

glossaries-extra.sty v1.25: documented code

Nicola L.C. Talbot

Dickimaw Books

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

2017-11-24

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

1	Main Package Code (glossaries-extra.sty)	5
1.1	Package Initialisation and Options	5
1.2	Extra Utilities	25
1.3	Modifications to Commands Provided by glossaries	33
1.3.1	Existence Checks	37
1.3.2	Document Definitions	45
1.3.3	Existing Glossary Style Modifications	50
1.3.4	Entry Formatting, Hyperlinks and Indexing	54
1.3.5	Entry Counting	89
1.3.6	Acronym Modifications	102
1.3.7	Indexing and Displaying Glossaries	104
1.3.8	Support for bib2gls	132
1.4	Integration with glossaries-accsupp	140
1.5	Categories	151
1.6	Abbreviations	176
1.6.1	Abbreviation Styles Setup	195
1.6.2	Predefined Styles (Default Font)	198
1.6.3	Predefined Styles (Small Capitals)	215
1.6.4	Predefined Styles (Fake Small Capitals)	229
1.6.5	Predefined Styles (Emphasized)	243
1.6.6	Predefined Styles (User Parentheses Hook)	265
1.6.7	Predefined Styles (Hyphen)	274
1.6.8	Predefined Styles (No Short on First Use)	288
1.7	Using Entries in Headings	291
1.8	Multi-Lingual Support	310
2	Style Adjustments (glossaries-extra-stylemods.sty)	312
2.1	Package Initialisation	312
2.2	List-Like Styles	313
2.3	Longtable Styles	316
2.4	Long Ragged Styles	318
2.5	Supertabular Styles	320
2.6	Super Ragged Styles	322
2.7	Inline Style	324
2.8	Tree Styles	324
2.9	Multicolumn Styles	341

3 bookindex style (glossary-bookindex.sty)	347
3.1 Package Initialisation and Options	347
Glossary	353
Change History	354
Index	370

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/11/24 v1.25 (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, which now indicates if
358%the post-description punctuation has been suppressed.
359%\changes{1.21}{2017-11-03}{new}
360% \begin{macrocode}
361\define@key{glossaries-extra.sty}{postpunc}{%
362 \glsxtr@doption{nopostdot=false}%
363 \ifstrequal{#1}{dot}%
364 {%
365 \renewcommand*{\@glsxtr@defpostpunc}{%
366 \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
367 }%
368 }%
369 {%
370 \ifstrequal{#1}{comma}%
371 {%
372 \renewcommand*{\@glsxtr@defpostpunc}{%
373 \renewcommand*{\glspostdescription}{,}%
374 }%
375 }%
376 {%
377 \ifstrequal{#1}{none}%
378 {%
379 \glsxtr@doption{nopostdot=true}%
380 \renewcommand*{\@glsxtr@defpostpunc}{%
381 \renewcommand*{\glspostdescription}{}%
382 }%
383 }%
384 {%
385 \renewcommand*{\@glsxtr@defpostpunc}{%
386 \renewcommand*{\glspostdescription}{#1}%
387 }%
388 }%
389 }%
390 }%
391 }

```

`glsxtrabbrvtype` Glossary type for abbreviations.

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

`abbreviationsdef` Set by abbreviations option.

```
393 \newcommand*{\@glsxtr@abbreviationsdef}{}%
```

`abbreviationsdef`

```

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

```



```

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

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

```

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

```

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

```

AbbreviationShortcuts Enable shortcut commands for the abbreviations. Unlike the analogous command provided by glossaries, this uses \newcommand instead of \let as a safety feature (except for \newabbr which is also provided with \GlsXtrDefineAcShortcuts).

```

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

437   \providecommand*\newabbr{\newabbreviation}%

```

Disable this command after it's been used.

```

438   \let\GlsXtrDefineAbbreviationShortcuts\relax

```

439 }

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

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

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

Disable this command after it's been used.

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

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

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

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

@setupshortcuts Command used to set the shortcuts option.

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

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

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

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

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

```

516      \glsacrshortcutstrue

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

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

    Leave none and false as last option.

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

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

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

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

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

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

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

```

stylemods

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

glstr@do@style

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

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

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

Defer actual style change:

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

Set this as the default style:

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

Set this style:

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

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

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

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

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

sxtrwrglossmark Does nothing by default.

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

debug Provide extra debug options.

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

Pass all other options to glossaries.

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

Process options.

```
615 \ProcessOptionsX
```

Load glossaries if not already loaded.

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

Load the glossaries-accsupp package if required.

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

Redefine \glspostdescription if required.

618 \let\@glstr@defpostpunc

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

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

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

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

r@doseeglossary

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

oindex@glossary

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

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

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

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

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

```

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

```

\@glo@autosee If \@glo@autosee has been defined (glossaries v4.30 onwards), redefine it to test the autoseeindex option.

```

657 \ifdef\@glo@autosee
658 {%
659 \renewcommand*{\@glo@autosee}{%
660 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
661 }%
662 {}

```

checkseeallowed Don't prohibit the use of the see key before the indexing files have been opened if the automatic see indexing has been disabled, since it's no longer an issue.

```

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

```

Define abbreviations glossaries if required.

```

666 \@glxtr@abbreviationsdef
667 \let\@glxtr@abbreviationsdef\relax

```

Setup shortcuts if required.

```

668 \@glxtr@setupshortcuts

```

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

```

669 \@glxtr@redef@for@gl@sentries

```

ariesextrasetup Allow user to set options after the package has been loaded. First modify \glxtr@doooption so that it now uses \setupglossaries:

```

670 \renewcommand{\glxtr@doooption}[1]{\setupglossaries{#1}}%

```

Now define the user command:

```

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

```


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

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

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

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

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

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

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

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

Set up record option if required.

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

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

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

1.2 Extra Utilities

`trifemptyglossary` `\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.)

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

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

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

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

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

vide@storagekey Unstarred version.

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

```

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

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

```

Provide the no-link command if not already defined.

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

vide@storagekey Starred version.

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

```

```

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

```

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

`\GlsXtrFmtField`

```

745 \newcommand{\GlsXtrFmtField}{useri}

```

`\GlsXtrFmtDefaultOptions`

```

746 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

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

```

`\@glstrfmt` Unstarred form.

```

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

```

`\s@glstrfmt` Starred form.

```

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

```

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

```

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

```

`\@@glstrfmt` Actual inner working.

```

754 \newcommand*{\@@glstrfmt}[4]{}

```

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

```

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

```

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

```

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

```

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

```

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

```

`\glxtrentryfmt` No link or indexing.

```

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

```

`@glxtrentryfmt`

```

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

```

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

```

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

```

trfieldlistgadd Similarly but uses \listcsgadd.

```

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

trfieldlistead Similarly but uses \listcseadd.

```

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

trfieldlistxadd Similarly but uses \listcsxadd.

```

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

Now provide commands to iterate over these lists.

fielddolistloop

```

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

fieldforlistloop

```

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

List element tests:

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

```

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

trfieldxifinlist Expands item.

```

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

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

```

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

```

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

```

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

```

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

```

`\lsxtrifhasfield` Unstarred version adds grouping.

```

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

```

`\lsxtrifhasfield` Starred version omits grouping.

```

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

```

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

Just uses `\ifcsundef`.

```

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

```

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

```

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

```

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

```

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

```

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

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

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

```
854 \newcommand*{\glxtredefield}[2]{\csedef{glo@\glstetoklabel{#1}@#2}}
```

`etfieldifexists`

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

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

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

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

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

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

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

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

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

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

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

`xGlsXtrSetField`

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

eGlsXtrSetField

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

XtrIfFieldEqStr

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

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

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

lossarypreamble

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

lossarypreamble

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



```

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

```

1.3 Modifications to Commands Provided by glossaries

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

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

`ewglossaryentry`

```

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

```

`ewglossaryentry` Starred version.

```

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

```

`ewglossaryentry` Unstarred version.

```

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

```

```

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

```

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

```

961 \newcommand*{\glxtrpostlongdescription}{\leavevmode\unskip\nopostdesc}

```

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

`\ignoredglossary` Redefine to check for star.

```

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

```

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

```

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

```

```

991 }%
992 }

```

ignoredglossary Starred form.

```

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

```

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

```

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

```

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

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

ignoredglossary Unstarred version.

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

ignoredglossary Starred form.

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

```

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

```

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

```

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

```

1.3.1 Existence Checks

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

```

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

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

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

```

There are now three types of cross-references: the see key (as original), the alias key (from glossaries-extra v1.12) and the seealso key (from glossaries-extra v1.16). The original see key

needs to have a corresponding field (which it doesn't with the base glossaries package).

`ryentryposthook` Hook into end of `\newglossaryentry` to add “see” value as a field.

```

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

```

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

```

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

```

`\glxtr@usesee`

```

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

```

`\@glxtr@usesee`

```

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

```

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

```

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

```

`lsseeitemformat` glossaries originally defined `\glsseeitemformat` to use `\glsentryname` but in v3.0 this was switched to use `\glsentrytext` due to problems occurring with the name field being sanitized. Since this is no longer a problem, glossaries-extra restores the original definition as it

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

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

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

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

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

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

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

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

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

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

`\glstrindexseealso` If `\@xdycrossrefhook` is defined, provide a `seealso` crossref class. Otherwise this just does `\glssee` with `\seealso` as the tag. The hook is only defined if both `xindy` and `glossaries v4.30+` are being used.

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



```

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

```

Append to class list.

```

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

```

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

```

1240   \newrobustcmd*{\glxtrindexseealso}{\glsosee[\seealso@name]}
1241 }

```

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

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

```

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

```

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

```

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

```

Add to the key mappings.

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

Set the default value.

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

Assign the field values.

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

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

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

Provide user-level commands to access the values.

\glstralias

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

trseealsolabels

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

Add to the \@glo@autosee hook.

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

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

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

```

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

```

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

`\glstralias`

```

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

```

`trseealsolabels`

```

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

```

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

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

```

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

```

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

```

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

```

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

```

1315 \AtEndDocument{\if@glstrindexcrossrefs\glstraddallcrossrefs\fi}

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

xtrunusedformat

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

1.3.2 Document Definitions

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

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

Add marker to \@do@seeglossary

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

Check for docdefs=restricted:

```
1372   \if@glxtrdocdefrestricted
```

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

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

Disable document definitions.

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

```

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

```

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

```

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

```

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

`\rEnableOnTheFly`

```

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

```

`\rEnableOnTheFly`

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

```

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

```

```

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

```

sxtrstarflywarn

```

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

```

rEnableOnTheFly

```

1438 \newcommand*\@GlsXtrEnableOnTheFly{%

```

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

These definitions are all assigned the category given by:

\glxtrcat

```

1439 \newcommand*\glxtrcat{general}

```

\glxtr

```

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

```

\@glxtr

```

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

```

```

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

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

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

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

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

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

\@Glsxtrpl

```



```

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

```

\GlsXtrWarning

```

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

```

Disable commands after the glossary:

```

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

```

abledflycommand

```

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

```

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

```

1527 \let\GlsXtrEnableOnTheFly\relax
1528 }
1529 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

`\current@style` Initialise the current style to the default style.

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

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

`\setglossarystyle`

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

Only set the current style if it exists.

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

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

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

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

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

Similarly for `\glstdescwidth`:

\glsdescwidth

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

and for \glspagelistwidth:

\glspagelistwidth

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

aryentrynumbers Has the nonumberlist option been used?

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

```

```

1603     }{}%
1604   }%
1605 \fi

```

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

```

1606 \newcommand*\GlsXtrFormatLocationList}[1]{#1}

```

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

`\ePreLocationTag`

```

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

```

`\glstrpreloctag`

```

1627 \newcommand*\@glstrpreloctag{%
1628   \let\@glstr@org@delimN\delimN
1629   \let\@glstr@org@delimR\delimR
1630   \let\@glstr@org@glsgignore\glsgignore

```

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

```

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

```

```

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

```

glxtrpreloctag

```
1643 \newcommand*{\glxtrpreloctag}{}
```

@glxtr@pagetag

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

glxtr@pagetag

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

lsxtrpostloctag

```

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

```

lsxtrpostloctag

```
1653 \newcommand*{\glxtrpostloctag}{}%
```

lsxtr@preloctag

```

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

```

glxtr@doloctag

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

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

```

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

```

```

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

```

1.3.4 Entry Formatting, Hyperlinks and Indexing

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

```

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

```

`sxtrregularfont` Font used for regular entries.

```

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

```

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

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

```

1686 \renewcommand{\@gls@field@link}[4] []{%

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

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

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

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

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

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

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

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



```

1766 \glsxtrinitwrglossbeforetrue
1767 }%
1768 }

```

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

```

1769 \newif\ifglsxtrinitwrglossbefore
1770 \glsxtrinitwrglossbeforetrue

```

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

```

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

```

```

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

```

```

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

```

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

```

1781 \define@boolkey{glslink}[glsxtr@]{hyperoutside}[true]{}
1782 \glsxtr@hyperoutsidettrue

```

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

```

1783 \newcommand*{\glsxtrinithyperoutside}{%
1784 \glsifattribute{glslabel}{hyperoutside}{false}%
1785 {%
1786 \glsxtr@hyperoutsidetfalse
1787 }%
1788 {%
1789 \glsxtr@hyperoutsidettrue
1790 }%
1791 }

```

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

```

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

```

Initialise thevalue and theHvalue (v1.19).

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

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

```
1803 \glxstrinitwrgloss
```

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

```
1804 \glxstrinithyperoutside
```

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

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

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

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

Check textformat attribute (new to v1.21).

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

Do write if it should occur before the link text:

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

Do the link text:

```

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

```

Do write if it should occur after the link text:

```

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

```

As the original definition:

```

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

```

```

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

```

```

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

```

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

```

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

```

```

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

```

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

```

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

```

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

assignfieldfont

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

ecknohyperfirst

```

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

```

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

```

1929 \def\@glsfirst@#1#2[#3]{%
1930   \glstrassignfieldfont{#2}%
1931   Ensure that \glsfirst honours the nohyperfirst attribute.
1932   \@gls@field@link
1933   [\let\glstrifwasfirstuse\@firstoftwo
1934   \glstrchecknohyperfirst{#2}%
1935   ]{#1}{#2}%
1936   {\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1937 }

```

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

```

1937 \def\@Glsfirst@#1#2[#3]{%
1938   \glstrassignfieldfont{#2}%
1939   Ensure that \Glsfirst honours the nohyperfirst attribute.
1940   \@gls@field@link
1941   [\let\glstrifwasfirstuse\@firstoftwo
1942   \let\glscapscase\@secondofthree
1943   \glstrchecknohyperfirst{#2}%
1944   ]%
1945   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1946 }

```

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

```

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

```

Ensure that \GLSfirst honours the nohyperfirst attribute.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```

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

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

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

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

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

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

```


\@glssymbol@

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

\@Glsymbol@ First letter uppercase version.

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

\@GLSsymbol@ All uppercase version.

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

lssymbolplural@ No case-changing version.

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

lssymbolplural@ First letter uppercase version.

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

LSsymbolplural@ All uppercase version.

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

\@Glsuseri@ First letter uppercase version.

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

\@GLSuseri@ All uppercase version.

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

\@Glsuserii@ First letter uppercase version.

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

\@GLSuserii@ All uppercase version.

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

\@Glsuseriii@ First letter uppercase version.

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

\@GLSuseriii@ All uppercase version.

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

\@Glsuseriv@ First letter uppercase version.

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

```

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

```

\@GLSuseriv@ All uppercase version.

```

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

```

\@Glsuserv@ First letter uppercase version.

```

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

```

\@GLSuserv@ All uppercase version.

```

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

```

\@Glsuservi@ First letter uppercase version.

```

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

```

\@GLSuservi@ All uppercase version.

```

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

```

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

\@acrshort No case change.

```

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

```

```

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

```

\@Acrshort First letter uppercase.

```

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

```

\@ACRshort All uppercase.

```

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

```

\@acrshortpl No case change.

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

\@Acrshortpl First letter uppercase.

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

\@ACRshortpl All uppercase.

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

2254 }

\@acrlong No case change.

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

\@Acrlong First letter uppercase.

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

\@ACRlong All uppercase.

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

```

2297 }%
2298 \glspostlinkhook
2299 }

```

\@acrlongpl No case change.

```

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

```

\@Acrlongpl First letter uppercase.

```

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

```

\@ACRlongpl All uppercase.

```

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

```

```

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

```

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

\@glsaddkey

```

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

```

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

```

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

```

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

```

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

```



```

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

```

Finally the all caps version (modified):

```

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

```

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

```

2424 \providecommand*\@gls@link@nocheckfirsthyper{}

```

checkfirsthyper Modify check to determine if the hyperlink should be automatically suppressed, but save the

original in case the acronyms are restored.

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

\ifglsused isn't useful in the post link hook as it's already been unset by then, so define a command that can be used in the post link hook. Since \@gls@link@checkfirsthyper is only used by commands like \gls but not by other commands, this seems the best place to put it.

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

Store the category label for convenience.

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

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

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

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

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

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

lt@glslink@opts

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

```

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

```

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

```

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

```

setaliasnoindex

```

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

```

xtrindexaliased

```

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

```

xtrindexaliased

```

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

```

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

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

tDefaultGlsOpts Set the default options for \glslink etc.

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

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

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

\glswriteentry Redefine to test for indexonlyfirst category attribute.

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

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

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

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

Similarly for the “noidx” version:

s@noidxglossary

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

xtr@do@@wrindex

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

owrglossaryhook Allow user to hook into indexing code. (Always used by \glsadd. Used by \gls when indexing, which may or may not occur depending on the indexing settings.)

```
2531 \newcommand*{\glsxtrdowrglossaryhook}[1]{}
```

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

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

alt@gls@hyp@opt User version

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

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

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

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

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

rSetAltModifier

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

org@dohyperlink

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

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

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

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

Scope:

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

`\glsdohyperlink` Unpleasant complications can occur if the text or first key etc contains `\gls`, particularly if there are hyperlinks. To get around this problem, patch `\glsdohyperlink` so that it temporarily makes `\gls` behave like `\glstext[hyper=false,noindex]`. (This will be overridden if the user explicitly cancels either of those options in the optional argument of `\gls` or using the plus version.) This also patches the short form commands like `\acrshort` and `\glsxtrshort` to use `\glsentryshort` and, similarly, the long form commands like `\acrlong` and `\glsxtrlong` to use `\glsentrylong`. Added attribute check.

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

```

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

```

Redirect link to the alias target.

```

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

```

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

```

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

```

`\glsdisablehyper` Redefine to set `\glslabel` (to allow it to be picked up by `\glsdohyperlink`). Also made it robust and added grouping to localise the definition of `\glslabel`. The original internal command `@glo@label` could probably be simply replaced with `\glslabel`, but it's retained in case its removal causes unexpected problems.

```

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

```

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

```

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

```

```

2625 \let\@glstarget\@secondoftwo
2626 }

```

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

```

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

```

`\glsdonohyperlink` This command was only introduced in glossaries v4.20, so it may not be defined (therefore use `\def`). For older glossaries versions, this won't be used if `hyperref` hasn't been loaded, which means the indexing will still take place. The generated text is scoped.

```

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

```

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

```

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

```

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

```

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

```



```

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

```

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

@glxstrp@text@

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

@Glsxtrp@text@

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

@GLSxtrp@text@

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

lsxtrp@plural@

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

lsxtrp@plural@

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

LSxtrp@plural@

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

glxstrp@short@

```

2680 \def\@glxstrp@short@#1#2[#3]{%
2681 {%
2682   \glissetabbrvfmt{\glscategory{#2}}%
2683   \glsabbrvfont{\glsentryshort{#2}}#3%
2684 }%
2685 }

```

Glsxtrp@short@

```

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

```

GLSxtr@p@short@

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

sxtr@p@shortpl@

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

Sxtr@p@shortpl@

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

Sxtr@p@shortpl@

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

@glxtr@p@long@

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

@Glsxtr@p@long@

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

@GLSxtr@p@long@

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

lsxtr@p@longpl@

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

lsxtr@p@longpl@

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

```

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

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

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

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

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

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

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

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

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

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

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

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

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

Commands to minimise conflict.

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

```

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

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

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

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

`\@@glsxtrp`

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

Add scope.

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

`\@glsxtrp`

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

`\@Glsxtrp`

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

```

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

```

\@GLSxtrp

```

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

```

\glxtr@entry@p

```

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

```

\glxtrp Not robust as it needs to expand somewhat.

```

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

```

```

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

```

Provide short synonyms for the most common option.

`\glsp`

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

`\glsp`

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

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

```

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

```

```

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

```

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

```

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

```

```

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

```


2958 }

1.3.5 Entry Counting

The entry counting mechanism from glossaries is adjusted here to work with category attributes. Provide a convenient command to enable entry counting, set the entrycount attribute for given categories and redefine \gls etc to use \cgl's instead.

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

\@glsunset Global unset.

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

glsxtrpostunset

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

\@glslocalunset Local unset.

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

glsxtrpostlocalunset

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

\@glsreset Global reset.

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

glsxtrpostreset

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

\@glslocalreset Local reset.

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

glsxtrpostlocalreset

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

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

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

Enable entry counting:

```
2980 \glsenableentrycount
```

Redefine \gls etc:

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

Set the entrycount attribute:

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

In case this command is used again:

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

ycountunsetattr

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

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

nableentrycount

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

Enable new fields:

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

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

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

New commands to access new fields:

```

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

```

Adjust post unset and reset:

```

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

```

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

```

3042 \let\@cgl@s\@cgl@s
3043 \let\@cgl spl\@cgl spl
3044 \let\@cGls\@cGls
3045 \let\@cGlspl\@cGlspl
3046 \let\@cGLS\@cGLS
3047 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

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

```

```

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

```

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

```

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

```

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

```

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

```

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

\@@cgl's@

```

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

\@@cgl'spl@

```

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

\@@cGl's@

```

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

\@@cGl'spl@

```

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

\@@cGLS@

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

\@@cGLSpl@

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

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

\@cgl's@

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

\@cGL's@

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

\@cgl'spl@

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

\@cGL'spl@

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

Add all upper case versions not provided by glossaries.

\cGLS

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

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

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

```

\@cGLS@
3163 \def\@cGLS@#1#2[#3]{\@GLS@{#1}{#2}[#3]}

\cGLSformat  Format used by \cGLS if entry only used once on previous run. The first argument is the label,
              the second argument is the insert text.
3164 \newcommand*\cGLSformat[2]{%
3165   \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
3166 }

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

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

\@cGLSpl@
3171 \def\@cGLSpl@#1#2[#3]{\@GLSpl@{#1}{#2}[#3]}

\cGLSplformat  Format used by \cGLSpl if entry only used once on previous run. The first argument is the
               label, the second argument is the insert text.
3172 \newcommand*\cGLSplformat[2]{%
3173   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
3174 }

      Modify the trigger formats to check for the regular attribute.

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

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

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

```

\cGlsplformat

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

New code similar to above for unit counting.

defunitcounters

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

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

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

@addunitcounter

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

r@ifunitcounter

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

urrentunitcount

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


previousunitcount

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

t@currunitcount

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

t@currunitcount

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

r@currunitcount

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

r@prevunitcount

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

entryunitcount

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

New commands to access new fields:

```
3290 \newcommand*{\glsentrycurrcount}[1]{%
3291   \@glxstr@currunitcount{##1}{\glsggetattribute{##1}{unitcount}}.%
3292   \csuse{glxstr@theunit@\glsggetattribute{##1}{unitcount}}}%
3293 }%
3294 \newcommand*{\glsentryprevcount}[1]{%
3295   \@glxstr@prevunitcount{##1}{\glsggetattribute{##1}{unitcount}}.%
3296   \csuse{glxstr@theunit@\glsggetattribute{##1}{unitcount}}}%
3297 }%
```

Access total count:

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

Access max value:

```

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

```

Adjust post unset and reset:

```

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

```

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

```

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

```

```

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

```

Write information to the aux file.

```

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

```

entry@unitcount

```

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

```

ryunitcounts@do

```

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

```

entryunitcounts

```

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

```

```

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

```

`tryUnitCounting` The first argument is the list of categories, the second argument is the value of the entrycount attribute and the third is the counter name.

