefullpl{#1}}
10900 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

10901 \ifdef\teorpdfstring
10902 {
10903   \newcommand*{\Glsfmtfullpl}[1]{%
10904     \teorpdfstring
10905     {\Glsxtrtitlefullpl{#1}}%
10906     {\Glsxtrinlinefullplformat{#1}{}}%
10907   }
10908 }
10909 {
10910   \newcommand*{\Glsfmtfullpl}[1]{%
10911     \Glsxtrtitlefullpl{#1}}
10912 }

```

## 1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```
10913 \newcommand*{\RequireGlossariesExtraLang}[1]{%
10914   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
10915 }
```

sariesExtraLang

```
10916 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
10917   \ProvidesFile{glossariesxtr-#1.ldf}%
10918 }
```

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.)

```
10919 \@ifpackageloaded{tracklang}
10920 {%
10921   \AnyTrackedLanguages
10922   {%
10923     \ForEachTrackedDialect{\this@dialect}{%
10924       \IfTrackedLanguageFileExists{\this@dialect}%
10925       {glossariesxtr-}% prefix
10926       {.ldf}%
10927       {%
10928         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
10929       }%
10930     }%
10931   }%
10932 }%
10933 }%
10934 {}%
10935 }
10936 {}
```

Load `glossaries-extra-stylemods` if required.

```
10937 \@glsxtr@redefstyles
```

and set the style:

```
10938 \@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:

```
10939 \NeedsTeXFormat{LaTeX2e}
10940 \ProvidesPackage{glossaries-extra-stylemods}[2017/11/03 v1.21 (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`.

`glxtr@loadstyles`

```
10941 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
10942 \DeclareOption{all}{%
10943   \appto\@glxtr@loadstyles{%
10944     \RequirePackage{glossary-inline}%
10945     \RequirePackage{glossary-list}%
10946     \RequirePackage{glossary-tree}%
10947     \RequirePackage{glossary-mcols}%
10948     \RequirePackage{glossary-long}%
10949     \RequirePackage{glossary-longragged}%
10950     \RequirePackage{glossary-longbooktabs}%
10951     \RequirePackage{glossary-super}%
10952     \RequirePackage{glossary-superragged}%
10953     \RequirePackage{glossary-bookindex}%
10954   }
10955 }

10956 \DeclareOption*{%
10957   \IfFileExists{glossary-\CurrentOption.sty}
10958   {\eappto\@glxtr@loadstyles{%
10959     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
10960   }%
10961   {%
10962     \PackageError{glossaries-extra-styles}%

```

```

10963     {Unknown option ‘\CurrentOption’}{}%
10964   }%
10965 }

```

Process the package options:

```
10966 \ProcessOptions
```

Load the required packages:

```
10967 \@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`.

```
10968 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

10969 \providecommand{\renewglossarystyle}[2]{%
10970   \ifcsundef{@glsstyle@#1}%
10971   {%
10972     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
10973   }%
10974   {%
10975     \csdef{@glsstyle@#1}{#2}%
10976   }%
10977 }

```

## 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.

```

10978 \ifdef{@glsstyle@listdotted}
10979 {%
10980   \renewglossarystyle{listdotted}{%
10981     \setglossarystyle{list}%
10982     \renewcommand*{\glossentry}[2]{%
10983       \item[]\makebox[\glslistdottedwidth][l]{%
10984         \glsentryitem{##1}%
10985         \glstarget{##1}{\glossentryname{##1}}%
10986         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10987         \glossentrydesc{##1}\glspostdescription}%
10988     \renewcommand*{\subglossentry}[3]{%
10989       \item[]\makebox[\glslistdottedwidth][l]{%
10990         \glssubentryitem{##2}%
10991         \glstarget{##2}{\glossentryname{##2}}%
10992         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
10993         \glossentrydesc{##2}\glspostdescription}%
10994   }

```

```
10995 }
10996 {%
```

Assume the style isn't required if it hasn't already been defined.

```
10997 }
```

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.

```
10998 \ifdef{\@glsstyle@list}
10999 {%
```

listprelocation Space before number list for top-level entries.

```
11000 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11001 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11002 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11003 \renewglossarystyle{list}{%
11004   \renewenvironment{theglossary}%
11005     {\begin{description}}{\end{description}}%
11006   \renewcommand*{\glossaryheader}{}%
11007   \renewcommand*{\glsgroupheading}[1]{}%
11008   \renewcommand*{\glossentry}[2]{%
11009     \item[\glsentryitem{##1}]%
11010       \glstarget{##1}{\glossentryname{##1}}]
11011     \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11012   \renewcommand*{\subglossentry}[3]{%
11013     \glssubentryitem{##2}%
11014     \glstarget{##2}{\strut}\space
11015     \glossentrydesc{##2}\glspostdescription
11016     \glslistchildprelocation ##3\glslistchildpostlocation}%
11017   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11018 }
11019 }
11020 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11021 \ifdef{\@glsstyle@altlist}
11022 {%
11023   \renewglossarystyle{altlist}{%
11024     \setglossarystyle{list}%
11025     \renewcommand*{\glossentry}[2]{%
11026       \item[\glsentryitem{##1}]%
```

```

11027      \glstarget{##1}{\glossentryname{##1}}}%
11028      \mbox{}\par\nobreak\@afterheading
11029      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11030 \renewcommand{\subglossentry}[3]{%
11031   \par
11032   \glssubentryitem{##2}%
11033   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11034   \glslistchildprelocation ##3}%
11035 }
11036 }
11037 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11038 \ifdef{\@glsstyle@listgroup}
11039 {%
11040   \renewglossarystyle{listgroup}{%
11041     \setglossarystyle{list}%
11042     \renewcommand*\glsgroupheading[1]{%
11043       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11044       \mbox{}\par\nobreak\@afterheading
11045     }%
11046   }
11047 }
11048 {}

```

Similarly for listhypergroup.

```

11049 \ifdef{\@glsstyle@listhypergroup}
11050 {%
11051   \renewglossarystyle{listhypergroup}{%
11052     \setglossarystyle{list}%
11053     \renewcommand*\glossaryheader{%
11054       \glslistnavigationitem{\glsnavigation}}%
11055     \renewcommand*\glsgroupheading[1]{%
11056       \item[\glslistgroupheaderfmt
11057         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11058       \mbox{}\par\nobreak\@afterheading
11059     }%
11060   }
11061 }
11062 {}

```

Similarly for altlistgroup.

```

11063 \ifdef{\@glsstyle@altlistgroup}
11064 {%
11065   \renewglossarystyle{altlistgroup}{%
11066     \setglossarystyle{altlist}%
11067     \renewcommand*\glsgroupheading[1]{%
11068       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11069       \mbox{}\par\nobreak\@afterheading
11070     }%
11071   }

```

```

11072 }
11073 {}

    Similarly for altlisthypergroup.
11074 \ifdef{\@glsstyle@altlisthypergroup}
11075 {%
11076   \renewglossarystyle{altlisthypergroup}{%
11077     \setglossarystyle{altlist}%
11078     \renewcommand*\{glossaryheader\}{%
11079       \glslistnavigationitem{\glsnavigation}}%
11080     \renewcommand*\{glsgroupheading\}[1]{%
11081       \item[\glslistgroupheaderfmt
11082         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11083       \mbox{}\par\nobreak\@afterheading
11084     }%
11085   }
11086 }
11087 {}

```

## 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`.

```

11088 \ifcsdef{@glsstyle@long}
11089 {%
11090   \renewglossarystyle{long}{%
11091     \renewenvironment{theglossary}%
11092       {\begin{longtable}{lp{\glsdescwidth}}}%
11093       {\end{longtable}}%
11094     \renewcommand*\{glossaryheader\}{}%
11095     \renewcommand*\{glsgroupheading\}[1]{}%
11096     \renewcommand{\glossentry}[2]{%
11097       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11098       \glossentrydesc{##1}\glspostdescription
11099       \glstrprelocation ##2\tabularnewline
11100     }%
11101     \renewcommand{\subglossentry}[3]{%
11102       &
11103       \glssubentryitem{##2}%
11104       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11105       \glstrprelocation ##3\tabularnewline
11106     }%
11107     \ifglsnogroupskip
11108       \renewcommand*\{glsgroupskip\}{}%
11109     \else
11110       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11111     \fi
11112   }

```

```
11113 }
11114 {}
```

Three column style:

```
11115 \ifcsdef{@glsstyle@long3col}
11116 {%
11117   \renewglossarystyle{long3col}{%
11118     \renewenvironment{theglossary}%
11119       {\begin{longtable}{lp{@glsdescwidth}p{@glspagelistwidth}}}%
11120       {\end{longtable}}}%
11121     \renewcommand*{\glossaryheader}{}%
11122     \renewcommand*{\glsgroupheading}[1]{}%
11123     \renewcommand{\glossentry}[2]{%
11124       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11125       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11126     }%
11127     \renewcommand{\subglossentry}[3]{%
11128       &
11129       \glssubentryitem{##2}%
11130       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11131       ##3\tabularnewline
11132     }%
11133   }
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11133   \ifglsnogroupskip
11134     \renewcommand*{\glsgroupskip}{}%
11135   \else
11136     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11137   \fi
11138 }
11139 }
11140 {}
```

Four column style:

```
11141 \ifcsdef{@glsstyle@long4col}
11142 {%
11143   \renewglossarystyle{long4col}{%
11144     \renewenvironment{theglossary}%
11145       {\begin{longtable}{llll}}%
11146       {\end{longtable}}}%
11147     \renewcommand*{\glossaryheader}{}%
11148     \renewcommand*{\glsgroupheading}[1]{}%
11149     \renewcommand{\glossentry}[2]{%
11150       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11151       \glossentrydesc{##1}\glspostdescription &
11152       \glossentrysymbol{##1} &
11153       ##2\tabularnewline
11154     }%
11155     \renewcommand{\subglossentry}[3]{%
11156       &
```

```

11157     \glssubentryitem{##2}%
11158     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11159     \glossentrysymbol{##2} & ##3\tabularnewline
11160 }%

11161 \ifglsnogroupskip
11162   \renewcommand*{\glsgroupskip}{}%
11163 \else
11164   \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11165 \fi
11166 }
11167 }
11168 {}

```

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`.

```

11169 \ifcsdef{@glstyle@longragged}
11170 {%
11171   \renewglossarystyle{longragged}{%
11172     \renewenvironment{theglossary}%
11173       {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}]}%
11174       {\end{longtable}}%
11175     \renewcommand*{\glossaryheader}{}%
11176     \renewcommand*{\glsgroupheading}[1]{}%
11177     \renewcommand{\glossentry}[2]{%
11178       \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11179       \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11180       \tabularnewline
11181     }%
11182     \renewcommand{\subglossentry}[3]{%
11183       &
11184       \glssubentryitem{##2}%
11185       \glstarget{##2}{\strut}\glossentrydesc{##2}%
11186       \glspostdescription\glxtrprelocation ##3%
11187       \tabularnewline
11188     }%
11189     \ifglsnogroupskip
11190       \renewcommand*{\glsgroupskip}{}%
11191     \else
11192       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11193     \fi
11194   }
11195 }

```

11196 {}

Three and four column styles don't use `\glstrprelocation` since the number list is in its own column.

```
11197 \ifcsdef{@glsstyle@longragged3col}
11198 {%
11199   \renewglossarystyle{longragged3col}{%
11200     \renewenvironment{theglossary}%
11201       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11202         >{\raggedright}p{\glspagelistwidth}}}%
11203       {\end{longtable}}}%
11204     \renewcommand*{\glossaryheader}{}%
11205     \renewcommand*{\glsgroupheading}[1]{}%
11206     \renewcommand{\glossentry}[2]{%
11207       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11208       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11209     }%
11210     \renewcommand{\subglossentry}[3]{%
11211       &
11212       \glssubentryitem{##2}%
11213       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11214       ##3\tabularnewline
11215     }%
11216   \ifglsgroupskip
11217     \renewcommand*{\glsgroupskip}{}%
11218   \else
11219     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11220   \fi
11221 }
11222 }
11223 {}
```

Four column style:

```
11224 \ifcsdef{@glsstyle@altlongragged4col}
11225 {%
11226   \renewglossarystyle{altlongragged4col}{%
11227     \renewenvironment{theglossary}%
11228       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11229         >{\raggedright}p{\glspagelistwidth}}}%
11230       {\end{longtable}}}%
11231     \renewcommand*{\glossaryheader}{}%
11232     \renewcommand*{\glsgroupheading}[1]{}%
11233     \renewcommand{\glossentry}[2]{%
11234       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11235       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11236       ##2\tabularnewline
11237     }%
11238     \renewcommand{\subglossentry}[3]{%
11239       &
```



```

11240     \glssubentryitem{##2}%
11241     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11242     \glossentrysymbol{##2} & ##3\tabularnewline
11243 }%

11244 \ifglsnogroupskip
11245     \renewcommand*{\glsgroupskip}{}%
11246 \else
11247     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11248 \fi
11249 }
11250 }
11251 {}

```

## 2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glxtrprelocation`.

```

11252 \ifcsdef{@glstyle@super}
11253 {%
11254     \renewglossarystyle{super}{%
11255         \renewenvironment{theglossary}%
11256             {\tablehead{}\tabletail}%
11257             \begin{supertabular}{lp{\glsdescwidth}}%
11258             {\end{supertabular}}%
11259         \renewcommand*{\glossaryheader}{}%
11260         \renewcommand*{\glsgroupheading}[1]{}%
11261         \renewcommand{\glossentry}[2]{%
11262             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11263             \glossentrydesc{##1}\glspostdescription
11264             \glxtrprelocation ##2\tabularnewline
11265         }%
11266         \renewcommand{\subglossentry}[3]{%
11267             &
11268             \glssubentryitem{##2}%
11269             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11270             \glxtrprelocation ##3\tabularnewline
11271         }%
11272         \ifglsnogroupskip
11273             \renewcommand*{\glsgroupskip}{}%
11274         \else
11275             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11276         \fi
11277     }
11278 }
11279 {}

```

Three column style:

```

11280 \ifcsdef{@glstyle@super3col}

```

```

11281 {%
11282   \renewglossarystyle{super3col}{%
11283     \renewenvironment{theglossary}%
11284       {\tablehead{}\tabletail{}}%
11285       \begin{supertabular}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
11286       {\end{supertabular}}%
11287     \renewcommand*{\glossaryheader}{}%
11288     \renewcommand*{\glsgroupheading}[1]{}%
11289     \renewcommand{\glossentry}[2]{%
11290       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11291       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11292     }%
11293     \renewcommand{\subglossentry}[3]{%
11294       &
11295       \glssubentryitem{##2}%
11296       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11297       ##3\tabularnewline
11298     }%
11299     \ifglsnogroupskip
11300       \renewcommand*{\glsgroupskip}{}%
11301     \else
11302       \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11303     \fi
11304   }
11305 }
11306 {}

```

Four column styles:

```

11307 \ifcsdef{@glstyle@super4col}
11308 {%
11309   \renewglossarystyle{super4col}{%
11310     \renewenvironment{theglossary}%
11311       {\tablehead{}\tabletail{}}%
11312       \begin{supertabular}{llll}}{%
11313       \end{supertabular}}%
11314     \renewcommand*{\glossaryheader}{}%
11315     \renewcommand*{\glsgroupheading}[1]{}%
11316     \renewcommand{\glossentry}[2]{%
11317       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11318       \glossentrydesc{##1}\glspostdescription &
11319       \glossentrysymbol{##1} & ##2\tabularnewline
11320     }%
11321     \renewcommand{\subglossentry}[3]{%
11322       &
11323       \glssubentryitem{##2}%
11324       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11325       \glossentrysymbol{##2} & ##3\tabularnewline
11326     }%

```

```

11327 \ifglsgroupskip
11328 \renewcommand*{\glsgroupskip}{}%
11329 \else
11330 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11331 \fi
11332 }
11333 }
11334 {}

```

## 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`.

```

11335 \ifcsdef{@glstyle@superragged}
11336 {%
11337 \renewglossarystyle{superragged}{%
11338 \renewenvironment{theglossary}%
11339 {\tablehead{}}\tabletail{}}%
11340 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11341 {\end{supertabular}}%
11342 \renewcommand*{\glossaryheader}{}%
11343 \renewcommand*{\glsgroupheading}[1]{}%
11344 \renewcommand{\glossentry}[2]{%
11345 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11346 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11347 \tabularnewline
11348 }%
11349 \renewcommand{\subglossentry}[3]{%
11350 &
11351 \glssubentryitem{##2}%
11352 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11353 \glstrprelocation ##3%
11354 \tabularnewline
11355 }%
11356 \ifglsgroupskip
11357 \renewcommand*{\glsgroupskip}{}%
11358 \else
11359 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11360 \fi
11361 }
11362 }
11363 {}

```

Three column style:

```

11364 \ifcsdef{@glstyle@superragged3col}
11365 {%
11366 \renewglossarystyle{superragged3col}{%

```

```

11367 \renewenvironment{theglossary}%
11368 {\tablehead{}\tabletail{}}%
11369 \begin{supertabular}{l>\raggedrightp{\glsdescwidth}%
11370 >\raggedrightp{\glspagelistwidth}}}%
11371 {\end{supertabular}}%
11372 \renewcommand*\glossaryheader{}%
11373 \renewcommand*\glsgroupheading}[1]{}%
11374 \renewcommand{\glossentry}[2]{%
11375 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11376 \glossentrydesc{##1}\glspostdescription &
11377 ##2\tabularnewline
11378 }%
11379 \renewcommand{\subglossentry}[3]{%
11380 &
11381 \glssubentryitem{##2}%
11382 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11383 ##3\tabularnewline
11384 }%

11385 \ifglsgroupskip
11386 \renewcommand*\glsgroupskip{}%
11387 \else
11388 \renewcommand*\glsgroupskip{ & \tabularnewline}%
11389 \fi
11390 }
11391 }
11392 {}

```

Four columns:

```

11393 \ifcsdef{@glsstyle@altsuperragged4col}
11394 {%
11395 \renewglossarystyle{altsuperragged4col}{%
11396 \renewenvironment{theglossary}%
11397 {\tablehead{}\tabletail{}}%
11398 \begin{supertabular}{l>\raggedrightp{\glsdescwidth}l%
11399 >\raggedrightp{\glspagelistwidth}}}%
11400 {\end{supertabular}}%
11401 \renewcommand*\glossaryheader{}%
11402 \renewcommand{\glossentry}[2]{%
11403 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11404 \glossentrydesc{##1}\glspostdescription &
11405 \glossentrysymbol{##1} & ##2\tabularnewline
11406 }%
11407 \renewcommand{\subglossentry}[3]{%
11408 &
11409 \glssubentryitem{##2}%
11410 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11411 \glossentrysymbol{##2} & ##3\tabularnewline
11412 }%

```

```

11413 \ifglsnogroupskip
11414 \renewcommand*{\glsgroupskip}{}%
11415 \else
11416 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11417 \fi
11418 }
11419 }
11420 {}

```

## 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`.

```

11421 \ifdef{\@glsstyle@inline}
11422 {%
11423 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11424 \renewcommand*{\glsinlinedescformat}[3]{%
11425 \space#1\glxtrpostdescription}
11426 \renewcommand*{\glsinlinesubdescformat}[3]{%
11427 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11428 }
11429 {}

```

## 2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11430 \ifdef{\@glsstyle@index}
11431 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11432 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11433 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

```

```

11434 \renewglossarystyle{index}{%
11435 \renewenvironment{theglossary}%
11436 {\setlength{\parindent}{0pt}%
11437 \setlength{\parskip}{0pt plus 0.3pt}%
11438 \let\item\glstreeitem
11439 \let\subitem\glstreesubitem}

```

```

11440     \let\subsubitem\glstreesubsubitem
11441   }%
11442 {\par}%
11443   \renewcommand*{\glossaryheader}{}%
11444   \renewcommand*{\glsgroupheading}[1]{}%
11445   \renewcommand*{\glossentry}[2]{%
11446     \item\glstentryitem{##1}%
11447     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11448     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11449     \glstreepredesc \glossentrydesc{##1}\glspostdescription
11450     \glstreeprelocation ##2%
11451   }%
11452   \renewcommand{\subglossentry}[3]{%
11453     \ifcase##1\relax
11454       \item
11455     \or
11456       \subitem
11457       \glssubentryitem{##2}%
11458     \else
11459       \subsubitem
11460     \fi
11461     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11462     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11463     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11464     \glstreechildprelocation ##3%
11465   }%
11466   \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11467 }
11468 }
11469 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11470 \ifdef{\@glstyle@indexgroup}
11471 {%
11472   \renewglossarystyle{indexgroup}{%
11473     \setglossarystyle{index}%
11474     \renewcommand*{\glsgroupheading}[1]{%
11475       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
11476       \nopagebreak\indexspace
11477       \nobreak\@afterheading
11478     }%
11479   }
11480 }
11481 {}

```

Similarly for indexhypergroup.

```

11482 \ifdef{\@glstyle@indexhypergroup}
11483 {%
11484   \renewglossarystyle{indexhypergroup}{%
11485     \setglossarystyle{index}%

```

```

11486 \renewcommand*{\glossaryheader}{%
11487 \item\glstreenavigationfmt{\glsnavigation}%
11488 \nobreak\@afterheading\indexspace}%
11489 \renewcommand*{\glsgroupheading}[1]{%
11490 \item\glstreegroupheaderfmt
11491 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
11492 \nopagebreak\indexspace
11493 \nobreak\@afterheading}%
11494 }%
11495 }
11496 {}

```

Adjust tree style to remove hard coded space before number list.

```

11497 \ifdef{\@glsstyle@tree}
11498 {%
11499 \renewglossarystyle{tree}{%
11500 \renewenvironment{theglossary}%
11501 {\setlength{\parindent}{0pt}%
11502 \setlength{\parskip}{0pt plus 0.3pt}}%
11503 {}%
11504 \renewcommand*{\glossaryheader}{}%
11505 \renewcommand*{\glsgroupheading}[1]{}%
11506 \renewcommand{\glossentry}[2]{%
11507 \hangindent0pt\relax
11508 \parindent0pt\relax
11509 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11510 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11511 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11512 \glstreeprelocation##2\par
11513 }%
11514 \renewcommand{\subglossentry}[3]{%
11515 \hangindent##1\glstreeindent\relax
11516 \parindent##1\glstreeindent\relax
11517 \ifnum##1=1\relax
11518 \glssubentryitem{##2}%
11519 \fi
11520 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11521 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11522 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11523 \glstreechildprelocation ##3\par
11524 }%
11525 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11526 }%
11527 }
11528 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11529 \ifdef{\@glsstyle@treegroup}
11530 {%
11531 \renewglossarystyle{treegroup}{%

```

```

11532 \setglossarystyle{tree}%
11533 \renewcommand{\glsgroupheading}[1]{\par
11534 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11535 \nopagebreak\indexspace\nobreak\@afterheading}%
11536 }
11537 }
11538 {}

```

Similarly for treehypergroup

```

11539 \ifdef{\@glsstyle@treehypergroup}
11540 {%
11541 \renewglossarystyle{treehypergroup}{%
11542 \setglossarystyle{tree}%
11543 \renewcommand*\glossaryheader{%
11544 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11545 \nobreak\@afterheading\indexspace}%
11546 \renewcommand*\glsgroupheading}[1]{%
11547 \par\noindent
11548 \glstreegroupheaderfmt
11549 {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
11550 \nopagebreak\indexspace\nobreak\@afterheading}%
11551 }
11552 }
11553 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11554 \ifdef{\@glsstyle@treenoname}
11555 {%
11556 \renewglossarystyle{treenoname}{%
11557 \renewenvironment{theglossary}%
11558 {\setlength{\parindent}{0pt}%
11559 \setlength{\parskip}{0pt plus 0.3pt}}%
11560 {}%
11561 \renewcommand*\glossaryheader{}%
11562 \renewcommand*\glsgroupheading}[1]{}%
11563 \renewcommand{\glossentry}[2]{%
11564 \hangindent0pt\relax
11565 \parindent0pt\relax
11566 \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11567 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11568 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11569 \glstreeprelocation##2\par
11570 }%
11571 \renewcommand{\subglossentry}[3]{%
11572 \hangindent##1\glstreeindent\relax
11573 \parindent##1\glstreeindent\relax
11574 \ifnum##1=1\relax
11575 \glssubentryitem{##2}%
11576 \fi
11577 \glstarget{##2}{\strut}%

```



```

11578     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11579 }%
11580 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11581 }
11582 }
11583 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11584 \ifdef{\@glsstyle@treenonamegroup}
11585 {%
11586   \renewglossarystyle{treenonamegroup}{%
11587     \setglossarystyle{treenoname}%
11588     \renewcommand{\glsgroupheading}[1]{\par
11589       \noindent\glstreegroupheaderfmt
11590       {\glsgrouptitle{##1}}}%
11591     \nopagebreak\indexspace\nobreak\@afterheading
11592   }%
11593 }
11594 }
11595 {}

```

Similarly for `treenonamehypergroup`

```

11596 \ifdef{\@glsstyle@treenonamehypergroup}
11597 {%
11598   \renewglossarystyle{treenonamehypergroup}{%
11599     \setglossarystyle{treenoname}%
11600     \renewcommand*{\glossaryheader}{%
11601       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11602       \nobreak\@afterheading\indexspace}%
11603     \renewcommand*{\glsgroupheading}[1]{%
11604       \par\noindent
11605       \glstreegroupheaderfmt
11606       {\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}%
11607     \nopagebreak\indexspace\nobreak\@afterheading}%
11608 }
11609 }
11610 {}

```

The `almtree` style is redefined to make it easier to make minor adjustments.

```

11611 \ifdef{\@glsstyle@almtree}
11612 {%

```

Only redefine this style if it's already been defined.

symbolDescLocation

```
\glxtralttreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

11613 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```

```

11614   {%
11615       \let\par\glxstrAltTreePar
11616       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11617       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11618   }%
11619 }

```

**trAltTreeIndent** Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11620 \newlength\glxstrAltTreeIndent

```

**lsxtrAltTreePar** Multi-paragraph descriptions need to keep the hanging indent.

```

11621 \newcommand{\glxstrAltTreePar}{%
11622     @@par
11623     \glxstrAltTreeSetHangIndent
11624     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11625 }

```

**symbolDescLocation** `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11626 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11627     \glxstralttreeSymbolDescLocation{#2}{#3}%
11628 }

```

**trreetopindent** 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.

```

11629 \newlength\glxstrtreetopindent

```

**sxtralttreeInit** User-level initialisation for the alttree style.

```

11630 \newcommand*{\glxstralttreeInit}{%
11631     \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
11632     \glxstrAltTreeIndent=\parindent
11633 }

```

**\glsglsetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\gdef`.

```

11634 \newcommand*{\glsglsetwidest}[2][0]{%
11635     \csgdef{@glswidestname\romannumeral#1}{#2}%
11636 }

```

**\eglssetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\protected@csedef`.

```

11637 \newcommand*{\eglssetwidest}[2][0]{%
11638     \protected@csedef{@glswidestname\romannumeral#1}{#2}%
11639 }

```

`\xglsssetwidest` Like the above but uses `\protected@csxdef`.

```

11640 \newcommand*{\xglsssetwidest}[2][0]{%
11641   \protected@csxdef{\glswidestname\romannumeral#1}{#2}%
11642 }

```

`\glssgetwidestname` Provide a user-level macro to obtain the widest top-level name.

```

11643 \newcommand*{\glssgetwidestname}{\@glswidestname}

```

`\glssgetwidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```

11644 \newcommand*{\glssgetwidestsubname}[1]{%
11645   \ifcsundef{\@glswidestname\romannumeral#1}%
11646   {\@glswidestname}%
11647   {\csuse{\@glswidestname\romannumeral#1}}%
11648 }

```

`\glssFindWidestTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glssfindwidesttoplevelname`.

```

11649 \let\glssFindWidestTopLevelName\glssfindwidesttoplevelname

```

`\glssUsedTopLevelName` Like `\glssfindwidesttoplevelname` 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.

```

11650 \newrobustcmd*{\glssFindWidestUsedTopLevelName}[1][\@glo@types]{%
11651   \dimen@=0pt\relax
11652   \gls@tmplen=0pt\relax
11653   \forallglossaries[#1]{\@gls@type}%
11654   {%
11655     \forallglsentries[\@gls@type]{\@glo@label}%
11656     {%
11657       \ifglssused{\@glo@label}%
11658       {%
11659         \ifglshasparent{\@glo@label}%
11660         {}%
11661         {%
11662           \settowidth{\dimen@}%
11663             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11664           \ifdim\dimen@>\gls@tmplen
11665             \gls@tmplen=\dimen@
11666             \eglssetwidest{\glsentryname{\@glo@label}}%
11667           \fi
11668         }%
11669       }%
11670     }%
11671   }%
11672 }%
11673 }

```

**destUsedAnyName** Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11674 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11675   \dimen@=0pt\relax
11676   \gls@tmplen=0pt\relax
11677   \forallglossaries[#1]{\@gls@type}%
11678   {%
11679     \forallgsentries[\@gls@type]{\@glo@label}%
11680     {%
11681       \ifglsused{\@glo@label}%
11682       {%
11683         \settowidth{\dimen@}%
11684           {\glstreenamefmt{\glssentryname{\@glo@label}}}%
11685         \ifdim\dimen@>\gls@tmplen
11686           \gls@tmplen=\dimen@
11687           \eglssetwidest{\glssentryname{\@glo@label}}%
11688         \fi
11689       }%
11690     }%
11691   }%
11692 }%
11693 }

```

**ndWidestAnyName** Like the above but doesn't check if the entry has been used.

```

11694 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
11695   \dimen@=0pt\relax
11696   \gls@tmplen=0pt\relax
11697   \forallglossaries[#1]{\@gls@type}%
11698   {%
11699     \forallgsentries[\@gls@type]{\@glo@label}%
11700     {%
11701       \settowidth{\dimen@}%
11702         {\glstreenamefmt{\glssentryname{\@glo@label}}}%
11703       \ifdim\dimen@>\gls@tmplen
11704         \gls@tmplen=\dimen@
11705         \eglssetwidest{\glssentryname{\@glo@label}}%
11706       \fi
11707     }%
11708   }%
11709 }

```

**estUsedLevelTwo** This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

11710 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
11711   \dimen@=0pt\relax
11712   \dimen@i=0pt\relax
11713   \dimen@ii=0pt\relax
11714   \forallglossaries[#1]{\@gls@type}%
11715   {%

```

```

11716 \forglsentries[\@gls@type]{\@glo@label}%
11717 {%
11718 \ifglsused{\@glo@label}%
11719 {%
11720 \ifglsashasparent{\@glo@label}%
11721 {%
11722 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
11723 \ifglsashasparent{\@glo@parent}%
11724 {%
11725 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
11726 \ifglsashasparent{\@glo@parent}%
11727 {}%
11728 {%
11729 \settowidth{\gls@tmplen}%
11730 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11731 \ifdim\gls@tmplen>\dimen@ii
11732 \dimen@ii=\gls@tmplen
11733 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
11734 \fi
11735 }%
11736 }%
11737 {%
11738 \settowidth{\gls@tmplen}%
11739 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11740 \ifdim\gls@tmplen>\dimen@i
11741 \dimen@i=\gls@tmplen
11742 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
11743 \fi
11744 }%
11745 }%
11746 {%
11747 \settowidth{\gls@tmplen}%
11748 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11749 \ifdim\gls@tmplen>\dimen@
11750 \dimen@=\gls@tmplen
11751 \eglssetwidest{\glsentryname{\@glo@label}}%
11752 \fi
11753 }%
11754 }%
11755 {}%
11756 }%
11757 }%
11758 }

```

**dWidestLevelTwo** This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

11759 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
11760 \dimen@=0pt\relax
11761 \dimen@i=0pt\relax
11762 \dimen@ii=0pt\relax

```

```

11763 \forall glossaries[#1]{\@gls@type}%
11764 {%
11765   \forall sentries[\@gls@type]{\@glo@label}%
11766   {%
11767     \ifglshasparent{\@glo@label}%
11768     {%
11769       \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@label}@parent}}%
11770       \ifglshasparent{\@glo@parent}%
11771       {%
11772         \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@parent}@parent}}%
11773         \ifglshasparent{\@glo@parent}%
11774         {}%
11775         {%
11776           \settowidth{\gls@tmplen}%
11777             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
11778           \ifdim\gls@tmplen>\dimen@ii
11779             \dimen@ii=\gls@tmplen
11780             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
11781           \fi
11782         }%
11783       }%
11784     {%
11785       \settowidth{\gls@tmplen}%
11786         {\glstreenamefmt{\glstentryname{\@glo@label}}}%
11787       \ifdim\gls@tmplen>\dimen@i
11788         \dimen@i=\gls@tmplen
11789         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
11790       \fi
11791     }%
11792   }%
11793   {%
11794     \settowidth{\gls@tmplen}%
11795       {\glstreenamefmt{\glstentryname{\@glo@label}}}%
11796     \ifdim\gls@tmplen>\dimen@
11797       \dimen@=\gls@tmplen
11798       \eglssetwidest{\glstentryname{\@glo@label}}%
11799     \fi
11800   }%
11801 }%
11802 }%
11803 }

```

**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.

```

11804 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
11805   \dimen@=0pt\relax
11806   \gls@tmplen=0pt\relax
11807   #2=0pt\relax
11808   \forall glossaries[#1]{\@gls@type}%

```

```

11809  {%
11810    \forglsentries[\@gls@type]{\@glo@label}%
11811    {%
11812      \ifglsused{\@glo@label}%
11813      {%
11814        \settowidth{\dimen@}%
11815          {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11816        \ifdim\dimen@>\gls@tmplen
11817          \gls@tmplen=\dimen@
11818          \eglssetwidest{\glsentryname{\@glo@label}}%
11819        \fi
11820        \settowidth{\dimen@}%
11821          {\glsentrysymbol{\@glo@label}}%
11822        \ifdim\dimen@>#2\relax
11823          #2=\dimen@
11824        \fi
11825      }%
11826    }%
11827  }%
11828 }%
11829 }

```

**stAnyNameSymbol** Like the above but doesn't check if the entry has been used.

```

11830 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
11831   \dimen@=0pt\relax
11832   \gls@tmplen=0pt\relax
11833   #2=0pt\relax
11834   \forallglossaries[#1]{\@gls@type}%
11835   {%
11836     \forglsentries[\@gls@type]{\@glo@label}%
11837     {%
11838       \settowidth{\dimen@}%
11839         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11840       \ifdim\dimen@>\gls@tmplen
11841         \gls@tmplen=\dimen@
11842         \eglssetwidest{\glsentryname{\@glo@label}}%
11843       \fi
11844       \settowidth{\dimen@}%
11845         {\glsentrysymbol{\@glo@label}}%
11846       \ifdim\dimen@>#2\relax
11847         #2=\dimen@
11848       \fi
11849     }%
11850   }%
11851 }

```

**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.

```

11852 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
11853   \dimen@=0pt\relax
11854   \gls@tmplen=0pt\relax
11855   #2=0pt\relax
11856   #3=0pt\relax
11857   \foralllglossaries[#1]{\@gls@type}%
11858   {%
11859     \forglsentries[\@gls@type]{\@glo@label}%
11860     {%
11861       \ifglsused{\@glo@label}%
11862       {%
11863         \settowidth{\dimen@}%
11864         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11865         \ifdim\dimen@>\gls@tmplen
11866           \gls@tmplen=\dimen@
11867           \eglssetwidest{\glsentryname{\@glo@label}}%
11868         \fi
11869         \settowidth{\dimen@}%
11870         {\glsentrysymbol{\@glo@label}}%
11871         \ifdim\dimen@>#2\relax
11872           #2=\dimen@
11873         \fi
11874         \settowidth{\dimen@}%
11875         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
11876         \ifdim\dimen@>#3\relax
11877           #3=\dimen@
11878         \fi
11879       }%
11880     }%
11881   }%
11882 }%
11883 }
```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

11884 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
11885   \dimen@=0pt\relax
11886   \gls@tmplen=0pt\relax
11887   #2=0pt\relax
11888   #3=0pt\relax
11889   \foralllglossaries[#1]{\@gls@type}%
11890   {%
11891     \forglsentries[\@gls@type]{\@glo@label}%
11892     {%
11893       \settowidth{\dimen@}%
11894       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11895       \ifdim\dimen@>\gls@tmplen
11896         \gls@tmplen=\dimen@
11897         \eglssetwidest{\glsentryname{\@glo@label}}%
```



```

11898      \fi
11899      \settowidth{\dimen@}%
11900      {\glstentrysymbol{\@glo@label}}}%
11901      \ifdim\dimen@>#2\relax
11902      #2=\dimen@
11903      \fi
11904      \settowidth{\dimen@}%
11905      {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
11906      \ifdim\dimen@>#3\relax
11907      #3=\dimen@
11908      \fi
11909    }%
11910  }%
11911 }

```

**AnyNameLocation** Like the `\glstFindWidestUsedAnyNameSymbolLocation` 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.

```

11912 \newrobustcmd*{\glstFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
11913   \dimen@=0pt\relax
11914   \glstmplen=0pt\relax
11915   #2=0pt\relax
11916   \forallglossaries[#1]{\@gls@type}%
11917   {%
11918     \forallglstentries[\@gls@type]{\@glo@label}%
11919     {%
11920       \ifglstused{\@glo@label}%
11921       {%
11922         \settowidth{\dimen@}%
11923         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
11924         \ifdim\dimen@>\glstmplen
11925         \glstmplen=\dimen@
11926         \eglstsetwidest{\glstentryname{\@glo@label}}%
11927         \fi
11928         \settowidth{\dimen@}%
11929         {\GlsXtrFormatLocationList{\glstentrynumberlist{\@glo@label}}}%
11930         \ifdim\dimen@>#2\relax
11931         #2=\dimen@
11932         \fi
11933       }%
11934     }%
11935   }%
11936 }%
11937 }

```

**AnyNameLocation** Like the `\glstFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

11938 \newrobustcmd*{\glstFindWidestAnyNameLocation}[2][\@glo@types]{%
11939   \dimen@=0pt\relax
11940   \glstmplen=0pt\relax

```

```

11941 #2=Opt\relax
11942 \foralllglossaries[#1]{\@gls@type}%
11943 {%
11944   \forglsentries[\@gls@type]{\@glo@label}%
11945   {%
11946     \settowidth{\dimen@}%
11947     {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11948     \ifdim\dimen@>\gls@tmplen
11949       \gls@tmplen=\dimen@
11950       \eglssetwidest{\glsentryname{\@glo@label}}%
11951     \fi
11952     \settowidth{\dimen@}%
11953     {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
11954     \ifdim\dimen@>#2\relax
11955       #2=\dimen@
11956     \fi
11957   }%
11958 }%
11959 }

```

**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`.

```

11960 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
11961   \glstreeindent=\glsxtrtreetopindent\relax
11962 }

```

**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.

```

11963 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
11964   \ifcsundef{@glswidestname\romannumeral#1}%
11965   {%
11966     \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
11967   }%
11968   {%
11969     \settowidth{#3}{\glstreenamefmt{%
11970       \csname @glswidestname\romannumeral#1\endcsname\space}}%
11971   }%
11972 }

```

**treeSetHangIndent** Set `\hangindent` for top-level entries:

```

11973 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
11974 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
11975 \renewglossarystyle{alttree}{%
11976   \renewenvironment{theglossary}%
11977     {%
11978       \glxtralttreeInit
11979       \def\@gls@prevlevel{-1}%
11980       \mbox{}\par}%
11981     {\par}%
11982   \renewcommand*{\glossaryheader}{}%
11983   \renewcommand*{\glsgroupheading}[1]{}%
11984   \renewcommand{\glossentry}[2]{%
11985     \ifnum\@gls@prevlevel=0\relax
11986     \else
11987       \glxtrComputeTreeIndent{##1}%
11988     \fi
11989     \parindent\glstreeindent
11990     \glxtrAltTreeSetHangIndent
11991     \makebox[0pt][r]%
11992     {%
11993       \glstreenamebox{\glstreeindent}%
11994       {%
11995         \glssentryitem{##1}%
11996         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11997       }%
11998     }%
11999     \glxtralttreeSymbolDescLocation{##1}{##2}%
12000     \def\@gls@prevlevel{0}%
12001   }
12002   \renewcommand{\subglossentry}[3]{%
12003     \ifnum##1=1\relax
12004       \glssubentryitem{##2}%
12005     \fi
12006     \ifnum\@gls@prevlevel=##1\relax
12007     \else
12008       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12009       \ifnum\@gls@prevlevel<##1\relax
12010         \setlength\glstreeindent\gls@tmplen
12011         \addtolength\glstreeindent\parindent
12012         \parindent\glstreeindent
12013       \else
12014         \ifnum\@gls@prevlevel=0\relax
12015           \glxtrComputeTreeIndent{##2}%
12016         \else
12017           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12018         \fi
12019         \addtolength\parindent{-\glstreeindent}%

```

```

12020         \setlength\glstreeindent\parindent
12021     \fi
12022 \fi
12023 \glxtrAltTreeSetSubHangIndent{##1}%
12024 \makebox[Opt][r]{\glstreenamebox{\glstmplen}{%
12025     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12026 \glxtralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12027 \def\@gls@prevlevel{##1}%
12028 }%
12029 \renewcommand*\@gls@groupskip{\ifglsnogroupskip\else\indexspace\fi}%
12030 }
12031 }%
12032 {%
12033 }

```

Redefine `almtreegroup` so that it discourages a break after group headings. Can't use `\@afterheading` here as it messes with the first item of the group.

```

12034 \ifdef{\@glsstyle@almtreegroup}
12035 {%
12036 \renewglossarystyle{almtreegroup}{%
12037 \setglossarystyle{almtree}%
12038 \renewcommand{\gls@groupheading}[1]{\par
12039 \def\@gls@prevlevel{-1}%
12040 \hangindentOpt\relax
12041 \parindentOpt\relax
12042 \glstreegroupheaderfmt{\gls@getgrouptitle{##1}}}%
12043 \nopagebreak\indexspace\nopagebreak
12044 }%
12045 }%
12046 }%
12047 {%
12048 }

```

Similarly for `almtreehypergroup`.

```

12049 \ifdef{\@glsstyle@almtreehypergroup}
12050 {%
12051 \renewglossarystyle{almtreehypergroup}{%
12052 \setglossarystyle{almtree}%
12053 \renewcommand*\@gls@groupheader{%
12054 \par
12055 \def\@gls@prevlevel{-1}%
12056 \hangindentOpt\relax
12057 \parindentOpt\relax
12058 \glstreenavigationfmt{\gls@navigation}\par\indexspace
12059 }%
12060 \renewcommand*\@gls@groupheading}[1]{%
12061 \par
12062 \def\@gls@prevlevel{-1}%
12063 \hangindentOpt\relax
12064 \parindentOpt\relax

```

```

12065 \glstreegroupheaderfmt
12066 {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12067 \nopagebreak\indexspace\nopagebreak
12068 }%
12069 }
12070 }%
12071 {%
12072 }

```

## 2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12073 \ifdef{\@glsstyle@mcolindexgroup}
12074 {%
12075 \renewglossarystyle{mcolindexgroup}{%
12076 \setglossarystyle{mcolindex}%
12077 \renewcommand*\{glsgroupheading}[1]{%
12078 \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12079 \nopagebreak\indexspace\nobreak\@afterheading
12080 }%
12081 }
12082 }%
12083 {%
12084 }

```

Similarly for mcolindexhypergroup.

```

12085 \ifdef{\@glsstyle@mcolindexhypergroup}
12086 {%
12087 \renewglossarystyle{mcolindexhypergroup}{%
12088 \setglossarystyle{mcolindex}%
12089 \renewcommand*\{glossaryheader}{%
12090 \item\glstreenavigationfmt{\glsnavigation}%
12091 \indexspace
12092 }%
12093 \renewcommand*\{glsgroupheading}[1]{%
12094 \item\glstreegroupheaderfmt
12095 {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12096 \nopagebreak\indexspace\nobreak\@afterheading
12097 }%
12098 }
12099 }%
12100 {%
12101 }

```

Similarly for mcolindexspannav.

```

12102 \ifdef{\@glsstyle@mcolindexspannav}
12103 {%
12104 \renewglossarystyle{mcolindexspannav}{%
12105 \setglossarystyle{index}%

```

```

12106 \renewenvironment{theglossary}%
12107 {%
12108 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12109 \setlength{\parindent}{0pt}%
12110 \setlength{\parskip}{0pt plus 0.3pt}%
12111 \let\item\glstreeitem}%
12112 {\end{multicols}}}%
12113 \renewcommand*{\glsgroupheading}[1]{%
12114 \item\glstreegroupheaderfmt
12115 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12116 \nopagebreak\indexspace\nobreak\@afterheading
12117 }%
12118 }
12119 }%
12120 {%
12121 }

```

Similarly for mcoltreegroup.

```

12122 \ifdef{\@glstyle@mcoltreegroup}
12123 {%
12124 \renewglossarystyle{mcoltreegroup}{%
12125 \setglossarystyle{mcoltree}%
12126 \renewcommand{\glsgroupheading}[1]{\par
12127 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12128 \nopagebreak\indexspace\nobreak\@afterheading
12129 }%
12130 }
12131 }%
12132 {%
12133 }

```

Similarly for mcoltreehypergroup.

```

12134 \ifdef{\@glstyle@mcoltreehypergroup}
12135 {%
12136 \renewglossarystyle{mcoltreehypergroup}{%
12137 \setglossarystyle{mcoltree}%
12138 \renewcommand*{\glossaryheader}{%
12139 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12140 }%
12141 \renewcommand*{\glsgroupheading}[1]{%
12142 \par\noindent
12143 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12144 \nopagebreak\indexspace\nobreak\@afterheading
12145 }%
12146 }
12147 }%
12148 {%
12149 }

```

Similarly for mcoltreespannav.

```

12150 \ifdef{\@glstyle@mcoltreespannav}

```

```

12151 {%
12152   \renewglossarystyle{mcoltreespannav}{%
12153     \setglossarystyle{tree}%
12154     \renewenvironment{theglossary}%
12155     {%
12156       \begin{multicols}{\glsmcols}%
12157       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12158       \setlength{\parindent}{0pt}%
12159       \setlength{\parskip}{0pt plus 0.3pt}%
12160     }%
12161     {\end{multicols}}}%
12162   \renewcommand*{\glsgroupheading}[1]{%
12163     \par\noindent
12164     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12165     \nopagebreak\indexspace\nobreak\@afterheading
12166   }%
12167 }
12168 }%
12169 {%
12170 }

```

Similarly for mcoltreenonamegroup.

```

12171 \ifdef{\@glsstyle@mcoltreenonamegroup}
12172 {%
12173   \renewglossarystyle{mcoltreenonamegroup}{%
12174     \setglossarystyle{mcoltreenoname}%
12175     \renewcommand*{\glsgroupheading}[1]{\par
12176       \noindent\glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%
12177     \nopagebreak\indexspace\nobreak\@afterheading
12178   }%
12179 }
12180 }%
12181 {%
12182 }

```

Similarly for mcoltreenonamehypergroup.

```

12183 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12184 {%
12185   \renewglossarystyle{mcoltreenonamehypergroup}{%
12186     \setglossarystyle{mcoltreenoname}%
12187     \renewcommand*{\glossaryheader}{%
12188       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12189     \renewcommand*{\glsgroupheading}[1]{%
12190       \par\noindent
12191       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12192       \nopagebreak\indexspace\nobreak\@afterheading}%
12193   }
12194 }%
12195 {%
12196 }

```

Similarly for mcoltreenonamespannav.

```

12197 \ifdef{\@glsstyle@mcoltreenonamespannav}
12198 {%
12199   \renewglossarystyle{mcoltreenonamespannav}{%
12200     \setglossarystyle{treenoname}%
12201     \renewenvironment{theglossary}%
12202     {%
12203       \begin{multicols}{\glsmcols}%
12204       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12205       \setlength{\parindent}{0pt}%
12206       \setlength{\parskip}{0pt plus 0.3pt}%
12207     }%
12208     {\end{multicols}}%
12209     \renewcommand*{\glsgroupheading}[1]{%
12210       \par\noindent
12211       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12212       \nopagebreak\indexspace\nobreak\@afterheading}%
12213   }
12214 }%
12215 {%
12216 }

```

mcolalmtree needs adjusting so that it uses \glxtralttreeInit This doesn't use \mbox{} \par which would unbalance the top of the columns.

```

12217 \ifdef{\@glsstyle@mcolalmtree}
12218 {%
12219   \renewglossarystyle{mcolalmtree}{%
12220     \setglossarystyle{almtree}%
12221     \renewenvironment{theglossary}%
12222     {%
12223       \glxtralttreeInit
12224       \def\@gls@prevlevel{-1}%
12225       \begin{multicols}{\glsmcols}%
12226     }%
12227     {\par\end{multicols}}%
12228   }
12229 }%
12230 {%
12231 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12232 \ifdef{\@glsstyle@mcolalmtreegroup}
12233 {%
12234   \renewglossarystyle{mcolalmtreegroup}{%
12235     \setglossarystyle{mcolalmtree}%
12236     \renewcommand{\glsgroupheading}[1]{\par
12237       \def\@gls@prevlevel{-1}%
12238       \hangindent0pt\relax
12239       \parindent0pt\relax
12240       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```



```

12241     \nopagebreak\indexspace\nopagebreak
12242   }%
12243 }
12244 }%
12245 {%
12246 }

```

Similarly for mcolalmtreehypergroup.

```

12247 \ifdef{\@glsstyle@mcolalmtreehypergroup}
12248 {%
12249   \renewglossarystyle{mcolalmtreehypergroup}{%
12250     \setglossarystyle{mcolalmtree}%
12251     \renewcommand*{\glossaryheader}{%
12252       \par
12253       \def\@gls@prevlevel{-1}%
12254       \hangindent0pt\relax
12255       \parindent0pt\relax
12256       \glstreenavigationfmt{\glsnavigation}%
12257       \par\indexspace
12258     }%
12259     \renewcommand*{\glsgroupheading}[1]{%
12260       \par
12261       \def\@gls@prevlevel{-1}%
12262       \hangindent0pt\relax
12263       \parindent0pt\relax
12264       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12265       \nopagebreak\indexspace\nopagebreak
12266     }%
12267   }
12268 }%
12269 {%
12270 }

```

Similarly for mcolaltreespannav.

```

12271 \ifdef{\@glsstyle@mcolaltreespannav}
12272 {%
12273   \renewglossarystyle{mcolaltreespannav}{%
12274     \setglossarystyle{almtree}%
12275     \renewenvironment{theglossary}%
12276     {%
12277       \glsxtralmtreeInit
12278       \def\@gls@prevlevel{-1}%
12279       \begin{multicols}{\glsncols}%
12280         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12281     }%
12282     {\par\end{multicols}}%
12283     \renewcommand*{\glsgroupheading}[1]{%
12284       \par
12285       \def\@gls@prevlevel{-1}%
12286       \hangindent0pt\relax

```

```

12287      \parindent0pt\relax
12288      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12289      \nopagebreak\indexspace\nopagebreak
12290    }%
12291  }
12292 }%
12293 {%
12294 }

Reset the default style
12295 \ifx\@glossary@default@style\relax
12296 \else
12297   \setglossarystyle{\@glxtr@current@style}
12298 \fi

```

# 3 bookindex style (glossary-bookindex.sty)

## 3.1 Package Initialisation and Options

```
12299 \NeedsTeXFormat{LaTeX2e}
12300 \ProvidesPackage{glossary-bookindex}[2017/11/03 v1.21 (NLCT)]
```

Load required packages.