```

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

```

Enable entry counting:

```

3412   \glsenableentryunitcount

```

Redefine `\gls` etc:

```

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

```

Set the entrycount attribute:

```

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

```

In case this command is used again:

```

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

```

`tcountunsetattr`

```

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

```

1.3.6 Acronym Modifications

It's more consistent to use the abbreviation code for acronyms, but make some adjustments to allow for continued use of the glossaries package's custom acronym format. (For example, user may already have defined some acronym styles with `\newacronymstyle` which they would like to continue to use.) The original glossaries acronym code can be restored with `\RestoreAcronyms`, but adjust `\SetGenericNewAcronym` so that `\newacronym` adds the category.

`\SetGenericNewAcronym`

```

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

```

```

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

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbreviations, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```

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

```

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

```

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

```

```

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

```

Switch acronyms to abbreviations:

```

3510 \MakeAcronymsAbbreviations

```

RestoreAcronyms Restore acronyms to glossaries interface.

```

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

```

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

```

\glsacspace Allow the user to customise the maximum value.

```

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

```

\glsacspacemax Value used in the above.

```

3530 \newcommand*{\glsacspacemax}{3em}

```

1.3.7 Indexing and Displaying Glossaries

From time-to-time users ask if they can have one glossary sorted normally and another sorted by definition or usage. With the base glossaries package this can only be achieved with the “noidx” commands (Option 1). This is an attempt to mix and match.

First we need a list of the glossaries that require **makeindex/xindy**.

r@reg@glosslist

```

3531 \newcommand*{\@glstr@reg@glosslist}{}

```


Save the original definition of `\makeglossaries`:

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

Redefine `\makeglossaries` to take an optional argument. This should be empty for the usual behaviour (all glossaries need processing with an indexing application) or a comma-separated list of glossary labels indicating those glossaries that should be processed with an indexing application. The optional argument version shouldn't be used with `record`.

`\makeglossaries`

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

Iterate through each supplied glossary type and activate it.

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

New glossaries must be created before `\makeglossaries`:

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

Any subsequent instances of this command should have no effect

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

```

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

```

```

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

```

Adjust display number list to check for type:

```

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

```

Adjust entry list:

```

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

```

Adjust number list loop

```

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

```

Only sanitize sort for normal indexing glossaries.

```

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

```

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

```

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

```

Only activate sort key for glossaries that aren't listed in #1 (glossary label is stored in `\@glo@type` but this defaults to `\glsdefaulttype` so some expansion is required).

```

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

```

Need to extract just the type value.

```

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

```

Check automake setting:

```

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

```

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

```

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

```

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

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

```

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

```

Add check here.

```

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

```

```

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

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

```

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

```

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

```

lsxtrnopostpunc

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

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

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

stpunc@postdesc

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

restore@postpunc

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

restorepostpunc Does nothing outside of glossary.

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

\@printglossary Redefine.

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

Add a key that switches off the entry targets:

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

```

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

```

hypernameprefix

```

3767 \newcommand{\@glxtrhypernameprefix}{}

```

New to v1.20:

```

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

```

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

```

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

```

@makeglossaries For the benefit of makeglossaries

```

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

```

@glxtr@gettype Get just the type.

```

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

```

@assign@sortkey Assign the sort key.

```

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

```

Display number list for the regular version:

displaynumberlist

```

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

```

Display number list for the “noidx” version:

splaynumberlist

```

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

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

And for the number list loop:

@numberlistloop

```

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

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



```

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

```

Same for entry number list.

entrynumberlist

```

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

```

entrynumberlist

```

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

```

x@getgrouptitle Patch.

```

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

```

```

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

```

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

```

3882 \let\glsxtr@org@getgrouptitle\@gls@getgrouptitle

```

trgetgrouptitle Provide a user-level command to fetch the group title. The first argument is the group label. The second argument is a control sequence in which to store the title.

```

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

```

trsetgrouptitle Sets the title for the given group label.

```

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

```

alsetgrouptitle As above put only locally defines the title.

```

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

```

\glsnavigation Redefine to use new user-level command.

```

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

```

```

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

```

@noidx@glossary

```

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

```

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

```

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

```

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

```

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

```

noidxdisplayloc Patch to check for range formations.

```

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

```

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

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

isplaysingleloc Single location.

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

By default the range identifiers are simply ignored. A custom list loop handler can be defined by the user to test for ranges by checking the definition of `\glxtrlocrangefmt`.

displaystartloc Start of a location range.

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

trdisplayendloc End of a location range.

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

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

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

`sxtrlocrangefmt` Current range format. Empty if not in a range.

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

`ls@removespaces` Redefine to allow adjustments to location hyperlink.

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

`cationhyperlink`

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

`supphypernumber`

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

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

`@print@glossary`

```

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

```

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

`\oGlsWarningHead` Header message.

```

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

```

```

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

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

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

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

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

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

```

```

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

```

WarningMismatch Mismatching \makenoidxglossaries.

```

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

```

WarningBuildInfo Build advice.

```

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

```

oGlsWarningTail Final paragraph.

```

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

```


GlsWarningNoOut No out file created. Build advice.

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

glossarywarning

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

```

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

```

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

glstrresourcefile Since it's dangerous for an external application to create a file with a .tex extension, as from v1.11 this enforces a .glstex extension to avoid conflict.

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

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

```

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

```

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

```

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

```

glstrresourcecount

```
4230 \newcount\glstrresourcecount
```

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

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

```

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

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

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

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

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

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

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

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

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

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

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

```

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

```

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

```

If `fontspec` has been loaded, assume UTF-8. (The encoding can be changed with `\XeTeXinputencoding`, but I can't work out how to determine the current encoding.)

```

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

```

Prefix deferred until the beginning of the document in case it's redefined later in the preamble. This is picked up by `bib2gls` when the external option is used.

```

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

```

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

```

4282 \ifglautomake
4283 \IfFileExists{\jobname.aux}%
4284 {\immediate\write18{bib2gls \jobname}}}%

```

If `\makeglossaries` is also used, allow `makeindex/xindy` to also be run, otherwise disable the error message about requiring `\makeglossaries` with `automake=true`.

```

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

```

`do@automake@err`

```

4290 \newcommand*{\@gls@doautomake@err}{%
4291 \PackageError{glossaries}{You must use

```

```

4292 \string\makeglossaries\space with automake=true}
4293 {%
4294     Either remove the automake=true setting or
4295     add \string\makeglossaries\space to your document preamble.%
4296 }%
4297 }

```

Allow locations specific to a particular counter to be recorded.

\glxtr@record

```

4298 \newcommand*{\glxtr@record}[5]{}

```

r@counterrecord Aux file command.

```

4299 \newcommand*{\glxtr@counterrecord}[3]{%
4300 \glxtrfieldlistgadd{#1}{record.#2}{#3}%
4301 }

```

unterrecordhook Hook used by \@glxtr@dorecord.

```

4302 \newcommand*{\@glxtr@counterrecordhook}{}

```

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

```

4303 \newcommand*{\GlsXtrRecordCounter}[1]{%
4304 \@glxtr@recordcounter{#1}%
4305 }
4306 \@onlypreamble\GlsXtrRecordCounter

```

docounterrecord

```

4307 \newcommand*{\@glxtr@docounterrecord}[1]{%
4308 \protected@write\@auxout{}\string\glxtr@counterrecord
4309 {\@glx@label}{#1}{\csuse{the#1}}}%
4310 }

```

lsxtrglossentry Users may prefer to have entries displayed throughout the document rather than gathered together in a list. This command emulates the way \glossentry behaves (without the style formatting commands like \item). This needs to define \currentglossary to the current glossary type (normally set at the start of \@printglossary) and needs to define \glscurrententrylabel to the entry's label (normally set before \glossentry and \subglossentry). This needs some protection in case it's used in a section heading.

```

4311 \newcommand*{\glxtrglossentry}[1]{%
4312 \glxtrtitleorpdforheading
4313 {\@glxtrglossentry{#1}}%
4314 {\glsentryname{#1}}%
4315 {\glxtrheadname{#1}}%
4316 }

```

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

```

4317 \newrobustcmd*{\@glxtrglossentry}[1]{%

```

```

4318 \glstrtitleorpdforheading
4319 {%
4320   \glsoifexists{#1}%
4321   {%
4322     \begingroup
4323     \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4324     \edef\currentglossary{\glsetentrytype{\glscurrententrylabel}}%
4325     \ifglshasparent{#1}%
4326     {\glssubentryitem{#1}}%
4327     {\glsetentryitem{#1}}%
4328     \glstarget{#1}{\glossentryname{#1}}%
4329   \endgroup
4330   }%
4331 }%
4332 {\glsetentryname{#1}}%
4333 {\glstrheadname{#1}}%
4334 }

```

`\glossentryother` As `\glstrglossentry` but uses a different field. First argument is command to use in the header. The second argument is the entry's label. The third argument is the internal field label. This needs to be expandable in case it occurs in a sectioning command so it can't have an optional argument.

```

4335 \newcommand*{\glstrglossentryother}[3]{%
4336   \ifstrempy{#1}%
4337   {%
4338     \ifcsdef{glstrhead#3}%
4339     {%
4340       \glstrtitleorpdforheading
4341       {\@glstrglossentryother{#2}{#3}{#1}}%
4342       {\@gls@entry@field{#2}{#3}}%
4343       {\csuse{glstrhead#3}{#2}}%
4344     }%
4345     {%
4346       \glstrtitleorpdforheading
4347       {\@glstrglossentryother{#2}{#3}{#1}}%
4348       {\@gls@entry@field{#2}{#3}}%
4349       {\@gls@entry@field{NoCaseChange{#2}}{#3}}%
4350     }%
4351   }%
4352   {%
4353     \glstrtitleorpdforheading
4354     {\@glstrglossentryother{#2}{#3}{#1}}%
4355     {\@gls@entry@field{#2}{#3}}%
4356     {#1}%
4357   }%
4358 }

```

`\glossentryother` As `\@glstrglossentry` but uses a different field.

```

4359 \newrobustcmd*{\@glstrglossentryother}[3]{%

```

```

4360 \glstrtitleorpdforheading
4361 {%
4362   \glsdoifexists{#1}%
4363   {%
4364     \beginngroup
4365     \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4366     \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4367     \ifglshasparent{#1}%
4368     {\glssubentryitem{#1}}%
4369     {\glsentryitem{#1}}%
4370     \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4371   \endgroup
4372   }%
4373 }%
4374 {\@gls@entry@field{#1}{#2}}%
4375 {#3}%
4376 }

```

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

```

4377 \newcommand*{\printunsrtglossary}{%
4378   \@ifstar\s@printunsrtglossary\@printunsrtglossary
4379 }

```

`\printunsrtglossary` Unstarred version.

```

4380 \newcommand*{\@printunsrtglossary}[1][ ]{%
4381   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4382 }

```

`\printunsrtglossary` Starred version.

```

4383 \newcommand*{\s@printunsrtglossary}[2][ ]{%
4384   \beginngroup
4385     #2%
4386     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4387   \endgroup
4388 }

```

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

```

4389 \newcommand*{\printunsrtglossaries}{%
4390   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4391 }

```

`\@unsrt@glossary`

```

4392 \newcommand*{\@print@unsrt@glossary}{%
4393   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4394   \glossarypreamble

```

check for empty list

```

4395 \glxtrifemptyglossary{\@glo@type}%
4396 {%
4397   \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4398 }%
4399 {%

4400   \key@ifundefined{glossentry}{group}%
4401   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4402   {\let\@gls@getgrouptitle\@glxtr@unsrt@getgrouptitle}%
4403   \def\@gls@currentlettergroup{}}%

```

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

```

4404 \def\@glxtr@doglossary{%
4405   \begin{theglossary}%
4406   \glossaryheader
4407   \glsresetentrylist
4408 }%
4409 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4410   :\expandafter=\csname glist@\@glo@type\endcsname\do{%
4411   \ifdefempty{\glscurrententrylabel}
4412   }%
4413   {%

```

Provide a hook (for example to measure width).

```

4414   \let\glxtr@process\@firstofone
4415   \let\printunsrtglossaryskipentry
4416       \@glxtr@printunsrtglossaryskipentry
4417   \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

4418   \glxtr@process
4419   {%
4420     \ifglshasparent{\glscurrententrylabel}{}%
4421     {%
4422       \@glxtr@checkgroup\glscurrententrylabel
4423       \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
4424         {\@glxtr@groupheading}%
4425     }%
4426     \eappto\@glxtr@doglossary{%
4427       \noexpand\@printunsrtglossary@handler{\glscurrententrylabel}}%
4428   }%
4429 }%
4430 }%
4431 \appto\@glxtr@doglossary{\end{theglossary}}%
4432 \printunsrtglossarypredoglossary
4433 \@glxtr@doglossary
4434 }%
4435 \glossarypostamble
4436 }

```


entryprocesshook

```
4437 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}
```

entryprocesshook

```
4438 \newcommand*{\printunsrtglossaryskipentry}{%
4439   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4440 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4441 }
```

entryprocesshook

```
4442 \newcommand*{\@glxtr@printunsrtglossaryskipentry}{%
4443   \let\glxtr@process\@gobble
4444 }
```

entrypredoglossary

```
4445 \newcommand*{\printunsrtglossarypredoglossary}{}%
```

glossary@handler

```
4446 \newcommand{\@printunsrtglossary@handler}[1]{%
4447   \xdef\glscurrententrylabel{#1}%
4448   \printunsrtglossaryhandler\glscurrententrylabel
4449 }
```

glossaryhandler

```
4450 \newcommand{\printunsrtglossaryhandler}[1]{%
4451   \glxtrunsrtdo{#1}%
4452 }
```

triflabelinlist Might be useful for the handler to check if an entry label or category label is contained in a list, so provide a user-level version of \@glx@ifinlist which ensures the label and list are fully expanded.

```
4453 \newrobustcmd*{\glxtriflabelinlist}[4]{%
4454   \protected@edef\@glxtr@doiflabelinlist{\noexpand\@glx@ifinlist{#1}{#2}}%
4455   \@glxtr@doiflabelinlist{#3}{#4}%
4456 }
```

unsrtglossaryunit

```
4457 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
4458   \s@printunsrtglossary[type=\gldefaulttype,#1]{%
4459     \printunsrtglossaryunitsetup{#2}%
4460   }%
4461 }
```

glossaryunitsetup

```
4462 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4463   \renewcommand{\printunsrtglossaryhandler}[1]{%
4464     \glxtrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
```

```

4465 {\glxtrunsrtdo{##1}}%
4466 {}%
4467 }%

```

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

```

4468 \ifcsundef{theH#1}%
4469 {%
4470 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4471 }%
4472 {%
4473 \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4474 }%
4475 \renewcommand*{\glossarysection}[2][{}]{%
4476 \appto\glossarypostamble{\glspar\medskip\glspar}%
4477 }

```

srtglossaryunit

```

4478 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
4479 \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4480 requires the record=only or record=alsoindex package option}{}%
4481 }

```

t@getgrouptitle

```

4482 \newrobustcmd*{\@glxtr@unsrt@getgrouptitle}[2]{%
4483 \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4484 \@onelevel@sanitize\@glxtr@titlelabel
4485 \ifcsdef{\@glxtr@titlelabel}
4486 {\letcs{#2}{\@glxtr@titlelabel}}%
4487 {\def#2{#1}}%
4488 }

```

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

```

4489 \newcommand{\glxtrunsrtdo}{\@glxtr@noidx@do}

```

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

```

4490 \newcommand*{\lsxtrgroupfield}{group}

```

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

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

```

4491 \newcommand*{\@glsxtr@checkgroup}[1]{%
4492   \def\@glsxtr@groupheading{}%
4493   \key@ifundefined{glossentry}{group}%
4494   {%
4495     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4496     \expandafter\glo@grabfirst\@gls@sort{}\@nil
4497   }%
4498   {%
4499     \protected@edef\@glo@thislettergrp{%
4500       \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
4501     }%
4502     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4503     {}%
4504     {%
4505       \ifdefempty{\@gls@currentlettergroup}{}%
4506       {\def\@glsxtr@groupheading{\gls@groupskip}}%
4507       \eappto\@glsxtr@groupheading{%
4508         \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
4509       }%
4510     }%
4511     \let\@gls@currentlettergroup\@glo@thislettergrp
4512 }

```

glsxtr@noidx@do Minor modification of \@gls@noidx@do to check for location field if present, but also need to check for the group field.

```

4513 \newcommand{\@glsxtr@noidx@do}[1]{%
4514   \ifglsentryexists{#1}%
4515   {%
4516     \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4517     \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
4518     \ifgls@hasparent{#1}%
4519     {%
4520       \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
4521       \ifdefvoid{\@gls@location}%
4522       {%
4523         \ifdefvoid{\@gls@loclist}%
4524         {%
4525           \subglossentry{\gls@level}{#1}{}%
4526         }%
4527         {%
4528           \subglossentry{\gls@level}{#1}%
4529           {%
4530             \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4531           }%
4532         }%
4533       }%
4534     }%
4535     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4536   }%

```

```

4537 }%
4538 {%

4539 \ifdefvoid{\@gls@location}%
4540 {%
4541 \ifdefvoid{\@gls@loclist}
4542 {%
4543 \glossentry{#1}{}%
4544 }%
4545 {%
4546 \glossentry{#1}%
4547 {%
4548 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4549 }%
4550 }%
4551 }%
4552 {%
4553 \glossentry{#1}%
4554 {%
4555 \glossaryentrynumbers{\@gls@location}%
4556 }%
4557 }%
4558 }%
4559 }%
4560 {}%
4561 }

```

1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4562 \newcommand*{\glshex}{\string\u}
```

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

```
4563 \newcommand*{\glxtrresourceinit}{}%
```

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

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

`\glxtrnewgls`

```
4564 \newcount\@glxtrnewgls@inner
```

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

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

```

4565 \newcommand*{\@glstrnewgls}[4]{%
4566   \ifdef{#3}%
4567   {%
4568     \PackageError{glossaries-extra}{Command \string#3\space already
4569 defined}{}%
4570   }%
4571   {%
4572     \ifcsdef{@#4like@#2}%
4573     {%
4574       \advance\@glstrnewgls@inner by \@ne
4575       \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
4576     }%
4577     {\def\@glstrnewgls@innercsname{@#4like@#2}}%
4578     \expandafter\newrobustcmd\expandafter*\expandafter
4579       #3\expandafter{\expandafter\@glshyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
4580     \ifstrempy{#1}%
4581     {%
4582       \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
4583         \new@ifnextchar[%
4584           {\csname @#4@\endcsname{##1}{#2##2}}%
4585           {\csname @#4@\endcsname{##1}{#2##2}[]}%
4586         }%
4587       }%
4588     }%
4589     \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
4590       \new@ifnextchar[%
4591         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4592         {\csname @#4@\endcsname{#1,##1}{#2##2}[]}%
4593       }%
4594     }%
4595   }%
4596 }

```

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

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

```

4597 \newrobustcmd*{\glstrnewgls}[3][]{%
4598   \@glstrnewgls{#1}{#2}{#3}{gls}%
4599 }

```

`lsxtrnewglslike` Provide a way to conveniently define commands that behave like `\gls`, `\glspl`, `\Gls` and

`\GLspl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
4600 \newrobustcmd*{\glxtrnewglslike}[6] [] {%
4601   \@glxtrnewgls{#1}{#2}{#3}{gls}%
4602   \@glxtrnewgls{#1}{#2}{#4}{glspl}%
4603   \@glxtrnewgls{#1}{#2}{#5}{Gls}%
4604   \@glxtrnewgls{#1}{#2}{#6}{Glspl}%
4605 }
```

`\glxtrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSpl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```
4606 \newrobustcmd*{\glxtrnewGLSlike}[4] [] {%
4607   \@glxtrnewgls{#1}{#2}{#3}{GLS}%
4608   \@glxtrnewgls{#1}{#2}{#4}{GLSpl}%
4609 }
```

`\glxtrnewrgls` As `\glxtrnewgls` but for `\rgls`.

```
4610 \newrobustcmd*{\glxtrnewrgls}[3] [] {%
4611   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4612 }
```

`\glxtrnewrglslike` As `\glxtrnewglslike` but for `\rgls` etc.

```
4613 \newrobustcmd*{\glxtrnewrglslike}[6] [] {%
4614   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4615   \@glxtrnewgls{#1}{#2}{#4}{rglspl}%
4616   \@glxtrnewgls{#1}{#2}{#5}{rGls}%
4617   \@glxtrnewgls{#1}{#2}{#6}{rGlspl}%
4618 }
```

`\glxtrnewGLSlike` As `\glxtrnewGLSlike` but for `\rGLS` etc.

```
4619 \newrobustcmd*{\glxtrnewrGLSlike}[4] [] {%
4620   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4621   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4622 }
```

Provide easy access to record count fields.

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

```
4623 \newcommand*{\GlsXtrTotalRecordCount}[1] {%
4624   \ifcsdef{glo@\glsdetoklabel{#1}@recordcount}%
4625   {\csname glo@\glsdetoklabel{#1}@recordcount\endcsname}%
4626   {0}%
4627 }
```

`\sXtrRecordCount` Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
4628 \newcommand*{\GlsXtrRecordCount}[2] {%
4629   \ifcsdef{glo@\glsdetoklabel{#1}@recordcount.#2}%
4630   {\csname glo@\glsdetoklabel{#1}@recordcount.#2\endcsname}%
4631   {0}%
4632 }
```

```

4630 {\csname glo@\glsdetoklabel{#1}@recordcount.#2\endcsname}%
4631 {0}%
4632 }

```

tionRecordCount Access record count for a particular counter and location. The first argument is the label. The second argument is the counter name. The third argument is the location. This command shouldn't be used if the location doesn't fully expand unless `\glsxtrdetoklocation` can be set to something sensible.

```

4633 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
4634 \ifcsdef{glo@\glsdetoklabel{#1}@recordcount.#2.\glsxtrdetoklocation{#3}}%
4635 {\csname glo@\glsdetoklabel{#1}@recordcount.#2.\glsxtrdetoklocation{#3}\endcsname}%
4636 {0}%
4637 }

```

trdetoklocation

```

4638 \newcommand*{\glsxtrdetoklocation}[1]{#1}

```

ablerecordcount

```

4639 \newcommand*{\glsxtrenablerecordcount}{%
4640 \renewcommand*{\gls}{\rgls}%
4641 \renewcommand*{\Gls}{\rGls}%
4642 \renewcommand*{\glspl}{\rglspl}%
4643 \renewcommand*{\Glspl}{\rGlspl}%
4644 \renewcommand*{\GLS}{\rGLS}%
4645 \renewcommand*{\GLSpl}{\rGLSpl}%
4646 }

```

ordtriggervalue The value used by the record trigger test. The argument is the entry's label.

```

4647 \newcommand*{\glsxtrrecordtriggervalue}[1]{%
4648 \GlsXtrTotalRecordCount{#1}%
4649 }

```

dCountAttribute

```

4650 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4651 \@for\@glsxtr@cat:=#1\do
4652 {%
4653 \ifdefempty{\@glsxtr@cat}{}%
4654 {%
4655 \glssetcategoryattribute{\@glsxtr@cat}{recordcount}{#2}%
4656 }%
4657 }%
4658 }

```

rifrecordtrigger `\glsxtrifrecordtrigger{<label>}{<trigger format>}{<normal>}`

```

4659 \newcommand*{\glxstrifrecordtrigger}[3]{%
4660   \glshasattribute{#1}{recordcount}%
4661   {%
4662     \ifnum\glxstrrecordtriggervalue{#1}>\glsetattribute{#1}{recordcount}\relax
4663     #3%
4664   \else
4665     #2%
4666   \fi
4667 }%
4668 {#3}%
4669 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

4670 \newcommand*{\@glxstr@rglstrigger@record}[3]{%
4671   \edef\glslabel{\glsetoklabel{#2}}%
4672   \let\@gls@link@label\glslabel
4673   \def\@glxstr@thevalue{%
4674     \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4675     \def\@glsnumberformat{glstriggerrecordformat}%
4676     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4677     \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4678     \def\@glxstr@thevalue{%
4679       \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4680       \glxstrinitwrgloss
4681       \setkeys{glslink}{#1}%
4682       \glslinkpostsetkeys
4683       \ifdefempty{\@glxstr@thevalue}%
4684       {%
4685         \@gls@saveentrycounter
4686       }%
4687       {%
4688         \let\thegl Sentrycounter\@glxstr@thevalue
4689         \def\theHgl Sentrycounter{\@glxstr@theHvalue}%
4690       }%
4691       \ifglxstrinitwrglossbefore
4692         \@do@wrglossary{#2}%
4693       \fi
4694       #3%
4695       \ifglxstrinitwrglossbefore
4696       \else
4697         \@do@wrglossary{#2}%
4698       \fi
4699       \ifKV@glslink@local
4700         \glslocalunset{#2}%
4701       \else
4702         \glsunset{#2}%
4703       \fi
4704 }

```


gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```
4705 \newcommand*{\glstriggerrecordformat}[1]{}
```

\rgls

```
4706 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}
```

\@rgls

```
4707 \newcommand*{\@rgls}[2][{}]{%
4708   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4709 }
```

\@rgls@

```
4710 \def\@rgls@#1#2[#3]{%
4711   \glstrifrecordtrigger{#2}%
4712   {%
4713     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4714   }%
4715   {%
4716     \@gls@{#1}{#2}[#3]%
4717   }%
4718 }%
```

\rglsp1

```
4719 \newrobustcmd*{\rglsp1}{\@gls@hyp@opt\@rglsp1}
```

\@rglsp1

```
4720 \newcommand*{\@rglsp1}[2][{}]{%
4721   \new@ifnextchar[{\@rglsp1@{#1}{#2}}{\@rglsp1@{#1}{#2}[]}%
4722 }
```

\@rglsp1@

```
4723 \def\@rglsp1@#1#2[#3]{%
4724   \glstrifrecordtrigger{#2}%
4725   {%
4726     \@glstr@rglstrigger@record{#1}{#2}{\rglsp1format{#2}{#3}}%
4727   }%
4728   {%
4729     \@glspl@{#1}{#2}[#3]%
4730   }%
4731 }%
```

\rGls

```
4732 \newrobustcmd*{\rGls}{\@gls@hyp@opt\@rGls}
```

\@rGls

```
4733 \newcommand*{\@rGls}[2][{}]{%
4734   \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4735 }
```

```

\@rGls@
4736 \def\@rGls@#1#2[#3]{%
4737   \glxtrifrecordtrigger{#2}%
4738   {%
4739     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4740   }%
4741   {%
4742     \@Gls@{#1}{#2}[#3]%
4743   }%
4744 }%

\rGlspl
4745 \newrobustcmd*{\rGlspl}{\@glshyp@opt\rGlspl}

\@rGlspl
4746 \newcommand*{\@rGlspl}[2][{}]{%
4747   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4748 }

\@rGlspl@
4749 \def\@rGlspl@#1#2[#3]{%
4750   \glxtrifrecordtrigger{#2}%
4751   {%
4752     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4753   }%
4754   {%
4755     \@Glspl@{#1}{#2}[#3]%
4756   }%
4757 }%

\rGLS
4758 \newrobustcmd*{\rGLS}{\@glshyp@opt\rGLS}

\@rGLS
4759 \newcommand*{\@rGLS}[2][{}]{%
4760   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}%
4761 }

\@rGLS@
4762 \def\@rGLS@#1#2[#3]{%
4763   \glxtrifrecordtrigger{#2}%
4764   {%
4765     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
4766   }%
4767   {%
4768     \@GLS@{#1}{#2}[#3]%
4769   }%
4770 }%

```

```
\rGLSpl
4771 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}
```

```
\@rGLSpl
4772 \newcommand*{\@rGLSpl}[2][\%
4773   \new@ifnextchar[\@rGLSpl@{#1}{#2}]{\@rGLSpl@{#1}{#2}[\%
4774 }
```

```
\@rGLSpl@
4775 \def\@rGLSpl@#1#2[#3]{\%
4776   \glstriferecordtrigger{#2}%
4777   {\%
4778     \@glstr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
4779   }%
4780   {\%
4781     \@GLSpl@{#1}{#2}[#3]%
4782   }%
4783 }%
```

```
\rglsformat
4784 \newcommand*{\rglsformat}[2]{\%
4785   \glsifregular{#1}
4786   {\glsentryfirst{#1}}%
4787   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4788 }
```

```
\rglsplformat
4789 \newcommand*{\rglsplformat}[2]{\%
4790   \glsifregular{#1}
4791   {\glsentryfirstplural{#1}}%
4792   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4793 }
```

```
\rGlsformat
4794 \newcommand*{\rGlsformat}[2]{\%
4795   \glsifregular{#1}
4796   {\Glsentryfirst{#1}}%
4797   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4798 }
```

```
\rGlsplformat
4799 \newcommand*{\rGlsplformat}[2]{\%
4800   \glsifregular{#1}
4801   {\Glsentryfirstplural{#1}}%
4802   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4803 }
```

`\rGLSformat`

```
4804 \newcommand*{\rGLSformat}[2]{%
4805   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSformat{#1}{#2}}%
4806 }
```

`\rGLSplformat`

```
4807 \newcommand*{\rGLSplformat}[2]{%
4808   \expandafter\mfirstucMakeUppercase\expandafter{\rGLSplformat{#1}{#2}}%
4809 }
```

1.4 Integration with glossaries-accsupp

Provide better integration with the glossaries-accsupp package. (Must be loaded before the main code of glossaries-extra either explicitly or through the accsupp package option.)

These commands have their definitions set according to whether or not glossaries-extra has been loaded.

```
4810 \@ifpackageloaded{glossaries-accsupp}
4811 {
```

Define (or redefine) commands to use the accessibility information.

`\glsaccessname` Display the name value (no link and no check for existence).

```
4812 \newcommand*{\glsaccessname}[1]{%
4813   \glsnameaccessdisplay
4814   {%
4815     \glsentryname{#1}%
4816   }%
4817   {#1}%
4818 }
```

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.

```
4819 \newcommand*{\Glsaccessname}[1]{%
4820   \glsnameaccessdisplay
4821   {%
4822     \Glsentryname{#1}%
4823   }%
4824   {#1}%
4825 }
```

`\GLSaccessname` Display the name value (no link and no check for existence) converted to upper case.

```
4826 \newcommand*{\GLSaccessname}[1]{%
4827   \glsnameaccessdisplay
4828   {%
4829     \mfirstucMakeUppercase{\glsentryname{#1}}%
4830   }%
4831   {#1}%
4832 }
```

`\glsaccesstext` Display the text value (no link and no check for existence).

```
4833 \newcommand*{\glsaccesstext}[1]{%
4834   \glstextaccessdisplay
4835   {%
4836     \glstrytext{#1}%
4837   }%
4838   {#1}%
4839 }
```

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.

```
4840 \newcommand*{\Glsaccesstext}[1]{%
4841   \glstextaccessdisplay
4842   {%
4843     \Glsstrytext{#1}%
4844   }%
4845   {#1}%
4846 }
```

`\GLSaccesstext` Display the text value (no link and no check for existence) converted to upper case.

```
4847 \newcommand*{\GLSaccesstext}[1]{%
4848   \glstextaccessdisplay
4849   {%
4850     \mfirstucMakeUppercase{\glstrytext{#1}}%
4851   }%
4852   {#1}%
4853 }
```

`\glsaccessplural` Display the plural value (no link and no check for existence).

```
4854 \newcommand*{\glsaccessplural}[1]{%
4855   \glspluralaccessdisplay
4856   {%
4857     \glstryplural{#1}%
4858   }%
4859   {#1}%
4860 }
```

`\Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```
4861 \newcommand*{\Glsaccessplural}[1]{%
4862   \glspluralaccessdisplay
4863   {%
4864     \Glsstryplural{#1}%
4865   }%
4866   {#1}%
4867 }
```

`\GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```

4868 \newcommand*{\GLSaccessplural}[1]{%
4869   \glspluralaccessdisplay
4870   {%
4871     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4872   }%
4873   {#1}%
4874 }

```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

4875 \newcommand*{\glsaccessfirst}[1]{%
4876   \glsfirstaccessdisplay
4877   {%
4878     \glsentryfirst{#1}%
4879   }%
4880   {#1}%
4881 }

```

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

4882 \newcommand*{\Glsaccessfirst}[1]{%
4883   \glsfirstaccessdisplay
4884   {%
4885     \Glsentryfirst{#1}%
4886   }%
4887   {#1}%
4888 }

```

`\GLSaccessfirst` Display the first value (no link and no check for existence) converted to upper case.

```

4889 \newcommand*{\GLSaccessfirst}[1]{%
4890   \glsfirstaccessdisplay
4891   {%
4892     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4893   }%
4894   {#1}%
4895 }

```

`\glsaccessfirstplural` Display the firstplural value (no link and no check for existence).

```

4896 \newcommand*{\glsaccessfirstplural}[1]{%
4897   \glsfirstpluralaccessdisplay
4898   {%
4899     \glsentryfirstplural{#1}%
4900   }%
4901   {#1}%
4902 }

```

`\Glsaccessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

4903 \newcommand*{\Glsaccessfirstplural}[1]{%

```

```

4904   \glsfirstpluralaccessdisplay
4905   {%
4906       \Glsentryfirstplural{#1}%
4907   }%
4908   {#1}%
4909   }

```

glsaccessfirstplural Display the firstplural value (no link and no check for existence) converted to upper case.

```

4910   \newcommand*{\Glsaccessfirstplural}[1]{%
4911       \glsfirstpluralaccessdisplay
4912       {%
4913           \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4914       }%
4915       {#1}%
4916   }

```

glsaccesssymbol Display the symbol value (no link and no check for existence).

```

4917   \newcommand*{\glsaccesssymbol}[1]{%
4918       \glssymbolaccessdisplay
4919       {%
4920           \glsentrysymbol{#1}%
4921       }%
4922       {#1}%
4923   }

```

Glsaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4924   \newcommand*{\Glsaccesssymbol}[1]{%
4925       \glssymbolaccessdisplay
4926       {%
4927           \Glsentrysymbol{#1}%
4928       }%
4929       {#1}%
4930   }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```

4931   \newcommand*{\GLSaccesssymbol}[1]{%
4932       \glssymbolaccessdisplay
4933       {%
4934           \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
4935       }%
4936       {#1}%
4937   }

```

glsaccesssymbolplural Display the symbolplural value (no link and no check for existence).

```

4938   \newcommand*{\glsaccesssymbolplural}[1]{%
4939       \glssymbolpluralaccessdisplay
4940       {%

```

```

4941     \glentrysymbolplural{#1}%
4942   }%
4943   {#1}%
4944 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4945 \newcommand*{\Glsaccesssymbolplural}[1]{%
4946   \glssymbolpluralaccessdisplay
4947   {%
4948     \glentrysymbolplural{#1}%
4949   }%
4950   {#1}%
4951 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) converted to upper case.

```

4952 \newcommand*{\GLSaccesssymbolplural}[1]{%
4953   \glssymbolpluralaccessdisplay
4954   {%
4955     \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
4956   }%
4957   {#1}%
4958 }

```

\glsaccessdesc Display the desc value (no link and no check for existence).

```

4959 \newcommand*{\glsaccessdesc}[1]{%
4960   \glsdescriptionaccessdisplay
4961   {%
4962     \glentrydesc{#1}%
4963   }%
4964   {#1}%
4965 }

```

\GLSaccessdesc Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

4966 \newcommand*{\GLSaccessdesc}[1]{%
4967   \glsdescriptionaccessdisplay
4968   {%
4969     \GLSentrydesc{#1}%
4970   }%
4971   {#1}%
4972 }

```

\GLSAccessdesc Display the desc value (no link and no check for existence) converted to upper case.

```

4973 \newcommand*{\GLSAccessdesc}[1]{%
4974   \glsdescriptionaccessdisplay
4975   {%
4976     \mfirstucMakeUppercase{\GLSentrydesc{#1}}%

```



```

4977     }%
4978     {#1}%
4979 }

```

`\glsaccessdescplural` Display the descplural value (no link and no check for existence).

```

4980 \newcommand*{\glsaccessdescplural}[1]{%
4981   \glsdescriptionpluralaccessdisplay
4982   {%
4983     \glsentrydescplural{#1}%
4984   }%
4985   {#1}%
4986 }

```

`\Glsaccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

```

4987 \newcommand*{\Glsaccessdescplural}[1]{%
4988   \glsdescriptionpluralaccessdisplay
4989   {%
4990     \Glsentrydescplural{#1}%
4991   }%
4992   {#1}%
4993 }

```

`\GLsaccessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

4994 \newcommand*{\GLsaccessdescplural}[1]{%
4995   \glsdescriptionpluralaccessdisplay
4996   {%
4997     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4998   }%
4999   {#1}%
5000 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5001 \newcommand*{\glsaccessshort}[1]{%
5002   \glsshortaccessdisplay
5003   {%
5004     \glsentryshort{#1}%
5005   }%
5006   {#1}%
5007 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5008 \newcommand*{\Glsaccessshort}[1]{%
5009   \glsshortaccessdisplay
5010   {%
5011     \Glsentryshort{#1}%
5012   }%

```

```

5013     {#1}%
5014 }

```

`\GLSaccessshort` Display the short value (no link and no check for existence) converted to upper case.

```

5015 \newcommand*{\GLSaccessshort}[1]{%
5016   \glsshortaccessdisplay
5017   {%
5018     \mfirstucMakeUppercase{\glsentryshort{#1}}%
5019   }%
5020   {#1}%
5021 }

```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

5022 \newcommand*{\lsaccessshortpl}[1]{%
5023   \glsshortpluralaccessdisplay
5024   {%
5025     \glsentryshortpl{#1}%
5026   }%
5027   {#1}%
5028 }

```

`lSaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5029 \newcommand*{\lSaccessshortpl}[1]{%
5030   \glsshortpluralaccessdisplay
5031   {%
5032     \lSentryshortpl{#1}%
5033   }%
5034   {#1}%
5035 }

```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence) converted to upper case.

```

5036 \newcommand*{\LSaccessshortpl}[1]{%
5037   \glsshortpluralaccessdisplay
5038   {%
5039     \mfirstucMakeUppercase{\lSentryshortpl{#1}}%
5040   }%
5041   {#1}%
5042 }

```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

5043 \newcommand*{\glsaccesslong}[1]{%
5044   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
5045 }

```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```

5046
5047 \newcommand*{\Glsaccesslong}[1]{%

```

```

5048   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
5049 }

```

`\GLSaccesslong` Display the long value (no link and no check for existence) converted to upper case.

```

5050   \newcommand*{\GLSaccesslong}[1]{%
5051     \glslongaccessdisplay
5052     {%
5053       \mfirstucMakeUppercase{\glsentrylong{#1}}%
5054     }%
5055     {#1}%
5056 }

```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5057   \newcommand*{\glsaccesslongpl}[1]{%
5058     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5059 }

```

`GLsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5060
5061   \newcommand*{\GLsaccesslongpl}[1]{%
5062     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5063 }

```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```

5064   \newcommand*{\GLSaccesslongpl}[1]{%
5065     \glslongpluralaccessdisplay
5066     {%
5067       \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5068     }%
5069     {#1}%
5070 }

```

End of if part

```

5071 }
5072 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

`\glsaccessname` Display the name value (no link and no check for existence).

```

5073   \newcommand*{\glsaccessname}[1]{\glsentryname{#1}}

```

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.

```

5074   \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

`\GLSAccessname` Display the name value (no link and no check for existence). converted to upper case.

```

5075   \newcommand*{\GLSAccessname}[1]{%
5076     \protect\mfirstucMakeUppercase{\glsentryname{#1}}%

```

`\glsaccesstext` Display the text value (no link and no check for existence).
5077 `\newcommand*{\glsaccesstext}[1]{\glsentrytext{#1}}`

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.
5078 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.
5079 `\newcommand*{\GLSaccesstext}[1]{%`
5080 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).
5081 `\newcommand*{\glsaccessplural}[1]{\glsentryplural{#1}}`

`Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.
5082 `\newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.
5083 `\newcommand*{\GLSaccessplural}[1]{%`
5084 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).
5085 `\newcommand*{\glsaccessfirst}[1]{\glsentryfirst{#1}}`

`\Glsaccessfirst` Display the first value (no link and no check for existence) with the first letter converted to upper case.
5086 `\newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.
5087 `\newcommand*{\GLSaccessfirst}[1]{%`
5088 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`glsaccessfirstplural` Display the firstplural value (no link and no check for existence).
5089 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`Glsaccessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.
5090 `\newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}`

`GLSaccessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
5091 `\newcommand*{\GLSaccessfirstplural}[1]{%`
5092 `\protect\mfirstucMakeUppercase{\glsentryfirstplural{#1}}}`

`\glsaccesssymbol` Display the symbol value (no link and no check for existence).
5093 `\newcommand*{\glsaccesssymbol}[1]{\glsentrysymbol{#1}}`

`\Glsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

5094 `\newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}`

`\GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

5095 `\newcommand*{\GLSaccesssymbol}[1]{%`
5096 `\protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}`

`\essymbolplural` Display the symbolplural value (no link and no check for existence).

5097 `\newcommand*{\glaccesssymbolplural}[1]{\glentrysymbolplural{#1}}`

`\essymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

5098 `\newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}`

`\essymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.

5099 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
5100 `\protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).

5101 `\newcommand*{\glsaccessdesc}[1]{\glentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.

5102 `\newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.

5103 `\newcommand*{\GLSaccessdesc}[1]{%`
5104 `\protect\mfirstucMakeUppercase{\glentrydesc{#1}}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence).

5105 `\newcommand*{\glaccessdescplural}[1]{\glentrydescplural{#1}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

5106 `\newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}`

`\ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.

5107 `\newcommand*{\GLSaccessdescplural}[1]{%`
5108 `\protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}`

`\glsaccessshort` Display the short form (no link and no check for existence).

5109 `\newcommand*{\glsaccessshort}[1]{\glentryshort{#1}}`

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```
5110 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}
```

`\GLSaccessshort` Display the short value (no link and no check for existence). converted to upper case.

```
5111 \newcommand*{\GLSaccessshort}[1]{%
5112 \protect\mfirstucMakeUppercase{\Glsentryshort{#1}}}
```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```
5113 \newcommand*{\lsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

`lSaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5114 \newcommand*{\lSaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.

```
5115 \newcommand*{\LSaccessshortpl}[1]{%
5116 \protect\mfirstucMakeUppercase{\Glsentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5117 \newcommand*{\glsaccesslong}[1]{\Glsentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5118 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

`\GLSaccesslong` Display the long value (no link and no check for existence). converted to upper case.

```
5119 \newcommand*{\GLSaccesslong}[1]{%
5120 \protect\mfirstucMakeUppercase{\Glsentrylong{#1}}}
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5121 \newcommand*{\glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5122 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.

```
5123 \newcommand*{\GLSaccesslongpl}[1]{%
5124 \protect\mfirstucMakeUppercase{\Glsentrylongpl{#1}}}
```

End of else part

```
5125 }
```

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5126 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5127 \newcommand{\glsifcategory}[4]{%
5128   \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5129 }
```

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
5130 \newcommand*{\glssetcategoryattribute}[3]{%
5131   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
5132 }
```

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5133 \newcommand*{\glsgetcategoryattribute}[2]{%
5134   \csuse{@glxtr@categoryattr@@#1@#2}%
5135 }
```

`categoryattribute` `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5136 \newcommand*{\glshascategoryattribute}[4]{%
5137   \ifcvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
5138 }
```

`\glssetattribute` `\glssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5139 \newcommand*{\glssetattribute}[3]{%
```

```

5140 \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5141 }

```

`\glsggetattribute` `\glsggetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```

5142 \newcommand*{\glsggetattribute}[2]{%
5143   \glsssetcategoryattribute{\glscategory{#1}}{#2}%
5144 }

```

`\glshasattribute` `\glshasattribute{<entry label>}{<attribute-label>}{<true>}{<false>}`

Short cut to test if the given attribute has been set where the category label is obtained from the entry information.

```

5145 \newcommand*{\glshasattribute}[4]{%
5146   \ifglssentryexists{#1}%
5147   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5148   {#4}%
5149 }

```

`categoryattribute` `\glssifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```

5150 \newcommand{\glssifcategoryattribute}[5]{%
5151   \ifglssundef{@glssxtr@categoryattr@#1@#2}%
5152   {#5}%
5153   {\ifcsstring{@glssxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5154 }

```

`\glssifattribute` `\glssifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

Short cut to determine if the given entry has a category with the given attribute set.

```

5155 \newcommand{\glssifattribute}[5]{%
5156   \ifglssentryexists{#1}%
5157   {\glssifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5158   {#5}%
5159 }

```


Set attributes for the default general category:

```
5160 \glsssetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5161 \glsssetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```
5162 \newcommand*{\glsssetregularcategory}[1]{%  
5163   \glsssetcategoryattribute{#1}{regular}{true}}%  
5164 }
```

`ifregularcategory` `\glssifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5165 \newcommand{\glssifregularcategory}[3]{%  
5166   \glssifcategoryattribute{#1}{regular}{true}{#2}{#3}}%  
5167 }
```

`ifnotregularcategory` `\glssifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5168 \newcommand{\glssifnotregularcategory}[3]{%  
5169   \glssifcategoryattribute{#1}{regular}{false}{#2}{#3}}%  
5170 }
```

`\glssifregular` `\glssifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5171 \newcommand{\glssifregular}[3]{%  
5172   \glssifregularcategory{\glsscategor{#1}}{#2}{#3}}%  
5173 }
```

`\glssifnotregular` `\glssifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5174 \newcommand{\glssifnotregular}[3]{%  
5175   \glssifnotregularcategory{\glsscategor{#1}}{#2}{#3}}%  
5176 }
```

oreachincategory

```
\glsforeachincategory[⟨glossary labels⟩]{⟨category-label⟩}
{⟨glossary-cs⟩}{⟨label-cs⟩}{⟨body⟩}
```

Iterates through all entries in all the glossaries (or just those listed in *⟨glossary labels⟩*) and does *⟨body⟩* if the category matches *⟨category-label⟩*. The control sequences *⟨glossary-cs⟩* and *⟨label-cs⟩* may be used in *⟨body⟩* to access the glossary label and entry label for the current iteration.

```
5177 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
5178   \forallglossaries[#1]{#3}%
5179   {%
5180     \forlgsentries[#3]{#4}%
5181     {%
5182       \glsifcategory{#4}{#2}{#5}{}%
5183     }%
5184   }%
5185 }
```

achwithattribute

```
\glsforeachwithattribute[⟨glossary labels⟩]{⟨attribute-label⟩}
{⟨attribute-value⟩}{⟨glossary-cs⟩}{⟨label-cs⟩}{⟨body⟩}
```

Iterates through all entries in all the glossaries (or just those listed in *⟨glossary labels⟩*) and does *⟨body⟩* if the category attribute *⟨attribute-label⟩* matches *⟨attribute-value⟩*. The control sequences *⟨glossary-cs⟩* and *⟨label-cs⟩* may be used in *⟨body⟩* to access the glossary label and entry label for the current iteration.

```
5186 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5187   \forallglossaries[#1]{#4}%
5188   {%
5189     \forlgsentries[#4]{#5}%
5190     {%
5191       \glsifattribute{#5}{#2}{#3}{#6}{}%
5192     }%
5193   }%
5194 }
```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to `index` and add `\glstrpostdescription`.

```
5195 \ifdef\newterm
5196 {%
```

`\newterm`

```
5197   \renewcommand*{\newterm}[2][ ]{%
5198     \newglossaryentry{#2}%
5199     {type={index},category=index,name={#2},%
```

```

5200     description={\glxtrpostdescription\nopostdesc},#1}%
5201 }

```

Indexed terms are regular by default.

```

5202 \glissetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5203 \newcommand*{\glxtrpostdescindex}{}

5204 }
5205 {}

```

If the symbols package option was used, define a similar command for symbols, but set the default sort to the label rather than the name as the symbols will typically contain commands that will confuse makeindex and xindy.

```

5206 \ifdef\printsymbols
5207 {%

```

glxtrnewsymbol Unlike \newterm, this has a separate argument for the label (since the symbol will likely contain commands).

```

5208 \newcommand*{\glxtrnewsymbol}[3] [] {%
5209     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5210 }

```

Symbols are regular by default.

```

5211 \glissetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

5212 \newcommand*{\glxtrpostdescsymbol}{}

5213 }
5214 {}

```

Similar for the numbers option.

```

5215 \ifdef\printnumbers
5216 {%

```

glxtrnewnumber

```

5217 \ifdef\printnumbers
5218 \newcommand*{\glxtrnewnumber}[3] [] {%
5219     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5220 }

```

Numbers are regular by default.