```
12301 \RequirePackage{multicol}
12302 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12303 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12304 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`ookindexsubname` Format used for sub entries.

```
12305 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`xsxtrprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12306 \providecommand*{\glsxtrprelocation}{\space}
```

`ndexprelocation` Separator used before location list for top-level entries.

```
12307 \newcommand*{\glsxtrbookindexprelocation}[1]{%
12308   \glsxtrifhasfield{location}{#1}%
12309   {\ifglsnopostdot,\fi\glsxtrprelocation}%
12310   {\glsxtrprelocation}%
12311 }
```

`xsubprelocation` Separator used before location list for sub-entries.

```
12312 \newcommand*{\glsxtrbookindexsubprelocation}[1]{%
12313   \glsxtrbookindexprelocation{#1}%
12314 }
```

`xparentchildsep` Separator used between top-level parent and child entry.

```
12315 \newcommand{\glsxtrbookindexparentchildsep}{\nopagebreak}
```

`rentsubchildsep` Separator used between sub-level parent and child entry.

```
12316 \newcommand{\glsxtrbookindexparentsubchildsep}{\glsxtrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
12317 \newcommand{\glstrbookindexbetween}[2] {}
```

`indexsubbetween` Between two level 1 entries identified by the labels in the arguments.

```
12318 \newcommand{\glstrbookindexsubbetween}[2] {}
```

`exsubsubbetween` Between two level 2 entries identified by the labels in the arguments.

```
12319 \newcommand{\glstrbookindexsubsubbetween}[2] {}
```

`indexatendgroup` At the end of a letter group. The argument is the index of the last top-level entry.

```
12320 \newcommand{\glstrbookindexatendgroup}[1] {}
```

`exsubatendgroup` At the end of a letter group. The argument is the index of the last level 1 entry.

```
12321 \newcommand{\glstrbookindexsubatendgroup}[1] {}
```

`subsubatendgroup` At the end of a letter group. The argument is the index of the last level 2 entry.

```
12322 \newcommand{\glstrbookindexsubsubatendgroup}[1] {}
```

`bookindexgroupskip` Group separator.

```
12323 \newcommand{\glstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

`indexformatheader` Group separator.

```
12324 \newcommand*{\glstrbookindexformatheader}[1] {%
12325   \par{\centering\glstreegroupheaderfmt{#1}\par}%
12326 }
```

`bookindexbookmark` Book mark group heading if supported.

```
12327 \ifdef\pdfbookmark
12328 {%
12329   \newcommand*{\glstrbookindexbookmark}[2] {%
12330     \ifdefstring{\@glossarysec}{chapter}%
12331     {\pdfbookmark[1]{#1}{#2}}%
12332     {\pdfbookmark[2]{#1}{#2}}%
12333   }
12334 }
12335 {%
12336   \newcommand*{\glstrbookindexbookmark}[2] {}
12337 }
```

`bookindexcolspread`

```
12338 \newcommand*{\glstrbookindexcolspread} {}
```

Define the style.

```
12339 \newglossarystyle{bookindex} {%
12340   \setglossarystyle{index}%
12341   \renewenvironment{theglossary}%
```

```

12342 {%
12343   \ifdefempty\glxstrbookindexcolspread
12344   {\begin{multicols}{\glxstrbookindexcols}}%
12345   {\begin{multicols}{\glxstrbookindexcols}[\glxstrbookindexcolspread]}%
12346   \setlength{\parindent}{0pt}%
12347   \setlength{\parskip}{0pt plus 0.3pt}%
12348   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12349   \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
12350   \let\@glxstr@bookindex@between\@gobble
12351   \let\@glxstr@bookindex@subbetween\@gobble
12352   \let\@glxstr@bookindex@subsubbetween\@gobble
12353   \let\@glxstr@bookindex@atendgroup\relax
12354   \let\@glxstr@bookindex@subatendgroup\relax
12355   \let\@glxstr@bookindex@subsubatendgroup\relax
12356   \let\@glxstr@bookindex@groupskip\relax
12357 }%
12358 {%

```

Do end group hooks.

```

12359   \@glxstr@bookindex@subsubatendgroup
12360   \@glxstr@bookindex@subatendgroup
12361   \@glxstr@bookindex@atendgroup

```

End multicols environment.

```

12362   \end{multicols}%
12363 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12364 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12365 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12366   \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12367   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12368   \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
12369   \let\@glxstr@bookindex@subbetween\@gobble
12370   \let\@glxstr@bookindex@subsubbetween\@gobble
12371   \edef\@glxstr@bookindex@between{%
12372     \noexpand\glxstrbookindexbetween{##1}%
12373   }%
12374   \edef\@glxstr@bookindex@atendgroup{%
12375     \noexpand\glxstrbookindexatendgroup{##1}%
12376   }%
12377   \let\@glxstr@bookindex@subatendgroup\relax
12378   \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12379   \glstreeitem

```

```

12380     \glstryitem{##1}%
12381     \glstarget{##1}{\glxtrbookindexname{##1}}%
12382     \glxtrbookindexprelocation{##1}##2%
12383 }%
12384 \renewcommand{\subglossentry}[3]{%
12385     \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12386     \glstreeitem
12387     \or

```

Level 1.

```

12388     \@glxtr@bookindex@sep
12389     \@glxtr@bookindex@subbetween{##2}%
12390     \let\@glxtr@bookindex@sep\relax

```

Update separators.

```

12391     \let\@glxtr@bookindex@subsubbetween\@gobble
12392     \let\@glxtr@bookindex@subsep\glxtrbookindexparentschildsep
12393     \edef\@glxtr@bookindex@subbetween{%
12394         \noexpand\glxtrbookindexsubbetween{##2}%
12395     }%
12396     \edef\@glxtr@bookindex@atsubendgroup{%
12397         \noexpand\glxtrbookindexatsubendgroup{##1}%
12398     }%

```

Start sub-item.

```

12399     \glstreesubitem
12400     \glssubentryitem{##2}%
12401     \else

```

All other levels.

```

12402     \@glxtr@bookindex@subsep
12403     \@glxtr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12404     \let\@glxtr@bookindex@subsep\relax
12405     \edef\@glxtr@bookindex@subsubbetween{%
12406         \noexpand\glxtrbookindexsubsubbetween{##2}%
12407     }%
12408     \edef\@glxtr@bookindex@atsubsubendgroup{%
12409         \noexpand\glxtrbookindexatsubsubendgroup{##1}%
12410     }%

```

Start sub-sub-item.

```

12411     \glstreesubsubitem
12412     \fi

```

Format entry.

```

12413     \glstarget{##2}{\glxtrbookindexsubname{##2}}%
12414     \glxtrbookindexsubprelocation{##2}##3%
12415 }%

```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12416 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12417 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12418 \@glstr@bookindex@subsubatendgroup
```

```
12419 \@glstr@bookindex@subatendgroup
```

```
12420 \@glstr@bookindex@atendgroup
```

```
12421 \@glstr@bookindexgroupskip
```

Update separators.

```
12422 \let\@glstr@bookindexgroupskip\glstrbookindexgroupskip
```

```
12423 \let\@glstr@bookindex@between\@gobble
```

```
12424 \let\@glstr@bookindex@atendgroup\relax
```

```
12425 \let\@glstr@bookindex@subatendgroup\relax
```

```
12426 \let\@glstr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12427 \glstrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12428 \glstrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12429 \glstrbookindexformatheader{\thisgrptitle}%
```

```
12430 \nopagebreak\indexspace\nopagebreak\@afterheading
```

```
12431 }%
```

```
12432 }
```

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 `\glstrbookindexthepage` 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.

```
12433 \newcommand{\glstrbookindexthepage}{%
```

```
12434 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
```

```
12435 }
```

`bookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
12436 \newcommand*{\glstrbookindexmarkentry}[1]{%
```

```
12437 \protected@write\@auxout
```

```
12438 {\let\glstrbookindexthepage\relax}%
```

```
12439 {\string\glstr@setbookindexmark{\glstrbookindexthepage}{#1}}%
```

```
12440 }
```

etbookindexmark

```
12441 \newcommand*{\glxtr@setbookindexmark}[2]{%
12442   \ifcsundef{glxtr@idxfirstmark@#1}%
12443   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
12444   {}%
12445   \csgdef{glxtr@idxlastmark@#1}{#2}%
12446 }
```

dexfirstmarkfmt

```
12447 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
12448   \glentryname{#1}%
12449 }
```

kindexfirstmark

```
12450 \newcommand*{\glxtrbookindexfirstmark}{%
12451   \letcs{glxtr@label}{glxtr@idxfirstmark@glxtrbookindexthepage}%
12452   \ifdefglxtr@label
12453   {\glxtrbookindexfirstmarkfmt{glxtr@label}}%
12454   {}%
12455 }
```

ndexlastmarkfmt

```
12456 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
12457   \glentryname{#1}%
12458 }
```

okindexlastmark

```
12459 \newcommand*{\glxtrbookindexlastmark}{%
12460   \letcs{glxtr@label}{glxtr@idxlastmark@glxtrbookindexthepage}%
12461   \ifdefglxtr@label
12462   {\glxtrbookindexlastmarkfmt{glxtr@label}}%
12463   {}%
12464 }
```



# 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	64
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	64
\Glsfmtshort: new	298	\@Glstext@: added redefinition	59
\glsfmtshort: new	297	\@Glsuseri@: added redefinition	64
\Glsfmtshortpl: new	298	\@Glsuserii@: added redefinition	65
\glsfmtshortpl: new	297	\@Glsuseriii@: added redefinition	65
short: switched inline full form to short		\@Glsuseriv@: added redefinition	65
(long)	201	\@Glsuserv@: added redefinition	66
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	66
\@ACRlong: added redefinition	69	\@acrlong: added redefinition	68
\@ACRlongpl: added redefinition	70	\@acrlongpl: added redefinition	69
\@ACRshort: added redefinition	67	\@acrshort: added redefinition	66
\@ACRshortpl: added redefinition	68	\@acrshortpl: added redefinition	67
\@Acrlong: added redefinition	69	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	70	argument	53
\@Acrshort: added redefinition	67	\@glsdescplural@: added redefinition	63
\@Acrshortpl: added redefinition	68	\@glsfirst@: added redefinition	60
\@GLSdesc@: added redefinition	63	\@glsfirstplural@: added redefinition	61
\@GLSdescplural@: added redefinition	63	\@glsplural@: added redefinition	61
\@GLSfirst@: added redefinition	60	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	62	redefinition	64
\@GLSname@: added redefinition	62	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	61	new	119
\@GLSSymbol@: added redefinition	64	\@glsxtr@field@linkdefs: new	59
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	171
redefinition	64	\@print@glossary: added redefinition	115
\@GLStext@: added redefinition	59	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	65	\abbrvdefaultfont	177
\@GLSuserii@: added redefinition	65	\glsaccessdesc: new	141
\@GLSuseriii@: added redefinition	65	\glsaccessdescplural: new	141
\@GLSuseriv@: added redefinition	66	\glsaccessfirst: new	138
\@GLSuserv@: added redefinition	66	\glsaccessfirstplural: new	139
\@GLSuservi@: added redefinition	66	\Glsaccesslong: new	143
\@Glsdesc@: added redefinition	63	\glsaccesslong: new	143
\@Glsdescplural@: added redefinition	63	\glsaccessname: new	137
\@Glsfirst@: added redefinition	60	\glsaccessplural: new	138
\@Glsfirstplural@: added redefinition	61	\Glsaccessshort: new	142
\@Glsname@: added redefinition	62	\glsaccessshort: new	142
\@Glsplural@: added redefinition	61	\Glsaccessshortpl: new	143

\glsaccesssshortpl: new .....	143	\cGLSpl: new .....	94
\glsaccessssymbol: new .....	140	\cGLSpl@: new .....	94
\glsaccessssymbolplural: new .....	140	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new .....	137	new .....	89
\glstentryfmt: added check for short ..	53	\cGLS: new .....	93
\glslongpltok: new .....	171	\cGLSformat: new .....	94
\glsshortpltok: new .....	171	\cGLSpl: new .....	94
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new .....	94
name in \setkeys .....	173	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new .....	15
for plural .....	168	\glsenableentrycount: new .....	89
\GLSxtrlongpl: new .....	186	\glsfirstabbrvdefaultfont: new ..	176
\Glsxtrlongpl: new .....	185	\glsfirstlongdefaultfont: new ...	177
\glsxtrlongpl: new .....	185	\Glsfmtfirst: new .....	300
\glsxtrNoGlossaryWarning: new ....	20	\glsfmtfirst: new .....	300
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new .....	301
new .....	168	\glsfmtfirstpl: new .....	301
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new .....	300
new .....	168	\glsfmtplural: new .....	299
\glsxtrpostlinkendsentence: new ..	167	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new .....	184	\Glsxtrtitleshort .....	298
\Glsxtrshortpl: new .....	184	renamed from \glstentryfmtshort ..	298
\glsxtrshortpl: new .....	183	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort .....	297
\glslabeltok .....	196	renamed from \glstentryfmtshort ..	297
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok .....	194	\Glsxtrtitleshortpl .....	298
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl .....	298
redefinition of \acronymtype .....	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl .....	297
\Glsxtrshort .....	298	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl .....	297
\glsxtrshort .....	297	\Glsfmttext: new .....	299
\Glsfmtshortpl: changed to use		\glsfmttext: new .....	299
\glsxtrshortpl .....	298	\glshasattribute: new .....	149
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	148
\glsxtrshortpl .....	297	\glsxtremsuffix: new .....	237
\glsxtrifemptyglossary: new .....	25	\GlsXtrEnableEntryCounting: new ..	88
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new .....	91
argument .....	152	\glsxtrscfont: new .....	209
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new .....	209
argument .....	152	\glsxtrsmfont: new .....	223
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new .....	223
default type to \acronymtype ....	102	short-em: new .....	244
\newterm: fixed name argument .....	151	short-em-desc: new .....	246
0.5 (2015-12-07)		short-em-footnote: new .....	255
\cGLS: new .....	93	short-em-long: new .....	241
\cGLS@: new .....	94	short-em-long-desc: new .....	242

short-em-postfootnote: new .....	257	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new .....	219	attribute .....	288
short-sc-postfootnote: new .....	221	\Glsxtrheadtext: now uses headuc	
short-sm: new .....	227	attribute .....	291
short-sm-desc: new .....	228	\glxtrheadtext: now uses headuc	
short-sm-footnote: new .....	234	attribute .....	290
short-sm-long: new .....	225	short-em-footnote: switch off regular	
short-sm-long-desc: new .....	226	attribute if set .....	255
short-sm-postfootnote: new .....	235	short-long: switch off regular attribute	
long-noshort-em: new .....	248	if set .....	195
long-noshort-em-desc: new .....	252	short-long-desc: switch off regular	
long-noshort-sm: new .....	230	attribute if set .....	197
long-noshort-sm-desc: new .....	232	short-sc-footnote: switch off regular	
long-short-em: new .....	238	attribute if set .....	220
long-short-em-desc: new .....	239	short-sm-footnote: switch off regular	
long-short-sm: new .....	223	attribute if set .....	234
long-short-sm-desc: new .....	225	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set .....	193
\Glsaccessstext: new .....	138	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set .....	195
\@glxtr@doaccsupp: new .....	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	288	attribute if set .....	211
\Glsaccessdesc: new .....	141	footnote: switch off regular attribute if	
\Glsaccessdescplural: new .....	142	set .....	198
\Glsaccessfirst: new .....	139	postfootnote: switch off regular	
\Glsaccessfirstplural: new .....	139	attribute if set .....	199
\Glsaccessname: new .....	137	0.5.2 (2015-12-08)	
\Glsaccessplural: new .....	138	\@GLSdesc@: added accessibility support	63
\Glsaccesssymbol: new .....	140	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new .....	140	support .....	63
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute .....	292	support .....	60
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute .....	292	support .....	62
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	62
headuc attribute .....	293	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support .....	61
headuc attribute .....	293	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support .....	64
attribute .....	292	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support .....	64
attribute .....	291	\@GLStext@: added accessibility support	59
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	63
attribute .....	289	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support .....	63
attribute .....	288	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support .....	60
attribute .....	289	\@GLSfirstplural@: added accessibility	
		support .....	61

\@Glsname@: add accessibility support ..	62	\GLSaccesssymbolplural: new ..	141, 146
\@Glsplural@: added accessibility support .....	61	\GLSaccessstext: new .....	138, 145
\@Glsymbol@: added accessibility support .....	64	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support .....	64	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	59	\glsxtrabbrvfmt to here .....	53
\@glsdesc@: added accessibility support	62	\GlsXtrEnableInitialTagging: new	164
\@glsdescplural@: added accessibility support .....	63	\glsxtrfieldtitlecase: new .....	153
\@glsfirst@: added accessibility support .....	60	\GlsXtrFormatLocationList: new ...	50
\@glsfirstplural@: added accessibility support .....	61	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	62	new .....	175
\@Glsplural@: added accessibility support .....	61	\glsxtrtagfont: new .....	166
\@Glsymbol@: added accessibility support .....	63	\KV@printgloss@nonumberlist: added	52
\@Glsymbolplural@: added accessibility support .....	64	\mfu@checkword@do: added .....	165
\@Glstext@: added accessibility support	59	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch ....	190
new .....	166	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	166	\@glsxtr@autoindex@at: new .....	161
\@glsxtr@tag: new .....	166	\@glsxtr@autoindex@encap: new ...	162
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new .....	162
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	162
\@ifpackageloaded .....	137	\@glsxtr@autoindex@setname: new ..	160
removed \glsxtrabbrvfmt .....	187	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added .....	50	General: removed	
\Glossentrydesc: added .....	164	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added .....	157	.....	118
\Glossentrysymbol: added .....	164	\glsdescwidth: added .....	49
\GLSaccessdesc: new .....	141, 146	\glspagelistwidth: added .....	50
\GLSaccessdescplural: new ...	142, 146	\glsxtrdoautoindexname: new .....	159
\GLSaccessfirst: new .....	139, 145	\glsxtrpostnamehook: new .....	159
\GLSaccessfirstplural: new ..	139, 145	\if@glsxtr@format@override: new ..	159
\GLSaccesslong: new .....	143, 147	\ProvidesGlossariesExtraLang: new	304
\GLSaccesslongpl: new .....	144, 147	\RequireGlossariesExtraLang: new	304
\Glsaccesslongpl: new .....	144	0.5.4 (2015-12-15)	
\glsaccesslongpl: new .....	144	\@newglossaryentry@defunitcounters:	
\GLSaccessname: new .....	137, 144	new .....	95
\GLSaccessplural: new .....	138, 145	\@GLSxtr@p@acrlong@: new .....	82
\GLSaccesssshort: new .....	142, 146	\@GLSxtr@p@acrlongpl@: new .....	82
\GLSaccesssshortpl: new .....	143, 147	\@GLSxtr@p@acrshort@: new .....	82
\GLSaccesssymbol: new .....	140, 145	\@GLSxtr@p@acrshortpl@: new .....	82
		\@GLSxtr@p@long@: new .....	81
		\@GLSxtr@p@longpl@: new .....	82
		\@GLSxtr@p@plural@: new .....	80
		\@GLSxtr@p@short@: new .....	81
		\@GLSxtr@p@shortpl@: new .....	81
		\@GLSxtr@p@text@: new .....	80
		\@GlsXtrEnableOnTheFly: new .....	46
		\@Glsxtr: new .....	46
		\@Glsxtr@p@acrlong@: new .....	82
		\@Glsxtr@p@acrlongpl@: new .....	82

\@Glsxtr@p@acrshort@: new .....	82	\glsenableentryunitcount: new ....	97
\@Glsxtr@p@acrshortpl@: new .....	82	\glshasattribute: added check for	
\@Glsxtr@p@long@: new .....	81	entry's existence .....	149
\@Glsxtr@p@longpl@: new .....	81	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new .....	80	entry's existence .....	149
\@Glsxtr@p@short@: new .....	80	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new .....	81	check .....	167
\@Glsxtr@p@text@: new .....	80	\Glsxtr: new .....	46
\@Glsxtrpl: new .....	47	\glxtr: new .....	46
\@alt@glshyp@opt: new .....	76	\glxtrcat: new .....	46
\@glscalt@hyp@opt: new .....	76	\glxtrdowrglossaryhook: new ....	76
\@glscalt@hyp@opt@char: new .....	76	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new .....	76	new .....	100
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new .....	45
new .....	96	\Glsxtrpl: new .....	47
\@glsc@local@increment@currunitcount:		\glxtrpl: new .....	47
new .....	96	\glxtrpostlocalreset: new .....	88
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new .....	88
new .....	73	\glxtrpostreset: new .....	88
\@glsxtr: new .....	46	\glxtrpostunset: new .....	88
\@glsxtr@addunitcounter: new ....	95	\glxtrprotectlinks: new .....	79
\@glsxtr@currunitcount: new ....	97	\GlsXtrSetAltModifier: new .....	76
\@glsxtr@ifunitcounter: new ....	95	\GlsXtrSetDefaultGlsOpts: new ....	75
\@glsxtr@p@acrlong@: new .....	82	\glxtrstarflywarn: new .....	46
\@glsxtr@p@acrlongpl@: new .....	82	\GlsXtrWarning: new .....	48
\@glsxtr@p@acrshort@: new .....	82	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new .....	82	disables \setacronymstyle .....	102
\@glsxtr@p@long@: new .....	81	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new .....	81	\@glsxtr@autoindexcrossrefs: new .	15
\@glsxtr@p@plural@: new .....	80	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new .....	80	new .....	109
\@glsxtr@p@shortpl@: new .....	81	\@glsxtr@idx@entrynumberlist: new	111
\@glsxtr@p@text@: new .....	80	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new .....	97	new .....	110
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new .....	100	new .....	111
\@glsxtr@unitcountlist: new .....	95	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new .....	47	new .....	110
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new ....	103
empty see value if not set and added		\makeglossaries: new .....	104
'see' to field key map .....	37	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new ....	45	\glxtrdiscardperiod: added check	
\cGlsformat: added .....	94	for first use .....	168
\cGlsformat: added .....	94	short-desc: fixed typo in	
\cGlsplformat: added .....	95	\glxtrinlinefullformat and	
\cGlsplformat: added .....	94	added missing second argument ...	203
\glstdisablehyper: added .....	78	1.02 (2016-04-25)	
\glstdohyperlink: added .....	77	\@glsxtr@current@style: new .....	48
\glstdonohyperlink: added .....	79	\Glsfmtfull: new .....	303

\glsfmtfull: new .....	302	\@GLSdescplural@: set abbreviation and regular format .....	63
\Glsfmtfullpl: new .....	303	\@GLSfirst@: set abbreviation format ..	60
\glsfmtfullpl: new .....	303	\@GLSfirstplural@: set abbreviation and regular format .....	62
\Glsfmtlong: new .....	301	\@GLSname@: set abbreviation and regular format .....	62
\glsfmtlong: new .....	301	\@GLSplural@: set abbreviation and regular format .....	61
\Glsfmtlongpl: new .....	302	\@GLSsymbol@: set regular format .....	64
\glsfmtlongpl: new .....	302	\@GLSsymbolplural@: set regular format	64
\Glsxtrheadfull: new .....	296	\@GLStext@: set abbreviation and regular format .....	59
\glsxtrheadfull: new .....	295	\@GLSuseri@: set regular format .....	65
\Glsxtrheadfullpl: new .....	297	\@GLSuserii@: set regular format .....	65
\glsxtrheadfullpl: new .....	296	\@GLSuseriii@: set regular format .....	65
\Glsxtrheadlong: new .....	295	\@GLSuseriv@: set regular format .....	66
\glsxtrheadlong: new .....	294	\@GLSuservi@: set regular format .....	66
\Glsxtrheadlongpl: new .....	295	\@GLSdesc@: set abbreviation and regular format .....	63
\glsxtrheadlongpl: new .....	294	\@GLSdescplural@: set abbreviation and regular format .....	63
\Glsxtrtitlefull: new .....	297	\@GLSfirst@: set abbreviation and regular format .....	60
\glsxtrtitlefull: new .....	296	\@GLSfirstplural@: set abbreviation and regular format .....	61
\Glsxtrtitlefullpl: new .....	297	\@GLSname@: set abbreviation and regular format .....	62
\glsxtrtitlefullpl: new .....	296	\@GLSplural@: set abbreviation and regular format .....	61
\Glsxtrtitlelong: new .....	295	\@GLsymbol@: set regular format .....	64
\glsxtrtitlelong: new .....	294	\@GLsymbolplural@: set regular format	64
\Glsxtrtitlelongpl: new .....	295	\@GLstext@: set abbreviation and regular format .....	59
\glsxtrtitlelongpl: new .....	294	\@GLSuseri@: set regular format .....	64
\ifglsxtrinsertinside: new .....	193	\@GLSuserii@: set regular format .....	65
postfootnote: added redef of		\@GLSuseriii@: set regular format .....	65
\glsxtrsetupfulldefs .....	200	\@GLSuseriv@: set regular format .....	65
stylemods: new .....	21	\@GLSuservi@: set regular format .....	66
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format .....	63
\@GLSfirstplural@: bug fix: misspelt cs name .....	62	\@GLSdescplural@: set abbreviation and regular format .....	61
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	61	\@GLSsymbol@: set regular format .....	64
\@GLSfirstplural@: bug fix: misspelt cs name .....	61	\@GLSsymbolplural@: set regular format	64
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	61	\@GLstext@: set abbreviation and regular format .....	59
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	61	\@GLSuseri@: set regular format .....	64
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	294	\@GLSuserii@: set regular format .....	65
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	289	\@GLSuseriii@: set regular format .....	65
1.04 (2015-04-30)		\@GLSuseriv@: set regular format .....	65
short-em-footnote: renamed from “footnote-em” .....	255	\@GLSuservi@: set regular format .....	66
1.04 (2016-05-02)		\@gls@preglossaryhook: added check for entry’s existence .....	166
\@glsxtrpostloctag: new .....	52	\@GLSdesc@: set abbreviation and regular format .....	62
\@GLSdesc@: set abbreviation and regular format .....	63	\@GLSdescplural@: set abbreviation and regular format .....	63
		\@GLSfirst@: set abbreviation and regular format .....	60

\@glsfirstplural@: set abbreviation and regular format .....	61	\glsxtruserparen: new .....	259
\@glsname@: set abbreviation and regular format .....	62	\glsxtrusersuffix: new .....	260
\@glsplural@: set abbreviation and regular format .....	61	\GlsXtrWarnDeprecatedAbbrStyle: new .....	192
\@glsymbol@: set regular format .....	63	short-em-long-em: new .....	243
\@glsymbolplural@: set regular format .....	64	short-em-long-em-desc: new .....	244
\@glstext@: set abbreviation and regular format .....	59	short-em-nolong: new .....	246
\@glsxtr@deprecated@abbrstyle: new .....	192	short-em-nolong-desc: new .....	247
\@glsxtr@do@style: new .....	21	short-em-postfootnote: renamed from “postfootnote-em” .....	257
\@glsxtr@doloctag: new .....	52	short-footnote: new .....	199
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand .....	111	short-long-user: new .....	266
\@glsxtr@pagetag: new .....	52	short-long-user-desc: new .....	267
\@glsxtr@pagetag: new .....	52	short-nolong: new .....	202
\@glsxtr@preloctag: new .....	52	short-nolong-desc: new .....	204
\@glsxtr@preloctag: new .....	51	short-postfootnote: new .....	201
\glossentrydesc: added glossdescfont attribute check .....	153	short-sc-footnote: renamed from “footnote-sc” .....	219
\Glossentryname: added glossnamefont attribute check .....	157	short-sc-nolong: new .....	214
\glossentryname: added glossnamefont attribute check .....	155	short-sc-nolong-desc: new .....	215
moved post name hook inside condition .....	157	short-sc-postfootnote: renamed from “postfootnote-sc” .....	221
\glsabbrvmfont: new .....	237	short-sm-footnote: renamed from “footnote-sm” .....	234
\glsabbrvuserfont: new .....	259	short-sm-nolong: new .....	228
\glsfirstabbrvmfont: new .....	237	short-sm-nolong-desc: new .....	230
\glsfirstabbrvuserfont: new .....	259	short-sm-postfootnote: renamed from “postfootnote-sm” .....	235
\glsfirstlongemfont: new .....	237	\letabbreviationstyle: new .....	192
\glsfirstlonguserfont: new .....	259	\newabbreviationstyle: bug fix: corrected test for existence .....	190
\glsifnotregularcategory: new ...	150	long-em-noshort-em: new .....	250
\glslongdefaultfont: new .....	177	long-em-noshort-em-desc: new ...	253
\glslongemfont: new .....	237	long-em-short-em: new .....	239
\glslongfont: new .....	177	long-em-short-em-desc: new .....	240
\glslonguserfont: new .....	259	long-noshort: new .....	208
\glsxtrassignfieldfont: new .....	59	long-noshort-desc: new .....	207
\GlsXtrEnablePreLocationTag: new .	51	long-noshort-em: renamed from “long-em” .....	248
\glsxtrfirstscfont: new .....	209	long-noshort-em-desc: renamed from “long-desc-em” .....	252
\glsxtrfirstsmfont: new .....	223	long-noshort-sc: renamed from “long-sc” .....	216
\glsxtrlongshortdescsort: new ...	194	long-noshort-sc-desc: renamed from “long-desc-sc” .....	218
\glsxtrpostnamehook: added category check .....	159	long-noshort-sm: renamed from “long-sm” .....	230
\glsxtrregularfont: new .....	53	long-noshort-sm-desc: renamed from \long-desc-sm .....	232
\glsxtruserfield: new .....	259		



long-short-user: new .....	260	docdef option changed to choice .....	14
long-short-user-desc: new .....	265	\glxtr@usesee: new .....	38
\renewabbreviationstyle: new ....	191	\glxtrusesee: new .....	38
style: new .....	21	\glxtruseseeformat: new .....	38
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	14
\eglssetwidest: new .....	322	1.07 (2016-08-15)	
\glsFindWidestAnyName: new .....	324	\@@glxtrp: new .....	83
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new .....	329	nohyperfirst attribute .....	60
\glsFindWidestAnyNameSymbol: new	327	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute .....	62
new .....	328	\@GLSxtrp: new .....	84
\glsFindWidestLevelTwo: new ....	325	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new ..	323	nohyperfirst attribute .....	60
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new .....	329	nohyperfirst attribute .....	61
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new .....	83
new .....	326	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts .....	166
new .....	327	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new .	324	nohyperfirst attribute .....	60
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new .....	323	nohyperfirst attribute .....	61
\glsfirstlongfootnotefont: new ..	197	\@glxtrinmark: new .....	286
\glsgetwidestname: new .....	323	\@glxtrnotinmark: new .....	286
\glsgetwidestsubname: new .....	323	\@glxtrp: new .....	83
\glslongfootnotefont: new .....	197	\@glxtrp@opt: new .....	82
\glxtrAltTreeIndent: new .....	322	\glossxtrsetpopts: new .....	83
\glxtralttreeInit: new .....	322	\glsp: new .....	85
\glxtrAltTreePar: new .....	322	\glsp: new .....	85
\glxtrAltTreeSetHangIndent: new	330	\glxtr@entry@p: new .....	84
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new .....	197
new .....	331	\glxtrchecknohyperfirst: new ....	60
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new ....	153
new .....	322	\glxtrifinmark: new .....	286
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new .....	86
new .....	321	\Glsxtrp: new .....	85
\glxtrComputeTreeIndent: new ...	330	\glxtrp: new .....	84
\glxtrComputeTreeSubIndent: new	330	\glxtrsetpopts: new .....	83
\glxtrtreetopindent: new .....	322	short-long-desc: added text key ....	197
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form .....	241	plural key .....	197
\xglsetwidest: new .....	323	long-short-desc: added missing text	
1.06 (2016-06-18)		key .....	195
\@glsdoifexistsorwarn: new .....	14	fixed misspelling of \glsabbrvfont .	195
\@glxtr@docdefval: new .....	14	footnote: changed first forms to use	
\@glxtr@usesee: new .....	38	\glsfirstlongfootnotefont ...	197
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document .....	25	from first keys .....	199