```

5221 \glissetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```

5222 \newcommand*{\glxtrpostdescnumber}{}

```

```
5223 }
5224 {}
```

`\glstrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
5225 \newcommand*{\glstrsetcategory}[2]{%
5226   \@for\@glstr@label:=#1\do
5227   {%
5228     \glsfieldxdef{\@glstr@label}{category}{#2}%
5229   }%
5230 }
```

`\glstrcategoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```
5231 \newcommand*{\glstrcategoryforall}[2]{%
5232   \forallglossaries[#1]{\@glstr@type}{%
5233     \forallsentries[\@glstr@type]{\@glstr@label}%
5234     {%
5235       \glsfieldxdef{\@glstr@label}{category}{#2}%
5236     }%
5237   }%
5238 }
```

`\glstrfieldtitlecase` `\glstrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```
5239 \newcommand*{\glstrfieldtitlecase}[2]{%
5240   \expandafter\glstrfieldtitlecasesecs\expandafter
5241   {\csname glo@glsdetoklabel{#1}@#2\endcsname}%
5242 }
```

`\glstrfieldtitlecasesecs` The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```
5243 \newcommand*{\glstrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}
```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

`\glossentrydesc` If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```
5244 \@ifpackageloaded{glossaries-accsupp}
5245 {
5246   \renewcommand*{\glossentrydesc}[1]{%
5247     \glsdoifexistsorwarn{#1}%
5248     {%
5249       \glssetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the glossdescfont attribute to determine the font applied.

```

5250 \glshasattribute{#1}{glossdescfont}%
5251 {%
5252 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5253 \ifcsdef{\@glxtr@attrval}%
5254 {%
5255 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5256 }%
5257 {%
5258 \GlossariesExtraWarning{Unknown control sequence name
5259 '\@glxtr@attrval' supplied in glossdescfont attribute
5260 for entry '#1'. Ignoring}%
5261 \let\@glxtr@glossdescfont\@firstofone
5262 }%
5263 }%
5264 {\let\@glxtr@glossdescfont\@firstofone}%
5265 \glusifattribute{#1}{glossdesc}{firstuc}%
5266 {%
5267 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5268 }%
5269 {%
5270 \glusifattribute{#1}{glossdesc}{title}%
5271 {%
5272 \@glxtr@do@titlecaps@warn
5273 \glsdescriptionaccessdisplay
5274 {%
5275 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5276 }%
5277 {#1}%
5278 }%
5279 {%
5280 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5281 }%
5282 }%
5283 }%
5284 }
5285 }
5286 {
5287 \renewcommand*\glossentrydesc[1]{%
5288 \glsdoifexistsorwarn{#1}%
5289 {%
5290 \glissetabbrvfmt{\glscategory{#1}}%
5291 \glshasattribute{#1}{glossdescfont}%
5292 {%
5293 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5294 \ifcsdef{\@glxtr@attrval}%
5295 {%
5296 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5297 }%

```

```

5298      {%
5299      \GlossariesExtraWarning{Unknown control sequence name
5300      '\@glsxtr@attrval' supplied in glossdescfont attribute
5301      for entry '#1'. Ignoring}%
5302      \let\@glsxtr@glossdescfont\@firstofone
5303      }%
5304    }%
5305    {\let\@glsxtr@glossdescfont\@firstofone}%
5306    \glsifattribute{#1}{glossdesc}{firstuc}%
5307    {%
5308      \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5309    }%
5310    {%
5311      \glsifattribute{#1}{glossdesc}{title}%
5312      {%
5313        \@glsxtr@do@titlecaps@warn
5314        \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5315      }%
5316      {%
5317        \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5318      }%
5319    }%
5320  }%
5321 }
5322 }

```

`\glossentryname` If the glossname attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5323 \ifpackageloaded{glossaries-accsupp}
5324 {
5325   \renewcommand*{\glossentryname}[1]{%
5326     \@glsdoifexistsorwarn{#1}%
5327     {%
5328       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5329   \glshasattribute{#1}{glossnamefont}%
5330   {%
5331     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5332     \ifcsdef{\@glsxtr@attrval}%
5333     {%
5334       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5335     }%
5336     {%
5337       \GlossariesExtraWarning{Unknown control sequence name
5338       '\@glsxtr@attrval' supplied in glossnamefont attribute
5339       for entry '#1'. Reverting to default \string\glsnamefont}%
5340       \let\@glsxtr@glossnamefont\glsnamefont
5341     }%
5342   }%

```

```

5343 {\let\@glsxtr@glossnamefont\glsnamefont}%
5344 \glsifattribute{#1}{glossname}{firstuc}%
5345 {%
5346   \glsnameaccessdisplay
5347   {%
5348     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5349   }%
5350   {#1}%
5351 }%
5352 {%
5353   \glsifattribute{#1}{glossname}{title}%
5354   {%
5355     \@glsxtr@do@titlecaps@warn
5356     \glsnameaccessdisplay
5357     {%
5358       \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
5359     }%
5360     {#1}%
5361   }%
5362   {%
5363     \glsifattribute{#1}{glossname}{uc}%
5364     {%
5365       \glsnameaccessdisplay
5366       {%

```

Hide the label from the upper-casing command.

```

5367       \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5368       \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5369     }%
5370     {#1}%
5371   }%
5372   {%
5373     \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5374     \glsnameaccessdisplay
5375     {%
5376       \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
5377     }%
5378     {#1}%
5379   }%
5380 }%
5381 }%

```

Do post-name hook:

```

5382   \glsxtrpostnamehook{#1}%
5383 }%
5384 }
5385 }
5386 {
5387   \renewcommand*{\glossentryname}[1]{%
5388     \@glsdoifexistsorwarn{#1}%

```

```

5389  {%
5390      \glsetabbrvfmt{\glscategory{#1}}%
5391      \glshasattribute{#1}{glossnamefont}%
5392  {%
5393      \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5394      \ifcsdef{\@glxtr@attrval}%
5395      {%
5396          \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5397      }%
5398      {%
5399          \GlossariesExtraWarning{Unknown control sequence name
5400              '\@glxtr@attrval' supplied in glossnamefont attribute
5401              for entry '#1'. Reverting to default \string\glnamefont}%
5402          \let\@glxtr@glossnamefont\glnamefont
5403      }%
5404  }%
5405  {\let\@glxtr@glossnamefont\glnamefont}%
5406  \gl@ifattribute{#1}{glossname}{firstuc}%
5407  {%
5408      \@glxtr@glossnamefont{\Glsentryname{#1}}%
5409  }%
5410  {%
5411      \gl@ifattribute{#1}{glossname}{title}%
5412      {%
5413          \@glxtr@do@titlecaps@warn
5414          \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
5415      }%
5416      {%
5417          \gl@ifattribute{#1}{glossname}{uc}%
5418      {%

```

Hide the label from the upper-casing command.

```

5419      \letcs{\glo@name}{\glo@glsetoklabel{#1}@name}%
5420      \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5421      }%
5422      {%

```

This little trick is used by glossaries to allow the user to redefine `\glnamefont` to use `\makefirstuc`. Support it even though they can now use the `firstuc` attribute.

```

5423      \letcs{\glo@name}{\glo@glsetoklabel{#1}@name}%
5424      \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5425      }%
5426      }%
5427      }%

```

Do post-name hook.

```

5428      \glxtrpostnamehook{#1}%
5429      }%
5430  }
5431 }

```


`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
5432 \@ifpackageloaded{glossaries-accsupp}
5433 {
5434   \renewcommand*{\Glossentryname}[1]{%
5435     \@glsdoifexistsorwarn{#1}%
5436     {%
5437       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
5438     \glshasattribute{#1}{glossnamefont}%
5439     {%
5440       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5441       \ifcsdef{\@glstr@attrval}%
5442       {%
5443         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5444       }%
5445       {%
5446         \GlossariesExtraWarning{Unknown control sequence name
5447           '\@glstr@attrval' supplied in glossnamefont attribute
5448           for entry '#1'. Reverting to default \string\glnamefont}%
5449         \let\@glstr@glossnamefont\glnamefont
5450       }%
5451     }%
5452     {\let\@glstr@glossnamefont\glnamefont}%
5453     \glnameaccessdisplay
5454     {%
5455       \@glstr@glossnamefont{\Glsentryname{#1}}%
5456     }%
5457     {#1}%
```

Do post-name hook:

```
5458     \glstrpostnamehook{#1}%
5459   }%
5460 }
5461 }
5462 {
5463   \renewcommand*{\Glossentryname}[1]{%
5464     \@glsdoifexistsorwarn{#1}%
5465     {%
5466       \glsetabbrvfmt{\glscategory{#1}}%
5467       \glshasattribute{#1}{glossnamefont}%
5468       {%
5469         \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5470         \ifcsdef{\@glstr@attrval}%
5471         {%
5472           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5473         }%
5474         {%
5475           \GlossariesExtraWarning{Unknown control sequence name
5476             '\@glstr@attrval' supplied in glossnamefont attribute
```

```

5477         for entry '#1'. Reverting to default \string\glsnamefont}%
5478         \let\@glsxtr@glossnamefont\glsnamefont
5479     }%
5480 }%
5481 {\let\@glsxtr@glossnamefont\glsnamefont}%
5482 \@glsxtr@glossnamefont{\Glsentryname{#1}}}%

```

Do post-name hook:

```

5483     \glsxtrpostnamehook{#1}%
5484 }%
5485 }
5486 }

```

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`xtrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```

5487 \newcommand*{\glsxtrpostnamehook}[1]{%
5488   \let\@glsnumberformat\@glsxtr@defaultnumberformat
5489   \glsxtrdoautoindexname{#1}{indexname}%

```

Allow additional code regardless of category:

```

5490   \glsextrapostnamehook{#1}%

```

Allow categories to hook in here.

```

5491   \csuse{glsxtrpostname\glscategory{#1}}%
5492 }

```

`trapostnamehook`

```

5493 \newcommand*{\glsextrapostnamehook}[1]{}%

```

`etaccessdisplay`

```

5494 \@ifpackageloaded{glossaries-accsupp}
5495 {
5496   \newcommand*{\glsxtr@setaccessdisplay}[1]{%
5497     \ifcsdef{gls#1accessdisplay}%
5498     {\letcs\@glsxtr@accessdisplay{gls#1accessdisplay}}%
5499     {%

```

This is essentially the reverse of `\@gls@fetchfield`, since the field supplied to `\glossentryname` has to be the internal label, but the `\gls<field>accessdisplay` commands use the key name.

```

5500     \edef\@gls@thisval{#1}%
5501     \@for\@gls@map:=\@gls@keymap\do{%
5502       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5503       \ifdefequal{\@this@key}{\@gls@thisval}%
5504       {%
5505         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5506         \@endfortrue

```

```

5507     }%
5508     {}%
5509     }%
5510     \ifcsdef{gls\@gls@thisval accessdisplay}%
5511     {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5512     {\let\@glxtr@accessdisplay\@firstoftwo}%
5513     }%
5514 }
5515 }
5516 {%
5517 \newcommand*{\glxtr@setaccessdisplay}[1]{%
5518 \let\@glxtr@accessdisplay\@firstoftwo}
5519 }

```

sentrynameother Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

5520 \newrobustcmd*{\glossentrynameother}[2]{%
5521 \@glsoifexistsorwarn{#1}%
5522 {%

```

Accessibility support:

```

5523 \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5524 \glsssetabbrfmt{\glscategory{#1}}%
5525 \glshasattribute{#1}{glossnamefont}%
5526 {%
5527 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5528 \ifcsdef{\@glxtr@attrval}%
5529 {%
5530 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5531 }%
5532 {%
5533 \GlossariesExtraWarning{Unknown control sequence name
5534 '\@glxtr@attrval' supplied in glossnamefont attribute
5535 for entry '#1'. Reverting to default \string\glsgnamefont}%
5536 \let\@glxtr@glossnamefont\glsgnamefont
5537 }%
5538 }%
5539 {\let\@glxtr@glossnamefont\glsgnamefont}%
5540 \glsgifattribute{#1}{glossname}{firstuc}%
5541 {%
5542 \@glxtr@accessdisplay
5543 {\@glxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5544 {#1}%
5545 }%
5546 {%
5547 \glsgifattribute{#1}{glossname}{title}%
5548 {%
5549 \@glxtr@do@titlecaps@warn

```

```

5550      \@glsxtr@accessdisplay
5551      {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
5552      {#1}%
5553  }%
5554  {%
5555      \glsifattribute{#1}{glossname}{uc}%
5556      {%
5557          \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5558          \@glsxtr@accessdisplay
5559          {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5560          {#1}%
5561      }%
5562      {%
5563          \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5564          \@glsxtr@accessdisplay
5565          {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
5566          {#1}%
5567      }%
5568  }%
5569  }%

  Do post-name hook.
5570      \glsxtrpostnamehook{#1}%
5571  }%
5572 }

```

`format@override` Determines if the `format` key should override the indexing attribute value.

```

5573 \newif\if@glsxtr@format@override
5574 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5575 \@ifpackageloaded{hyperref}
5576 {
  If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
  don't add it.
5577   \ifHy@hyperindex
5578     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5579       \@glsxtr@format@override true
5580       \appto\theindex{\let\glshypernumber\@firstofone}%
5581     }
5582   \else
5583     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5584       \@glsxtr@format@override true
5585       \appto\theindex{\let\glshypernumber\hyperpage}%
5586     }
5587   \fi

```

```

5588 }
5589 {
5590   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5591     \@glsxtr@format@override true
5592   }
5593 }
5594 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

5595 \newcommand*{\glxtrdoautoindexname}[2]{%
5596   \glshasattribute{#1}{#2}%
5597   {%

```

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.

```

5598   \@glsxtr@autoindex@setname{#1}%

```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```

5599   \protected@edef\@glsxtr@attrval{\glsggetattribute{#1}{#2}}%
5600   \if@glsxtr@format@override

5601     \ifx\@glsnumberformat\@glsxtr@defaultnumberformat
5602     \else
5603       \let\@glsxtr@attrval\@glsnumberformat
5604     \fi
5605   \fi
5606   \ifdefstring{\@glsxtr@attrval}{true}%
5607   {}%
5608   {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
5609   \expandafter\glxtrautoindex\expandafter{\@glo@name}%
5610 }%
5611 {}%
5612 }

```

glxtrautoindex

```

5613 \newcommand*{\glxtrautoindex}{\index}

```

toindex@setname Assign \@glo@name for use with indexname attribute.

```

5614 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
5615   \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
5616   \glxtrautoindexassignsort{\@glo@sort}{#1}%
5617   \@gls@checkmkidxchars\@glo@sort
5618   \@glsxtr@autoindex@doextra@esc\@glo@sort
5619   \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
5620 }

```

rautoindexentry Command used for the actual part when auto-indexing.

```

5621 \newcommand*{\glxtrautoindexentry}[1]{\string\glstryname{#1}}

```

trautoindexsort Used to assign the sort value when auto-indexing.

```
5622 \newcommand*{\glxtrautoindexassignsort}[2]{%
5623   \glslentryfield{#1}{#2}{sort}%
5624 }
```

dex@doextra@esc

```
5625 \newcommand*{\@glxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5626   \ifx\@glxtr@autoindex@esc\@gl@quotechar
5627   \else
5628     \def\@gl@checkedmkidx{}%
5629     \edef\@glxtr@checkspch{%
5630       \noexpand\@glxtr@autoindex@escquote\expandonce{#1}%
5631       \noexpand\@empty\@glxtr@autoindex@esc\noexpand\@nnil
5632       \@glxtr@autoindex@esc\noexpand\@empty\noexpand\@glxtr@endescspch}%
5633     \@glxtr@checkspch
5634     \let#1\@gl@checkedmkidx\relax
5635   \fi
```

Escape actual character unless it has already been escaped.

```
5636   \ifx\@glxtr@autoindex@at\@gl@actualchar
5637   \else
5638     \def\@gl@checkedmkidx{}%
5639     \edef\@glxtr@checkspch{%
5640       \noexpand\@glxtr@autoindex@escat\expandonce{#1}%
5641       \noexpand\@empty\@glxtr@autoindex@at\noexpand\@nnil
5642       \@glxtr@autoindex@at\noexpand\@empty\noexpand\@glxtr@endescspch}%
5643     \@glxtr@checkspch
5644     \let#1\@gl@checkedmkidx\relax
5645   \fi
```

Escape level character unless it has already been escaped.

```
5646   \ifx\@glxtr@autoindex@level\@gl@levelchar
5647   \else
5648     \def\@gl@checkedmkidx{}%
5649     \edef\@glxtr@checkspch{%
5650       \noexpand\@glxtr@autoindex@esclevel\expandonce{#1}%
5651       \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
5652       \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
5653     \@glxtr@checkspch
5654     \let#1\@gl@checkedmkidx\relax
5655   \fi
```

Escape encap character unless it has already been escaped.

```
5656   \ifx\@glxtr@autoindex@encap\@gl@encapchar
5657   \else
5658     \def\@gl@checkedmkidx{}%
5659     \edef\@glxtr@checkspch{%
5660       \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
5661       \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
```

```

5662      \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5663      \@glsxtr@checkspch
5664      \let#1\@gls@checkedmkidx\relax
5665      \fi
5666 }

```

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`.

```
5667 \newcommand*{\@glsxtr@autoindex@at}{}

```

`\trSetActualChar` Set the actual character.

```

5668 \newcommand*{\GlsXtrSetActualChar}[1]{%
5669   \gdef\@glsxtr@autoindex@at{#1}%
5670   \def\@glsxtr@autoindex@escat##1#1##2#1##3\@glsxtr@endescspch{%
5671     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
5672   }%
5673 }
5674 \@onlypreamble\GlsXtrSetActualChar
5675 \makeatother
5676 \GlsXtrSetActualChar{}
5677 \makeatletter

```

`\autoindex@encap` Encap character for use with `\index`.

```
5678 \newcommand*{\@glsxtr@autoindex@encap}{}

```

`\XtrSetEncapChar` Set the encap character.

```

5679 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5680   \gdef\@glsxtr@autoindex@encap{#1}%
5681   \def\@glsxtr@autoindex@escencap##1#1##2#1##3\@glsxtr@endescspch{%
5682     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
5683   }%
5684 }
5685 \GlsXtrSetEncapChar{}
5686 \@onlypreamble\GlsXtrSetEncapChar

```

`\autoindex@level` Level character for use with `\index`.

```
5687 \newcommand*{\@glsxtr@autoindex@level}{}

```

`\XtrSetLevelChar` Set the encap character.

```

5688 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5689   \gdef\@glsxtr@autoindex@level{#1}%
5690   \def\@glsxtr@autoindex@esclevel##1#1##2#1##3\@glsxtr@endescspch{%
5691     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@esclevel}{##1}{##2}{##3}%
5692   }%
5693 }
5694 \GlsXtrSetLevelChar{!}
5695 \@onlypreamble\GlsXtrSetLevelChar

```

r@autoindex@esc Escape character for use with \index.

```
5696 \newcommand*{\@glxtr@autoindex@esc}{"
```

lsXtrSetEscChar Set the escape character.

```
5697 \newcommand*{\GlsXtrSetEscChar}[1]{%
5698   \gdef\@glxtr@autoindex@esc{#1}%
5699   \def\@glxtr@autoindex@escquote##1##2##3\@glxtr@endescspch{%
5700     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
5701   }%
5702 }
5703 \GlsXtrSetEscChar{"}
5704 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
5705 \ifdef\actualchar
5706 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5707 {}
```

Quote character \quotechar:

```
5708 \ifdef\quotechar
5709 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5710 {}
```

Level character \levelchar:

```
5711 \ifdef\levelchar
5712 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5713 {}
```

Encap character \encapchar:

```
5714 \ifdef\encapchar
5715 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5716 {}
```

leto@endescspch

```
5717 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

<code>\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}</code>
--

```
5718 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
5719   \@glstmpb=\expandafter{\@glstmpb}%
5720   \toks@={#3}%
5721   \ifx\@nnil#3\relax
5722     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
5723   \else
5724     \ifx\@nnil#4\relax
5725       \edef\@glstmpb{\the\@glstmpb\the\toks@}%
5726       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
```



```

5727      #4#5\@glxtr@endescspch}%
5728  \else
5729    \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
5730      \@glxtr@autoindex@esc#1}%
5731    \def\@@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
5732  \fi
5733 \fi
5734 \@glxtr@checkspch
5735 }

```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```

5736 \renewcommand*{\Glossentrydesc}[1]{%
5737   \glsdoifexistsorwarn{#1}%
5738   {%
5739     \glssetabbrvfmt{\glscategory{#1}}%
5740     \Glsaccessdesc{#1}%
5741   }%
5742 }

```

\Glossentrysymbol Redefine to set the abbreviation format and accessibility support.

```

5743 \renewcommand*{\Glossentrysymbol}[1]{%
5744   \glsdoifexistsorwarn{#1}%
5745   {%
5746     \glssetabbrvfmt{\glscategory{#1}}%
5747     \Glsaccesssymbol{#1}%
5748   }%
5749 }

```

\Glossentrysymbol Redefine to set the abbreviation format and accessibility support.

```

5750 \renewcommand*{\Glossentrysymbol}[1]{%
5751   \glsdoifexistsorwarn{#1}%
5752   {%
5753     \glssetabbrvfmt{\glscategory{#1}}%
5754     \Glsaccesssymbol{#1}%
5755   }%
5756 }

```

Allow initials to be marked but only use the formatting for the tag in the glossary.

\GlsXtrEnableInitialTagging 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.

```

5757 \newcommand*{\GlsXtrEnableInitialTagging}{%
5758   \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
5759 }
5760 \@onlypreamble\GlsXtrEnableInitialTagging

```

\GlsXtrEnableInitialTagging Starred version undefines command.

```

5761 \newcommand*{\s@glxtr@enabletagging}[2]{%
5762   \undef#2%
5763   \@glxtr@enabletagging{#1}{#2}%
5764 }

```

r@enabletagging Internal command.

```

5765 \newcommand*{\@glxtr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
5766   \@for\@glxtr@cat:=#1\do
5767   {%
5768     \ifdefempty\@glxtr@cat
5769     {}%
5770     {\glsssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%
5771   }%
5772   \newrobustcmd*#2[1]{##1}%
5773   \def\@glxtr@taggingcs{#2}%
5774   \renewcommand*{\@glxtr@activate@initialtagging{%
5775     \let#2\@glxtr@tag
5776   }%
5777   \ifundef\@gl@preglossaryhook
5778   {\GlossariesExtraWarning{Initial tagging requires at least
5779     glossaries.sty v4.19 to work correctly}}%
5780   {}%
5781 }

```

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

```

5782 \ifundef\mfu@checkword@do
5783 {
5784   \newcommand*{\mfu@checkword@do}[1]{%
5785     \ifdefstring{\mfu@checkword@arg}{#1}%
5786     {%
5787       \let\@mfu@domakefirstuc\@firstofone
5788       \listbreak
5789     }%
5790     {}%
5791   }

```

\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.

```

5792 \ifundef\mfu@checkword
5793 {
5794   \newcommand{\@glxtr@do@titlecaps@warn}{%
5795     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5796       support not available}%

```

One warning should suffice.

```
5797 \let\@glxstr@do@titlecaps@warn\relax
5798 }
5799 }
5800 {
5801 \renewcommand*{\mfu@checkword}[1]{%
5802 \def\mfu@checkword@arg{#1}%
5803 \let\@mfu@domakefirstuc\makefirstuc
5804 \forlistloop\mfu@checkword@do\@mfu@nocaplist
5805 }
5806 }
5807 }
5808 {}% no patch required
```

@titlecaps@warn Do warning if title case not supported.

```
5809 \newcommand*{\@glxstr@do@titlecaps@warn}{}%
```

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.

```
5810 \newcommand*{\@glxstr@activate@initialtagging}{}%
```

\@glxstr@tag Definition of tagging command when used in glossary.

```
5811 \newrobustcmd*{\@glxstr@tag}[1]{%
5812 \gl@ifattribute{\gl@currententrylabel}{tagging}{true}%
5813 {\glxstr@tagfont{#1}}{#1}%
5814 }
```

\glxstrtagfont Used in the glossary.

```
5815 \newcommand*{\glxstrtagfont}[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.