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	200	1.10 (2016-12-17)	\@GLSp1@: fixed bug caused by typo in command name .....	54
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse .....	103	1.11 (2017-01-19)	\@glsxtr@do@redef@for@gl@sentries: new .....	6
1.08 (2016-12-13)			\@glsxtr@noidx@do: new .....	128
\@glsxtr@record: new .....	8		\@glsxtr@redef@for@gl@sentries: new .	6
\@GLS@: added \@glsxtr@record .....	54		\@glsxtr@shortcutsval: new .....	19
\@GLSp1@: added \@glsxtr@record ...	54		\@glsxtr@unsrt@getgroup@title: new	127
\@Gls@: added \@glsxtr@record .....	54		\@print@noidx@glossary: added redefinition .....	113
\@Glspl@: added \@glsxtr@record ...	54		\glsxtr@addloc@listfield: added group key .....	12
\@gls@: added \@glsxtr@record .....	54		added location key .....	12
\@gls@@link@: added \@glsxtr@record .....	55		\glsxtr@fields: new .....	121
\@gls@field@link: added \@glsxtr@record .....	53		\glsxtr@linkprefix: new .....	121
\@gls@saveentrycounter: new .....	25		\glsxtr@org@new@ignored@glossary: new .....	33
\@glsdisp: added \@glsxtr@record ..	55		\glsxtr@s@new@ignored@glossary: new	33
\@glspl@: added \@glsxtr@record ...	54		\glsxtr@shortcutsval: new .....	121
\@glsxtr@dorecord: new .....	10		\glsxtr@texencoding: new .....	121
\@glsxtr@err@undefaction: new .....	6		\glsxtr@writefields: new .....	121
\@glsxtr@record: new .....	7		\GlsXtrLoadResources: new .....	120
\@glsxtr@warn@onexists@do: new ...	6		\glsxtr@resourcefile: changed extension to .glstex .....	120
\@glsxtr@warn@undefaction: new ....	6		\new@ignored@glossary: added starred version .....	33
\@print@unsrt@glossary: new .....	124	1.12 (2017-02-03)		
General: added record package option ...	12		\@glsxtr@recordcounter: new .....	10
\glsadd: added \@glsxtr@record ....	58		\@gls@preglossaryhook: check for definition .....	166
\glsdoifexists: now defines \glslabel .....	36		\@glsxtr@counter@recordhook: new .	123
\glsxtr@do@wrglossary: new .....	25		\@glsxtr@display@loc: new .....	113
\glsxtr@addloc@listfield: new .....	11		\@glsxtr@docounter@record: new ...	123
\glsxtr@indexonly@saveentrycounter: new .....	11		\@glsxtr@longnew@glossaryentry: new .....	32
\glsxtr@record: new .....	123		\@glsxtr@noop@recordcounter: new .	11
\glsxtr@resource: new .....	121		\@glsxtr@op@recordcounter: new ...	11
\glsxtr@saveentrycounter: new ....	25		\@glsxtr@provide@storagekey: new .	26
\glsxtr@setup@record: new .....	11		\@glsxtr@s@longnew@glossaryentry: new .....	32
\glsxtr@assignfieldfont: added check for existence .....	59		\@glsxtrentryfmt: new .....	27
\glsxtr@resourcefile: new .....	120		\@glsxtr@index@aliased: new .....	74
\printunsrt@glossaries: new .....	124		\@glsxtr@set@alias@noindex: new ....	74
\printunsrt@glossary: new .....	124		\@new@glossaryentry@posthook: added check for alias key .....	42
1.09 (2016-12-16)			\@no@glsxtr@index@aliased: new .....	74
\@glsxtr@gettype: new .....	109		\@printunsrt@glossary: new .....	124
\@glsxtr@mixed@assign@sortkey: new .....	109			
\@print@glossary: redefined to save options .....	108			
\glsxtr@make@glossaries: new .....	109			

General: added target key to printgloss family .....	109	\GlsXtrLoadResources: removed restriction on only one per document	120
\apptoglossarypreamble: new .....	31	\glxtrlocrangefmt: new .....	114
\csGlsXtrLetField: new .....	30	\glxtrpostlongdescription: new ..	33
\eGlsXtrSetField: new .....	30	\glxtrprovidestoragekey: new ....	26
\gGlsXtrSetField: new .....	30	\GlsXtrRecordCounter: new .....	123
\glsdohyperlink: added check for alias field .....	77	\glxtrresourcecount: new .....	120
\glsnoidxdisplayloc: added redefinition .....	113	\glxtrresourcefile: added catcode change for @ .....	120
\glissettoctitle: added patch .....	34	\glxtrsetaliasnoindex: new .....	74
\glxtr@counterrecord: new .....	123	\GlsXtrSetField: new .....	30
\glxtr@langtag: new .....	121	\glxtrsetfieldifexists: new ....	29
\glxtr@newabbreviation: new ....	173	\glxtrunsrtdo: new .....	127
\glxtr@org@newignoredglossary: Added check for existence .....	33	\Glsxtrusefield: new .....	29
\glxtr@pluralsuffixes: new .....	121	\glxtrusefield: new .....	29
\glxtr@provideignoredglossary: new .....	34	short-postlong-user: new .....	263
\glxtr@s@newignoredglossary: Added check for existence .....	33	short-postlong-user-desc: new ...	265
\glxtr@s@provideignoredglossary: new .....	35	\longnewglossaryentry: added starred version .....	32
\glxtrabbrvpluralsuffix: new ...	177	long-postshort-user: new .....	261
\glxtralias: new .....	42	long-postshort-user-desc: new ...	262
\glxtrcopytoglossary: new .....	36	postdot: new .....	15
\glxtrdeffield: new .....	29	\pretoglossarypreamble: new .....	31
\glxtrdisplayendloc: new .....	114	\print@noop@unsrtglossaryunit: new .....	127
\glxtrdisplayendlochook: new ...	114	\print@op@unsrtglossaryunit: new	126
\glxtrdisplaysingleloc: new ....	114	\printunsrtglossary: added starred form .....	124
\glxtrdisplaystartloc: new ....	114	\printunsrtglossaryhandler: new .	126
\glxtrdeffield: new .....	29	\printunsrtglossaryunit: new ....	11
\glxtrentryfmt: new .....	27	\printunsrtglossaryunitsetup: new	126
\glxtrfielddolistloop: new .....	28	\provideignoredglossary: new ....	34
\glxtrfieldforlistloop: new ....	28	\s@glxtr@provide@storagekey: new	26
\glxtrfielddifylist: new .....	28	\s@printunsrtglossary: new .....	124
\glxtrfieldlistadd: new .....	28	\xGlsXtrSetField: new .....	30
\glxtrfieldlistead: new .....	28	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new .....	28	\@glsdisp: removed	
\glxtrfieldlistxadd: new .....	28	\@glxtr@org@glsdisp .....	55
\glxtrfieldxifylist: new .....	29	\glxtrsetaliasnoindex: switched to \providecommand .....	74
\glxtrfmt: new .....	27	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new ....	27	\@gls@link: added redefinition .....	56
\GlsXtrFmtField: new .....	27	\@gls@noidx@getgrouptitle: new ..	111
\glxtrifkeydefined: new .....	26	\@gls@removespaces: new .....	114
\glxtrindexaliased: new .....	74	\@glxtr@do@automake@err: new ...	122
\GlsXtrLetField: new .....	30	\@glxtr@org@gloautosee: new ....	23
\GlsXtrLetFieldToField: new .....	30	\@glxtr@record: added third arg ....	7
		\@glxtr@recordsee: new .....	11

General: added \glsadd option	1.16 (2017-06-15)
theHvalue ..... 58	\@glo@autosee: added redefinition .... 24
added \glsadd option thevalue .... 58	\@gls@noidx@getgrouptitle: fixed
\glsdisablehyper: added redefinition . 78	bug ..... 111
\glsenableentrycount: fixed	\@glsxtr@addunusedxrefs: added
assignment of \@cGls@ ..... 90	check for seealso field ..... 43
\glsenableentryunitcount: fixed	\@glsxtr@checkgroup: use \csuse
assignment of \@cGls@ ..... 98	instead of \csname ..... 128
\glsnavigation: new ..... 112	\@glsxtr@dorecordnodefer: new .... 10
\glsxtr@org@getgrouptitle: new .. 112	\@print@unsrt@glossary: corrected
\glsxtr@recordsee: new ..... 7	misspelt command ..... 124
\glsxtr@writefields: added check for	\@printunsrt@glossary@handler:
automake ..... 122	new ..... 126
\glsxtrdisplayendloc: added check	General: added check for
for empty format ..... 114	\@gls@setupsort@none ..... 13
\glsxtrgetgrouptitle: new ..... 112	\@gls@checkseeallowed: added
\glsxtrinitwrgloss: new ..... 55	redefinition ..... 24
\glsxtrlocationhyperlink: new ... 115	\@glsxtr@writefields: added
\glsxtrsetgrouptitle: new ..... 112	\providecommand lines ..... 121
\glsxtrsupphypernumber: new .... 115	\@glsxtrautoindex: new ..... 160
\ifglsxtrwrglossbefore: new ..... 55	\@glsxtrautoindexentry: new ..... 160
1.15 (2017-05-10)	\@glsxtrautoindexsort: new ..... 160
\@glsxtr@dorecord: corrected	\@glsxtrindexseealso: new ..... 39
premature expansion of \@glslocref 10	\@glsxtrseealsolabels: new ..... 42
short-em-long-em: fixed spelling of	\@glsxtrseelist: new ..... 39
\glsabbrvfont ..... 243	\@glsxtruseseealso: new ..... 38
short-long: fixed spelling of	\@glsxtruseseealsoformat: new .... 39
\glsabbrvfont ..... 195	\@seealso: new ..... 39
short-long-user: fixed spelling of	autoseeindex: new ..... 15
\glsabbrvfont ..... 266	1.17 (2017-08-09)
short-postlong-user: fixed spelling of	\@glsxtr@mark@wordseps: new .... 172
\glsabbrvfont ..... 263	\@glsxtr@markwordseps: new ..... 172
short-postlong-user-desc: fixed	\@glsxtr@noidx@displaynumberlist:
spelling of \glsabbrvfont ..... 265	replace hard-coded ?? with
long-em-short-em: fixed spelling of	\glsxtrundeftag ..... 110
\glsabbrvfont ..... 240	\@glsxtr@noidx@entrynumberlist:
long-postshort-user: fixed spelling of	replace hard-coded ?? with
\glsabbrvfont ..... 261	\glsxtrundeftag ..... 111
long-postshort-user-desc: fixed	\@glsxtr@noidx@numberlistloop:
spelling of \glsabbrvfont ..... 262	replace hard-coded ?? with
long-short: fixed spelling of	\glsxtrundeftag ..... 110
\glsabbrvfont ..... 193	\@glsxtrifhyphenstart: new ..... 268
long-short-user: fixed spelling of	General: removed some inconsistencies
\glsabbrvfont ..... 260	in the abbreviation styles ..... 193
footnote: fixed spelling of	\@glsabbrvhyphenfont: new ..... 269
\glsabbrvfont ..... 198	\@glsabbrvonlyfont: new ..... 282
postfootnote: fixed spelling of	\@glsabbrvscfont: new ..... 209
\glsabbrvfont ..... 199	\@glsabbrvsmfont: new ..... 223

\glsabbrvuserfont: initialised to default font .....	259	short-long-user-desc: corrected first forms .....	267
\glsfirstabbrvhyphenfont: new ...	269	short-nolong-desc-noreg: new ...	204
\glsfirstabbrvonlyfont: new .....	282	short-nolong-noreg: new .....	202
\glsfirstabbrvscfont: new .....	209	long-em-noshort-em-desc-noreg: new .....	255
\glsfirstabbrvsmfont: new .....	223	long-em-noshort-em-noreg: new ...	251
\glsfirstlonghyphenfont: new ...	269	long-hyphen-noshort-desc-noreg: new .....	271
\glsfirstlongonlyfont: new .....	282	long-hyphen-postshort-hyphen: new	274
\glslonghyphenfont: new .....	269	long-hyphen-postshort-hyphen-desc: new .....	276
\glslongonlyfont: new .....	282	long-hyphen-short-hyphen: new ...	269
\glslonguserfont: initialised to default font .....	259	long-hyphen-short-hyphen-desc: new .....	270
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong .....	173	long-noshort-desc-noreg: new ...	207
\GlsXtrDefineAcShortcuts: new ....	18	long-noshort-noreg: new .....	208
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside .....	187	long-only-short-only: new .....	282
\glsxtrhyphensuffix: new .....	269	long-only-short-only-desc: new ..	284
\glsxtrifhyphenstart: new .....	268	long-short-user-desc: corrected first forms .....	265
\glsxtrlonghyphen: new .....	273	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	270	stylemods: changed default value to "default" .....	21
\glsxtrlonghyphenshort: new ....	268	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	194	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new .....	284	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref .....	10
\glsxtronlydescsort: new .....	283	\@glsxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new .....	282	\@glsxtr@record@setting: new .....	12
\glsxtrparen: new .....	175	\@glsxtr@record@setting@alsoindex: new .....	12
\glsxtrposthyphenlong: new .....	279	\@glsxtrifhasfield: new .....	29
\glsxtrposthyphenshort: new .....	273	General: added \glslink option theHvalue .....	56
\glsxtrposthyphensubsequent: new	274	added \glslink option thevalue ...	56
\glsxtrshortdescname: new .....	203	\glsxtr@writefields: removed double-quotes around \jobname ..	122
\glsxtrshorthyphen: new .....	278	\glsxtrdoautoindexname: changed format test .....	160
\glsxtrshorthyphenlong: new ....	276	\glsxtrhyperlink: new .....	78
\glsxtrshortlongdescname: new ...	196	\glsxtrifhasfield: new .....	29
\glsxtrshortlongdescsort: new ...	196	\GlsXtrSetDefaultNumberFormat: new .....	7
\Glsxtrsubsequentfmt: new .....	189	\s@glsxtrifhasfield: new .....	29
\glsxtrsubsequentfmt: new .....	189		
\Glsxtrsubsequentplfmt: new ....	189		
\glsxtrsubsequentplfmt: new ....	189		
\glsxtrword: new .....	172		
\glsxtrwordsep: new .....	172		
short-hyphen-long-hyphen: new ...	277		
short-hyphen-long-hyphen-desc: new .....	278		
short-hyphen-postlong-hyphen: new	279		
short-hyphen-postlong-hyphen-desc: new .....	281		



headings .....	335	\glxtrbookindexbetween: new ....	340
redefined		\glxtrbookindexbookmark: new ...	340
mcoltreenonamehypergroup to		\glxtrbookindexcols: new .....	339
discourage breaks after group		\glxtrbookindexcolspread: new ..	340
headings .....	335	\glxtrbookindexfirstmark: new ..	344
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	344
discourage breaks after group		\glxtrbookindexformatheader: new	340
headings .....	336	\glxtrbookindexgroupskip: new ..	340
redefined mcoltreepannav to		\glxtrbookindexlastmark: new ...	344
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	344
headings .....	334	\glxtrbookindexmarkentry: new ..	343
redefined treegroup to discourage		\glxtrbookindexname: new .....	339
breaks after group headings .....	319	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new .....	339
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings .....	320	new .....	339
redefined treenonamegroup to		\glxtrbookindexprelocation: new	339
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings .....	321	new .....	340
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	340
discourage breaks after group		\glxtrbookindexsubname: new ....	339
headings .....	321	\glxtrbookindexsubprelocation:	
debug: new .....	22	new .....	339
\gglsetwidest: new .....	322	\glxtrbookindexsubsubatendgroup:	
\gl:disablehyper: added check for		new .....	340
existence .....	78	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	78	new .....	340
\gl:enablehyper: changed to use \def		\glxtrbookindexthepage: new ....	343
rather than \let .....	78	\glxtrdetoklocation: new .....	132
\Glsfmtname: new .....	299	\glxtrenablerecordcount: new ...	132
\gl:fmtname: new .....	298	\glxtrglossentry: new .....	123
\glshex: new .....	129	\glxtrgroupfield: new .....	127
\gl:listchildpostlocation: new ..	307	\GlsXtrheadname: new .....	290
\gl:listchildprelocation: new ...	307	\glxtrheadname: new .....	290
\gl:listprelocation: new .....	307	\GlsXtrIfFieldEqStr: new .....	30
\gl:navhyperlink: patched .....	76	\glxtriflabelinlist: new .....	126
\gl:seeitemformat: new .....	38	\glxtrifrecordtrigger: new ....	132
\gl:showtarget: new .....	23	\glxtrindexseealso: added check	
\gl:treechildprelocation: new ...	317	that the entry exists .....	39
\gl:treeprelocation: new .....	317	\glxtrtrinityhyperoutside: new ....	56
\gl:triggerrecordformat: new ....	133	\GlsXtrLocationRecordCount: new ..	131
\gl:useabbrvfont: new .....	187	\glxtrnewgls: new .....	129, 130
\gl:uselongfont: new .....	187	\glxtrnewGLSlike: new .....	131
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new .....	130
new .....	8	\glxtrnewrgls: new .....	131
\glxtr@org@@do@wrglossary: new ..	25	\glxtrnewrGLSlike: new .....	131
\glxtr@org@dohyperlink: new ....	76	\glxtrnewrglslike: new .....	131
\glxtr@setbookindexmark: new ...	344	\glxtrprelocation: new ....	306, 339
\glxtrbookindexatendgroup: new ..	340	\GlsXtrRecordCount: new .....	131

\glxtrrecordtriggervalue:new ..	132	nolong-short-sm:new .....	230
\glxtrresourcefile: now disables		nopostdot:new .....	15
record key .....	120	\printunsrtglossaryentryprocesshook:	
\glxtrresourceinit:new .....	129	new .....	125, 126
\GlsXtrSetRecordCountAttribute:		\printunsrtglossarypredoglossary:	
new .....	132	new .....	126
\glxtrtrtitlename:new .....	290	\rGLS:new .....	135
\glxtrtrtitleorpdforheading:new ..	286	\rGls:new .....	134
\GlsXtrTotalRecordCount:new ....	131	\rgls:new .....	133
\glxtrwrglossmark:new .....	22	\rGLSformat:new .....	136
short-em:new .....	245	\rGlsformat:new .....	136
short-sc: corrected first letter		\rglsformat:new .....	136
uppercasing .....	213	\rGLSpl:new .....	135
short-sm: corrected first letter		\rGlspl:new .....	135
uppercasing .....	228	\rglspl:new .....	134
\ifglxtr@hyperoutside:new .....	56	\rGLSplformat:new .....	136
all:new .....	305	\rGlsplformat:new .....	136
nolong-short:new .....	205	\rglsplformat:new .....	136
nolong-short-em:new .....	247	\s@glxtrifhasfield: switched from	
nolong-short-noreg:new .....	205	\ifdef to \ifundef .....	29
nolong-short-sc:new .....	215		



# 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, 168, 317
\@	120
\@cGLS@	90, 99
\@cGLSpl@	90, 99
\@cGls@	90, 98
\@cGlspl@	90, 99
\@cgls@	90, 98
\@cglspl@	90, 98
\@do@wrglossary	8, 9, 105
\@do@wrglossary	11, 13, 25, 58, 74
\@glo@assign@sortkey	109
\@glo@list	6
\@glo@type	124
\@glossarysec	340
\@gls@expand@field	26
\@glslocalreset	88
\@glslocalunset	88
\@glsreset	88
\@glsunset	88
\@glsxtr@autoindex@escspch	162, 163
\@glsxtr@checkspch	160, 161, 163
\@glsxtr@disabledflycommand	48
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 123
\@glsxtrp	83, 84
\@glsxtrpostloctag	51
\@glsxtrpreloctag	51
\@glsxtrwrglossmark	8, 11, 23, 25, 40, 44, 105
\@newglossaryentry@defcounters	89
\@newglossaryentry@defunitcounters	97
\@par	322
\@ACRlong	80
\@ACRlongpl	80
\@ACRshort	79
\@ACRshortpl	80
\@Acrlong	80
\@Acrlongpl	80
\@Acrshort	79
\@Acrshortpl	80
\@Gls@	79, 92, 93, 134
\@Gls@acronymname	101
\@Gls@entry@field	71, 86
\@Gls@entryname	101
\@GlsXtrEnableOnTheFly	45
\@Glspl@	79, 92, 93, 135
\@Glsplural@	80
\@Glsstext@	80
\@Glsxtr	46, 48
\@Glsxtr@full	178
\@Glsxtr@fullpl	179
\@Glsxtr@full	178
\@Glsxtr@fullpl	180
\@Glsxtr@p@acrlong@	80
\@Glsxtr@p@acrlongpl@	80
\@Glsxtr@p@acrshort@	79
\@Glsxtr@p@acrshortpl@	80
\@Glsxtr@p@long@	79
\@Glsxtr@p@longpl@	79
\@Glsxtr@p@plural@	79
\@Glsxtr@p@short@	79
\@Glsxtr@p@shortpl@	79
\@Glsxtr@p@text@	79
\@Glsxtrlong	79, 183
\@Glsxtrlongpl	79, 186
\@Glsxtrp	87
\@Glsxtrshort	79, 181
\@Glsxtrshortpl	79, 184

\@Glsxtr@p@acrlong@	80	\@firstofthree	55,
\@Glsxtr@p@acrlongpl@	80		59, 67–70, 76, 177, 179, 180, 182, 184, 185
\@Glsxtr@p@acrshort@	79	\@firstoftwo	60–64, 68, 70, 73, 76,
\@Glsxtr@p@acrshortpl@	80		103, 169, 170, 178–180, 184–186, 286, 287
\@Glsxtr@p@long@	79	\@for	6, 21,
\@Glsxtr@p@longpl@	79		43, 89, 101, 104, 107, 112, 125, 132, 152, 164
\@Glsxtr@p@plural@	79	\@glo@alias	40, 41
\@Glsxtr@p@short@	79	\@glo@assign@sortkey	107
\@Glsxtr@p@shortpl@	79	\@glo@autosee	23
\@Glsxtr@p@text@	79	\@glo@autoseehook	41
\@Glsxtrlong	79, 182	\@glo@category	95
\@Glsxtrlongpl	79, 186	\@glo@check@sortallowed	107
\@Glsxtrp	86	\@glo@counterprefix	10, 115
\@Glsxtrpl	47, 48	\@glo@countunit	95
\@Glsxtrshort	79, 181	\@glo@default@sorttype	107
\@Glsxtrshortpl	79, 184	\@glo@desc	32
\@acrlong	80	\@glo@descplural	32
\@acrlongpl	80	\@glo@group	12
\@acrshort	79	\@glo@label	
\@acrshortpl	80		11, 12, 26, 37, 38, 40–42, 71, 78, 323–330
\@afterheading		\@glo@location	12
	308, 309, 318–321, 333–336, 343	\@glo@loclist	11
\@alt@glshyp@opt	76	\@glo@name	160
\@auxout	10, 11,	\@glo@no@assign@sortkey	109
	44, 52, 91, 99, 104, 105, 116, 120–123, 343	\@glo@parent	325, 326
\@bibgl@restoreat	120	\@glo@see	37–39, 41–43
\@cGLS	93	\@glo@seealso	40, 41
\@cGLS@	90, 93, 99	\@glo@sort	160
\@cGLSpl	94	\@glo@sorttype	107, 113
\@cGLSpl@	90, 94, 99	\@glo@text	55
\@cGls@	90, 98	\@glo@thislettergrp	128
\@cGlspl@	90, 99	\@glo@thisvalue	259
\@cgl@	90, 98	\@glo@tmp	26, 39, 71
\@cglspl@	90, 98	\@glo@type	42, 77, 101, 104, 107–
\@disable@onlypremakeg	105		109, 112, 113, 115, 116, 119, 120, 124, 125
\@do@auxoutstuff	116	\@glo@types	150, 151, 323–329
\@do@gl@getcounterprefix	10	\@glossary@default@style	48, 49, 108, 338
\@do@gl@see	41, 42	\@glossarystyle	108
\@do@newglossaryentry	101, 102, 174, 175	\@gl@	79, 92, 93, 134
\@do@seeglossary	13, 23, 44, 105	\@gl@@link	55
\@do@wrglossary	57, 58, 133	\@gl@ReturnAfterFi	115
\@empty	59, 67–70, 161, 177–186	\@gl@actualchar	161
\@end@glxtr@addunused	43	\@gl@adjustmode	58
\@end@glxtr@gettype	107, 109	\@gl@alt@hyp@opt	76
\@end@glxtr@usesee	38	\@gl@alt@hyp@opt@char	76
\@end@glxtr@ifhyphenstart	268	\@gl@alt@hyp@opt@keys	76
\@endfortrue	190	\@gl@automake	107
\@firstofone	59, 125, 154, 159, 165	\@gl@between	112
		\@gl@checkedmkidx	160, 161, 163

<code>\@gls@checkmkidxchars</code> .....	40, 160	<code>\@gls@org@glssseeformat</code> .....	110, 111
<code>\@gls@codepage</code> .....	116	<code>\@gls@preglossaryhook</code> .....	108, 165
<code>\@gls@counter</code> .....	9, 10, 56, 58, 74, 133	<code>\@gls@prevlevel</code> .....	331, 332, 336, 337
<code>\@gls@currentlettergroup</code> ...	113, 125, 128	<code>\@gls@quotechar</code> .....	160
<code>\@gls@declareoption</code> .....	5	<code>\@gls@reference</code> .....	44, 104, 105
<code>\@gls@default@longpl</code> .....	173, 174	<code>\@gls@saveentrycounter</code> .	13, 25, 57, 58, 133
<code>\@gls@doautomake</code> .....	107, 122	<code>\@gls@see@noindex</code> .....	24, 120
<code>\@gls@doautomake@err</code> .....	122	<code>\@gls@setdefault@glslink@opts</code> .	9, 57, 75
<code>\@gls@encapchar</code> .....	161	<code>\@gls@setsort</code> .....	57
<code>\@gls@entry@count</code> .....	90, 91	<code>\@gls@setupsort@none</code> .....	13
<code>\@gls@entry@field</code> ..	26, 29, 41, 71, 84–87, 90	<code>\@gls@short</code> .....	173, 174
<code>\@gls@entry@unitcount</code> .....	99	<code>\@gls@shortpl</code> .....	171, 174
<code>\@gls@field@font</code> .....	59–66	<code>\@gls@sort</code> .....	127
<code>\@gls@field@link</code> .....	59–66, 71, 72	<code>\@gls@tmp</code> .....	112
<code>\@gls@getcounterprefix</code> .....	10	<code>\@gls@tmpb</code> .....	163
<code>\@gls@getgrouptitle</code> .....	112, 124, 125	<code>\@gls@type</code> .....	105, 107, 190, 323–330
<code>\@gls@grptitle</code> .....	77, 112	<code>\@gls@write@entrycounts</code> .....	90
<code>\@gls@hyp@opt</code> .....		<code>\@gls@write@entryunitcounts</code> .....	99
. 71, 72, 76, 93, 94, 130, 133–135, 177–186		<code>\@gls@write@entryunitcounts@do</code> ....	100
<code>\@gls@hyp@opt@cs</code> .....	76	<code>\@gls@xref</code> .....	11, 40
<code>\@gls@ifinlist</code> .....	126	<code>\@gls@abbrv@current@abbreviation</code> 173, 187	
<code>\@gls@increment@currcount</code> .....	90	<code>\@gls@acronymlists</code> .....	101
<code>\@gls@increment@currunitcount</code> .....	98	<code>\@gls@doifexistsorwarn</code> .....	14, 155–158
<code>\@gls@keymap</code> .....	11, 12, 26, 38, 40, 71, 121	<code>\@gls@entry</code> .....	91, 99, 100
<code>\@gls@label</code> ...	8–10, 44, 75, 76, 105, 123, 190	<code>\@gls@link</code> .....	57, 77–79
<code>\@gls@levelchar</code> .....	161	<code>\@gls@nextpages</code> .....	108
<code>\@gls@link</code> .....	27, 53, 55, 67–70, 177–186	<code>\@gls@nonextpages</code> .....	108
<code>\@gls@link@checkfirsthyper</code> .....	55, 103	<code>\@gls@numberformat</code> .....	
<code>\@gls@link@label</code> .....	56, 133	. 9, 10, 56, 58, 74, 133, 159, 160	
<code>\@gls@link@nocheckfirsthyper</code> .....		<code>\@gls@order</code> .....	104
. 53, 66–70, 177–186		<code>\@gls@pl@</code> .....	79, 92, 93, 134
<code>\@gls@link@opts</code> .....	56	<code>\@gls@plural@</code> .....	80
<code>\@gls@list</code> .....	112	<code>\@gls@punc@token</code> .....	169, 170
<code>\@gls@local@increment@currcount</code> ....	90	<code>\@gls@recordlocref</code> .....	10
<code>\@gls@local@increment@currunitcount</code> 98		<code>\@gls@showtarget</code> .....	78
<code>\@gls@location</code> .....	128, 129	<code>\@gls@style@altlist</code> .....	307
<code>\@gls@loclist</code> .....	110, 111, 128, 129	<code>\@gls@style@altlistgroup</code> .....	308
<code>\@gls@long</code> .....	173	<code>\@gls@style@altlisthypergroup</code> .....	309
<code>\@gls@longpl</code> .....	171, 173, 174	<code>\@gls@style@alttree</code> .....	321
<code>\@gls@nohyperlist</code> .....	33, 35	<code>\@gls@style@alttreegroup</code> .....	332
<code>\@gls@noidx@do</code> .....	113	<code>\@gls@style@alttreehypergroup</code> .....	332
<code>\@gls@noidx@getgrouptitle</code> .....	124	<code>\@gls@style@index</code> .....	317
<code>\@gls@noidx@nosanitizesort</code> .....	106	<code>\@gls@style@indexgroup</code> .....	318
<code>\@gls@noidx@sanitizesort</code> .....	106	<code>\@gls@style@indexhypergroup</code> .....	318
<code>\@gls@noidx@loclist@finalsep</code> .....	110	<code>\@gls@style@inline</code> .....	317
<code>\@gls@noidx@loclist@prev</code> .....	110	<code>\@gls@style@list</code> .....	307
<code>\@gls@noidx@loclist@sep</code> .....	110	<code>\@gls@style@listdotted</code> .....	306
<code>\@gls@noref@warn</code> .....	105, 113	<code>\@gls@style@listgroup</code> .....	308
<code>\@gls@org@glsnoidxdisplayloc</code> ..	110, 111	<code>\@gls@style@listhypergroup</code> .....	308

<code>\@glsstyle@mcollalttree</code> .....	336	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcollalttreegroup</code> .....	336	.....	341, 343
<code>\@glsstyle@mcollalttreehypergroup</code> ..	337	<code>\@glsxtr@bookindex@subbetween</code> .	341, 342
<code>\@glsstyle@mcollalttreespannav</code> .....	337	<code>\@glsxtr@bookindex@subsep</code> .....	341, 342
<code>\@glsstyle@mcollindexgroup</code> .....	333	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcollindexhypergroup</code> .....	333	.....	341, 343
<code>\@glsstyle@mcollindexspannav</code> .....	333	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcolltreegroup</code> .....	334	.....	341, 342
<code>\@glsstyle@mcolltreehypergroup</code> .....	334	<code>\@glsxtr@bookindexgroupskip</code> ...	341, 343
<code>\@glsstyle@mcolltreenonamegroup</code> .....	335	<code>\@glsxtr@cat</code> .....	89, 101, 132, 164, 165
<code>\@glsstyle@mcolltreenonamehypergroup</code>	335	<code>\@glsxtr@checkgroup</code> .....	125
<code>\@glsstyle@mcolltreenonamespannav</code> ..	336	<code>\@glsxtr@counterrecordhook</code> .....	10, 11
<code>\@glsstyle@mcolltreespannav</code> .....	334	<code>\@glsxtr@csname</code> .....	96, 98
<code>\@glsstyle@mtree</code> .....	319	<code>\@glsxtr@current@style</code> .....	49, 338
<code>\@glsstyle@treegroup</code> .....	319	<code>\@glsxtr@currentunitcount</code> .....	96, 98
<code>\@glsstyle@treehypergroup</code> .....	320	<code>\@glsxtr@currunitcount</code> .....	97, 99
<code>\@glsstyle@treenoname</code> .....	320	<code>\@glsxtr@declareoption</code> .....	5, 15, 17, 20
<code>\@glsstyle@treenonamegroup</code> .....	321	<code>\@glsxtr@defaultnoglossarywarning</code> ..	20
<code>\@glsstyle@treenonamehypergroup</code> ..	321	<code>\@glsxtr@defaultnumberformat</code> .....	
<code>\@glstarget</code> .....	78, 79, 109	.....	7, 9, 56, 58, 74, 159, 160
<code>\@glstext@</code> .....	80	<code>\@glsxtr@defpostpunc</code> .....	15, 16, 23
<code>\@glswidestname</code> .....	323, 330	<code>\@glsxtr@deprecated@abbrstyle</code> .....	
<code>\@glsxtr</code> .....	46, 48	.....	218, 219, 221,
<code>\@glsxtr@do@wrglossary</code> .....	105	223, 232, 234, 235, 237, 250, 253, 257, 259	
<code>\@glsxtr@abbreviationsdef</code> .....	17, 24	<code>\@glsxtr@disabledflycommand</code> .....	48
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@display@loc</code> .....	113
.....	165, 166	<code>\@glsxtr@do@wrrindex</code> .....	75
<code>\@glsxtr@addunitcounter</code> .....	95	<code>\@glsxtr@do@glstdisablehyperinlist</code> ..	73
<code>\@glsxtr@addunused</code> .....	43	<code>\@glsxtr@do@record@wrglossary</code> ....	8, 13
<code>\@glsxtr@addunusedxrefs</code> .....	42, 43	<code>\@glsxtr@do@redef@forglsentries</code> .....	7
<code>\@glsxtr@attrval</code> .....	57, 153–158, 160	<code>\@glsxtr@do@style</code> .....	21, 304
<code>\@glsxtr@autoindex@at</code> .....	160–162	<code>\@glsxtr@do@titlecaps@warn</code> .	154–157, 165
<code>\@glsxtr@autoindex@doextra@esc</code> ....	160	<code>\@glsxtr@doabbreviationsdef</code> .....	17
<code>\@glsxtr@autoindex@encap</code> .....	160–162	<code>\@glsxtr@doaccsupp</code> .....	20, 22
<code>\@glsxtr@autoindex@esc</code> .....	160–163	<code>\@glsxtr@docdefval</code> .....	14, 44
<code>\@glsxtr@autoindex@escat</code> .....	161, 162	<code>\@glsxtr@docounterrecord</code> .....	11
<code>\@glsxtr@autoindex@escencap</code> ...	161, 162	<code>\@glsxtr@doglossary</code> .....	125
<code>\@glsxtr@autoindex@esclevel</code> ...	161, 162	<code>\@glsxtr@doiflabelinlist</code> .....	126
<code>\@glsxtr@autoindex@escquote</code> ...	161, 162	<code>\@glsxtr@doloctag</code> .....	51
<code>\@glsxtr@autoindex@level</code> .....	161, 162	<code>\@glsxtr@dorecord</code> .....	8, 9
<code>\@glsxtr@autoindex@setname</code> .....	160	<code>\@glsxtr@dorecordnodefer</code> .....	8, 9
<code>\@glsxtr@autoindexcrossrefs</code> 13, 14, 38, 41		<code>\@glsxtr@dosee@alsoindex@glossary</code> ..	13
<code>\@glsxtr@autoseeindexfalse</code> .....	13	<code>\@glsxtr@doseeglossary</code> .....	13, 23
<code>\@glsxtr@autoseeindextrue</code> .....	15	<code>\@glsxtr@dostylewarn</code> .....	190
<code>\@glsxtr@bookindex@atendgroup</code> .	341, 343	<code>\@glsxtr@enabletagging</code> .....	164
<code>\@glsxtr@bookindex@atsubendgroup</code> ..	342	<code>\@glsxtr@end@</code> .....	45
<code>\@glsxtr@bookindex@atsubsubendgroup</code>	342	<code>\@glsxtr@endescspch</code> .....	161–163
<code>\@glsxtr@bookindex@between</code> ....	341, 343	<code>\@glsxtr@entrycount@org@localreset</code> .	90
<code>\@glsxtr@bookindex@sep</code> .....	341, 342	<code>\@glsxtr@entrycount@org@localunset</code> .	90

\@glsxtr@entrycount@org@reset .....	90	\@glsxtr@org@Glsxtrtitlefirstplural	
\@glsxtr@entrycount@org@unset .....	90	.....	286, 288
\@glsxtr@entryunitcount@org@localreset		\@glsxtr@org@Glsxtrtitlefull ..	287, 288
.....	98	\@glsxtr@org@Glsxtrtitlefullpl	287, 288
\@glsxtr@entryunitcount@org@localunset		\@glsxtr@org@Glsxtrtitlelong ..	287, 288
.....	98	\@glsxtr@org@Glsxtrtitlelongpl	287, 288
\@glsxtr@entryunitcount@org@reset ..	98	\@glsxtr@org@Glsxtrtitlename ..	286, 287
\@glsxtr@entryunitcount@org@unset ..	98	\@glsxtr@org@Glsxtrtitleplural	286, 288
\@glsxtr@err@undefaction .....	7, 13	\@glsxtr@org@Glsxtrtitleshort ..	286, 287
\@glsxtr@field@linkdefs .....	53	\@glsxtr@org@Glsxtrtitleshortpl	286, 287
\@glsxtr@format@overridefalse .....	159	\@glsxtr@org@Glsxtrtitletext ..	286, 288
\@glsxtr@format@overridetrue .....	159	\@glsxtr@org@MakeUppercase ....	286, 287
\@glsxtr@foundinlist .....	170	\@glsxtr@org@checkfirsthyper ...	72, 103
\@glsxtr@full .....	177	\@glsxtr@org@delimN .....	51, 52
\@glsxtr@fullpl .....	179	\@glsxtr@org@delimR .....	51, 52
\@glsxtr@gettype .....	107	\@glsxtr@org@doseeglossary .....	23, 105
\@glsxtr@glossdescfont .....	153–155	\@glsxtr@org@gloautosee .....	24
\@glsxtr@glossnamefont .....	155–158	\@glsxtr@org@gls@ .....	54
\@glsxtr@gobbleto@endescspch .....	163	\@glsxtr@org@glsdohypertarget .....	109
\@glsxtr@groupheading .....	125, 127, 128	\@glsxtr@org@glsignore .....	51, 52
\@glsxtr@idx@displaynumberlist ....	106	\@glsxtr@org@glspl@ .....	54
\@glsxtr@idx@entrynumberlist .....	106	\@glsxtr@org@glsxtrtitlefirst ..	286, 288
\@glsxtr@ifcsstart .....	45	\@glsxtr@org@glsxtrtitlefirstplural	
\@glsxtr@ifpunctoken .....	170	.....	286, 288
\@glsxtr@ifunitcounter .....	95	\@glsxtr@org@glsxtrtitlefull ..	287, 288
\@glsxtr@insert@dots .....	171	\@glsxtr@org@glsxtrtitlefullpl	287, 288
\@glsxtr@insert@dots@next .....	172	\@glsxtr@org@glsxtrtitlelong ..	286, 288
\@glsxtr@insertdots .....	173	\@glsxtr@org@glsxtrtitlelongpl	286, 288
\@glsxtr@label .....	43, 152, 153	\@glsxtr@org@glsxtrtitlename ..	286, 287
\@glsxtr@loadstyles .....	305, 306	\@glsxtr@org@glsxtrtitleorpdforheading	
\@glsxtr@longnewglossaryentry .....	32	.....	286, 287
\@glsxtr@mark@wordseps .....	172	\@glsxtr@org@glsxtrtitleplural	286, 288
\@glsxtr@mark@wordseps@next .....	172	\@glsxtr@org@glsxtrtitleshort ..	286, 287
\@glsxtr@markwordseps .....	173, 174	\@glsxtr@org@glsxtrtitleshortpl	286, 287
\@glsxtr@mixed@assign@sortkey .....	107	\@glsxtr@org@glsxtrtitletext ..	286, 288
\@glsxtr@noidx@displaynumberlist ..	106	\@glsxtr@org@makeglossaries .....	104
\@glsxtr@noidx@do .....	127	\@glsxtr@org@markboth .....	285, 286
\@glsxtr@noidx@entrynumberlist ....	106	\@glsxtr@org@markright .....	285
\@glsxtr@noidx@numberlistloop .....	106	\@glsxtr@org@newacronymstyle ..	102, 103
\@glsxtr@noop@recordcounter .....	10, 13	\@glsxtr@org@postdescription .....	166
\@glsxtr@notfoundinlist .....	170	\@glsxtr@org@see@noindex .....	120
\@glsxtr@op@recordcounter .....	13	\@glsxtr@org@setacronymstyle ..	102, 103
\@glsxtr@optlist .....	48	\@glsxtr@org@theHvalue .....	8, 9
\@glsxtr@org@@starttoc .....	285, 286	\@glsxtr@org@prefix .....	10
\@glsxtr@org@GLS@ .....	54	\@glsxtr@org@printglossary .....	48, 108
\@glsxtr@org@GLSpl@ .....	54	\@glsxtr@org@warndep .....	171
\@glsxtr@org@Gls@ .....	54	\@glsxtr@p@acrlong@ .....	80
\@glsxtr@org@Glspl@ .....	54	\@glsxtr@p@acrlongpl@ .....	80
\@glsxtr@org@Glsxtrtitlefirst ..	286, 288	\@glsxtr@p@acrshort@ .....	79

\@glxstr@p@acrshortpl@	80	\@glxtrentryfmt	27
\@glxstr@p@long@	79	\@glxstrglossentry	123
\@glxstr@p@longpl@	79	\@glxstrhypernameprefix	109, 126, 127
\@glxstr@p@plural@	79	\@glxstrifhasfield	29
\@glxstr@p@short@	79	\@glxstrifhyphenstart	268
\@glxstr@p@shortpl@	79	\@glxstrindexaliased	74
\@glxstr@p@text@	79	\@glxstrindexcrossrefsfalse	14
\@glxstr@p@pagetag	51	\@glxstrindexcrossrefstrue	15
\@glxstr@p@pagetag	51	\@glxstrinmark	285
\@glxstr@p@prevunitcount	97	\@glxstrlong	79, 182
\@glxstr@p@printglossopts	48, 107, 108	\@glxstrlongpl	79, 185
\@glxstr@p@printunsrtglossaryskipentry	125, 126	\@glxstrnewgls	130, 131
\@glxstr@p@provide@addstoragekey	27	\@glxstrnewgls@inner	129, 130
\@glxstr@p@provide@storagekey	26	\@glxstrnewgls@innercsname	130
\@glxstr@p@record	13, 53–55, 58	\@glxstrnotinmark	285
\@glxstr@p@record@setting	8, 9, 12, 39, 43, 44, 104	\@glxstrp	85
\@glxstr@p@record@setting@alsoindex	8, 9, 39, 104	\@glxstrp@opt	83
\@glxstr@p@record@setting@off	43	\@glxstrpl	47, 48
\@glxstr@p@record@setting@only	104	\@glxstrpostloctag	50–52
\@glxstr@p@recordsee	13, 23, 39	\@glxstrpreloctag	50–52
\@glxstr@p@redef@forglsentries	7, 24	\@glxstrsetaliasnoindex	73–75
\@glxstr@p@redef@styles	21, 304	\@glxstrshort	79, 180
\@glxstr@p@reg@glosslist	104–107, 109	\@glxstrshortpl	79, 183
\@glxstr@p@rglstrigger@record	134–136	\@glxstrundeftag	6, 25
\@glxstr@p@s@longnewglossaryentry	32	\@glxstrwrglossmark	22
\@glxstr@p@savepreloctag	51, 52	\@gobble	7, 13, 15, 59, 126, 127, 172, 341–343
\@glxstr@p@setentrycountunsetattr	89	\@gobbletwo	171
\@glxstr@p@setentryunitcountunsetattr	100	\@ifnextchar	76
\@glxstr@p@setupshortcuts	19, 20, 24	\@ifpackageloaded	5, 16, 122, 137, 153, 155, 157, 159, 304
\@glxstr@p@shortcutsval	19, 122	\@ifstar	26, 29, 32–34, 45, 76, 124, 164
\@glxstr@p@swaptwo	170	\@ifundefined	304
\@glxstr@p@tag	165	\@ignored@glossaries	33–35
\@glxstr@p@taggingcs	165	\@input	120
\@glxstr@p@textformat	57, 58	\@input@	115
\@glxstr@p@theHvalue	8, 9, 56–58, 133	\@istfilename	104
\@glxstr@p@thevalue	8, 9, 56–58, 133	\@makeglossary	104
\@glxstr@p@thisloctag	51, 52	\@mfu@domakefirstuc	165
\@glxstr@p@titlelabel	111, 112, 127	\@mfu@nocaplist	165
\@glxstr@p@tmp	21, 114	\@one	91, 100, 130
\@glxstr@p@type	153	\@newglossaryentry@defcounters	89, 97
\@glxstr@p@unitcountlist	95	\@newglossaryentryposthook	11, 12, 26, 40, 71
\@glxstr@p@unsrt@getgrouptitle	125	\@newglossaryentryprehook	11, 12, 26, 32, 40, 71
\@glxstr@p@usesee	38	\@nil	114, 115, 127
\@glxstr@p@warn@onexistsordo	7, 13	\@nnil	161, 163, 170–172
\@glxstr@p@warn@undefaction	7, 13	\@no@glxstrindexaliased	74
\@glxstr@p@docdeffalse	44	\@no@makeglossaries	119





<code>\acronymsort</code> .....	101	<code>entrycount</code> .....	88, 89, 91, 100
<code>\acronymstype</code> .....	17, 101, 102	<code>firstuc</code> .....	157
<code>\acrpluralsuffix</code> .....	101, 122	<code>glossdesc</code> .....	153
<code>\ACS</code> .....	18	<code>glossdescfont</code> .....	153
<code>\Acs</code> .....	18	<code>glossname</code> .....	155
<code>\acs</code> .....	18	<code>glossnamefont</code> .....	155, 157
<code>\ACSP</code> .....	18	<code>headuc</code> .....	288
<code>\Acsp</code> .....	18	<code>indexname</code> .....	160
<code>\acsp</code> .....	18	<code>indexonlyfirst</code> .....	75
<code>\actualchar</code> .....	163	<code>insertdots</code> .....	173
<code>\addtolength</code> .....	331	<code>markshortwords</code> .....	173
<code>\advance</code> .....	91, 100, 121, 130	<code>markwords</code> .....	173, 174, 268, 269, 277
<code>\AF</code> .....	17	<code>nohyper</code> .....	73
<code>\Af</code> .....	17	<code>nohyperfirst</code> .....	60–62
<code>\af</code> .....	17	<code>noshortplural</code> .....	174
<code>\AFP</code> .....	17	<code>regular</code> 53, 94, 193, 195, 197–199, 202–205,	
<code>\Afp</code> .....	17	207, 208, 211, 212, 214, 215, 217, 219–	
<code>\afp</code> .....	17	221, 225, 227–229, 232–234, 236, 239–	
<code>\AL</code> .....	17	245, 247, 249, 251, 253–255, 257, 260,	
<code>\Al</code> .....	17	266, 267, 269–271, 273, 277, 278, 282, 284	
<code>\al</code> .....	17	<code>textformat</code> .....	57
<code>\ALP</code> .....	17	<code>\cdot</code> .....	22
<code>\Alp</code> .....	17	<code>\centering</code> .....	340
<code>\alp</code> .....	17	<code>\cGLS</code> .....	17, 18, 89, 100
<code>\AnyTrackedLanguages</code> .....	304	<code>\cGLs</code> .....	17, 18, 89, 100
<code>\appto</code> .....	11, 12, 21, 26, 37–42,	<code>\cglS</code> .....	17, 18, 89, 100
	71, 75, 89, 97, 125, 127, 159, 169, 172, 305	<code>\cGLSformat</code> .....	93
<code>\arabic</code> .....	343	<code>\cGLsformat</code> .....	92
<code>\AS</code> .....	17	<code>\cglSformat</code> .....	92, 94
<code>\As</code> .....	17	<code>\cGLSpl</code> .....	17, 18, 89, 100
<code>\as</code> .....	17	<code>\cGLspl</code> .....	17, 18, 89, 100
<code>\ASP</code> .....	17	<code>\cglSpl</code> .....	17, 18, 89, 100
<code>\Asp</code> .....	17	<code>\cGLSplformat</code> .....	93
<code>\asp</code> .....	17	<code>\cGLsplformat</code> .....	92
<code>\AtBeginDocument</code> .....	22, 25, 49, 50, 122	<code>\cglSplformat</code> .....	92, 94
<code>\AtEndDocument</code> .....	42, 90, 99, 116	<code>\changes</code> .....	16, 290
<b>B</b>			
<code>babel package</code> .....	160, 161, 169	<code>\char</code> .....	111
<code>\begin</code> .....	15, 16, 113,	<code>\columnwidth</code> .....	49, 50
	117, 118, 125, 307, 309–316, 334–337, 341	<code>\count@</code> .....	91, 99, 100
<code>\begingroup</code> .....	8, 9, 74, 123, 124	<code>\cs</code> .....	16
<code>\bgroup</code> .....	32, 108	<code>\csappto</code> .....	31
<code>bib2gls</code> .....	3, 127, 129, 133	<code>\csdef</code> .....	26, 29–31, 71, 72, 90,
<b>C</b>			
<code>\catcode</code> .....	120		95, 96, 98, 148, 190–192, 199, 221, 236,
<code>category attributes:</code>			257, 261, 263, 265, 274, 276, 279, 281, 306
<code>aposplural</code> .....	174	<code>\cseappto</code> .....	36
<code>discardperiod</code> .....	168	<code>\csedef</code> .....	29, 96
		<code>\csgdef</code> 30, 33–35, 44, 51, 90, 96, 98, 99, 322, 344	
		<code>\cslet</code> .....	30, 32, 108
		<code>\csletcs</code> .....	30, 192



<code>\csname</code> .....	6, 26, 34, 40, 44, 49, 52, 55, 56, 58, 67–72, 74, 83, 96, 105, 112, 115, 116, 119, 120, 125, 130–133, 153, 171, 177–187, 192, 330
<code>\cspreto</code> .....	31
<code>\csuse</code> .....	27, 28, 34, 42, 51, 71, 72, 84– 86, 95–99, 107, 111, 113, 114, 123, 126– 128, 148, 159, 167, 190, 192, 323, 325, 326
<code>\csxdef</code> .....	37, 38, 40, 41, 96, 99, 112
<code>\currentglossary</code> .....	108, 123, 343
<code>\CurrentOption</code> .....	22, 305, 306
<code>\CurrentTrackedLanguageTag</code> .....	121
<code>\CurrentTrackedTag</code> .....	304
<code>\CustomAbbreviationFields</code> .....	175, 193–197, 199, 201, 203, 206, 208–214, 216, 219, 221, 223, 225– 228, 230, 234, 235, 238–244, 246, 248, 250, 253, 255, 257, 260–263, 265–267, 269–271, 273, 274, 276–279, 281, 282, 284
<b>D</b>	
<code>\DeclareAcronymList</code> .....	101
<code>\DeclareOption</code> .....	5, 305
<code>\DeclareOptionX</code> .....	5, 22
<code>\def</code> .....	9–13, 23, 25, 32, 34, 38, 40, 43, 45–48, 50, 52–70, 76, 78–82, 92–94, 101, 105, 107–110, 112– 115, 125, 127, 128, 130, 133–136, 160– 163, 165, 169–174, 177–186, 190, 268, 271, 273, 274, 277, 279, 331, 332, 336, 337
<code>\defglstryfmt</code> .....	33–35
<code>\define@boolkey</code> .....	14, 15, 56, 73
<code>\define@choicekey</code> .....	7, 12, 14, 15, 19, 20, 22, 56, 109
<code>\define@key</code> .....	11, 12, 16, 21, 26, 40, 56, 58, 71, 109, 171
<code>\DefineAcronymSynonyms</code> .....	19
<code>\delimN</code> .....	51, 52
<code>\delimR</code> .....	51, 52
<code>\detokenize</code> .....	45
<code>\dimen@</code> .....	103, 323–330
<code>\dimen@i</code> .....	324–326
<code>\dimen@ii</code> .....	324–326
<code>\dimexpr</code> .....	49, 50, 322
<code>\disable@keys</code> .....	17, 25, 44, 120
<code>\do</code> .....	6, 21, 43, 89, 101, 104, 107, 112, 125, 132, 152, 164
<code>\do@glsl@link@checkfirsthyper</code> .....	27, 53, 55, 57, 66–70, 177–186
<code>\do@glsl@disablehyperinlist</code> .....	57, 74
<code>doc package</code> .....	163
<code>\dolistcsloop</code> .....	28
<code>\DTLifinlist</code> .....	105, 106, 109
<code>\DTLifint</code> .....	111
<b>E</b>	
<code>\eappto</code> .....	11, 21, 33–35, 125, 128, 160, 305
<code>\edef</code> .....	6, 8–10, 33–36, 39, 41, 42, 44, 56–58, 73, 74, 77, 78, 95, 96, 98, 104, 105, 109, 111, 114–116, 120, 123, 133, 153–156, 158, 160, 161, 163, 171, 325, 326, 341, 342
<code>\eglssetwidest</code> .....	323–330
<code>\egroup</code> .....	32, 33, 108
<code>\else</code> .....	8– 11, 14, 15, 17, 19, 20, 23, 31, 44, 45, 50, 52, 55, 57, 58, 74, 75, 91, 103, 104, 106, 108, 109, 111, 113–115, 117–120, 132, 133, 159–161, 163, 170, 172, 174, 180– 186, 189, 194, 196, 198–207, 210, 212– 224, 226–240, 242–258, 260–262, 264, 267, 268, 271, 274–277, 279, 281, 283, 307, 309–319, 321, 331, 332, 338, 340, 342
<code>\emph</code> .....	237
<code>\empty</code> .....	113–115
<code>\encapchar</code> .....	163
<code>\end</code> .....	113, 117, 118, 125, 307, 309–316, 334–337, 341
<code>\end@glxtr@display@loc</code> .....	113
<code>\endcsname</code> .....	6, 26, 34, 40, 44, 49, 52, 55, 56, 58, 67–72, 74, 83, 96, 105, 112, 115, 116, 119, 120, 125, 130–133, 153, 171, 177–187, 192, 330
<code>\endgroup</code> .....	8, 10, 74, 124
<code>\ensuremath</code> .....	22
entry categories:	
abbreviation .....	187
general .....	147, 149
index .....	151
<code>\epreto</code> .....	160
<code>\equal</code> .....	119
<code>etoolbox package</code> .....	5
<code>\expandafter</code> .....	22, 26, 27, 38, 39, 42, 43, 45–47, 71, 72, 76, 83, 94, 95, 105– 107, 109, 112, 114, 125, 127, 130, 136, 153, 156, 157, 160, 163, 170, 173, 174, 268
<code>\expandonce</code> .....	101, 128, 161, 194, 271
<b>F</b>	
<code>\fi</code> ..	7–11, 13–15, 17, 19, 20, 22–24, 31, 38,



<code>\glossentryname</code> .....	<code>\GLSaccessfirstplural</code> .....	62
.... 124, 306–316, 318–320, 331, 332, 339	<code>\Glsaccessfirstplural</code> .....	62
<code>\glossentrysymbol</code> 310–314, 316, 318–320, 322	<code>\glsaccessfirstplural</code> .....	61
<code>\glossxtrsetpopts</code> .....	<code>\Glsaccesslong</code> .....	69,
166	175, 183, 194, 202, 206, 207, 210, 216–	
<code>\GLS</code> .....	219, 224, 230, 231, 233, 238, 240, 248–	
89, 100, 132	252, 254, 260, 262, 270, 272, 275, 278, 283	
<code>\Gls</code> .....	<code>\glsaccesslong</code> .....	
47, 89, 100, 132	.. 69, 175, 182, 183, 194, 196, 198–201,	
<code>\gls</code> .....	203–207, 210, 212, 213, 215–221, 223,	
31, 46, 48, 89, 100, 106, 117, 132	224, 226–235, 237, 238, 240, 242–258,	
<code>\gls@assign@desc</code> .....	260–262, 264, 267, 270–272, 275, 278, 283	
32	<code>\Glsaccesslongpl</code> .....	
<code>\gls@assign@field</code> .....	.. 70, 176, 186, 194, 202, 206, 207, 210,	
11, 12, 26, 71	216–219, 224, 230–233, 238, 240, 248–	
<code>\gls@checkseeallowed</code> .....	254, 261, 262, 270, 272, 275, 276, 278, 283	
45, 105	<code>\glsaccesslongpl</code> .....	70, 175,
<code>\gls@codepage</code> .....	185, 186, 194, 196, 198, 199, 201, 202,	
116	204–207, 210, 212–221, 223, 224, 226,	
<code>\gls@defdocnewglossaryentry</code> .....	228–235, 237, 238, 240, 242, 244–258,	
89, 97	260, 262, 264, 267, 270, 272, 275, 278, 283	
<code>\gls@defglossaryentry</code> .....	<code>\GLSaccessname</code> .....	62
32, 46, 47	<code>\Glsaccessname</code> .....	62
<code>\gls@dotocitle</code> .....	<code>\glsaccessname</code> .....	38, 62
108	<code>\GLSaccessplural</code> .....	61
<code>\gls@glossary</code> .....	<code>\Glsaccessplural</code> .....	61
40	<code>\glsaccessplural</code> .....	61
<code>\gls@grplabel</code> .....	<code>\Glsaccessshort</code> .....	67, 181, 189, 196,
77	198–201, 204, 205, 212, 213, 215, 220–	
<code>\gls@level</code> .....	223, 226, 228, 229, 235–237, 242, 244,	
128	245, 247, 256, 258, 264, 267, 275, 280, 281	
<code>\gls@noidxglossary</code> .....	<code>\glsaccessshort</code> .....	
105	..... 67, 175, 180, 181, 189, 194, 196,	
<code>\gls@org@glossaryentryfield</code> .....	198, 200–205, 207, 210, 212–217, 219–	
108	222, 224, 226–231, 233–238, 240, 242,	
<code>\gls@org@glossarysubentryfield</code> ....	243, 245–252, 254, 256, 258, 260–262,	
108	264, 267, 270, 272, 275, 278, 280, 281, 283	
<code>\gls@save@numberlist</code> .....	<code>\Glsaccessshorttpl</code> 68, 184, 189, 196, 198–	
50, 52	201, 204, 205, 212, 213, 215, 221–223,	
<code>\gls@set@xr@key</code> .....	226, 228, 229, 235–237, 242, 244, 245,	
40	247, 256–258, 264, 267, 275, 280, 281, 283	
<code>\gls@tmplen</code> .....	<code>\glsaccessshorttpl</code> 68, 175, 176, 184, 185,	
323–332	189, 194, 196, 198–200, 202–205, 207,	
<code>\gls@type</code> .....	210, 212–217, 219–224, 226–240, 242,	
105	244–249, 251–254, 256, 258, 260–262,	
<code>\glsabbrvdefaultfont</code> ... 176, 193, 195,	264, 267, 270, 272, 275, 278, 280, 281, 283	
198, 200, 201, 203, 206, 259, 269, 271, 282	<code>\GLSaccesssymbol</code> .....	64
<code>\glsabbrvmfont</code> .....	<code>\Glsaccesssymbol</code> .....	64, 164
.... 237–246, 248, 250, 252, 253, 255–258	<code>\glsaccesssymbol</code> .....	63, 164, 168
<code>\glsabbrvfont</code> .....		
80, 81, 102, 176, 180,		
181, 184, 185, 187, 189, 193–201, 203,		
206, 208, 210, 211, 213, 214, 217, 218,		
220, 222, 224, 226, 227, 229, 231, 232,		
234, 236, 238, 240, 242, 243, 245, 246,		
248, 250, 252, 253, 256, 258, 260, 261,		
264–267, 269, 271, 273–275, 277, 280, 282		
<code>\glsabbrvhyphenfont</code> .... 269, 270, 274–281		
<code>\glsabbrvonlyfont</code> .....		
282, 284		
<code>\glsabbrvscfont</code> . 209–214, 216–218, 220–222		
<code>\glsabbrvsmfont</code> .....		
223–232, 234–236		
<code>\glsabbrvuserfont</code> .....		
259–266		
<code>\GLSaccessdesc</code> .....		
63		
<code>\Glsaccessdesc</code> .....		
63, 154, 164		
<code>\glsaccessdesc</code> .....		
62, 154, 168		
<code>\GLSaccessdescplural</code> .....		
63		
<code>\Glsaccessdescplural</code> .....		
63		
<code>\glsaccessdescplural</code> .....		
63		
<code>\GLSaccessfirst</code> .....		
60		
<code>\Glsaccessfirst</code> .....		
60		
<code>\glsaccessfirst</code> .....		
60		

<code>\GLSaccesssymbolplural</code> .....	64	<code>\glstentrydesc</code> .....	141, 146, 155
<code>\Glsaccesssymbolplural</code> .....	64	<code>\Glstentrydescplural</code> .....	142, 146
<code>\glsaccesssymbolplural</code> .....	64	<code>\glstentrydescplural</code> .....	141, 142, 146
<code>\GLSaccesstext</code> .....	59	<code>\Glstentryfirst</code> .....	94, 136, 139, 145
<code>\Glsaccesstext</code> .....	60	<code>\glstentryfirst</code> .....	94, 136, 139, 145, 300