```
5816 \ifdef\@gls@preglossaryhook
5817 {
5818 \renewcommand*{\@gls@preglossaryhook}{%
5819 \@glxstr@activate@initialtagging
```

Since the glossaries are automatically scoped, \@glxstr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

```
5820 \ifundef\@glxstr@org@postdescription
5821 {%
5822 \let\@glxstr@org@postdescription\glspostdescription
5823 \renewcommand*{\glspostdescription}{%
5824 \ifgl@entryexists{\gl@currententrylabel}%
5825 {%
5826 \glxstrpostdescription
5827 \@glxstr@org@postdescription
```

```

5828     }%
5829     {}%
5830     }%
5831     }%
5832     {}%

```

Enable the options used by \@glsxtrp:

```

5833     \glossxtrsetpopts
5834     }%
5835 }
5836 {}

```

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.

```

5837 \newcommand*{\glsxtrpostdescription}{%
5838   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5839 }

```

postdescgeneral

```

5840 \newcommand*{\glsxtrpostdescgeneral}{}

```

xtrpostdescterm

```

5841 \newcommand*{\glsxtrpostdescterm}{}

```

postdescacronym

```

5842 \newcommand*{\glsxtrpostdescacronym}{}

```

escabbreviation

```

5843 \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.

```

5844 \renewcommand*{\glspostlinkhook}{%
5845   \ifglsentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
5846 }

```

xtrpostlinkhook The entry label should already be stored in \glslabel by \@gls@link.

```

5847 \newcommand*{\glsxtrpostlinkhook}{%
5848   \glsxtrdiscardperiod{\glslabel}%
5849   {\glsxtrpostlinkendsentence}%
5850   {\glsxtrifcustomdiscardperiod
5851     {\glsxtrifperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}}%
5852   {\glsxtrpostlink}%
5853   }%
5854 }

```

`omdiscardperiod` Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
5855 \newcommand*{\glxtrifcustomdiscardperiod}[2]{#2}
```

`\glxtrpostlink`

```
5856 \newcommand*{\glxtrpostlink}{%
5857 \csuse{\glxtrpostlink\glscategory{\glslabel}}%
5858 }
```

`linkendsentence` Done by `\glxtrpostlinkhook` if a full stop is discarded.

```
5859 \newcommand*{\glxtrpostlinkendsentence}{%
5860 \ifcsdef{\glxtrpostlink\glscategory{\glslabel}}
5861 {%
5862 \csuse{\glxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
5863 .\spacefactor\sfcode'\. \relax
5864 }%
5865 {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```
5866 \spacefactor\sfcode'\. \relax
5867 }%
5868 }
```

`dDescOnFirstUse` 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.

```
5869 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
5870 \glxtrifwasfirstuse{\space\glxtrparen{\glssaccessdesc{\glslabel}}}{}%
5871 }
```

`ymbolOnFirstUse` 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.

```
5872 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5873 \glxtrifwasfirstuse
5874 {%
5875 \ifglshassymbol{\glslabel}%
5876 {\space\glxtrparen{\glssaccesssymbol{\glslabel}}}%
5877 {}%
5878 }%
5879 {}%
5880 }
```

`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).

```

5881 \newcommand*{\glxtrdiscardperiod}[3]{%
5882   \glxtrifwasfirstuse
5883   {%
5884     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5885     {#3}%
5886     {%
5887       \glusifattribute{#1}{discardperiod}{true}%
5888       {%
5889         \glusifplural
5890         {%
5891           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5892           {\glxtrifperiod{#2}{#3}}%
5893           {#3}%
5894         }%
5895         {%
5896           \glxtrifperiod{#2}{#3}%
5897         }%
5898       }%
5899     }%
5900   }%
5901 }%
5902 {%
5903   \glusifattribute{#1}{discardperiod}{true}%
5904   {%
5905     \glusifplural
5906     {%
5907       \glusifattribute{#1}{pluraldiscardperiod}{true}%
5908       {\glxtrifperiod{#2}{#3}}%
5909       {#3}%
5910     }%
5911     {%
5912       \glxtrifperiod{#2}{#3}%
5913     }%
5914   }%
5915   {#3}%
5916 }%
5917 }

```

`\glxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

5918 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.\{\@firstoftwo{#1}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```

5919 \newcommand*{\glxtr@punclist}{.,;?!}

```

`punctuationmark` Add character to punctuation list.

```

5920 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}

```

unctuationmarks Reset the punctuation list.

```
5921 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

```
\glxtrifpunc    \glxtrifnextpunc{<true part>}{<false part>}
```

Test if this is followed by a punctuation mark. (Adapted from \new@ifnextchar.)

```
5922 \newcommand*{\glxtrifnextpunc}[2]{%
5923   \def\reserved@a{#1}%
5924   \def\reserved@b{#2}%
5925   \futurelet\@glspunc@token\glxtr@ifnextpunc
5926 }
```

glxtr@ifnextpunc

```
5927 \newcommand*{\glxtr@ifnextpunc}{%
5928   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5929   \reserved@b
5930 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
5931 \newcommand*{\glxtr@ifpunctoken}[1]{%
5932   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5933 }
```

glxtr@ifpunctoken

```
5934 \def\@glxtr@ifpunctoken#1#2{%
5935   \let\reserved@d=#2%
5936   \ifx\reserved@d\@nnil
5937     \let\glxtr@next\@glxtr@notfoundinlist
5938   \else
5939     \ifx#1\reserved@d
5940       \let\glxtr@next\@glxtr@foundinlist
5941     \else
5942       \let\glxtr@next\@glxtr@ifpunctoken
5943     \fi
5944   \fi
5945   \glxtr@next#1%
5946 }
```

glxtr@foundinlist

```
5947 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5948 \def\@glxtr@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.

```
5949 \newcommand{\glsxtrdopostpunc}[1]{%
5950   \glsxtrifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
5951 }
```

@glsxtr@swaptwo

```
5952 \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.

```
5953 \define@key{glsxtrabbrv}{category}{%
5954   \edef\glscategorylabel{#1}%
5955   \ifcsdef{@glsabbrv@current@#1}%
5956   {%
```

Warning should already have been issued.

```
5957   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5958   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5959   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
5960   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
5961 }%
5962 {}%
5963 }
```

Save the short plural form. This may be needed before the entry is defined.

```
5964 \define@key{glsxtrabbrv}{shortplural}{%
5965   \def\@gls@shortpl{#1}%
5966 }
```

Similarly for the long plural form.

```
5967 \define@key{glsxtrabbrv}{longplural}{%
5968   \def\@gls@longpl{#1}%
5969 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
5970 \newtoks\glsshortpltok
```



```

\glslongpltok
5971 \newtoks\glslongpltok

sxt@insertdots  Provided in case user wants to automatically insert dots between each letter of the abbrevi-
                  ation. 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.
5972 \newcommand*{\@glsxt@insertdots}[2]{%
5973   \def#1{}%
5974   \@glsxt@insert@dots#1#2\@nnil
5975 }

xt@insert@dots
5976 \newcommand*{\@glsxt@insert@dots}[2]{%
5977   \ifx\@nnil#2\relax
5978   \let\@glsxt@insert@dots@next\@gobble
5979   \else
5980   \ifx\relax#2\relax
5981   \else
5982     \appto#1{#2.}%
5983   \fi
5984   \let\@glsxt@insert@dots@next\@glsxt@insert@dots
5985   \fi
5986   \@glsxt@insert@dots@next#1%
5987 }

        Similarly provide a way of replacing spaces with \glsxtwordsep, which first needs to be
        defined:

\glsxtwordsep
5988 \newcommand*{\glsxtwordsep}{\space}

        Each word is marked with

\glsxtword
5989 \newcommand*{\glsxtword}[1]{#1}

tr@markwordseps
5990 \newcommand*{\@glsxt@markwordseps}[2]{%
5991   \def#1{}%
5992   \@glsxt@mark@wordseps#1#2 \@nnil
5993 }

r@mark@wordseps
5994 \def\@glsxt@mark@wordseps#1#2 #3{%
5995   \ifdefempty{#1}%

```

```

5996 {\def#1{\protect\glstrword{#2}}}%
5997 {\appto#1{\protect\glstrwordsep\protect\glstrword{#2}}}%
5998 \ifx\@nnil#3\relax
5999 \let\@glstr@mark@wordseps@next\relax
6000 \else
6001 \def\@glstr@mark@wordseps@next{%
6002   \@glstr@mark@wordseps#1#3}%
6003 \fi
6004 \@glstr@mark@wordseps@next
6005 }

```

newabbreviation Define a new generic abbreviation.

```

6006 \newcommand*{\newabbreviation}[4][]{%
6007   \glstr@newabbreviation{#1}{#2}{#3}{#4}%
6008 }

```

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.)

```

6009 \newcommand*{\glstr@newabbreviation}[4]{%
6010   \glscopylisttok{#1}%
6011   \glscopylabeltok{#2}%
6012   \glscopyshorttok{#3}%
6013   \glscopylongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

6014 \def\glstr@orgshort{#3}%
6015 \def\glstr@orglong{#4}%

```

Get the category.

```

6016 \def\glscategorylabel{abbreviation}%
6017 \glstr@applyabbrvstyle{\@glsabrv@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```

6018 \setkeys*{\glstr@abbrv}{shortplural,longplural}{#1}%

```

Set the default long plural

```

6019 \def\@glslongpl{#4\glspluralsuffix}%
6020 \let\@glslongpl\@glslongpl

```

Has the markwords attribute been set?

```

6021 \glscopycategoryattribute{\glscopylabel}{markwords}{true}%
6022 {%
6023   \@glstr@markwordseps\@glslong{#4}%
6024   \expandafter\def\expandafter\@glslongpl\expandafter
6025     {\@glslong\glspluralsuffix}%
6026   \let\@glslongpl\@glslongpl

```

Update \glslongtok.

```

6027   \expandafter\glslongtok\expandafter{\@glslong}%
6028 }%
6029 {}%

```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
6030 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
6031 {%
6032   \@glstr@markwordseps\@gls@short{#3}%
6033 }%
6034 {%
```

Has the insertdots attribute been set?

```
6035 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
6036 {%
6037   \@glstr@insertdots\@gls@short{#3}%
6038   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6039 }%
6040 {\def\@gls@short{#3}}%
6041 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
6042 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
6043 {%
6044   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6045     '\abbrvpluralsuffix}%
6046 }%
6047 {%
```

Has the noshortplural attribute been set?

```
6048 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
6049 {%
6050   \let\@gls@shortpl\@gls@short
6051 }%
6052 {%
6053   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6054     \abbrvpluralsuffix}%
6055 }%
6056 }%
```

Update \glsshorttok:

```
6057 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
6058 \glstrnewabbrvprehook{#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).

```
6059 \setkeys*{glstrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
6060 \ifx\@gls@default@longpl\@gls@longpl
6061 \else
```

Has the markwords attribute been set?

```
6062 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6063 {%
```

```

6064      \expandafter\@glxtr@markwordseps\expandafter\@gls@longpl\expandafter
6065      {\@gls@longpl}%
6066      }%
6067      {}%
6068      \fi

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

6069      \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6070      \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

6071      \newabbreviationhook

```

Define this entry:

```

6072      \protected@edef\@do@newglossaryentry{%
6073      \noexpand\newglossaryentry{\the\glslabeltok}%
6074      {%
6075          type=\glxtrabbrvtype,%
6076          category=abbreviation,%
6077          short={\the\glsshorttok},%
6078          shortplural={\the\glsshortpltok},%
6079          long={\the\glslongtok},%
6080          longplural={\the\glslongpltok},%
6081          name={\the\glsshorttok},%
6082          \CustomAbbreviationFields,%
6083          \the\glskeylisttok
6084      }%
6085      }%
6086      \@do@newglossaryentry
6087      \GlsXtrPostNewAbbreviation
6088      }

```

`evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```

6089 \newcommand*{\glxtrnewabbrevpresetkeyhook}[3]{}

```

`NewAbbreviation` Hook used by abbreviation styles.

```

6090 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`bbreviationhook` Hook for use with `\newabbreviation`.

```

6091 \newcommand*{\newabbreviationhook}{}

```

`reviationFields`

```

6092 \newcommand*{\CustomAbbreviationFields}{}

```

`\glxtrparen` For the parenthetical styles.

```

6093 \newcommand*{\glxtrparen}[1]{(#1)}

```

`lsxtrfullformat` Full format without case change.

```

6094 \newcommand*{\glxtrfullformat}[2]{%

```

```

6095 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6096 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6097 }

```

`\glsxtrfullformat` Full format with case change.

```

6098 \newcommand*{\Glsxtrfullformat}[2]{%
6099 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6100 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6101 }

```

`\glsxtrfullplformat` Plural full format without case change.

```

6102 \newcommand*{\Glsxtrfullplformat}[2]{%
6103 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6104 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6105 }

```

`\glsxtrfullplformat` Plural full format with case change.

```

6106 \newcommand*{\Glsxtrfullplformat}[2]{%
6107 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6108 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6109 }

```

`\glsxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```

6110 \newcommand*{\glsxtrfullsep}[1]{\space}

```

In-line formats in case first use isn't compatible with `\glsentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glsxtrinfullformat` Full format without case change.

```

6111 \newcommand*{\Glsxtrinfullformat}{\glsxtrfullformat}

```

`\glsxtrinfullformat` Full format with case change.

```

6112 \newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}

```

`\glsxtrfullplformat` Plural full format without case change.

```

6113 \newcommand*{\Glsxtrinfullplformat}{\glsxtrfullplformat}

```

`\glsxtrinfullplformat` Plural full format with case change.

```

6114 \newcommand*{\Glsxtrinfullplformat}{\Glsxtrfullplformat}

```

Redefine `\glsentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glsxtrfull` set of commands instead.

`\glsentryfull`

```

6115 \renewcommand*{\glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}

```

`\Glsentryfull`

```

6116 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}

```

`\glsentryfullpl`
6117 `\renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}`

`\Glsentryfullpl`
6118 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\sfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.
6119 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\abbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.
6120 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.
6121 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\abbrvdefaultfont`
6122 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.
6123 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`\longdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.
6124 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`\lsfirstlongfont` Font changing command used for the long form on first use or in the full format.
6125 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`\longdefaultfont`
6126 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`\brvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.
6127 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

`\brvpluralsuffix` Default plural suffix.
6128 `\newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).
6129 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}`
6130 `\newcommand*{\ns@glsxtrfull}[2][{}]{%`
6131 `\new@ifnextchar[{\@glsxtr@full{#1}{#2}}{%`
6132 `{\@glsxtr@full{#1}{#2}}[{}]{%`
6133 `}`

\@glxtr@full Low-level macro:

```

6134 \def\@glxtr@full#1#2[#3]{%
6135   \glstoifexists{#2}%
6136   {%
6137     \glsetabbrvfmt{\glscategory{#2}}%
6138     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6139     \let\gl@ifplural\@secondoftwo
6140     \let\glscapscase\@firstofthree
6141     \let\glinsert\@empty
6142     \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%

```

What should \glxtrifwasfirstuse be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```

6143   \glxtrsetupfulldefs
6144   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6145   }%
6146   \glspostlinkhook
6147 }

```

trsetupfulldefs

```

6148 \newcommand*{\glxtrsetupfulldefs}{%
6149   \let\glxtrifwasfirstuse\@firstoftwo
6150 }

```

\Glsxtrfull Full form (first letter uppercase).

```

6151 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\@ns@Glsxtrfull}
6152 \newcommand*\ns@Glsxtrfull[2][ ]{%
6153   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
6154   {\@Glsxtr@full{#1}{#2} [ ]}%
6155 }

```

\@Glsxtr@full Low-level macro:

```

6156 \def\@Glsxtr@full#1#2[#3]{%
6157   \glstoifexists{#2}%
6158   {%
6159     \glsetabbrvfmt{\glscategory{#2}}%
6160     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6161     \let\gl@ifplural\@secondoftwo
6162     \let\glscapscase\@secondofthree
6163     \let\glinsert\@empty
6164     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6165     \glxtrsetupfulldefs
6166     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6167   }%
6168   \glspostlinkhook
6169 }

```

`\GLSxtrfull` Full form (all uppercase).

```
6170 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
6171 \newcommand*\ns@GLSxtrfull[2][]{%
6172   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}{%
6173     {\@GLSxtr@full{#1}{#2}[]}%
6174 }
```

`\@GLSxtr@full` Low-level macro:

```
6175 \def\@GLSxtr@full#1#2[#3]{%
6176   \glsdoifexists{#2}%
6177   {%
6178     \glssetabbrvfmt{\glscategory{#2}}%
6179     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6180     \let\glsifplural\@secondoftwo
6181     \let\glsapscase\@thirdofthree
6182     \let\glsinsert\@empty
6183     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
6184     \glsxtrsetupfulldefs
6185     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6186   }%
6187   \glspostlinkhook
6188 }
```

`\glsxtrfullpl` Plural full form (no case-change).

```
6189 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
6190 \newcommand*\ns@glsxtrfullpl[2][]{%
6191   \new@ifnextchar[{\@glsxtr@fullpl{#1}{#2}}{%
6192     {\@glsxtr@fullpl{#1}{#2}[]}%
6193 }
```

`\@glsxtr@fullpl` Low-level macro:

```
6194 \def\@glsxtr@fullpl#1#2[#3]{%
6195   \glsdoifexists{#2}%
6196   {%
6197     \glssetabbrvfmt{\glscategory{#2}}%
6198     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6199     \let\glsifplural\@firstoftwo
6200     \let\glsapscase\@firstofthree
6201     \let\glsinsert\@empty
6202     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
6203     \glsxtrsetupfulldefs
6204     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6205   }%
6206   \glspostlinkhook
6207 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
6208 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
6209 \newcommand*\ns@Glsxtrfullpl[2][]{%
```



```

6210 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6211         {\@Glsxtr@fullpl{#1}{#2}[]}%
6212 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6213 \def\@Glsxtr@fullpl#1#2[#3]{%
6214   \glsdoifexists{#2}%
6215   {%
6216     \glssetabbrvfmt{\glscategory{#2}}%
6217     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6218     \let\glsifplural\@firstoftwo
6219     \let\glsupcase\@secondofthree
6220     \let\glsinsert\@empty
6221     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6222     \glsxtrsetupfulldefs
6223     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6224   }%
6225   \glspostlinkhook
6226 }

```

`\Glsxtrfullpl` Plural full form (all upper case).

```

6227 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6228 \newcommand*\ns@Glsxtrfullpl[2][]{%
6229   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6230   {\@Glsxtr@fullpl{#1}{#2}[]}%
6231 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6232 \def\@Glsxtr@fullpl#1#2[#3]{%
6233   \glsdoifexists{#2}%
6234   {%
6235     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6236     \let\glsifplural\@firstoftwo
6237     \let\glsupcase\@thirdofthree
6238     \let\glsinsert\@empty
6239     \def\glscustomtext{%
6240       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}%
6241     }
6242     \glsxtrsetupfulldefs
6243     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6244   }%
6245   \glspostlinkhook
6246 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

6246 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\@ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6247 \newcommand*{\ns@glstrshort}[2] [] {%
6248   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} []}%
6249 }

```

Read in the final optional argument:

```

6250 \def\@glstrshort#1#2[#3] {%
6251   \glsd@ifexists{#2}%
6252   {%

```

Need to make sure \glabbrvfont is set correctly.

```

6253     \glsetabbrvfmt{\glscategory{#2}}%
6254     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6255     \let\glstrifwasfirstuse\@secondoftwo
6256     \let\gl@ifplural\@secondoftwo
6257     \let\glscapscase\@firstofthree
6258     \let\glinsert\@empty
6259     \def\glscustomtext{%
6260       \glabbrvfont{\glaccessshort{#2}\ifglstrinsertinside#3\fi}%
6261       \ifglstrinsertinside\else#3\fi
6262     }%
6263     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6264   }%
6265   \glspostlinkhook
6266 }

```

\Glsxtrshort

```

6267 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6268 \newcommand*{\ns@Glsxtrshort}[2] [] {%
6269   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
6270 }

```

Read in the final optional argument:

```

6271 \def\@Glsxtrshort#1#2[#3] {%
6272   \glsd@ifexists{#2}%
6273   {%
6274     \glsetabbrvfmt{\glscategory{#2}}%
6275     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6276     \let\glstrifwasfirstuse\@secondoftwo
6277     \let\gl@ifplural\@secondoftwo
6278     \let\glscapscase\@secondofthree
6279     \let\glinsert\@empty
6280     \def\glscustomtext{%
6281       \glabbrvfont{\Glsaccessshort{#2}\ifglstrinsertinside#3\fi}%
6282       \ifglstrinsertinside\else#3\fi
6283     }%
6284     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6285   }%
6286   \glspostlinkhook
6287 }

```

\GLSxtrshort

```
6288 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
6289 \newcommand*{\ns@GLSxtrshort}[2] [] {%
6290   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}]%
6291 }

    Read in the final optional argument:
6292 \def\@GLSxtrshort#1#2[#3] {%
6293   \glsdoifexists{#2}%
6294   {%
6295     \glssetabbrvfmt{\glscategory{#2}}%
6296     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6297     \let\glsxtrifwasfirstuse\@secondoftwo
6298     \let\glsifplural\@secondoftwo
6299     \let\glsapscase\@thirdofthree
6300     \let\glsinsert\@empty
6301     \def\glscustomtext{%
6302       \mfirstucMakeUppercase
6303       {\glsabbrvfont{\glsaccessshort{#2}}\ifglsxtrininsertinside#3\fi}%
6304       \ifglsxtrininsertinside\else#3\fi
6305     }%
6306   }%
6307   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6308 }%
6309 \glspostlinkhook
6310 }
```

\glsxtrlong

```
6311 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6312 \newcommand*{\ns@glsxtrlong}[2] [] {%
6313   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}]%
6314 }

    Read in the final optional argument:
6315 \def\@glsxtrlong#1#2[#3] {%
6316   \glsdoifexists{#2}%
6317   {%
6318     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6319     \let\glsxtrifwasfirstuse\@secondoftwo
6320     \let\glsifplural\@secondoftwo
6321     \let\glsapscase\@firstofthree
6322     \let\glsinsert\@empty
6323     \def\glscustomtext{%
6324       \glsfont{\glsaccesslong{#2}}\ifglsxtrininsertinside#3\fi}%
6325       \ifglsxtrininsertinside\else#3\fi
6326     }%
6327     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6328   }%
6329 }
```

```

6328 }%
6329 \glspostlinkhook
6330 }

```

\Glsxtrlong

```

6331 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6332 \newcommand*{\ns@Glsxtrlong}[2] [] {%
6333   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
6334 }

    Read in the final optional argument:
6335 \def\@Glsxtrlong#1#2[#3]{%
6336   \glsdoifexists{#2}%
6337   {%
6338     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6339     \let\glstrifwasfirstuse\@secondoftwo
6340     \let\glsifplural\@secondoftwo
6341     \let\glscapscase\@thirdofthree
6342     \let\glsinsert\@empty
6343     \def\glscustomtext{%
6344       \glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6345       \ifglstrinsertinside\else#3\fi
6346     }%
6347     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6348   }%
6349   \glspostlinkhook
6350 }

```

\GLSxtrlong

```

6351 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6352 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6353   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6354 }

    Read in the final optional argument:
6355 \def\@GLSxtrlong#1#2[#3]{%
6356   \glsdoifexists{#2}%
6357   {%
6358     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6359     \let\glstrifwasfirstuse\@secondoftwo
6360     \let\glsifplural\@secondoftwo
6361     \let\glscapscase\@thirdofthree
6362     \let\glsinsert\@empty
6363     \def\glscustomtext{%
6364       \mfirstucMakeUppercase
6365       {\glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6366       \ifglstrinsertinside\else#3\fi

```

```

6367     }%
6368     }%
6369     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6370 }%
6371 \glspostlinkhook
6372 }

```

Plural short forms:

\glsxtrshortpl

```

6373 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\@ns@glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6374 \newcommand*{\ns@glsxtrshortpl}[2][ ]{%
6375   \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2}[]}%
6376 }

```

Read in the final optional argument:

```

6377 \def\@glsxtrshortpl#1#2[#3]{%
6378   \glsdoifexists{#2}%
6379   {%
6380     \glssetabbrvfmt{\glscategory{#2}}%
6381     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6382     \let\glsxtrifwasfirstuse\@secondoftwo
6383     \let\glsifplural\@firstoftwo
6384     \let\glscapscase\@firstofthree
6385     \let\glsinsert\@empty
6386     \def\glscustomtext{%
6387       \glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6388       \ifglsxtrininsertinside\else#3\fi
6389     }%
6390     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6391   }%
6392   \glspostlinkhook
6393 }

```

\Glsxtrshortpl

```

6394 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\@ns@Glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6395 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
6396   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[]}%
6397 }

```

Read in the final optional argument:

```

6398 \def\@Glsxtrshortpl#1#2[#3]{%
6399   \glsdoifexists{#2}%
6400   {%
6401     \glssetabbrvfmt{\glscategory{#2}}%
6402     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6403     \let\glsxtrifwasfirstuse\@secondoftwo

```

```

6404 \let\glsifplural\@firstoftwo
6405 \let\glscapscase\@secondofthree
6406 \let\glsinsert\@empty
6407 \def\glscustomtext{%
6408     \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
6409     \ifglsxtrinsertinside\else#3\fi
6410 }%
6411 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6412 }%
6413 \glspostlinkhook
6414 }

```

\GLSxtrshortpl

```

6415 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}

    Define the un-starred form. Need to determine if there is a final optional argument
6416 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6417     \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6418 }

    Read in the final optional argument:
6419 \def\@GLSxtrshortpl#1#2[#3] {%
6420     \glsdoifexists{#2}%
6421     {%
6422         \glssetabbrvfmt{\glscategory{#2}}%
6423         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6424         \let\glxtrifwasfirstuse\@secondoftwo
6425         \let\glsifplural\@firstoftwo
6426         \let\glscapscase\@thirdofthree
6427         \let\glsinsert\@empty
6428         \def\glscustomtext{%
6429             \mfirstucMakeUppercase
6430             {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
6431             \ifglsxtrinsertinside\else#3\fi
6432         }%
6433     }%
6434     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6435 }%
6436 \glspostlinkhook
6437 }

```

Plural long forms:

\glxtrlongpl

```

6438 \newrobustcmd*{\glxtrlongpl}{\@gls@hyp@opt\ns@glxtrlongpl}

    Define the un-starred form. Need to determine if there is a final optional argument
6439 \newcommand*{\ns@glxtrlongpl}[2] [] {%
6440     \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2} []}%
6441 }

```

Read in the final optional argument:

```
6442 \def\@glxstrlongpl#1#2[#3]{%
6443   \glstoifexists{#2}%
6444   {%
6445     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6446     \let\glxstrifwasfirstuse\@secondoftwo
6447     \let\gl@ifplural\@firstoftwo
6448     \let\glscapscase\@firstofthree
6449     \let\glinsert\@empty
6450     \def\glscustomtext{%
6451       \glslongfont{\glaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
6452       \ifglxstrinsertinside\else#3\fi
6453     }%
6454     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6455   }%
6456   \glspostlinkhook
6457 }
```

\Glsxtrlongpl

```
6458 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\ns@Glsxtrlongpl}
  Define the un-starred form. Need to determine if there is a final optional argument
6459 \newcommand*{\ns@Glsxtrlongpl}[2][]{%
6460   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
6461 }
```

Read in the final optional argument:

```
6462 \def\@Glsxtrlongpl#1#2[#3]{%
6463   \glstoifexists{#2}%
6464   {%
6465     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6466     \let\glxstrifwasfirstuse\@secondoftwo
6467     \let\gl@ifplural\@firstoftwo
6468     \let\glscapscase\@secondofthree
6469     \let\glinsert\@empty
6470     \def\glscustomtext{%
6471       \glslongfont{\Glsaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
6472       \ifglxstrinsertinside\else#3\fi
6473     }%
6474     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6475   }%
6476   \glspostlinkhook
6477 }
```

\GLSxtrlongpl

```
6478 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
  Define the un-starred form. Need to determine if there is a final optional argument
6479 \newcommand*{\ns@GLSxtrlongpl}[2][]{%
6480   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2}[]}%
6481 }
```

Read in the final optional argument:

```
6482 \def\@GLSxtrlongpl#1#2[#3]{%
6483   \glsdoifexists{#2}%
6484   {%
6485     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6486     \let\glxtrifwasfirstuse\@secondoftwo
6487     \let\glsifplural\@firstoftwo
6488     \let\glscapscase\@thirdofthree
6489     \let\glinsert\@empty
6490     \def\glscustomtext{%
6491       \mfirstucMakeUppercase
6492       {\glslongfont{\glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6493       \ifglxtrininsertinside\else#3\fi
6494     }%
6495   }%
6496   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6497 }%
6498 \glspostlinkhook
6499 }
```

`\glsetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```
6500 \newcommand*{\glsetabbrvfmt}[1]{%
6501   \ifcsdef{@glsabbrv@current@#1}%
6502   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6503   {\glxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6504 }
```

`glsuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```
6505 \newrobustcmd*{\glsuseabbrvfont}[2]{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}
```

`\glsuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```
6506 \newrobustcmd*{\glsuselongfont}[2]{\glsetabbrvfmt{#2}\glslongfont{#1}}
```

`sxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```
6507 \newcommand*{\glsxtrgenabbrvfmt}{%
6508   \ifdefempty\glscustomtext
6509   {%
6510     \ifglused\glslabel
6511     {%
```

Subsequent use:

```
6512     \glsifplural
6513     {%
```

Subsequent plural form:

```
6514     \glscapscase
6515     {%
```


Subsequent plural form, don't adjust case:

```
6516      \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6517      }%  
6518      {%
```

Subsequent plural form, make first letter upper case:

```
6519      \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6520      }%  
6521      {%
```

Subsequent plural form, all caps:

```
6522      \mfirstucMakeUppercase  
6523      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%  
6524      }%  
6525      }%  
6526      {%
```

Subsequent singular form

```
6527      \glscapscase  
6528      {%
```

Subsequent singular form, don't adjust case:

```
6529      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6530      }%  
6531      {%
```

Subsequent singular form, make first letter upper case:

```
6532      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6533      }%  
6534      {%
```

Subsequent singular form, all caps:

```
6535      \mfirstucMakeUppercase  
6536      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6537      }%  
6538      }%  
6539      }%  
6540      {%
```

First use:

```
6541      \glsifplural  
6542      {%
```

First use plural form:

```
6543      \glscapscase  
6544      {%
```

First use plural form, don't adjust case:

```
6545      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6546      }%  
6547      {%
```

First use plural form, make first letter upper case:

```
6548      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
6549      }%
6550      {%
```

First use plural form, all caps:

```
6551      \mfirstucMakeUppercase
6552      {\glsxtrfullplformat{\glslabel}{\glsinsert}}%
6553      }%
6554      }%
6555      {%
```

First use singular form

```
6556      \glscapscase
6557      {%
```

First use singular form, don't adjust case:

```
6558      \glsxtrfullformat{\glslabel}{\glsinsert}%
6559      }%
6560      {%
```

First use singular form, make first letter upper case:

```
6561      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6562      }%
6563      {%
```

First use singular form, all caps:

```
6564      \mfirstucMakeUppercase
6565      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6566      }%
6567      }%
6568      }%
6569      }%
6570      {%
```

User supplied text.

```
6571      \glscustomtext
6572      }%
6573 }
```

`trsubsequentfmt` Subsequent use format (singular no case change).

```
6574 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6575   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6576   \ifglsxtrininsertinside \else#2\fi
6577 }
6578 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

`subsequentplfmt` Subsequent use format (plural no case change).

```
6579 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6580   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6581   \ifglsxtrininsertinside \else#2\fi
```

```

6582 }
6583 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt
trsubsequentfmt Subsequent use format (singular, first letter uppercase).
6584 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6585   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
6586   \ifglxtrininsertinside \else#2\fi
6587 }
6588 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
subsequentplfmt Subsequent use format (plural, first letter uppercase).
6589 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6590   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
6591   \ifglxtrininsertinside \else#2\fi
6592 }
6593 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.6.1 Abbreviation Styles Setup

```

breviationstyle
6594 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6595   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6596   {%
6597     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6598   }%
6599   {%
6600     \ifcsstring{@glsabbrv@current@#1}{#2}%
6601     {%
6602       style already set.
6603       {%
6604         \def\@glxtr@dostylewarn{%
6605           \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6606           {%
6607             \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
6608               style has been switched \MessageBreak
6609               for category ‘#1’, \MessageBreak
6610               but there have already been entries \MessageBreak
6611               defined for this category. Unwanted \MessageBreak
6612               side-effects may result}}%
6613             \@endfortrue
6614           }%
6615           \@glxtr@dostylewarn
6616           \csdef{@glsabbrv@current@#1}{#2}%
6617           \glxtr@applyabbrvstyle{#2}%

```

```

6618     }%
6619 }%
6620 }

```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```

6621 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
6622   \csuse{@glsabbrv@dispstyle@setup@#1}%
6623   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6624 }

```

`\r@applyabbrvfmt` Only apply the style formats.

```

6625 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
6626   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6627 }

```

`\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.

```

6628 \newcommand*{\newabbreviationstyle}[3]{%
6629   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6630     {%
6631       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6632         defined}{}%
6633     }%
6634     {%
6635       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```

6636       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6637       #2}%
6638       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

6639       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6640       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6641       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6642       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glxtrsubsequentfmt` etc in case a style changes this.

```

6643       \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
6644       \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
6645       \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6646       \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6647       #3}%
6648     }%
6649 }

```

`\abbreviationstyle`

```

6650 \newcommand*{\renewabbreviationstyle}[3]{%
6651   \ifcsundef{@glsabbrv@dispstyle@setup@#1}

```

```

6652 {%
6653   \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6654 }%
6655 {%
6656   \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6657     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6658     #2}%
6659   \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6660     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6661     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6662     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6663     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6664     #3}%
6665   }%
6666 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

6667 \newcommand*{\letabbreviationstyle}[2]{%
6668   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6669   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6670 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

6671 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6672   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6673     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6674     \csuse{@glsabbrv@dispstyle@setup@#2}%
6675   }%
6676   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6677 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

6678 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6679   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6680     use ‘#2’ instead}%
6681 }

```

UseAbbrStyleSetup

```

6682 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6683   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%

```

```

6684 {%
6685     \PackageError{glossaries-extra}%
6686     {Unknown abbreviation style definitions ‘#1’}{}%
6687 }%
6688 {%
6689     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6690 }%
6691 }

```

seAbbrStyleFmts

```

6692 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
6693     \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6694     {%
6695         \PackageError{glossaries-extra}%
6696         {Unknown abbreviation style formats ‘#1’}{}%
6697     }%
6698     {%
6699         \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6700     }%
6701 }

```

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, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

xtrinsertinside Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```

6702 \newif\ifglsxtrinsertinside
6703 \glsxtrinsertinsidefalse

```

trlongshortname

```

6704 \newcommand*{\glsxtrlongshortname}{%
6705     \protect\glsabbrvfont{\the\glsshorttok}%
6706 }

```

long-short

```

6707 \newabbreviationstyle{long-short}%
6708 {%
6709     \renewcommand*{\CustomAbbreviationFields}{%
6710         name={\glsxtrlongshortname},
6711         sort={\the\glsshorttok},

```

```

6712 first={\protect\glsfirstlongfont{\the\glslongtok}%
6713 \protect\glsxtrfullsep{\the\glslabeltok}%
6714 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6715 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6716 \protect\glsxtrfullsep{\the\glslabeltok}%
6717 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

6718 plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6719 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

6720 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6721 \glsasattribute{\the\glslabeltok}{regular}%
6722 {%
6723 \glssetattribute{\the\glslabeltok}{regular}{false}%
6724 }%
6725 {}%
6726 }%
6727 }%
6728 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6729 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6730 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6731 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6732 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6733 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6734 \renewcommand*{\glsxtrfullformat}[2]{%
6735 \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6736 \ifglsxtrininsertinside\else##2\fi
6737 \glsxtrfullsep{##1}%
6738 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6739 }%
6740 \renewcommand*{\glsxtrfullplformat}[2]{%
6741 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6742 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6743 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6744 }%
6745 \renewcommand*{\Glsxtrfullformat}[2]{%
6746 \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6747 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6748 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6749 }%
6750 \renewcommand*{\Glsxtrfullplformat}[2]{%
6751 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6752 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6753 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6754 }%
6755 }

```

Set this as the default style for general abbreviations:

```
6756 \setabbreviationstyle{long-short}
```

ngshortdescsort

```
6757 \newcommand*{\glxtrlongshortdescsort}{%
6758   \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6759 }
```

ngshortdescname

```
6760 \newcommand*{\glxtrlongshortdescname}{%
6761   \protect\glslongfont{\the\glslongtok}
6762   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6763 }
```

long-short-desc User supplies description. The long form is included in the name.

```
6764 \newabbreviationstyle{long-short-desc}%
6765 {%
6766   \renewcommand*{\CustomAbbreviationFields}{%
6767     name={\glxtrlongshortdescname},
6768     sort={\glxtrlongshortdescsort},%
6769     first={\protect\glsfirstlongfont{\the\glslongtok}%
6770           \protect\glxtrfullsep{\the\glslabeltok}%
6771           \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6772     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6773                 \protect\glxtrfullsep{\the\glslabeltok}%
6774                 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
6775     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6776     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6777   }%
```

The text key should only have the short form.

```
6775     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6776     plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6777   }%
```

Unset the regular attribute if it has been set.

```
6778 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6779   \glshasattribute{\the\glslabeltok}{regular}%
6780   {%
6781     \glssetattribute{\the\glslabeltok}{regular}{false}%
6782   }%
6783   {}%
6784 }%
6785 }%
6786 {%
6787   \GlsXtrUseAbbrStyleFmts{long-short}%
6788 }
```

trshortlongname

```
6789 \newcommand*{\glxtrshortlongname}{%
6790   \protect\glsabbrvfont{\the\glsshorttok}%
6791 }
```


short-long Short form followed by long form in parenthesis on first use.

```
6792 \newabbreviationstyle{short-long}%
6793 {%
6794   \renewcommand*{\CustomAbbreviationFields}{%
6795     name={\glxtrshortlongname},
6796     sort={\the\glsshorttok},
6797     description={\the\glslongtok},%
6798     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6799       \protect\glxtrfullsep{\the\glslabeltok}%
6800       \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6801     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6802       \protect\glxtrfullsep{\the\glslabeltok}%
6803       \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6804     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6805 }
```

Unset the regular attribute if it has been set.

```
6805 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6806   \glshasattribute{\the\glslabeltok}{regular}%
6807   {%
6808     \glissetattribute{\the\glslabeltok}{regular}{false}%
6809   }%
6810   {}%
6811 }%
6812 }%
6813 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6814 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6815 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6816 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6817 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6818 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6819 \renewcommand*{\glxtrfullformat}[2]{%
6820   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6821   \ifglxtrininsertinside\else##2\fi
6822   \glxtrfullsep{##1}%
6823   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6824 }%
6825 \renewcommand*{\glxtrfullplformat}[2]{%
6826   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6827   \ifglxtrininsertinside\else##2\fi
6828   \glxtrfullsep{##1}%
6829   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6830 }%
6831 \renewcommand*{\Glsxtrfullformat}[2]{%
6832   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6833   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6834 }
```

```

6834 \glstrparen{\glfirstlongfont{\glaccesslong{##1}}}%
6835 }%
6836 \renewcommand*{\Glsxtrfullplformat}[2]{%
6837 \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6838 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6839 \glstrparen{\glfirstlongfont{\glaccesslongpl{##1}}}%
6840 }%
6841 }

```

ortlongdescsort

```

6842 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

6843 \newcommand*{\glxtrshortlongdescname}{%
6844 \protect\glsabbrvfont{\the\glsshorttok}
6845 \glstrparen{\protect\glslongfont{\the\glslongtok}}}%
6846 }

```

short-long-desc User supplies description. The long form is included in the name.

```

6847 \newabbreviationstyle{short-long-desc}%
6848 {%
6849 \renewcommand*{\CustomAbbreviationFields}{%
6850 name={\glxtrshortlongdescname},
6851 sort={\glxtrshortlongdescsort},
6852 first={\protect\glfirstabbrvfont{\the\glsshorttok}}%
6853 \protect\glxtrfullsep{\the\glslabeltok}}%
6854 \glstrparen{\protect\glfirstlongfont{\the\glslongtok}}},%
6855 firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}}%
6856 \protect\glxtrfullsep{\the\glslabeltok}}%
6857 \glstrparen{\protect\glfirstlongfont{\the\glslongpltok}}},%
6858 text={\protect\glsabbrvfont{\the\glsshorttok}},%
6859 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6860 }%

```

Unset the regular attribute if it has been set.

```

6861 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6862 \glshasattribute{\the\glslabeltok}{regular}%
6863 {%
6864 \glissetattribute{\the\glslabeltok}{regular}{false}%
6865 }%
6866 {}%
6867 }%
6868 }%
6869 {%
6870 \GlsXtrUseAbbrStyleFmts{short-long}%
6871 }

```

ongfootnotefont Only used by the “footnote” styles.

```
6872 \newcommand*{\glfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6873 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote `\glxtrabbrvfootnote{<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`).

```
6874 \newcommand*{\glxtrabbrvfootnote}[2]{\footnote{#2}}
```

xtrfootnotename

```
6875 \newcommand*{\glxtrfootnotename}{%
6876   \protect\glsabbrvfont{\the\glsshorttok}%
6877 }
```

footnote Short form followed by long form in footnote on first use.

```
6878 \newabbreviationstyle{footnote}%
6879 {%
6880   \renewcommand*{\CustomAbbreviationFields}{%
6881     name={\glxtrfootnotename},
6882     sort={\the\glsshorttok},
6883     description={\the\glslongtok},%
6884     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6885       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6886       {\protect\glslongfootnotefont{\the\glslongtok}}},%
6887     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6888       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6889       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
6890     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6891 }
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
6891 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6892   \glsssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6893   \glshasattribute{\the\glslabeltok}{regular}%
6894   {%
6895     \glsssetattribute{\the\glslabeltok}{regular}{false}%
6896   }%
6897   {}%
6898 }
```

6899 }%
 6900 {%

In case the user wants to mix and match font styles, these are redefined here.

```
6901 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6902 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6903 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6904 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6905 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
6906 \renewcommand*\glxtrfullformat[2]{%
6907   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6908   \ifglxtrininsertinside\else##2\fi
6909   \protect\glxtrabbrvfootnote{##1}%
6910   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6911 }%
6912 \renewcommand*\glxtrfullplformat[2]{%
6913   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6914   \ifglxtrininsertinside\else##2\fi
6915   \protect\glxtrabbrvfootnote{##1}%
6916   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6917 }%
6918 \renewcommand*\Glsxtrfullformat[2]{%
6919   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6920   \ifglxtrininsertinside\else##2\fi
6921   \protect\glxtrabbrvfootnote{##1}%
6922   {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6923 }%
6924 \renewcommand*\Glsxtrfullplformat[2]{%
6925   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6926   \ifglxtrininsertinside\else##2\fi
6927   \protect\glxtrabbrvfootnote{##1}%
6928   {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6929 }%
```

The first use full form and the inline full form use the short (long) style.

```
6930 \renewcommand*\glxtrininlinefullformat[2]{%
6931   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6932   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6933   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6934 }%
6935 \renewcommand*\glxtrininlinefullplformat[2]{%
6936   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6937   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6938   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6939 }%
6940 \renewcommand*\Glsxtrininlinefullformat[2]{%
6941   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6942   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6943   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6944 }
```

```

6944 }%
6945 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6946   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6947   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6948   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6949 }%
6950 }

```

short-footnote

```

6951 \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.

```

6952 \newabbreviationstyle{postfootnote}%
6953 {%
6954   \renewcommand*{\CustomAbbreviationFields}{%
6955     name={\glxtrfootnotename},
6956     sort={\the\glsshorttok},
6957     description={\the\glslongtok},%
6958     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6959     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6960     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.

```

6961 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6962   \csdef{glxtrpostlink\glscategorylabel}{%
6963     \glxtrifwasfirstuse
6964     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6965       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
6966       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
6967     }%
6968   }%
6969 }%
6970 \glshasattribute{\the\glslabeltok}{regular}%
6971 {%
6972   \glissetattribute{\the\glslabeltok}{regular}{false}%
6973 }%
6974 {}%
6975 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

6976 \renewcommand*{\glxtrsetupfulldefs}{%
6977   \let\glxtrifwasfirstuse\@secondoftwo

```

```

6978 }%
6979 }%
6980 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6981 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6982 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6983 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6984 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6985 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6986 \renewcommand*\glxtrfullformat[2]{%
6987   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6988   \ifglxtrininsertinside\else##2\fi
6989 }%
6990 \renewcommand*\glxtrfullplformat[2]{%
6991   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6992   \ifglxtrininsertinside\else##2\fi
6993 }%
6994 \renewcommand*\Glsxtrfullformat[2]{%
6995   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6996   \ifglxtrininsertinside\else##2\fi
6997 }%
6998 \renewcommand*\Glsxtrfullplformat[2]{%
6999   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7000   \ifglxtrininsertinside\else##2\fi
7001 }%

```

The first use full form and the inline full form use the short (long) style.