<code>\glsaccesstext</code> .....	38, 59	<code>\Glstentryfirstplural</code> ....	95, 136, 139, 145
<code>\glsacrshortcutstrue</code> .....	19, 20	<code>\glstentryfirstplural</code> .....	
<code>\glsacspacemax</code> .....	103	.....	94, 136, 139, 140, 145, 301
<code>\glsadd</code> .....	43, 117	<code>\glstentryfmt</code> .....	33–35
<code>\glsadd options</code>		<code>\Glstentryfull</code> .....	102
<code>theHvalue</code> .....	9	<code>\glstentryfull</code> .....	102
<code>thevalue</code> .....	9	<code>\Glstentryfullpl</code> .....	102
<code>\glsaddstoragekey</code> .....	42, 147	<code>\glstentryfullpl</code> .....	102
<code>\glsbackslash</code> .....	45	<code>\glstentryitem</code> .....	
<code>\glscapscase</code> .....	55, 59–70, 72, 177–188	.....	124, 306, 307, 309–316, 318–320, 331, 342
<code>\glscategory</code> .....	53, 59, 73, 80, 81, 148–	<code>\Glstentrylong</code> .....	81, 82, 94, 136, 143, 147
150, 153–159, 164, 167, 177–181, 183–185		<code>\glstentrylong</code> ..	81, 82, 94, 136, 143, 147,
<code>\glscategorylabel</code> .....		.....	199, 222, 236, 257, 263, 265, 279, 301, 302
.....	73, 171, 173, 174, 199, 221,	<code>\Glstentrylongpl</code> .....	81, 82, 95, 144, 147
236, 257, 261, 263, 265, 274, 276, 279, 281		<code>\glstentrylongpl</code> ....	81, 82, 94, 144, 147, 302
<code>\glsclosebrace</code> .....	39, 118, 119	<code>\Glstentrylongplural</code> .....	136
<code>\glscurrententrylabel</code> .....		<code>\glstentrylongplural</code> .....	136
..	50–52, 108, 115, 123, 125, 126, 166, 167	<code>\Glstentryname</code> .....	137, 144, 155, 157, 158
<code>\glscurrentfieldvalue</code> .....	27–29, 31, 259	<code>\glstentryname</code> .....	123,
<code>\glscustomtext</code> ..	53, 55, 67–70, 177–187, 189	.....	124, 137, 144, 160, 298, 299, 323–330, 344
<code>\glsdefaulttype</code> ..	6, 16, 31, 107, 117, 124, 126	<code>\glstentrynumberlist</code> ....	106, 111, 328–330
<code>\glsdescriptionaccessdisplay</code> ..	141, 154	<code>\Glstentryplural</code> .....	138, 145
<code>\glsdescriptionpluralaccessdisplay</code>		<code>\glstentryplural</code> .....	138, 145, 300
.....	141, 142	<code>\glstentryprevcount</code> .....	90, 91, 97
<code>\glsdescwidth</code> .....	309–316	<code>\glstentryprevmaxcount</code> .....	98
<code>\glsdetoklabel</code> .....		<code>\glstentryprevtotalcount</code> .....	97
..	8, 9, 28–32, 36–39, 43–45, 56, 58, 74,	<code>\Glstentryshort</code> .....	80, 82, 142, 146
78, 90, 95–100, 105, 108, 110, 111, 123,		<code>\glstentryshort</code> .....	80–
127, 128, 131–133, 153, 156, 157, 325, 326		.....	82, 103, 142, 146, 261, 263, 274, 297, 298
<code>\glsdisplaynumberlist</code> .....	106, 109	<code>\Glstentryshortpl</code> .....	81, 82, 143, 147
<code>\glsdohyperlink</code> .....	76, 77, 79	<code>\glstentryshortpl</code> ..	81, 82, 143, 146, 147, 298
<code>\glsdohypertarget</code> .....	79, 109	<code>\Glstentrysymbol</code> .....	140, 145
<code>\glsdoifexists</code> ..	14, 23, 29, 36, 38–40, 53, 55,	<code>\glstentrysymbol</code> .....	140, 145, 327–329
58, 66–70, 78, 105, 110, 111, 123, 177–186		<code>\Glstentrysymbolplural</code> .....	141, 146
<code>\glsdoifexistsordo</code> .....	27, 55	<code>\glstentrysymbolplural</code> .....	140, 141, 146
<code>\glsdoifexistsorwarn</code> ....	14, 153, 154, 164	<code>\Glstentrytext</code> .....	138, 144
<code>\glsdoifnoexists</code> .....	32	<code>\glstentrytext</code> ....	78, 137, 138, 144, 145, 299
<code>\glsdonohyperlink</code> .....	58, 78, 79	<code>\glstentrytype</code> .....	123
<code>\glsdosanitizesort</code> .....	106	<code>\Glstentryuseri</code> .....	65
<code>\glsenableentrycount</code> .....	89, 91, 99	<code>\glstentryuseri</code> .....	65
<code>\glsenableentryunitcount</code> .....	90, 100	<code>\Glstentryuserii</code> .....	65
<code>\glstentrycounter</code> .....	115	<code>\glstentryuserii</code> .....	65
<code>\glstentrycurrcount</code> .....	90, 91, 97	<code>\Glstentryuseriii</code> .....	65
<code>\Glstentrydesc</code> .....	141, 146, 154	<code>\glstentryuseriii</code> .....	65

<code>\Glsentryuseriv</code> .....	65	<code>\glsgenentryfmt</code> .....	53
<code>\glsentryuseriv</code> .....	66	<code>\glsgetattribute</code> .....	57, 77, 91, 95–97, 115, 132, 153–156, 158, 160
<code>\Glsentryuserv</code> .....	66	<code>\glsgetcategoryattribute</code> .....	148
<code>\glsentryuserv</code> .....	66	<code>\glsgetgrouptitle</code> .....	308, 309, 318–321, 332–338
<code>\Glsentryuservi</code> .....	66	<code>\glsgetwidestname</code> .....	322
<code>\glsentryuservi</code> .....	66	<code>\glsgroupheading</code> .....	.... 128, 307–316, 318–321, 331–337, 343
<code>\glsfieldfetch</code> .....	78	<code>\glsgroupskip</code> .....	128, 307, 309–319, 321, 332, 343
<code>\glsfieldxdef</code> .....	152, 153	<code>\glshasattribute</code> .....	57, 77, 91, 96, 98, 100, 115, 132, 153–159, 193, 195, 197, 198, 200, 203–205, 208, 209, 211, 212, 220, 222, 224, 225, 227, 234, 236, 238–244, 251, 255, 257, 260, 261, 263, 265–267, 269–271, 273, 274, 276–278, 280–282, 284
<code>\glsfindwidesttoplevelname</code> .....	323	<code>\glshascategoryattribute</code> .....	149
<code>\GLSfirst</code> .....	292, 293	<code>\glshyperlink</code> .....	78
<code>\Glsfirst</code> .....	293	<code>\glshypernavsep</code> .....	112
<code>\glsfirst</code> .....	292	<code>\glshypernumber</code> .....	115, 159
<code>\glsfirstabbrvdefaultfont</code> .....	176, 193, 196, 198, 200, 201, 203, 206, 271	<code>\glsifattribute</code> .....	55, 56, 60, 73, 75, 84, 151, 154–157, 166, 168, 169, 288–297
<code>\glsfirstabbrvmfont</code> .....	238–258	<code>\glsifcategory</code> .....	151
<code>\glsfirstabbrvfont</code> .....	102, 175, 176, 193–207, 210, 211, 213, 214, 217, 218, 220, 222, 224, 226, 227, 229, 231, 232, 234, 236, 238, 240, 242, 243, 245, 246, 248, 250, 252, 253, 256, 258, 260, 261, 264, 266, 269, 271, 272, 275, 277, 280, 283	<code>\glsifcategoryattribute</code> .....	73, 149, 150, 173, 174
<code>\glsfirstabbrvhyphenfont</code> .....	268–270, 274, 275, 277–281	<code>\glsifnotregular</code> .....	59
<code>\glsfirstabbrvonlyfont</code> .....	283	<code>\glsifnotregularcategory</code> .....	150
<code>\glsfirstabbrvscfont</code> .....	209–223	<code>\glsifplural</code> .....	.. 55, 59, 61–64, 66–70, 168, 169, 177–188
<code>\glsfirstabbrvsmfont</code> .....	224–237	<code>\glsifregular</code> .....	38, 53, 59, 94, 95, 136
<code>\glsfirstabbrvuserfont</code> .....	260–267	<code>\glsifregularcategory</code> .....	150
<code>\glsfirstaccessdisplay</code> .....	139	<code>\glsifusetranslator</code> .....	34
<code>\glsfirstlongdefaultfont</code> .....	193, 196, 201, 203, 206, 209–219, 224– 233, 238, 239, 241, 242, 245–249, 252, 253	<code>\glsignore</code> .....	51, 52
<code>\glsfirstlongemfont</code> .....	239–241, 243, 244, 250, 251, 253–255	<code>\glsinlinedescformat</code> .....	317
<code>\glsfirstlongfont</code> .....	175, 176, 193–196, 198, 200–208, 210, 211, 213, 214, 217, 218, 220, 222, 224, 226, 227, 229, 231, 232, 234, 236, 238, 240, 242, 243, 245, 246, 248, 250, 252, 253, 256, 258, 260, 261, 264, 266, 269, 271, 275, 277, 280, 283	<code>\glsinlinesubdescformat</code> .....	317
<code>\glsfirstlongfootnotefont</code> .....	197–201, 220–223, 234–237, 255–258	<code>\glsinsert</code> .....	55, 59, 67–70, 177–189, 268, 274, 276, 279–281
<code>\glsfirstlonghyphenfont</code> .....	268–271, 273–280	<code>\glskeylisttok</code> .....	101, 102, 173, 175
<code>\glsfirstlongonlyfont</code> .....	282–284	<code>\glslabel</code> .....	8, 9, 36, 53, 55–58, 73, 74, 77, 78, 103, 133, 167, 168, 187–189, 199, 222, 236, 257, 261, 263, 265, 274, 276, 279–281
<code>\glsfirstlonguserfont</code> .....	260–267	<code>\glslabeltok</code> .....	101, 173, 174, 193–198, 200, 201, 203– 206, 208–214, 216, 220, 222, 224–227, 229, 231, 234, 236, 238–246, 248, 250, 251, 253, 255, 257, 260–263, 265–267, 269–271, 273, 274, 276–278, 280–282, 284
<code>\GLSfirstplural</code> .....	293	<code>\glsletentryfield</code> .....	160
<code>\Glsfirstplural</code> .....	293, 294	<code>\glslink</code> .....	102
<code>\glsfirstplural</code> .....	293		
<code>\glsfirstpluralaccessdisplay</code> ..	139, 140		
<code>\glsforeachincategory</code> .....	190		

<code>\glslink options</code>	<code>\glsmcols</code> .....	334–337
counter .....	<code>\GLSname</code> .....	290
format .....	<code>\Glsname</code> .....	290
hyper .....	<code>\glsname</code> .....	290
hyperoutside .....	<code>\glsnameaccessdisplay</code> ..	137, 155, 156, 158
noindex .....	<code>\glsnamefont</code> .....	155, 157, 158
theHvalue .....	<code>\glsnavhyperlink</code> .....	112
thevalue .....	<code>\glsnavhyperlinkname</code> .....	77
wrgloss .....	<code>\glsnavhypertarget</code> .....	
<code>\glslinkcheckfirsthyperhook</code> .....	.....	308, 309, 319–321, 333–338
<code>\glslinkpostsetkeys</code> .....	<code>\glsnavigation</code> ..	308, 309, 319–321, 332–337
<code>\glslinkvar</code> .....	<code>\glsnextpages</code> .....	108
<code>\glslistchildpostlocation</code> .....	<code>\glsnoidxdisplayloc</code> .....	110, 111
<code>\glslistchildprelocation</code> .....	<code>\glsnoidxdisplaylocclsthandler</code> ....	110
<code>\glslistdottedwidth</code> .....	<code>\glsnoidxloclist</code> .....	111, 128, 129
<code>\glslistgroupheaderfmt</code> .....	<code>\glsnoidxnumberlistloophandler</code> ....	110
<code>\glslistnavigationitem</code> .....	<code>\glsnonextpages</code> .....	108
<code>\glslistprelocation</code> .....	<code>\glsnonumberlistfalse</code> .....	50
<code>\glslocalunset</code> .....	<code>\glsnonumberlisttrue</code> .....	50
<code>\glslongaccessdisplay</code> .....	<code>\glsnumberlistloop</code> .....	106
<code>\glslongdefaultfont</code> ..	<code>\glsnumlistlastsep</code> .....	110
177, 193, 196, 197,	<code>\glsnumlistsep</code> .....	110
201, 203, 206, 210, 211, 213, 214, 216–	<code>\glsopenbrace</code> .....	39, 118, 119
218, 224, 226, 227, 229, 231–233, 238,	<code>\glsorder</code> .....	104
242, 245, 246, 248, 249, 252, 259, 269, 282	<code>\glspagelistwidth</code> .....	310, 312, 314, 316
<code>\glslongemfont</code> ..	<code>\glspar</code> .....	127
<code>\glslongfont</code> ..	<code>\GLSpl</code> .....	89, 100, 132
81, 82, 177, 182, 183, 185–	<code>\Glspl</code> .....	47, 89, 100, 132
187, 193, 194, 196, 198, 200, 201, 203,	<code>\glspl</code> .....	47, 89, 100, 132
206, 208, 210, 211, 213, 214, 217, 218,	<code>\GLSplural</code> .....	291, 292
220, 222, 224, 226, 227, 229, 231, 232,	<code>\Glsplural</code> .....	292
234, 236, 238, 240, 242, 243, 245, 246,	<code>\glsplural</code> .....	291, 292
248, 250, 252, 253, 256, 258, 260, 261,	<code>\glspluralaccessdisplay</code> .....	138
264, 266, 269, 271, 275, 277, 280, 283, 284	<code>\glspluralsuffix</code> .....	122, 173, 177
<code>\glslongfootnotefont</code> .....	<code>\glspostdescription</code> .....	
197, 198, 200, 220, 222, 234, 236, 256, 258	.....	15, 16, 166, 306–316, 318–322
<code>\glslonghyphenfont</code> .....	<code>\glspostinline</code> .....	317
.....	<code>\glspostlinkhook</code> ..	54, 55, 67–70, 83, 177–186
<code>\glslongonlyfont</code> .....	<code>\glsprestandardsort</code> .....	106
<code>\glslongpltok</code> .....	<code>\glsresetentrylist</code> .....	113, 125
. 174, 175, 193–197, 206, 208–212, 216,	<code>\glssee</code> .....	40–42
220, 224–226, 231, 234, 238, 239, 241–	<code>\glsseeformat</code> .....	38, 39, 44, 105, 110, 111
244, 248, 250, 253, 255, 260–263, 265–	<code>\glsseelist</code> .....	39
267, 269–271, 273, 274, 276–278, 282, 284	<code>\glssetabbrvfmt</code> .....	53, 59, 80,
<code>\glslongpluralaccessdisplay</code> .....	81, 153–158, 164, 177–181, 183–185, 187	
<code>\glslongtok</code> ..	<code>\glssetattribute</code> ..	193, 195, 197, 198, 200,
101, 173, 175, 193–197, 199,	201, 203, 204, 206, 208, 209, 211–214,	
201, 203, 206, 208–214, 216, 220, 221,	216, 220, 222, 224–227, 229, 231, 234,	
224–227, 229–231, 234, 235, 238–246,	236, 238–241, 243–246, 248, 250, 251,	
248, 250, 253, 255, 257, 260–263, 265–		
267, 269–271, 273, 274, 276–279, 282, 284		
<code>\glslonguserfont</code> .....		259–266

253, 255, 257, 260, 261, 263, 265–267, 269–271, 273, 274, 276–278, 280–282, 284	\glstxt@do@wrglossary . . . . . 8, 9, 11, 13
\glsssetcategoryattribute . . . . .	\glstxt@addloccllistfield . . . . . 13
. 89, 101, 103, 132, 148, 149, 151, 152, 165	\glstxt@addunused . . . . . 43
\glsssetnoexpandfield . . . . . 11, 12	\glstxt@applyabbrvfmt . . . . . 187
\glsssettoctitle . . . . . 108	\glstxt@applyabbrvstyle . . . . 171, 173, 190
\glssshortaccessdisplay . . . . . 142	\glstxt@counterrecord . . . . . 123
\glssshortpltok . . . . .	\glstxt@do@alsoindex@wrglossary . . . . 13
. . . . 174, 175, 193, 195–199, 201, 203, 209–214, 220, 221, 224–228, 234, 235, 238–246, 255, 257, 260, 261, 263, 265– 267, 269, 270, 274, 276–279, 281, 282, 284	\glstxt@doooption . . . . . 5, 15, 16, 22, 24
\glssshortpluralaccessdisplay . . . . 143	\glstxt@fields . . . . . 121
\glssshorttok . 101, 173–175, 193–197, 199, 201, 203, 208–214, 216, 220, 221, 224– 228, 230, 234, 235, 238, 239, 241–246, 248, 250, 255, 257, 260–263, 265–267, 269, 270, 273, 274, 276–279, 281, 282, 284	\glstxt@headentry@p . . . . . 84, 85
\glsssubentryitem . . . . .	\glstxt@hyperoutsidefalse . . . . . 56
. . . . . 124, 306–316, 318–320, 331, 342	\glstxt@hyperoutsidettrue . . . . . 56
\glsssymbolaccessdisplay . . . . . 140	\glstxt@ifnextpunc . . . . . 169
\glssymbolpluralaccessdisplay . 140, 141	\glstxt@ifpunctoken . . . . . 170
\glstarget 124, 306–316, 318–320, 331, 332, 342	\glstxt@indexonly@saveentrycounter . . . . . 13, 25
\GLstext . . . . . 291	\glstxt@keylist . . . . . 46, 47
\Glstext . . . . . 291	\glstxt@label . . . . . 344
\glstext . . . . . 291	\glstxt@langtag . . . . . 121
\glstextaccessdisplay . . . . . 137, 138	\glstxt@linkprefix . . . . . 121, 122
\glstextformat . . . . . 55, 57	\glstxt@makeglossaries . . . . . 104
\glstextup . . . . . 209	\glstxt@newabbreviation . . . . . 102, 172
\glstreechildpredesc . . . . . 318, 319	\glstxt@next . . . . . 170
\glstreechildprelocation . . . 318, 319, 321	\glstxt@org@@do@wrglossary . . . . . 25
\glstreegroupheaderfmt . . . . .	\glstxt@org@dohyperlink . . . . . 77
. . . . . 318–321, 332–338, 340	\glstxt@org@getgrouptitle . . . . . 112
\glstreeindent . . . . . 319, 320, 330–332	\glstxt@org@newignoredglossary . . . . 33
\glstreeitem . . . . . 317, 334, 341, 342	\glstxt@orgmakenoidxglossaries . . 43, 44
\glstreenamebox . . . . . 331, 332	\glstxt@pluralsuffixes . . . . . 121, 122
\glstreenamefmt . . . . . 318–320, 322–332	\glstxt@process . . . . . 125, 126
\glstreenavigationfmt . . 319–321, 332–337	\glstxt@provideignoredglossary . . . . 34
\glstreepredesc . . . . . 318–320	\glstxt@punclist . . . . . 169, 170
\glstreeprelocation . . . . . 317–320, 322	\glstxt@record . . . . . 10
\glstreesubitem . . . . . 317, 342	\glstxt@recordsee . . . . . 11
\glstreesubsubitem . . . . . 318, 342	\glstxt@resource . . . . . 120, 121
\GlstrLetField . . . . . 30	\glstxt@s@newignoredglossary . . . . . 33
\glstype . . . . . 55, 56, 67–70, 133, 177–186	\glstxt@s@provideignoredglossary . . 34
\glunset . . . . . 43, 55, 92, 93, 133	\glstxt@saveentrycounter . . . . . 8, 9, 11, 74
\glswrite . . . . . 39, 104	\glstxt@setbookindexmark . . . . . 343
\glswriteentry . . . . . 8, 9	\glstxt@setup@record . . . . . 12, 13, 24, 25
\Glsxtr . . . . . 48	\glstxt@shortcutsval . . . . . 121, 122
\glxtr . . . . . 48	\glstxt@texencoding . . . . . 122
	\glstxt@usesee . . . . . 38
	\glstxt@warnonexistsordo . . . . . 7, 13, 37
	\glstxt@writefields . . . . . 120
	\glstxt@abbrvfootnote . . . . .
	. . . . 197–199, 220–222, 234–236, 255–257



<code>\glxtrabbrvpluralsuffix</code> .....	122,	<code>\GlsXtrDefineAcShortcuts</code> .....	19, 20
	177, 193, 195, 198, 200, 201, 203, 206,	<code>\GlsXtrDefineOtherShortcuts</code> .....	19, 20
	209, 223, 237, 260, 269, 271, 275, 280, 282	<code>\glxtrdetoklocation</code> .....	131, 132
<code>\glxtrabbrvtype</code> .....	17, 175	<code>\glxtrdiscardperiod</code> .....	167
<code>\glxtraddallcrossrefs</code> .....	42	<code>\glxtrdisplayendloc</code> .....	113
<code>\glxtralias</code> .....	74	<code>\glxtrdisplayendloohook</code> .....	114
<code>\glxtrAltTreeIndent</code> .....	322	<code>\glxtrdisplaysingleloc</code> .....	114
<code>\glxtralttreeInit</code> .....	331, 336, 337	<code>\glxtrdisplaystartloc</code> .....	113
<code>\glxtrAltTreePar</code> .....	322	<code>\glxtrdoautoindexname</code> .....	75, 76, 159
<code>\glxtrAltTreeSetHangIndent</code> ...	322, 331	<code>\glxtrdopostpunc</code> .....	199, 222, 236, 257
<code>\glxtrAltTreeSetSubHangIndent</code> ...	332	<code>\glxtrdowrglossaryhook</code> .....	75
<code>\glxtralttreeSubSymbolDescLocation</code>	332	<code>\glxtrtremsuffix</code> .....	238, 240, 241,
<code>\glxtralttreeSymbolDescLocation</code> ..			243, 245, 246, 248, 250, 252, 253, 256, 258
	322, 331	<code>\GlsXtrEnableEntryCounting</code> .....	100
<code>\glxtrassignfieldfont</code> .....	59–66	<code>\GlsXtrEnableEntryUnitCounting</code> .....	89
<code>\glxtrautoindex</code> .....	160	<code>\GlsXtrEnableOnTheFly</code> .....	46, 48
<code>\glxtrautoindexassignsort</code> .....	160	<code>\glxtrfieldlistgadd</code> .....	123
<code>\glxtrautoindexentry</code> .....	160	<code>\glxtrfieldtitlecase</code> .....	154–157
<code>\glxtrbookindexatendgroup</code> .....	341	<code>\glxtrfieldtitlecasecs</code> .....	153
<code>\glxtrbookindexatsubendgroup</code> .....	342	<code>\glxtrfieldxifinlist</code> .....	126
<code>\glxtrbookindexatsubsubendgroup</code> ..	342	<code>\glxtrfirstscfont</code> .....	209
<code>\glxtrbookindexbetween</code> .....	341	<code>\glxtrfirstsmfont</code> .....	223
<code>\glxtrbookindexbookmark</code> .....	343	<code>\GlsXtrFmtDefaultOptions</code> .....	27
<code>\glxtrbookindexcols</code> .....	341	<code>\GlsXtrFmtField</code> .....	27
<code>\glxtrbookindexcolspread</code> .....	341	<code>\GlsXtrFormatLocationList</code>	50, 52, 328–330
<code>\glxtrbookindexfirstmarkfmt</code> .....	344	<code>\GLSxtrfull</code> .....	17, 18, 296
<code>\glxtrbookindexformatheader</code> .....	343	<code>\Glsxtrfull</code> .....	17, 18, 296, 297
<code>\glxtrbookindexgroupskip</code> .....	343	<code>\glxtrfull</code> .....	17, 18, 296
<code>\glxtrbookindexlastmarkfmt</code> .....	344	<code>\Glsxtrfullformat</code>	176, 189, 191, 194, 196,
<code>\glxtrbookindexname</code> .....	339, 342		198, 200, 202, 204, 207, 210, 212, 214,
<code>\glxtrbookindexparentchildsep</code>	339, 341		215, 218–220, 222, 224, 226, 228, 229,
<code>\glxtrbookindexparentsubchildsep</code> .			232–234, 236, 238, 240, 242, 244, 246,
	341, 342		247, 249, 251, 253, 255, 256, 258, 260,
<code>\glxtrbookindexprelocation</code> ...	339, 342		262, 264, 267, 270, 272, 275, 278, 280, 283
<code>\glxtrbookindexsubbetween</code> .....	342	<code>\glxtrfullformat</code> .....	
<code>\glxtrbookindexsubname</code> .....	342		176, 188, 189, 191, 194, 196, 198,
<code>\glxtrbookindexsubprelocation</code> ...	342		200, 202, 204, 207, 210, 212, 214, 215,
<code>\glxtrbookindexsubsubbetween</code> .....	342		217, 219, 220, 222, 224, 226, 228, 229,
<code>\glxtrbookindexthepage</code> .....	343, 344		232–234, 236, 238, 240, 242, 243, 245,
<code>\glxtrcat</code> .....	46, 47		247, 249, 251, 253, 254, 256, 258, 260,
<code>\glxtrchecknohyperfirst</code> .....	60–62		261, 264, 267, 270, 272, 275, 278, 280, 283
<code>\glxtrComputeTreeIndent</code> .....	331	<code>\GLSxtrfullpl</code> .....	17, 18, 296, 297
<code>\glxtrComputeTreeSubIndent</code> .....	331	<code>\Glsxtrfullpl</code> .....	17, 18, 297
<code>\Glsxtrdefaultsubsequentfmt</code> ...	189, 191	<code>\glxtrfullpl</code> .....	17, 18, 296
<code>\glxtrdefaultsubsequentfmt</code> ...	189, 191	<code>\Glsxtrfullplformat</code> .....	
<code>\Glsxtrdefaultsubsequentplfmt</code> .	189, 191		176, 188, 191, 194, 196, 198,
<code>\glxtrdefaultsubsequentplfmt</code> .	189, 191		200, 202, 204, 207, 210, 212, 214, 215,
<code>\GlsXtrDefineAbbreviationShortcuts</code>			218–220, 222, 224, 226, 228, 229, 232,
	19, 20		233, 235, 236, 238, 240, 242, 244, 246,



247, 249, 251, 253, 255, 256, 258, 261, 262, 264, 267, 270, 272, 275, 278, 280, 283	\glsxtrifwasfirstuse .....
\glsxtrfullplformat . 188, 191, 194, 196, 198, 200, 202, 204, 207, 210, 212, 214, 215, 218–220, 222, 224, 226, 228, 229, 232–234, 236, 238, 240, 242, 243, 246, 247, 249, 251, 253, 254, 256, 258, 260, 262, 264, 267, 270, 272, 275, 278, 280, 283	..... 59–62, 66–70, 73, 103, 168, 178, 180–186, 199, 200, 221, 222, 236, 257, 261, 263, 265, 274, 276, 279, 281
\glsxtrfullsep .....	\glsxtrindexaliased .....
.... 175, 176, 193–196, 199–205, 207, 209–217, 219, 221–233, 235, 237–254, 256–259, 268–270, 272, 274, 277–279, 283	..... 74
\glsxtrgenabbrvfmt .....	\glsxtrindexseealso .....
..... 53	..... 41, 42
\glsxtrgetgroupitle .....	\glsxtrinithyperoutside .....
..... 112, 343	..... 57
\glsxtrgroupfield .....	\glsxtrinitwrgloss .....
..... 128	..... 56, 133
\Glsxtrheadfirst .....	\glsxtrinitwrglossbeforefalse ...
..... 287	..... 55, 56
\glsxtrheadfirst .....	\glsxtrinitwrglossbeforetrue ....
..... 287	..... 55, 56
\Glsxtrheadfirstplural .....	\Glsxtrinlinefullformat 176, 178, 191, 199, 201, 202, 204, 205, 207, 213, 215– 217, 219, 221, 223, 228–231, 233, 235, 237, 245, 246, 248, 249, 251, 252, 254, 256, 258, 262, 264, 272, 275, 281, 283, 303
\glsxtrheadfirstplural .....	\glsxtrinlinefullformat 176–178, 191, 198, 200, 201, 203, 205, 207, 213, 215– 217, 219, 221, 222, 227, 229–231, 233, 235, 237, 245–247, 249, 250, 252, 254, 256, 258, 262, 264, 272, 275, 280, 283, 302
\Glsxtrheadfull .....	\Glsxtrinlinefullplformat .. 176, 179, 191, 199, 201, 202, 204, 205, 207, 213, 215–217, 219, 221, 223, 228–231, 233, 235, 237, 245, 247–249, 251, 253, 254, 257, 258, 262, 264, 272, 276, 281, 283, 303
\glsxtrheadfull .....	\glsxtrinlinefullplformat .....
..... 287	..... 176, 179, 180, 191, 199, 200, 202, 203, 205, 207, 213, 215– 217, 219, 221, 223, 227, 229–231, 233, 235, 237, 245–247, 249, 250, 252, 254, 256, 258, 262, 264, 272, 275, 281, 283, 303
\Glsxtrheadfullpl .....	\glsxtrininsertinsidefalse .....
..... 287	..... 193
\glsxtrheadfullpl .....	\glsxtrlocationhyperlink .....
..... 287	..... 115
\Glsxtrheadlong .....	\glsxtrlocrangefmt .....
..... 287	..... 114
\glsxtrheadlong .....	\GLSxtrlong .....
..... 287	..... 17, 18, 294, 295
\Glsxtrheadlongpl .....	\Glsxtrlong .....
..... 287	..... 17, 18, 295
\glsxtrheadlongpl .....	\glsxtrlong .....
..... 287	..... 17, 18, 294
\Glsxtrheadname .....	\glsxtrlonghyphen .....
..... 287	..... 275
\glsxtrheadname .....	\glsxtrlonghyphennoshort .....
..... 123, 124, 287	..... 271, 272
\Glsxtrheadplural .....	\glsxtrlonghyphenshort .....
..... 287	..... 270
\glsxtrheadplural .....	\GLSxtrlongpl .....
..... 287	..... 17, 18, 294, 295
\Glsxtrheadshort .....	\Glsxtrlongpl .....
..... 287	..... 17, 18, 295
\glsxtrheadshort .....	\glsxtrlongpl .....
..... 287	..... 17, 18, 294
\Glsxtrheadshortpl .....	\glsxtrlongshortdescname .....
..... 287	..... 194, 210, 225, 239, 240, 265, 270, 276
\glsxtrheadshortpl .....	\glsxtrlongshortdescsort .....
..... 287	..... 194, 210, 225, 239, 240, 265, 270, 276
\Glsxtrheadtext .....	\glsxtrmarkhook .....
..... 287	..... 285
\glsxtrheadtext .....	\glsxtrnewabbrevpresetkeyhook .....
..... 287	..... 174
\glsxtrhyperlink .....	
..... 78, 115	
\glsxtrhyphensuffix .....	
..... 269, 277	
\glsxtrifcounttrigger .....	
..... 92, 93	
\glsxtrifemptyglossary .....	
..... 113, 119, 124	
\glsxtrifhasfield .....	
..... 31, 74, 339	
\glsxtrifhyphenstart .....	