```

7002 \renewcommand*\glxtrininlinefullformat[2]{%
7003   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7004   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7005   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7006 }%
7007 \renewcommand*\glxtrininlinefullplformat[2]{%
7008   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7009   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7010   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7011 }%
7012 \renewcommand*\Glsxtrininlinefullformat[2]{%
7013   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7014   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7015   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7016 }%
7017 \renewcommand*\Glsxtrininlinefullplformat[2]{%
7018   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7019   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7020   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7021 }%
7022 }

```

rt-postfootnote

```
7023 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

shortnolongname

```
7024 \newcommand*{\glxtrshortnolongname}{%
7025   \protect\glsabbrvfont{\the\glsshorttok}%
7026 }
```

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.

```
7027 \newabbreviationstyle{short}%
7028 {%
7029   \renewcommand*{\CustomAbbreviationFields}{%
7030     name={\glxtrshortnolongname},
7031     sort={\the\glsshorttok},
7032     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7033     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7034     text={\protect\glsabbrvfont{\the\glsshorttok}},
7035     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7036     description={\the\glslongtok}}%
7037   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7038     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7039 }%
7040 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7041 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7042 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7043 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7044 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7045 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7046 \renewcommand*{\glxtrinlinefullformat}[2]{%
7047   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7048   \ifglxtrininsertinside##2\fi}%
7049   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7050   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7051 }%
7052 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7053   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
7054   \ifglxtrininsertinside##2\fi}%
7055   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7056   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7057 }%
7058 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7059   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
```

```

7060     \ifglxtrinsertinside##2\fi}%
7061     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7062     \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7063 }%
7064 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7065     \protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}%
7066         \ifglxtrinsertinside##2\fi}%
7067     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7068     \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7069 }%

```

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.

```

7070 \renewcommand*{\glxtrfullformat}[2]{%
7071     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7072     \ifglxtrinsertinside\else##2\fi
7073 }%
7074 \renewcommand*{\glxtrfullplformat}[2]{%
7075     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7076     \ifglxtrinsertinside\else##2\fi
7077 }%
7078 \renewcommand*{\Glsxtrfullformat}[2]{%
7079     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7080     \ifglxtrinsertinside\else##2\fi
7081 }%
7082 \renewcommand*{\Glsxtrfullplformat}[2]{%
7083     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7084     \ifglxtrinsertinside\else##2\fi
7085 }%
7086 }

```

Set this as the default style for acronyms:

```

7087 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7088 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7089 \newabbreviationstyle{short-nolong-noreg}%
7090 {%
7091     \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7092 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7093     \glshasattribute{\the\glslabeltok}{regular}%
7094     {%
7095         \glissetattribute{\the\glslabeltok}{regular}{false}%
7096     }%
7097     {}%
7098 }%

```



```

7099 }%
7100 {%
7101   \GlsXtrUseAbbrStyleFmts{short-nolong}%
7102 }

```

trshortdescname

```

7103 \newcommand*{\glxtrshortdescname}{%
7104   \protect\glsabbrvfont{\the\glsshorttok}%
7105 }

```

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.

```

7106 \newabbreviationstyle{short-desc}%
7107 {%
7108   \renewcommand*{\CustomAbbreviationFields}{%
7109     name={\glxtrshortdescname},
7110     sort={\the\glsshorttok},
7111     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7112     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7113     text={\protect\glsabbrvfont{\the\glsshorttok}},
7114     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7115     description={\the\glslongtok}}%
7116   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7117     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7118 }%
7119 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7120 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7121 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7122 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7123 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7124 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7125 \renewcommand*{\glxtrinlinefullformat}[2]{%
7126   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7127   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7128   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7129 }%
7130 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7131   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7132   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7133   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7134 }%
7135 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7136   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7137   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7138   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7139 }%

```

```

7140 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7141   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7142   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7143   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7144 }%

```

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.

```

7145 \renewcommand*{\glsxtrfullformat}[2]{%
7146   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7147   \ifglxtrinsertinside\else##2\fi
7148 }%
7149 \renewcommand*{\glsxtrfullplformat}[2]{%
7150   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7151   \ifglxtrinsertinside\else##2\fi
7152 }%
7153 \renewcommand*{\Glsxtrfullformat}[2]{%
7154   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7155   \ifglxtrinsertinside\else##2\fi
7156 }%
7157 \renewcommand*{\Glsxtrfullplformat}[2]{%
7158   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7159   \ifglxtrinsertinside\else##2\fi
7160 }%
7161 }

```

ort-nolong-desc

```

7162 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7163 \newabbreviationstyle{short-nolong-desc-noreg}%
7164 {%
7165   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
7166   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7167     \glshasattribute{\the\glslabeltok}{regular}%
7168     {%
7169       \glssetattribute{\the\glslabeltok}{regular}{false}%
7170     }%
7171     {}%
7172   }%
7173 }%
7174 {%
7175   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7176 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7177 \newabbreviationstyle{nolong-short}%
7178 {%
7179   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7180 }%
7181 {%
7182   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7183 \renewcommand*{\glxtrinlinefullformat}[2]{%
7184   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7185   \ifglxtrininsertinside##2\fi}%
7186   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7187   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7188 }%
7189 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7190   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7191   \ifglxtrininsertinside##2\fi}%
7192   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7193   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7194 }%
7195 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7196   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7197   \ifglxtrininsertinside##2\fi}%
7198   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7199   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7200 }%
7201 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7202   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7203   \ifglxtrininsertinside##2\fi}%
7204   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7205   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7206 }%
7207 }

```

nolong-short-noreg Like nolong-short but doesn't set the regular attribute.

```

7208 \newabbreviationstyle{nolong-short-noreg}%
7209 {%
7210   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7211 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7212   \glshasattribute{\the\glslabeltok}{regular}%
7213   {%
7214     \glissetattribute{\the\glslabeltok}{regular}{false}%
7215   }%
7216   {}%
7217 }%
7218 }%
7219 {%
7220   \GlsXtrUseAbbrStyleFmts{nolong-short}%

```

7221 }

noshortdescname

```
7222 \newcommand*{\glxtrlongnoshortdescname}{%
7223   \protect\glslongfont{\the\glslongtok}%
7224 }
```

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.

```
7225 \newabbreviationstyle{long-desc}%
7226 {%
7227   \renewcommand*{\CustomAbbreviationFields}{%
7228     name={\glxtrlongnoshortdescname},
7229     sort={\the\glslongtok},
7230     first={\protect\glsfirstlongfont{\the\glslongtok}},
7231     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7232     text={\glslongfont{\the\glslongtok}},
7233     plural={\glslongfont{\the\glslongpltok}}}%
7234 }%
7235 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7236   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7237 }%
7238 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7239 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7240 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7241 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7242 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7243 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7244 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7245   \glslongfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7246   \ifglxtrininsertinside \else##2\fi
7247 }%
7248 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7249   \glslongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7250   \ifglxtrininsertinside \else##2\fi
7251 }%
7252 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7253   \glslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7254   \ifglxtrininsertinside \else##2\fi
7255 }%
7256 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7257   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7258   \ifglxtrininsertinside \else##2\fi
7259 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```

7260 \renewcommand*{\glxtrinlinefullformat}[2]{%
7261   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7262   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7263   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7264 }%
7265 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7266   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7267   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7268   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7269 }%
7270 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7271   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7272   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7273   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7274 }%
7275 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7276   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7277   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7278   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7279 }%

```

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.

```

7280 \renewcommand*{\glxtrfullformat}[2]{%
7281   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7282   \ifglxtrinsertinside\else##2\fi
7283 }%
7284 \renewcommand*{\glxtrfullplformat}[2]{%
7285   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7286   \ifglxtrinsertinside\else##2\fi
7287 }%
7288 \renewcommand*{\Glsxtrfullformat}[2]{%
7289   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7290   \ifglxtrinsertinside\else##2\fi
7291 }%
7292 \renewcommand*{\Glsxtrfullplformat}[2]{%
7293   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7294   \ifglxtrinsertinside\else##2\fi
7295 }%
7296 }

```

`ng-noshort-desc` Provide a synonym that matches similar styles.

```
7297 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

`short-desc-noreg` Like `long-noshort-desc` but doesn't set the regular attribute.

```

7298 \newabbreviationstyle{long-noshort-desc-noreg}%
7299 {%
7300   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

```

Unset the regular attribute if it has been set.

```

7301 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7302   \glshasattribute{\the\glslabeltok}{regular}%
7303   {%
7304     \glsselattribute{\the\glslabeltok}{regular}{false}%
7305   }%
7306   {}%
7307 }%
7308 }%
7309 {%
7310   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7311 }

```

longnoshortname

```

7312 \newcommand*{\glsxtrlongnoshortname}{%
7313   \protect\glsabbrvfont{\the\glsshorttok}%
7314 }

```

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.

```

7315 \newabbreviationstyle{long}%
7316 {%
7317   \renewcommand*{\CustomAbbreviationFields}{%
7318     name={\glsxtrlongnoshortname},
7319     sort={\the\glsshorttok},
7320     first={\protect\glsfirstlongfont{\the\glslongtok}},
7321     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7322     text={\glslongfont{\the\glslongtok}},
7323     plural={\glslongfont{\the\glslongpltok}},%
7324     description={\the\glslongtok}%
7325   }%
7326   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7327     \glsselattribute{\the\glslabeltok}{regular}{true}}%
7328 }%
7329 {%
7330   \GlsXtrUseAbbrStyleFmts{long-desc}%
7331 }

```

long-noshort Provide a synonym that matches similar styles.

```

7332 \letabbreviationstyle{long-noshort}{long}

```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```

7333 \newabbreviationstyle{long-noshort-noreg}%
7334 {%
7335   \GlsXtrUseAbbrStyleSetup{long-noshort}%

```

Unset the regular attribute if it has been set.

```

7336 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

7337 \glshasattribute{\the\glslabeltok}{regular}%
7338 {%
7339 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7340 }%
7341 {}%
7342 }%
7343 }%
7344 {%
7345 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7346 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
7347 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

`\glssabbrvscfont` Added for consistent naming.

```
7348 \newcommand*{\glssabbrvscfont}{\glsxtrscfont}
```

`\glsxtrfirstscfont` Maintained for backward-compatibility.

```
7349 \newcommand*{\glsxtrfirstscfont}[1]{\glssabbrvscfont{#1}}
```

`\glssfirstabbrvscfont` Added for consistent naming.

```
7350 \newcommand*{\glssfirstabbrvscfont}{\glsxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix`

```
7351 \newcommand*{\glsxtrscsuffix}{\glstextup{\glsxtrabbrvpluralsuffix}}
```

`long-short-sc`

```

7352 \newabbreviationstyle{long-short-sc}%
7353 {%
7354 \renewcommand*{\CustomAbbreviationFields}{%
7355   name={\glsxtrlongshortname},
7356   sort={\the\glssshorttok},
7357   first={\protect\glssfirstlongdefaultfont{\the\glslongtok}%
7358     \protect\glsxtrfullsep{\the\glslabeltok}%
7359     \glsxtrparen{\protect\glssfirstabbrvscfont{\the\glssshorttok}}},%
7360   firstplural={\protect\glssfirstlongdefaultfont{\the\glslongpltok}%
7361     \protect\glsxtrfullsep{\the\glslabeltok}%
7362     \glsxtrparen{\protect\glssfirstabbrvscfont{\the\glssshortpltok}}},%
7363   plural={\protect\glssabbrvscfont{\the\glssshortpltok}}},%
7364   description={\the\glslongtok}}%
7365 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7366   \glshasattribute{\the\glslabeltok}{regular}%
7367   {%

```

```

7368     \glsetattribute{\the\glslabeltok}{regular}{false}%
7369   }%
7370   {}%
7371 }%
7372 }%
7373 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7374 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
7375 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
7376 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

7377 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7378 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7379 \renewcommand*\glsxtrfullformat[2]{%
7380   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7381   \ifglsxtrinsertinside\else##2\fi
7382   \glsxtrfullsep{##1}%
7383   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7384 }%
7385 \renewcommand*\glsxtrfullplformat[2]{%
7386   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7387   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7388   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7389 }%
7390 \renewcommand*\Glsxtrfullformat[2]{%
7391   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7392   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7393   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7394 }%
7395 \renewcommand*\Glsxtrfullplformat[2]{%
7396   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7397   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7398   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7399 }%
7400 }

```

g-short-sc-desc

```

7401 \newabbreviationstyle{long-short-sc-desc}%
7402 {%
7403   \renewcommand*\CustomAbbreviationFields{%
7404     name={\glsxtrlongshortdescname},
7405     sort={\glsxtrlongshortdescsort},%
7406     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7407       \protect\glsxtrfullsep{\the\glslabeltok}%
7408       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7409     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%

```



```

7410 \protect\glxtrfullsep{\the\glslabeltok}%
7411 \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7412 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7413 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7414 }%

```

Unset the regular attribute if it has been set.

```

7415 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7416 \glshasattribute{\the\glslabeltok}{regular}%
7417 {%
7418 \glissetattribute{\the\glslabeltok}{regular}{false}%
7419 }%
7420 {}%
7421 }%
7422 }%
7423 {%

```

As long-short-sc style:

```

7424 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7425 }

```

Now the short (long) version

```

7426 \newabbreviationstyle{short-sc-long}%
7427 {%
7428 \renewcommand*{\CustomAbbreviationFields}{%
7429 name={\glxtrshortlongname},
7430 sort={\the\glsshorttok},
7431 description={\the\gslongtok},%
7432 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7433 \protect\glxtrfullsep{\the\glslabeltok}%
7434 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
7435 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7436 \protect\glxtrfullsep{\the\glslabeltok}%
7437 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
7438 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7439 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7440 \glshasattribute{\the\glslabeltok}{regular}%
7441 {%
7442 \glissetattribute{\the\glslabeltok}{regular}{false}%
7443 }%
7444 {}%
7445 }%
7446 }%
7447 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7448 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7449 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvscfont{##1}}%
7450 \renewcommand*{\glsfirstabbrvfont[1]}{\glsfirstabbrvscfont{##1}}%

```

```

7451 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7452 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7453 \renewcommand*{\glsxtrfullformat}[2]{%
7454   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7455   \ifglsxtrininsertinside\else##2\fi
7456   \glsxtrfullsep{##1}%
7457   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7458 }%
7459 \renewcommand*{\glsxtrfullplformat}[2]{%
7460   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7461   \ifglsxtrininsertinside\else##2\fi
7462   \glsxtrfullsep{##1}%
7463   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7464 }%
7465 \renewcommand*{\Glsxtrfullformat}[2]{%
7466   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7467   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7468   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7469 }%
7470 \renewcommand*{\Glsxtrfullplformat}[2]{%
7471   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7472   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7473   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7474 }%
7475 }

```

As before but user provides description

```

7476 \newabbreviationstyle{short-sc-long-desc}%
7477 {%
7478   \renewcommand*{\CustomAbbreviationFields}{%
7479     name={\glsxtrshortlongdescname},
7480     sort={\glsxtrshortlongdescsort},
7481     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7482       \protect\glsxtrfullsep{\the\glslabeltok}%
7483       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7484     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7485       \protect\glsxtrfullsep{\the\glslabeltok}%
7486       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7487     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7488     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7489   }%

```

Unset the regular attribute if it has been set.

```

7490 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7491   \glshasattribute{\the\glslabeltok}{regular}%
7492   {%
7493     \glssetattribute{\the\glslabeltok}{regular}{false}%
7494   }%

```

```

7495 {}%
7496 }%
7497 }%
7498 {%
    As short-sc-long style:
7499 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
7500 }

```

short-sc

```

7501 \newabbreviationstyle{short-sc}%
7502 {%
7503   \renewcommand*{\CustomAbbreviationFields}{%
7504     name={\glxtrshortnolongname},
7505     sort={\the\glsshorttok},
7506     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7507     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7508     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7509     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7510     description={\the\glslongtok}}%
7511   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7512     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7513 }%
7514 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7515 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7516 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7517 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7518 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7519 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7520 \renewcommand*{\glxtrinlinefullformat}[2]{%
7521   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
7522   \ifglxtrininsertinside##2\fi}%
7523   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7524   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7525 }%
7526 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7527   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
7528   \ifglxtrininsertinside##2\fi}%
7529   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7530   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7531 }%

7532 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7533   \protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}%
7534   \ifglxtrininsertinside##2\fi}%
7535   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7536   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%

```

```

7537 }%
7538 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7539   \protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}%
7540   \ifglxtrinsertinside##2\fi}%
7541   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7542   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7543 }%

```

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.

```

7544 \renewcommand*{\glxtrfullformat}[2]{%
7545   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7546   \ifglxtrinsertinside\else##2\fi
7547 }%
7548 \renewcommand*{\glxtrfullplformat}[2]{%
7549   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7550   \ifglxtrinsertinside\else##2\fi
7551 }%
7552 \renewcommand*{\Glsxtrfullformat}[2]{%
7553   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7554   \ifglxtrinsertinside\else##2\fi
7555 }%
7556 \renewcommand*{\Glsxtrfullplformat}[2]{%
7557   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7558   \ifglxtrinsertinside\else##2\fi
7559 }%
7560 }

```

short-sc-nolong

```

7561 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7562 \newabbreviationstyle{short-sc-desc}%
7563 {%
7564   \renewcommand*{\CustomAbbreviationFields}{%
7565     name={\glxtrshortdescname},
7566     sort={\the\glsshorttok},
7567     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7568     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7569     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7570     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7571     description={\the\glslongtok}}%
7572   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7573     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7574 }%
7575 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7576 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7577 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%

```

```

7578 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
7579 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7580 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7581 \renewcommand*\glsxtrinlinefullformat}[2]{%
7582   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7583   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7584   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7585 }%
7586 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7587   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7588   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7589   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7590 }%
7591 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7592   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7593   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7594   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7595 }%
7596 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7597   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7598   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7599   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7600 }%

```

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.

```

7601 \renewcommand*\glsxtrfullformat}[2]{%
7602   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7603   \ifglsxtrininsertinside\else##2\fi
7604 }%
7605 \renewcommand*\glsxtrfullplformat}[2]{%
7606   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7607   \ifglsxtrininsertinside\else##2\fi
7608 }%
7609 \renewcommand*\Glsxtrfullformat}[2]{%
7610   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7611   \ifglsxtrininsertinside\else##2\fi
7612 }%
7613 \renewcommand*\Glsxtrfullplformat}[2]{%
7614   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7615   \ifglsxtrininsertinside\else##2\fi
7616 }%
7617 }

```

-sc-nolong-desc

```

7618 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7619 \newabbreviationstyle{nolong-short-sc}%
7620 {%
7621   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
7622 }%
7623 {%
7624   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7625 \renewcommand*{\glxtrinlinefullformat}[2]{%
7626   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%
7627   \ifglxtrinsertinside##2\fi}%
7628 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7629 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7630 }%
7631 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7632   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%
7633   \ifglxtrinsertinside##2\fi}%
7634 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7635 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7636 }%
7637 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7638   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%
7639   \ifglxtrinsertinside##2\fi}%
7640 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7641 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7642 }%
7643 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7644   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%
7645   \ifglxtrinsertinside##2\fi}%
7646 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7647 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7648 }%
7649 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7650 \newabbreviationstyle{long-noshort-sc}%
7651 {%
7652   \renewcommand*{\CustomAbbreviationFields}{%
7653     name={\glxtrlongnoshortname},
7654     sort={\the\glsshorttok},
7655     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7656     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7657     text={\protect\glslongdefaultfont{\the\glslongtok}},
7658     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7659     description={\the\glslongtok}%
7660   }%
7661   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7662     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7663 }%

```

7664 {%

Use smallcaps and adjust the plural suffix to revert to upright.

```
7665 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7666 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7667 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7668 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7669 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7670 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7671   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7672   \ifglxtrininsertinside \else##2\fi
7673 }%
7674 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7675   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7676   \ifglxtrininsertinside \else##2\fi
7677 }%
7678 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7679   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7680   \ifglxtrininsertinside \else##2\fi
7681 }%
7682 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7683   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7684   \ifglxtrininsertinside \else##2\fi
7685 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
7686 \renewcommand*{\glxtrinlinefullformat}[2]{%
7687   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7688   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7689   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7690 }%
7691 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7692   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7693   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7694   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7695 }%
7696 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7697   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7698   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7699   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7700 }%
7701 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7702   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7703   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7704   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7705 }%
```

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.

```

7706 \renewcommand*{\glxtrfullformat}[2]{%
7707   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7708   \ifglxtrinsertinside\else##2\fi
7709 }%
7710 \renewcommand*{\glxtrfullplformat}[2]{%
7711   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7712   \ifglxtrinsertinside\else##2\fi
7713 }%
7714 \renewcommand*{\Glsxtrfullformat}[2]{%
7715   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7716   \ifglxtrinsertinside\else##2\fi
7717 }%
7718 \renewcommand*{\Glsxtrfullplformat}[2]{%
7719   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7720   \ifglxtrinsertinside\else##2\fi
7721 }%
7722 }

```

long-sc Backward compatibility:

```

7723 \@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.

```

7724 \newabbreviationstyle{long-noshort-sc-desc}%
7725 {%
7726   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7727 }%
7728 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7729 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7730 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7731 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7732 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7733 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7734 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7735   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7736   \ifglxtrinsertinside \else##2\fi
7737 }%
7738 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7739   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7740   \ifglxtrinsertinside \else##2\fi
7741 }%
7742 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7743   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7744   \ifglxtrinsertinside \else##2\fi
7745 }%
7746 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```



```

7747 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7748 \ifglxtrinsertinside \else##2\fi
7749 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7750 \renewcommand*{\glxtrinlinefullformat}[2]{%
7751 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7752 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7753 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7754 }%
7755 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7756 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7757 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7758 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7759 }%
7760 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7761 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7762 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7763 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7764 }%
7765 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7766 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7767 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7768 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7769 }%

```

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.

```

7770 \renewcommand*{\glxtrfullformat}[2]{%
7771 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7772 \ifglxtrinsertinside\else##2\fi
7773 }%
7774 \renewcommand*{\glxtrfullplformat}[2]{%
7775 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7776 \ifglxtrinsertinside\else##2\fi
7777 }%
7778 \renewcommand*{\Glsxtrfullformat}[2]{%
7779 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7780 \ifglxtrinsertinside\else##2\fi
7781 }%
7782 \renewcommand*{\Glsxtrfullplformat}[2]{%
7783 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7784 \ifglxtrinsertinside\else##2\fi
7785 }%
7786 }

```

long-desc-sc Backward compatibility:

```

7787 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

7788 \newabbreviationstyle{short-sc-footnote}%

7789 {%

7790 \renewcommand*{\CustomAbbreviationFields}{%

7791 name={\glxtrfootnotename},

7792 sort={\the\glsshorttok},

7793 description={\the\glslongtok},%

7794 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%

7795 \protect\glxtrabbrvfootnote{\the\glslabeltok}%

7796 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%

7797 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%

7798 \protect\glxtrabbrvfootnote{\the\glslabeltok}%

7799 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

7800 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.

7801 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

7802 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%

7803 \glsattribute{\the\glslabeltok}{regular}%

7804 {%

7805 \glssetattribute{\the\glslabeltok}{regular}{false}%

7806 }%

7807 {}%

7808 }%

7809 }%

7810 {%

Use smallcaps and adjust the plural suffix to revert to upright.

7811 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%

7812 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%

7813 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%

7814 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%

7815 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

The full format displays the short form followed by the long form as a footnote.

7816 \renewcommand*{\glxtrfullformat}[2]{%

7817 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%

7818 \ifglxtrininsertinside\else##2\fi

7819 \protect\glxtrabbrvfootnote{##1}%

7820 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%

7821 }%

7822 \renewcommand*{\glxtrfullplformat}[2]{%

7823 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

7824 \ifglxtrininsertinside\else##2\fi

7825 \protect\glxtrabbrvfootnote{##1}%

7826 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%

7827 }%

7828 \renewcommand*{\Glsxtrfullformat}[2]{%

7829 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%

7830 \ifglxtrininsertinside\else##2\fi

7831 \protect\glxtrabbrvfootnote{##1}%

```

7832     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7833 }%
7834 \renewcommand*{\Glsxtrfullplformat}[2]{%
7835   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7836   \ifglxtrinsertinside\else##2\fi
7837   \protect\glxtrabbrvfootnote{##1}%
7838   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7839 }%

```

The first use full form and the inline full form use the short (long) style.

```

7840 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7841   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7842   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7843   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7844 }%
7845 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7846   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7847   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7848   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7849 }%
7850 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7851   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7852   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7853   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7854 }%
7855 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7856   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7857   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7858   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7859 }%
7860 }

```

footnote-sc Backward compatibility:

```

7861 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

7862 \newabbreviationstyle{short-sc-postfootnote}%
7863 {%
7864   \renewcommand*{\CustomAbbreviationFields}{%
7865     name={\glxtrfootnotename},
7866     sort={\the\glsshorttok},
7867     description={\the\glslongtok},%
7868     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7869     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7870     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.

```

7871 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7872   \csdef{glxtrpostlink\glscategorylabel}{%

```

```

7873 \glxtrifwasfirstuse
7874 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7875 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7876 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7877 }%
7878 {%
7879 }%
7880 \glshasattribute{\the\glslabeltok}{regular}%
7881 {%
7882 \glissetattribute{\the\glslabeltok}{regular}{false}%
7883 }%
7884 {}%
7885 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7886 \renewcommand*{\glxtrsetupfulldefs}{%
7887 \let\glxtrifwasfirstuse\@secondoftwo
7888 }%
7889 }%
7890 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7891 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7892 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
7893 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
7894 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
7895 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7896 \renewcommand*{\glxtrfullformat}[2]{%
7897 \glfirstabbrvscfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7898 \ifglxtrininsertinside\else##2\fi
7899 }%
7900 \renewcommand*{\glxtrfullplformat}[2]{%
7901 \glfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7902 \ifglxtrininsertinside\else##2\fi
7903 }%
7904 \renewcommand*{\Glsxtrfullformat}[2]{%
7905 \glfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7906 \ifglxtrininsertinside\else##2\fi
7907 }%
7908 \renewcommand*{\Glsxtrfullplformat}[2]{%
7909 \glfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7910 \ifglxtrininsertinside\else##2\fi
7911 }%

```

The first use full form and the inline full form use the short (long) style.

```

7912 \renewcommand*{\glxtrinlinefullformat}[2]{%
7913   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7914   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7915   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7916 }%
7917 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7918   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7919   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7920   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7921 }%
7922 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7923   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7924   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7925   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7926 }%
7927 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7928   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7929   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7930   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7931 }%
7932 }

```

postfootnote-sc Backward compatibility:

```
7933 \@glxtr@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:

`\glxtrsmfont` Maintained for backward compatibility.