..... 268, 271, 273, 274, 277, 279	
\glsxtrifindexing .....	
..... 75	
\glsxtrifinmark .....	
..... 58, 84–87, 286, 287	
\glsxtrifnextpunc .....	
..... 169, 170	
\glsxtrifperiod .....	
..... 168, 169	
\glsxtrifrecordtrigger .....	
..... 134–136	

\glstrnewnumber .....	18	\glstrprelocation .....	
\glstrnewsymbol .....	18	.....	307, 309, 311, 313, 315, 317, 339
\glstrNoGlossaryWarning .....	20, 115	\glstrprotectlinks .....	77–79
\GlsXtrNoGlsWarningAutoMake .....	119	\GlsXtrRecordCounter .....	11
\GlsXtrNoGlsWarningBuildInfo .....	119	\glstrrecordtriggervalue .....	132
\GlsXtrNoGlsWarningCheckFile .....	119	\glstrregularfont .....	53, 59
\GlsXtrNoGlsWarningEmptyMain .....	119	\glstrresourcecount .....	120, 121
\GlsXtrNoGlsWarningEmptyNotMain ...	119	\glstrresourcefile .....	120
\GlsXtrNoGlsWarningEmptyStart .....	119	\glstrresourceinit .....	120
\GlsXtrNoGlsWarningHead .....	119	\glstrrestoremarkhook .....	285
\GlsXtrNoGlsWarningMisMatch .....	119	\glstrscfont .....	209
\GlsXtrNoGlsWarningNoOut .....	120	\glstrscsuffix .....	
\GlsXtrNoGlsWarningTail .....	120	....	210, 211, 213, 214, 217, 218, 220, 222
\glstronlydescname .....	284	\GlsXtrSetActualChar .....	163
\glstronlydescsort .....	284	\glstrsetaliasnoindex .....	13, 74
\glstronlysuffix .....	282	\GlsXtrSetEncapChar .....	163
\glstrorg@ifKV@glslink@hyper .....	53	\GlsXtrSetEscChar .....	163
\glstrorglong .....	173, 194, 271	\glstrsetfieldifexists .....	30
\glstrorgshort .....	173, 194	\GlsXtrSetLevelChar .....	163
\GLSxtrp .....	84	\glstrsetpopts .....	83
\Glsxtrp .....	84	\glstrsetupfulldefs .....	
\glsxtrp .....	83, 85	.....	177–180, 200, 222, 236, 257
\glstrparen .....		\GLSxtrshort .....	17, 18, 87, 288, 289
.....	175, 176, 193–196, 199–205, 207, 209–	\Glsxtrshort .....	17, 18, 289
.....	217, 219, 221, 223–233, 235, 237–254,	\glstrshort .....	17, 18, 288
.....	256–259, 268–270, 272, 274, 277–279, 283	\glstrshortdescname ...	203, 214, 228, 246
\Glsxtrpl .....	48	\glstrshorthyphen .....	280
\glxtrpl .....	48	\glstrshorthyphenlong .....	278
\glstrpostdescription .....	151, 166, 317	\glstrshortlongdescname .....	
\glstrposthyphenlong .....	279, 281	....	196, 212, 226, 242, 244, 267, 278, 281
\glstrposthyphenshort .....	274, 276	\glstrshortlongdescsort .....	
\glstrposthyphensubsequent .....		....	196, 212, 226, 242, 244, 267, 278, 281
.....	274, 276, 280, 281	\GLSxtrshortpl .....	17, 18, 288, 289
\glstrpostlink .....	167	\Glsxtrshortpl .....	17, 18, 289, 290
\glstrpostlinkendsentence .....	167	\glstrshortpl .....	17, 18, 289
\glstrpostlinkhook .....	167	\glstrsmfont .....	223
\glstrpostlocalreset .....	88, 90, 98	\glstrsmsuffix .....	
\glstrpostlocalunset .....	88, 90, 98	....	224, 226, 227, 229, 231, 232, 234, 236
\glstrpostlongdescription .....	32	\Glsxtrsubsequentfmt .....	
\glstrpostnamehook .....	156–158	.....	188, 191, 206, 217, 218,
\GlsXtrPostNewAbbreviation .....		.....	231, 233, 248, 250, 252, 254, 272, 275, 280
.....	175, 191, 193, 195,	\glxtrsubsequentfmt .....	
.....	197–199, 201, 203–206, 208, 209, 211–	.....	188, 191, 206, 217, 218,
.....	214, 216, 220, 221, 224, 225, 227, 229,	.....	231, 232, 248, 250, 252, 254, 271, 275, 280
.....	231, 234, 236, 238–246, 248, 250, 251,	\Glsxtrsubsequentplfmt .....	
.....	253, 255, 257, 260, 261, 263, 265–267,	.....	187, 191, 206, 217, 218,
.....	269–271, 273, 274, 276–279, 281, 282, 284	.....	231, 233, 249, 250, 252, 254, 272, 275, 280
\glstrpostreset .....	88, 90, 98		
\glstrpostunset .....	88, 90, 98		

<code>\glxtrsubsequentplfmt</code> .....	<code>\glxtrwordsep</code> 172, 268, 271, 273, 274, 277, 279
..... 187, 191, 206, 217, 218, 231, 232, 248, 250, 252, 254, 272, 275, 280	<code>\glxtrwrglossmark</code> ..... 22
<code>\glxtrsuppllocationurl</code> ..... 115	<b>H</b>
<code>\glxtrtagfont</code> ..... 166	<code>\hangindent</code> . 319, 320, 322, 330–332, 336, 337
<code>\Glsxtrtitlefirst</code> ..... 286–288, 300, 301	<code>\hbox</code> ..... 306
<code>\glxtrtitlefirst</code> ..... 286–288, 300	<code>\hfill</code> ..... 306
<code>\Glsxtrtitlefirstplural</code> .... 286–288, 301	<code>\href</code> ..... 77
<code>\glxtrtitlefirstplural</code> .... 286–288, 301	<code>\hsize</code> ..... 49, 50
<code>\Glsxtrtitlefull</code> ..... 287, 288, 303	<code>\hss</code> ..... 306
<code>\glxtrtitlefull</code> ..... 287, 288, 302, 303	<code>\hyperlink</code> ..... 78
<code>\Glsxtrtitlefullpl</code> ..... 287, 288, 303	<code>\hyperpage</code> ..... 159
<code>\glxtrtitlefullpl</code> ..... 287, 288, 303	<code>\hyperref</code> ..... 77, 115
<code>\Glsxtrtitlelong</code> ..... 287, 288, 302	hyperref package ..... 79, 159, 284, 297
<code>\glxtrtitlelong</code> ..... 286–288, 301	<b>I</b>
<code>\Glsxtrtitlelongpl</code> ..... 287, 288, 302	<code>\if</code> ..... 45
<code>\glxtrtitlelongpl</code> ..... 286–288, 302	<code>\if@glxtr@autoseeindex</code> .... 23, 24, 38, 41
<code>\Glsxtrtitlename</code> ..... 286, 287, 299	<code>\if@glxtr@format@override</code> ..... 160
<code>\glxtrtitlename</code> ..... 286, 287, 298, 299	<code>\if@glxtrdocdefrestricted</code> ..... 44
<code>\glxtrtitleorpdforheading</code> .....	<code>\if@glxtrindexcrossrefs</code> ..... 14, 42
..... 23, 123, 286, 287	<code>\ifblank</code> ..... 26, 46, 47, 104
<code>\Glsxtrtitleplural</code> ..... 286–288, 300	<code>\ifcase</code> .. 7, 12, 19, 20, 22, 44, 56, 109, 318, 342
<code>\glxtrtitleplural</code> ..... 286–288, 300	<code>\ifcsdef</code> ..... 25, 31, 33–36, 57, 71, 72, 83–87, 95, 107, 112, 113, 127, 130, 131, 153–156, 158, 167, 171, 187, 191, 309–316
<code>\Glsxtrtitleshort</code> ..... 286, 287, 298	<code>\ifcsstring</code> ..... 25, 149, 190
<code>\glxtrtitleshort</code> ..... 286, 287, 297	<code>\ifcsundef</code> ..... 31, 33– 35, 44, 49, 51, 79, 90, 95–99, 111, 112, 116, 126, 149, 190–192, 306, 323, 330, 344
<code>\Glsxtrtitleshortpl</code> ..... 286, 287, 298	<code>\ifcsvoid</code> ..... 42, 148
<code>\glxtrtitleshortpl</code> ..... 286, 287, 298	<code>\ifdef</code> ..... 13, 18, 24, 27, 36, 37, 39, 40, 49, 50, 73, 77, 78, 84–86, 107, 110, 111, 121, 122, 129, 151, 152, 163, 166, 259, 286, 297–303, 306–309, 317–321, 332–337, 340, 343, 344
<code>\Glsxtrtitletext</code> ..... 286–288, 299	<code>\ifdefempty</code> ..... 7–9, 29, 33–35, 38, 39, 57, 58, 89, 101, 104, 107, 114, 125, 128, 132, 133, 165, 172, 187, 341
<code>\glxtrtitletext</code> ..... 286–288, 299	<code>\ifdefequal</code> ..... 43, 119, 128
<code>\GlsXtrTotalRecordCount</code> ..... 132	<code>\ifdefstring</code> ..... 6, 31, 160, 165, 340
<code>\glxtrtreetopindent</code> ..... 322, 330	<code>\ifdefvoid</code> ... 37, 40–43, 78, 95, 111, 115, 128
<code>\glxtrundefaction</code> .. 7, 13, 25, 33, 34, 36, 37	<code>\ifdim</code> ..... 49, 50, 103, 323–330
<code>\glxtrundeftag</code> ..... 25, 110, 111	<code>\IfFileExists</code> .... 21, 115, 119, 120, 122, 305
<code>\glxtrunsrtdo</code> ..... 126	<code>\ifglossaryexists</code> ..... 37
<code>\GlsXtrUseAbbrStyleFmts</code> .....	<code>\ifglsacronym</code> ..... 17, 119
..... 195, 197, 203, 205, 206, 208, 209, 211, 213, 216, 225, 227, 230, 239, 241, 243, 244, 247, 252, 255, 263, 265, 266, 268, 270, 271, 273, 276, 278, 282, 284	<code>\ifglsacrshortcuts</code> ..... 19
<code>\GlsXtrUseAbbrStyleSetup</code> 202, 204, 205, 208, 215, 218, 230, 232, 247, 251, 252, 255	<code>\ifglsautomake</code> ..... 107, 119, 122
<code>\glxtruserfield</code> ..... 259	<code>\ifglsentrycounter</code> ..... 31
<code>\glxtruserparen</code> ..... 260–267	
<code>\glxtrusersuffix</code> ..... 260, 261, 264, 266	
<code>\glxtruseseealsoformat</code> ..... 39	
<code>\glxtruseseeeformat</code> ..... 38	
<code>\GlsXtrWarnDeprecatedAbbrStyle</code> 171, 192	
<code>\GlsXtrWarning</code> ..... 46, 47	
<code>\glxtrword</code> ..... 172	

<code>\ifglstryexists</code> .....	8,		
	36, 37, 46, 47, 50, 59, 128, 149, 166, 167		
<code>\ifglshasfield</code> .....	147		
<code>\ifglshaslong</code> .....	27, 259		
<code>\ifglshasparent</code> .	123, 125, 128, 323, 325, 326		
<code>\ifglshasshort</code> .....	53, 59		
<code>\ifglshassymbol</code> .....	168, 318–320, 322		
<code>\ifglshindexonlyfirst</code> .....	75		
<code>\ifglshnogroupskip</code>	307, 309–319, 321, 332, 340		
<code>\ifglshnonumberlist</code> .....	52		
<code>\ifglshnopostdot</code> .....	15, 339		
<code>\ifglssanitizesort</code> .....	106		
<code>\ifglssubentrycounter</code> .....	31		
<code>\ifglssused</code> .....	42, 43,		
	73, 75, 91, 100, 103, 187, 323–325, 327–329		
<code>\ifglshxindy</code> .....	115, 117, 118		
<code>\ifglshxtr@hyperoutside</code> .....	57, 58		
<code>\ifglshxtrinitwrglossbefore</code>	55, 57, 58, 133		
<code>\ifglshxtrininsertinside</code> .....			
	. 180–186, 189, 194, 196, 198–207, 210,		
	212–224, 226–240, 242–258, 260–262,		
	264, 267, 268, 271, 274–277, 279, 281, 283		
<code>\ifHy@hyperindex</code> .....	159		
<code>\ifinlistcs</code> .....	28, 44		
<code>\ifinner</code> .....	23		
<code>\ifKV@glslink@hyper</code> .....	53, 56–58		
<code>\ifKV@glslink@local</code> .....	55, 133		
<code>\ifKV@glslink@noindex</code> ....	8, 9, 11, 74, 75		
<code>\ifmmode</code> .....	23		
<code>\ifnum</code>	14, 91, 99, 100, 111, 120, 132, 319, 320, 331		
<code>\ifstrempy</code> .....	130		
<code>\ifstrequal</code> .....	16, 21		
<code>\ifthenelse</code> .....	119		
<code>\IfTrackedLanguageFileExists</code> ....	304		
<code>\ifundef</code> .....	29, 104, 165, 166		
<code>\ifx</code> ...	8–10, 39, 49, 50, 104, 108, 113–115,		
	122, 160, 161, 163, 170, 172, 174, 268, 338		
<code>\immediate</code> .....	91, 99, 116, 122		
<code>\index</code> .....	160		
<code>\indexspace</code> .	307, 318–321, 332–338, 340, 343		
<code>\input</code> .....	304		
<code>\inputencodingname</code> .....	122		
<code>\istfilename</code> .....	104		
<code>\item</code> ....	117, 118, 306–309, 317–319, 333, 334		
<b>J</b>			
<code>\jobname</code> .....	115, 117–120, 122		
<b>K</b>			
<code>\key@ifundefined</code> ....	11, 12, 26, 71, 124, 127		
<code>\KV@glslink@hyperfalse</code> ....	60, 73, 78, 79		
<code>\KV@glslink@hypertrue</code> .....	79		
<code>\KV@glslink@noindexfalse</code> ....	73, 74		
<code>\KV@glslink@noindextrue</code> ....	74, 79		
<b>L</b>			
<code>\LaTeX</code> .....	117, 118		
<code>\leaders</code> .....	306		
<code>\leavevmode</code> .....	33, 56		
<code>\let</code> .....	5, 7–13,		
	15, 17–19, 23–25, 27, 32, 43, 45, 48, 49,		
	51–70, 72–74, 76–80, 83, 89, 90, 98–105,		
	107–112, 120, 122, 124–126, 128, 133,		
	154, 155, 157–161, 165, 166, 170–174,		
	177–186, 189, 191, 200, 222, 236, 257,		
	285–288, 317, 318, 322, 323, 334, 341–343		
<code>\letabbreviationstyle</code> ..	199, 201, 202,		
	204, 207, 208, 214, 215, 228, 230, 246, 247		
<code>\letcs</code> .....	26, 29, 38, 39,		
	43, 57, 71, 110–112, 127, 128, 153–158, 344		
<code>\levelchar</code> .....	163		
<code>\listadd</code> .....	95		
<code>\listbreak</code> .....	165		
<code>\listcsadd</code> .....	28		
<code>\listcseadd</code> .....	28, 96		
<code>\listcsgadd</code> .....	28, 44		
<code>\listcsxadd</code> .....	28, 96		
<code>\loadglstentries</code> .....	45, 117		
<code>\long</code> .....	32		
<b>M</b>			
<code>\MakeAcronymsAbbreviations</code> .....	103		
<code>\makeatletter</code> .....	115, 120, 162		
<code>\makeatother</code> .....	162		
<code>\makebox</code> .....	306, 331, 332		
<code>\makefirstuc</code> .....	165		
<code>makeglossaries</code> .....	109		
<code>\makeglossaries</code> ....	104, 116, 118, 119, 122		
<code>\makeglossary</code> .....	104		
<code>makeindex</code> .....	345		
<code>makeindex</code> .....	103		
<code>\makenoidxglossaries</code> .....	118		
<code>\MakeTextUppercase</code> .....	287		
<code>\MakeUppercase</code> .....	286, 287		
<code>\marginpar</code> .....	23		
<code>\markboth</code> .....	286		
<code>\markright</code> .....	285		
<code>\maxdimen</code> .....	49, 50		

<code>\mbox</code> .....	308, 309, 331	<code>longplural</code> .....	144, 147, 302
<code>\medskip</code> .....	119, 127	<code>name</code> .....	38, 137, 144, 160, 290, 298
<code>\MessageBreak</code> .....		<code>plural</code> .....	138, 145, 193, 291, 292, 299
....	44, 45, 48, 91, 100, 104, 107, 108, 190	<code>see</code> .....	15, 24, 37, 38, 40, 43, 45, 105
<code>mfistuc</code> package .....	165	<code>seealso</code> .....	15, 37, 38, 40, 42, 43, 356
<code>\mfistucMakeUppercase</code> .....		<code>short</code> .....	142, 146, 171
59–70, 72, 81, 82, 84, 87, 94, 102, 136–		<code>shortplural</code> .....	143, 147, 171
147, 156, 157, 178, 180, 181, 183, 185–189		<code>symbol</code> .....	140, 145
<code>\mfu@checkword@arg</code> .....	165	<code>symbolplural</code> .....	140, 141, 146
<code>\mfu@checkword@do</code> .....	165	<code>text</code> .....	77,
		137, 138, 144, 145, 193, 195, 290, 291, 299	
		<code>\newglossarystyle</code> .....	340
		<code>\newif</code> .....	55, 159, 193
		<code>\newlength</code> .....	322
		<code>\newnum</code> .....	18
		<code>\newrobustcmd</code> .....	27,
		29, 30, 39, 40, 71, 72, 83, 84, 93, 94, 112,	
		123, 126, 127, 130, 131, 133–135, 165,	
		166, 177–187, 268, 286, 288–297, 323–329	
		<code>\newsym</code> .....	18
		<code>\newterm</code> .....	151
		<code>\newtoks</code> .....	171
		<code>\newwrite</code> .....	104
		<code>\nobreak</code> .....	308, 309, 318–321, 333–336
		<code>\NoCaseChange</code> .....	84–87, 288–297
		<code>\noexpand</code> ..	10, 11, 21, 39, 41, 42, 101, 116,
		120, 125, 126, 128, 161, 174, 305, 341, 342	
		<code>\nofiles</code> .....	119
		<code>\noindent</code> .....	119, 320, 321, 334–337
		<code>\nopagebreak</code> .....	318–321, 332–339, 343
		<code>\nopostdesc</code> .....	33, 46, 47, 108, 151
		<code>\nr</code> .....	7, 12, 14, 19, 20, 22, 56, 109
		<code>\ns@GLSxtrfull</code> .....	178
		<code>\ns@Glsxtrfull</code> .....	178
		<code>\ns@glxtrfull</code> .....	177
		<code>\ns@GLSxtrfullpl</code> .....	180
		<code>\ns@Glsxtrfullpl</code> .....	179
		<code>\ns@glxtrfullpl</code> .....	179
		<code>\ns@GLSxtrlong</code> .....	183
		<code>\ns@Glsxtrlong</code> .....	182
		<code>\ns@glxtrlong</code> .....	182
		<code>\ns@GLSxtrlongpl</code> .....	186
		<code>\ns@Glsxtrlongpl</code> .....	185, 186
		<code>\ns@glxtrlongpl</code> .....	185
		<code>\ns@GLSxtrshort</code> .....	181
		<code>\ns@Glsxtrshort</code> .....	181
		<code>\ns@glxtrshort</code> .....	180
		<code>\ns@GLSxtrshortpl</code> .....	184
		<code>\ns@Glsxtrshortpl</code> .....	184
<b>N</b>			
<code>\NeedsTeXFormat</code> .....	5, 305, 339		
<code>\new@glossaryentry</code> .....	45, 107		
<code>\new@ifnextchar</code> .....			
71, 72, 93, 94, 130, 133–135, 169, 177–186			
<code>\newabbr</code> .....	17, 18		
<code>\newabbreviation</code> .....	17, 18		
<code>\newabbreviationhook</code> .....	174		
<code>\newabbreviationstyle</code> ..	193–197, 199,		
201–216, 218, 219, 221, 223, 225–228,			
230, 232, 234, 235, 238–244, 246–248,			
250–253, 255, 257, 260–263, 265–267,			
269–271, 273, 274, 276–279, 281, 282, 284			
<code>\newacronym</code> .....	101, 102		
<code>\newacronymhook</code> .....	101		
<code>\newacronymstyle</code> .....	102, 103		
<code>\newcommand</code> .....	5–8, 10–12, 14–		
29, 31–39, 41–43, 45–48, 50–53, 55, 56,			
59, 60, 71, 72, 74–76, 78, 79, 82–91, 93–			
103, 107, 109–112, 114–153, 159–173,			
175–187, 189–192, 194, 196, 197, 203,			
209, 223, 237, 259, 260, 268, 269, 271,			
273, 274, 277, 279, 282, 284–305, 307,			
317, 321–323, 330, 331, 339, 340, 343, 344			
<code>\newcount</code> .....	14, 120, 129		
<code>\newentry</code> .....	18		
<code>\newglossary</code> .....	16, 104		
<code>\newglossaryentry</code> .....			
18, 45, 89, 97, 101, 151, 152, 174			
<code>\newglossaryentry options</code>			
alias .....	15, 37, 40–43		
desc .....	141, 146		
descplural .....	141, 142, 146		
first 77, 138, 139, 145, 193, 292–294, 300, 345			
firstplural .....	139, 145, 193, 293, 301, 345		
group .....	127, 128		
loclist .....	28		
long .....	143, 147, 301		

<code>\ns@glstrshortpl</code> .....	183	<code>\PackageError</code> .....	6, 11, 21, 24, 44, 45, 48, 49, 71, 72, 74, 83, 84, 89, 90, 97, 99, 100, 102, 104, 106, 107, 113, 122, 125, 127, 129, 190–192, 305, 306
<code>\null</code> .....	20	<code>\PackageWarning</code> .....	15
<code>\number</code> .....	96–99, 120, 130	<code>\PackageWarningNoLine</code> .....	15
<code>\numexpr</code> .....	96, 99	<code>\pageref</code> .....	31
<b>O</b>			
<code>\or</code> .....	7, 13, 19, 20, 22, 45, 56, 318, 342	<code>\par</code> ..	119, 120, 308, 309, 318–322, 331–337, 340
<code>\org@glossaryentrynumbers</code> .....	50, 108	<code>\parindent</code> .....	317, 319, 320, 322, 331, 332, 334–338, 341
<code>\org@glossarytitle</code> .....	108	<code>\parskip</code> .....	317, 319, 320, 334–336, 341
<code>\org@ifKV@glslink@hyper</code> .....	56, 58	<code>\PassOptionsToPackage</code> .....	5, 21
<b>P</b>			
<code>\p@glshyp@opt</code> .....	76	<code>\pdfbookmark</code> .....	340
package options:		<code>\preglossarypreamble</code> .....	31
abbreviations .....	16, 17	<code>\preto</code> .....	74
accsupp .....	20, 137	<code>\print@noop@unsrtglossaryunit</code> ...	11, 13
acronym .....	17	<code>\print@op@unsrtglossaryunit</code> .....	13
automake .....	107, 117, 122	<code>\printabbreviations</code> .....	17
true .....	122	<code>\printglossaries</code> .....	105, 118
autoseeindex .....	24	<code>\printglossary</code> .....	17, 105, 118
false .....	23	<code>\printglossary options</code>	
debug		nonumberlist .....	52
showtargets .....	78	type .....	107
docdef .....	14, 43, 44, 89, 97	<code>\printnoidxglossaries</code> .....	118
false .....	45	<code>\printnoidxglossary</code> .....	105, 106, 118
restricted .....	14	<code>\printnumbers</code> .....	18, 152
true .....	45	<code>\printsymbols</code> .....	18, 152
docdefs		<code>\printunsrtglossary</code> .....	124
restricted .....	44	<code>\printunsrtglossaryentryprocesshook</code>	125
nonumberlist .....	50	<code>\printunsrtglossaryhandler</code> .....	126
nopostdot .....	15	<code>\printunsrtglossarypredoglossary</code> ..	125
false .....	15	<code>\printunsrtglossaryskipentry</code> .....	125
numbers .....	18	<code>\printunsrtglossaryunit</code> .....	13, 127
postdot .....	15	<code>\printunsrtglossaryunitsetup</code> .....	126
record .....	7, 12, 43, 53, 104, 120, 354	<code>\ProcessOptions</code> .....	306
alsoindex .....	8, 10	<code>\ProcessOptionsX</code> .....	22
only .....	7, 8	<code>\protect</code> .....	84–87, 144–147, 172, 175, 176, 193–199, 201–203, 205–214, 216– 222, 224–236, 238–258, 260–263, 265– 267, 269–274, 276–279, 281–284, 288–297
shortcuts .....	19	<code>\protected@csedef</code> .....	30, 322
ac .....	19	<code>\protected@csxdef</code> .....	30, 323
all .....	19	<code>\protected@edef</code> .....	.. 49, 77, 101, 111, 112, 126–128, 160, 174
false .....	19	<code>\protected@write</code> .....	.... 10, 11, 44, 52, 104, 105, 120–123, 343
none .....	19	<code>\providecommand</code> .....	16, 26, 39, 52, 72, 74, 91, 99, 104, 116, 121, 306, 339
true .....	19	<code>\ProvidesFile</code> .....	304
style .....	21		
stylemods .....	21		
symbols .....	18, 152		
undefaction .....	36		
error .....	6		
warn .....	6		
xindy .....	39, 40		

<code>\ProvidesPackage</code> .....	5, 305, 339	<code>\setabbreviationstyle</code> .....	102, 194, 202
<b>Q</b>		<code>\setacronymstyle</code> .....	102, 103
<code>\quotechar</code> .....	163	<code>\setentrycounter</code> .....	113
<b>R</b>		<code>\SetGenericNewAcronym</code> .....	103
<code>\raggedright</code> .....	311, 312, 315, 316, 341	<code>\setglossarystyle</code> .....	22, 108, 306–309, 318, 320, 321, 332–338, 340
<code>\relax</code> .....	7, 10, 12–14, 17–20, 22, 24, 27, 45, 48–51, 55, 56, 76, 83, 90, 91, 99, 104, 105, 108, 110, 111, 113, 120–122, 128, 132, 161, 163, 165, 168, 172, 173, 268, 318–320, 323–332, 336–338, 341–343	<code>\setkeys</code> .....	9, 21, 24, 57, 58, 75, 101, 108, 133, 173, 174
<code>relsize package</code> .....	223	<code>\setlength</code> .....	49, 50, 317, 319, 320, 322, 331, 332, 334–336, 341
<code>\renewcommand</code> .....	6, 7, 13–17, 19–22, 24, 32–38, 43–45, 48–53, 55, 71–73, 75, 77–79, 83, 88–91, 94, 95, 97–109, 111– 113, 115, 116, 126, 127, 132, 151, 153– 158, 164–167, 176, 191, 193–258, 260– 267, 269–285, 306–321, 331–337, 341–343	<code>\settowidth</code> .....	103, 322–330
<code>\renewenvironment</code> .....	307, 309–317, 319, 320, 331, 334–337, 340	<code>\setupglossaries</code> .....	5, 24
<code>\renewglossarystyle</code> ....	306–321, 331–337	<code>\sfcode</code> .....	15, 16, 168, 317
<code>\renewrobustcmd</code> .....	58, 78	<code>\small</code> .....	23
<code>\RequireGlossariesExtraLang</code> .....	304	<code>\space</code> .....	6, 11, 39, 44, 46, 48, 74, 89–91, 97, 99, 100, 102–106, 108, 116, 119, 122, 125, 127, 129, 168, 172, 176, 194, 306, 307, 317–320, 322, 330, 339
<code>\RequirePackage</code> .....	5, 20–22, 305, 339	<code>\spacefactor</code> .....	15, 16, 168, 173, 317
<code>\reserved@a</code> .....	169, 170	<code>\string</code> .....	6, 10, 11, 39, 40, 44–46, 48, 52, 57, 71, 72, 74, 83, 84, 89–91, 97, 99, 100, 102–108, 116– 123, 125, 127, 129, 155, 157, 158, 160, 343
<code>\reserved@b</code> .....	169, 170	<code>\strut</code> .....	306–316, 320
<code>\reserved@d</code> .....	170	<code>\sty</code> .....	16
<code>\RestoreAcronyms</code> .....	102, 103	<code>\subglossentry</code> .....	108, 128, 306–316, 318–320, 331, 342
<code>\rGLS</code> .....	132	<code>\subitem</code> .....	317, 318
<code>\rGls</code> .....	132	<code>\subsubitem</code> .....	318
<code>\rgls</code> .....	132	<b>T</b>	
<code>\rGLSformat</code> .....	135	<code>\tablehead</code> .....	313–316
<code>\rGlsformat</code> .....	134	<code>\tabletail</code> .....	313–316
<code>\rglsformat</code> .....	134, 136	<code>\tabularnewline</code> .....	309–317
<code>\rGLSpl</code> .....	132	<code>\TeX</code> .....	117
<code>\rGlspl</code> .....	132	<code>\texorpdfstring</code> ....	27, 84–87, 286, 297–303
<code>\rglspl</code> .....	132	<code>textcase package</code> .....	285
<code>\rGLSplformat</code> .....	136	<code>\textsc</code> .....	209
<code>\rGlsplformat</code> .....	135	<code>\textsmaller</code> .....	223
<code>\rglsplformat</code> .....	134, 136	<code>\texttt</code> .....	23, 116–119
<code>\romannumeral</code> .....	322, 323, 330	<code>\the</code> .....	101, 102, 114, 115, 120, 163, 174, 175, 193–201, 203–206, 208–214, 216, 220–222, 224–231, 234–236, 238–246, 248, 250, 251, 253, 255, 257, 260–263, 265–267, 269–271, 273, 274, 276–282, 284
<b>S</b>		<code>\theGlsentrycounter</code> ....	8–10, 57, 58, 133
<code>\s@glshyp@opt</code> .....	76	<code>\theHglentrycounter</code> ....	8–10, 57, 58, 133
<code>\s@glsextr@enabletagging</code> .....	164	<code>\theindex</code> .....	159
<code>\s@glsextrifhasfield</code> .....	29	<code>\this@dialect</code> .....	304
<code>\s@printunsrtglossary</code> .....	124, 126		
<code>\seealso name</code> .....	39, 40		
<code>\seename</code> .....	38		

\thisgrptitle .....	343	\warn@noprintglossary .....	105, 108
\toks@ .....	114, 115, 163	\write .....	39, 91, 99, 104, 116, 122
tracklang package .....	121		
		<b>X</b>	
<b>U</b>		\x .....	115
\u .....	129	\xcapitalisewords .....	153
\undef .....	13, 164	\xdef .....	108, 126
\underline .....	166	\xifinlist .....	95
\unskip .....	33, 43, 306	\xifinlistcs .....	29
\usepackage .....	118, 119	xindy .....	345
		xindy .....	103
<b>V</b>		xkeyval package .....	5
\val .....	7, 12, 14, 19, 20, 22, 56, 109	\XKV@checkchoice .....	52
		\XKV@plfalse .....	52
<b>W</b>		\XKV@resa .....	52
\warn@nomakeglossaries .....	105	\XKV@sttrue .....	52