```
7934 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

`\glsabbrvsmfont` Added for consistent naming.

```
7935 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}
```

`\glxtrfirstsmfont` Maintained for backward compatibility.

```
7936 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}
```

`\glfirstabbrvsmfont` Added for consistent naming.

```
7937 \newcommand*{\glfirstabbrvsmfont}{\glxtrfirstsmfont}
```

and for the default short form suffix:

`\glxtrsmsuffix`

```
7938 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}
```

long-short-sm

```
7939 \newabbreviationstyle{long-short-sm}%
7940 {%
7941   \renewcommand*{\CustomAbbreviationFields}{%
7942     name={\glxtrlongshortname},
7943     sort={\the\glsshorttok},
7944     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7945       \protect\glxtrfullsep{\the\glslabeltok}%
7946       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7947     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7948       \protect\glxtrfullsep{\the\glslabeltok}%
7949       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7950     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
7951     description={\the\glslongtok}}%
7952   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7953     \glshasattribute{\the\glslabeltok}{regular}%
7954     {%
7955       \glissetattribute{\the\glslabeltok}{regular}{false}%
7956     }%
7957   }%
7958 }%
7959 }%
7960 {%
7961   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7962   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7963   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%

```

Use the default long fonts.

```
7964 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7965 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
7966 \renewcommand*{\glxtrfullformat}[2]{%
7967   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7968   \ifglxtrininsertinside\else##2\fi
7969   \glxtrfullsep{##1}%
7970   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7971 }%
7972 \renewcommand*{\glxtrfullplformat}[2]{%
7973   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7974   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7975   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7976 }%
7977 \renewcommand*{\Glsxtrfullformat}[2]{%
7978   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7979   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7980   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7981 }%
7982 \renewcommand*{\Glsxtrfullplformat}[2]{%
7983   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

7984 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7985 \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
7986 }%
7987 }

```

g-short-sm-desc

```

7988 \newabbreviationstyle{long-short-sm-desc}%
7989 {%
7990 \renewcommand*{\CustomAbbreviationFields}{%
7991   name={\glxtrlongshortdescname},
7992   sort={\glxtrlongshortdescsort},%
7993   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7994     \protect\glxtrfullsep{\the\glslabeltok}%
7995     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7996   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7997     \protect\glxtrfullsep{\the\glslabeltok}%
7998     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7999   text={\protect\glssabbrvsmfont{\the\glsshorttok}},%
8000   plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}%
8001 }%

```

Unset the regular attribute if it has been set.

```

8002 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8003   \glshasattribute{\the\glslabeltok}{regular}%
8004   {%
8005     \glissetattribute{\the\glslabeltok}{regular}{false}%
8006   }%
8007   {}%
8008 }%
8009 }%
8010 {%

```

As long-short-sm style:

```

8011 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
8012 }

```

short-sm-long Now the short (long) version

```

8013 \newabbreviationstyle{short-sm-long}%
8014 {%
8015 \renewcommand*{\CustomAbbreviationFields}{%
8016   name={\glxtrshortlongname},
8017   sort={\the\glsshorttok},
8018   description={\the\glslongtok},%
8019   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8020     \protect\glxtrfullsep{\the\glslabeltok}%
8021     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8022   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8023     \protect\glxtrfullsep{\the\glslabeltok}%
8024     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8025   plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}%

```

Unset the regular attribute if it has been set.

```

8026 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8027   \glshasattribute{\the\glslabeltok}{regular}%
8028   {%
8029     \glsselattribute{\the\glslabeltok}{regular}{false}%
8030   }%
8031   {}%
8032 }%
8033 }%
8034 {%
8035 \renewcommand*{\glabbrvfont}[1]{\glabbrvsmfont{##1}}%
8036 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvsmfont{##1}}%
8037 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8038 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
8039 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8040 \renewcommand*{\glxtrfullformat}[2]{%
8041   \glfirstabbrvsmfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8042   \ifglxtrininsertinside\else##2\fi
8043   \glxtrfullsep{##1}%
8044   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
8045 }%
8046 \renewcommand*{\glxtrfullplformat}[2]{%
8047   \glfirstabbrvsmfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8048   \ifglxtrininsertinside\else##2\fi
8049   \glxtrfullsep{##1}%
8050   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8051 }%
8052 \renewcommand*{\Glsxtrfullformat}[2]{%
8053   \glfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8054   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8055   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
8056 }%
8057 \renewcommand*{\Glsxtrfullplformat}[2]{%
8058   \glfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8059   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8060   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8061 }%
8062 }

```

rt-sm-long-desc As before but user provides description

```

8063 \newabbreviationstyle{short-sm-long-desc}%
8064 {%
8065 \renewcommand*{\CustomAbbreviationFields}{%
8066   name={\glxtrshortlongdescname},
8067   sort={\glxtrshortlongdescsort},
8068   first={\protect\glfirstabbrvsmfont{\the\glsshorttok}}%
8069   \protect\glxtrfullsep{\the\glslabeltok}%

```



```

8070 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8071 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8072 \protect\glxtrfullsep{\the\glslabeltok}%
8073 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8074 text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
8075 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
8076 }%

```

Unset the regular attribute if it has been set.

```

8077 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8078 \glshasattribute{\the\glslabeltok}{regular}%
8079 {%
8080 \glssetattribute{\the\glslabeltok}{regular}{false}%
8081 }%
8082 {}%
8083 }%
8084 }%
8085 {%

```

As short-sm-long style:

```

8086 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8087 }

```

short-sm

```

8088 \newabbreviationstyle{short-sm}%
8089 {%
8090 \renewcommand*{\CustomAbbreviationFields}{%
8091 name={\glxtrshortnolongname},
8092 sort={\the\glsshorttok},
8093 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8094 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8095 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8096 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8097 description={\the\glslongtok}}%
8098 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8099 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8100 }%
8101 {%
8102 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8103 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8104 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8105 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8106 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8107 \renewcommand*{\glxtrinlinefullformat}[2]{%
8108 \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8109 \ifglxtrininsertinside##2\fi}%
8110 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8111 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%

```

```

8112 }%
8113 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8114   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}%
8115     \ifglxtrininsertinside##2\fi}%
8116   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8117   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8118 }%

8119 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8120   \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}%
8121     \ifglxtrininsertinside##2\fi}%
8122   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8123   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8124 }%

8125 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8126   \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}%
8127     \ifglxtrininsertinside##2\fi}%
8128   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8129   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8130 }%

```

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.

```

8131 \renewcommand*{\glxtrfullformat}[2]{%
8132   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8133   \ifglxtrininsertinside\else##2\fi
8134 }%
8135 \renewcommand*{\glxtrfullplformat}[2]{%
8136   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8137   \ifglxtrininsertinside\else##2\fi
8138 }%
8139 \renewcommand*{\Glsxtrfullformat}[2]{%
8140   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8141   \ifglxtrininsertinside\else##2\fi
8142 }%
8143 \renewcommand*{\Glsxtrfullplformat}[2]{%
8144   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8145   \ifglxtrininsertinside\else##2\fi
8146 }%
8147 }

```

short-sm-nolong

```
8148 \letabbreviationstyle{short-sm-nolong}{short-sm}
```

short-sm-desc

```

8149 \newabbreviationstyle{short-sm-desc}%
8150 {%
8151   \renewcommand*{\CustomAbbreviationFields}{%
8152     name={\glxtrshortdescname},

```

```

8153     sort={\the\glsshorttok},
8154     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8155     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8156     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8157     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8158     description={\the\glslongtok}}%
8159 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8160   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8161 }%
8162 {%
8163   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8164   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8165   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrmsuffix}%
8166   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8167   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8168   \renewcommand*{\glsxtrinlinefullformat}[2]{%
8169     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8170     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8171     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8172   }%
8173   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8174     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8175     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8176     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8177   }%
8178   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8179     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8180     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8181     \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8182   }%
8183   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8184     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8185     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8186     \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8187   }%

```

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.

```

8188   \renewcommand*{\glsxtrfullformat}[2]{%
8189     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8190     \ifglsxtrininsertinside\else##2\fi
8191   }%
8192   \renewcommand*{\glsxtrfullplformat}[2]{%
8193     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8194     \ifglsxtrininsertinside\else##2\fi
8195   }%
8196   \renewcommand*{\Glsxtrfullformat}[2]{%
8197     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

8198     \ifglxtrinsertinside\else##2\fi
8199 }%
8200 \renewcommand*{\Glsxtrfullplformat}[2]{%
8201     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8202     \ifglxtrinsertinside\else##2\fi
8203 }%
8204 }

```

-sm-nolong-desc

```

8205 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8206 \newabbreviationstyle{nolong-short-sm}%
8207 {%
8208     \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8209 }%
8210 {%
8211     \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8212 \renewcommand*{\glxtrinlinefullformat}[2]{%
8213     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8214         \ifglxtrinsertinside##2\fi}%
8215     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8216     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8217 }%
8218 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8219     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8220         \ifglxtrinsertinside##2\fi}%
8221     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8222     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8223 }%
8224 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8225     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8226         \ifglxtrinsertinside##2\fi}%
8227     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8228     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8229 }%
8230 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8231     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8232         \ifglxtrinsertinside##2\fi}%
8233     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8234     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8235 }%
8236 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

8237 \newabbreviationstyle{long-noshort-sm}%

```

```

8238 {%
8239   \renewcommand*{\CustomAbbreviationFields}{%
8240     name={\glxtrlongnosshortname},
8241     sort={\the\glssshorttok},
8242     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8243     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8244     text={\protect\glslongdefaultfont{\the\glslongtok}},
8245     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8246     description={\the\glslongtok}%
8247   }%
8248   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8249     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8250 }%
8251 {%
8252   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8253   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8254   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8255   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8256   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8257   \renewcommand*{\glxtrsubsequentfmt}[2]{%
8258     \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
8259     \ifglxtrininsertinside \else##2\fi
8260   }%
8261   \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8262     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
8263     \ifglxtrininsertinside \else##2\fi
8264   }%
8265   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8266     \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
8267     \ifglxtrininsertinside \else##2\fi
8268   }%
8269   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8270     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
8271     \ifglxtrininsertinside \else##2\fi
8272   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8273   \renewcommand*{\glxtrininlinefullformat}[2]{%
8274     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8275     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8276     \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8277   }%
8278   \renewcommand*{\glxtrininlinefullplformat}[2]{%
8279     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8280     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8281     \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8282   }%
8283   \renewcommand*{\Glsxtrininlinefullformat}[2]{%

```

```

8284 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8285 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8286 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8287 }%
8288 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8289 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8290 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8291 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8292 }%

```

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.

```

8293 \renewcommand*{\glxtrfullformat}[2]{%
8294 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8295 \ifglxtrinsertinside\else##2\fi
8296 }%
8297 \renewcommand*{\glxtrfullplformat}[2]{%
8298 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8299 \ifglxtrinsertinside\else##2\fi
8300 }%
8301 \renewcommand*{\Glsxtrfullformat}[2]{%
8302 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8303 \ifglxtrinsertinside\else##2\fi
8304 }%
8305 \renewcommand*{\Glsxtrfullplformat}[2]{%
8306 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8307 \ifglxtrinsertinside\else##2\fi
8308 }%
8309 }

```

long-sm Backward compatibility:

```

8310 \@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`.

```

8311 \newabbreviationstyle{long-noshort-sm-desc}%
8312 {%
8313 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8314 }%
8315 {%
8316 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8317 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8318 \renewcommand*\{abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8319 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8320 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8321 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8322 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8323 \ifglxtrinsertinside \else##2\fi

```

```

8324 }%
8325 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8326   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8327   \ifglxtrinsertinside \else##2\fi
8328 }%
8329 \renewcommand*{\Glsxtrsubsequentffmt}[2]{%
8330   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8331   \ifglxtrinsertinside \else##2\fi
8332 }%
8333 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8334   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8335   \ifglxtrinsertinside \else##2\fi
8336 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8337 \renewcommand*{\glxtrinlinefullformat}[2]{%
8338   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8339   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8340   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8341 }%
8342 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8343   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8344   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8345   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8346 }%
8347 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8348   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8349   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8350   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8351 }%
8352 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8353   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8354   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8355   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8356 }%

```

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.

```

8357 \renewcommand*{\glxtrfullformat}[2]{%
8358   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8359   \ifglxtrinsertinside\else##2\fi
8360 }%
8361 \renewcommand*{\glxtrfullplformat}[2]{%
8362   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8363   \ifglxtrinsertinside\else##2\fi
8364 }%
8365 \renewcommand*{\Glsxtrfullformat}[2]{%
8366   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8367   \ifglxtrinsertinside\else##2\fi
8368 }%

```

```

8369 \renewcommand*{\Glsxtrfullplformat}[2]{%
8370   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8371   \ifglxtrinsertinside\else##2\fi
8372 }%
8373 }

```

long-desc-sm Backward compatibility:

```

8374 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

8375 \newabbreviationstyle{short-sm-footnote}%
8376 {%
8377   \renewcommand*{\CustomAbbreviationFields}{%
8378     name={\glsxtrfootnotename},
8379     sort={\the\glsshorttok},
8380     description={\the\glslongtok},%
8381     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8382       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8383       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8384     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8385       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8386       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8387     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8388 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8389   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8390   \glsattribute{\the\glslabeltok}{regular}%
8391   {%
8392     \glssetattribute{\the\glslabeltok}{regular}{false}%
8393   }%
8394   {}%
8395 }%
8396 }%
8397 {%
8398   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8399   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8400   \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
8401   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8402   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8403 \renewcommand*{\glsxtrfullformat}[2]{%
8404   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8405   \ifglxtrinsertinside\else##2\fi
8406   \protect\glxtrabbrvfootnote{##1}%
8407   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8408 }%
8409 \renewcommand*{\glsxtrfullplformat}[2]{%

```



```

8410 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8411 \ifglxtrinsertinside\else##2\fi
8412 \protect\glxtrabbrvfootnote{##1}%
8413 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8414 }%
8415 \renewcommand*{\Glsxtrfullformat}[2]{%
8416 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8417 \ifglxtrinsertinside\else##2\fi
8418 \protect\glxtrabbrvfootnote{##1}%
8419 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8420 }%
8421 \renewcommand*{\Glsxtrfullplformat}[2]{%
8422 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8423 \ifglxtrinsertinside\else##2\fi
8424 \protect\glxtrabbrvfootnote{##1}%
8425 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8426 }%

```

The first use full form and the inline full form use the short (long) style.

```

8427 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8428 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8429 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8430 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8431 }%
8432 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8433 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8434 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8435 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8436 }%
8437 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8438 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8439 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8440 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8441 }%
8442 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8443 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8444 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8445 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8446 }%
8447 }

```

footnote-sm Backward compatibility:

```
8448 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}
```

sm-postfootnote

```

8449 \newabbreviationstyle{short-sm-postfootnote}%
8450 {%
8451 \renewcommand*{\CustomAbbreviationFields}{%
8452 name={\glxtrfootnotename},
8453 sort={\the\glsshorttok},

```

```

8454   description={\the\glslongtok},%
8455   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8456   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8457   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.

```

8458   \renewcommand*\GlsXtrPostNewAbbreviation{%
8459     \csdef{glsxtrpostlink\glscategorylabel}{%
8460       \glsxtrifwasfirstuse
8461       {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8462         \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
8463         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
8464     }%
8465   }%
8466 }%
8467 \glsattribute{\the\glslabeltok}{regular}%
8468 {%
8469   \glssetattribute{\the\glslabeltok}{regular}{false}%
8470 }%
8471 {}%
8472 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

8473   \renewcommand*\glsxtrsetupfulldefs{%
8474     \let\glsxtrifwasfirstuse\@secondoftwo
8475   }%
8476 }%
8477 {%
8478   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8479   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8480   \renewcommand*\glsabbrvpluralsuffix{\protect\glsxtrrmsuffix}%
8481   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8482   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8483   \renewcommand*\glsxtrfullformat[2]{%
8484     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8485     \ifglsxtrininsertinside\else##2\fi
8486   }%
8487   \renewcommand*\glsxtrfullplformat[2]{%
8488     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8489     \ifglsxtrininsertinside\else##2\fi
8490   }%
8491   \renewcommand*\Glsxtrfullformat[2]{%
8492     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8493     \ifglsxtrininsertinside\else##2\fi

```

```

8494 }%
8495 \renewcommand*{\Glsxtrfullplformat}[2]{%
8496   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8497   \ifglxtrinsertinside\else##2\fi
8498 }%

```

The first use full form and the inline full form use the short (long) style.

```

8499 \renewcommand*{\glxtrinlinefullformat}[2]{%
8500   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8501   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8502   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8503 }%
8504 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8505   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8506   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8507   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8508 }%
8509 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8510   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8511   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8512   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8513 }%
8514 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8515   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8516   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8517   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8518 }%
8519 }

```

postfootnote-sm Backward compatibility:

```

8520 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

8521 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%

```

`\glsfirstabbrvemfont`

```

8522 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%

```

The default short form suffix:

`\glxtremsuffix`

```

8523 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```

8524 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```
8525 \newcommand*{\glslongemfont}[1]{\emph{#1}}%
```

`long-short-em` The long form is just set in the default long font.

```
8526 \newabbreviationstyle{long-short-em}%
8527 {%
8528   \renewcommand*{\CustomAbbreviationFields}{%
8529     name={\glsxtrlongshortname},
8530     sort={\the\glsshorttok},
8531     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8532       \protect\glsxtrfullsep{\the\glslabeltok}%
8533       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8534     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8535       \protect\glsxtrfullsep{\the\glslabeltok}%
8536       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8537     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8538     description={\the\glslongtok}}%
8539   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8540     \glsattribute{\the\glslabeltok}{regular}%
8541     {%
8542       \glssetattribute{\the\glslabeltok}{regular}{false}%
8543     }%
8544   }%
8545 }%
8546 }%
8547 {%
8548   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8549   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8550   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```
8551 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8552 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8553 \renewcommand*{\glsxtrfullformat}[2]{%
8554   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8555   \ifglsxtrininsertinside\else##2\fi
8556   \glsxtrfullsep{##1}%
8557   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8558 }%
8559 \renewcommand*{\glsxtrfullplformat}[2]{%
8560   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8561   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8562   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8563 }%
8564 \renewcommand*{\Glsxtrfullformat}[2]{%
8565   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8566   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8567   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

8568 }%
8569 \renewcommand*{\Glsxtrfullplformat}[2]{%
8570   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8571   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8572   \glsxtrparen{\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8573 }%
8574 }

```

g-short-em-desc

```

8575 \newabbreviationstyle{long-short-em-desc}%
8576 {%
8577   \renewcommand*{\CustomAbbreviationFields}{%
8578     name={\glsxtrlongshortdescname},
8579     sort={\glsxtrlongshortdescsort},%
8580     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8581       \protect\glsxtrfullsep{\the\glslabeltok}%
8582       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8583     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8584       \protect\glsxtrfullsep{\the\glslabeltok}%
8585       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8586     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8587     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8588 }%

```

Unset the regular attribute if it has been set.

```

8589 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8590   \glsattribute{\the\glslabeltok}{regular}%
8591   {%
8592     \glssetattribute{\the\glslabeltok}{regular}{false}%
8593   }%
8594   {}%
8595 }%
8596 }%
8597 {%

```

As long-short-em style:

```

8598 \GlsXtrUseAbbrStyleFmts{long-short-em}%
8599 }

```

ong-em-short-em

```

8600 \newabbreviationstyle{long-em-short-em}%
8601 {%
8602   \glslongemfont is used in the description since \glsdesc doesn't set the style.
8603   \renewcommand*{\CustomAbbreviationFields}{%
8604     name={\glsxtrlongshortname},
8605     sort={\the\glsshorttok},
8606     first={\protect\glsfirstlongemfont{\the\glslongtok}%
8607       \protect\glsxtrfullsep{\the\glslabeltok}%
8608       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%

```

```

8608 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}}%
8609 \protect\glsxtrfullsep{\the\glslabeltok}}%
8610 \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8611 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8612 description={\protect\glsfirstlongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8613 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8614 \glsattribute{\the\glslabeltok}{regular}}%
8615 {%
8616 \glssetattribute{\the\glslabeltok}{regular}{false}}%
8617 }%
8618 {}%
8619 }%
8620 }%
8621 {%
8622 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8623 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8624 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8625 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8626 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8627 \renewcommand*{\glsxtrfullformat}[2]{%
8628 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8629 \ifglsxtrininsertinside\else##2\fi
8630 \glsxtrfullsep{##1}}%
8631 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8632 }%
8633 \renewcommand*{\glsxtrfullplformat}[2]{%
8634 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8635 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8636 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8637 }%
8638 \renewcommand*{\Glsxtrfullformat}[2]{%
8639 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8640 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8641 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8642 }%
8643 \renewcommand*{\Glsxtrfullplformat}[2]{%
8644 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8645 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8646 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8647 }%
8648 }

```

m-short-em-desc

```

8649 \newabbreviationstyle{long-em-short-em-desc}%
8650 {%

```

```

8651 \renewcommand*{\CustomAbbreviationFields}{%
8652   name={\glxtrlongshortdescname},
8653   sort={\glxtrlongshortdescsort},%
8654   first={\protect\glsfirstlongemfont{\the\glslongtok}%
8655     \protect\glxtrfullsep{\the\glslabeltok}%
8656     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8657   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8658     \protect\glxtrfullsep{\the\glslabeltok}%
8659     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8660   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8661   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8662 }%

```

Unset the regular attribute if it has been set.

```

8663 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8664   \glshasattribute{\the\glslabeltok}{regular}%
8665   {%
8666     \glissetattribute{\the\glslabeltok}{regular}{false}%
8667   }%
8668   {}%
8669 }%
8670 }%
8671 {%
8672   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8673 }

```

short-em-long Now the short (long) version

```

8674 \newabbreviationstyle{short-em-long}%
8675 {%
8676   \renewcommand*{\CustomAbbreviationFields}{%
8677     name={\glxtrshortlongname},
8678     sort={\the\glsshorttok},
8679     description={\the\glslongtok},%
8680     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8681       \protect\glxtrfullsep{\the\glslabeltok}%
8682       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8683     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8684       \protect\glxtrfullsep{\the\glslabeltok}%
8685       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8686     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8687 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8688   \glshasattribute{\the\glslabeltok}{regular}%
8689   {%
8690     \glissetattribute{\the\glslabeltok}{regular}{false}%
8691   }%
8692   {}%
8693 }%
8694 }%

```

8695 {%

Mostly as short-long style:

```
8696 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
8697 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8698 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8699 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8700 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8701 \renewcommand*{\glsxtrfullformat}[2]{%
8702   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8703   \ifglsxtrininsertinside\else##2\fi
8704   \glsxtrfullsep{##1}%
8705   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8706 }%
8707 \renewcommand*{\glsxtrfullplformat}[2]{%
8708   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8709   \ifglsxtrininsertinside\else##2\fi
8710   \glsxtrfullsep{##1}%
8711   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8712 }%
8713 \renewcommand*{\Glsxtrfullformat}[2]{%
8714   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8715   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8716   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8717 }%
8718 \renewcommand*{\Glsxtrfullplformat}[2]{%
8719   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8720   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8721   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8722 }%
8723 }
```

rt-em-long-desc As before but user provides description

```
8724 \newabbreviationstyle{short-em-long-desc}%
8725 {%
8726   \renewcommand*{\CustomAbbreviationFields}{%
8727     name={\glsxtrshortlongdescname},
8728     sort={\glsxtrshortlongdescsort},
8729     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8730     \protect\glsxtrfullsep{\the\glslabeltok}%
8731     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8732     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8733     \protect\glsxtrfullsep{\the\glslabeltok}%
8734     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8735     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8736     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8737   }%
```

Unset the regular attribute if it has been set.


```

8738 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8739   \glshasattribute{\the\glslabeltok}{regular}%
8740   {%
8741     \glissetattribute{\the\glslabeltok}{regular}{false}%
8742   }%
8743   {}%
8744 }%
8745 }%
8746 {%
8747   \GlsXtrUseAbbrStyleFmts{short-em-long}%
8748 }

```

hort-em-long-em

```

8749 \newabbreviationstyle{short-em-long-em}%
8750 {%

```

\glslongemfont is used in the description since \glsgdesc doesn't set the style.

```

8751 \renewcommand*{\CustomAbbreviationFields}{%
8752   name={\glsxtrshortlongname},
8753   sort={\the\glsshorttok},
8754   description={\protect\glslongemfont{\the\glslongtok}},%
8755   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8756     \protect\glsxtrfullsep{\the\glslabeltok}%
8757     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8758   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8759     \protect\glsxtrfullsep{\the\glslabeltok}%
8760     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8761   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}

```

Unset the regular attribute if it has been set.

```

8762 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8763   \glshasattribute{\the\glslabeltok}{regular}%
8764   {%
8765     \glissetattribute{\the\glslabeltok}{regular}{false}%
8766   }%
8767   {}%
8768 }%
8769 }%
8770 {%
8771 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8772 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8773 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8774 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8775 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8776 \renewcommand*{\glsxtrfullformat}[2]{%
8777   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8778   \ifglsxtrininsertinside\else##2\fi
8779   \glsxtrfullsep{##1}%

```

```

8780 \glstrparen{\glfirstlongemfont{\glaccesslong{##1}}}%
8781 }%
8782 \renewcommand*{\glstrfullplformat}[2]{%
8783 \glfirstabbrvemfont{\glaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
8784 \ifglstrinsertinside\else##2\fi
8785 \glstrfullsep{##1}%
8786 \glstrparen{\glfirstlongemfont{\glaccesslongpl{##1}}}%
8787 }%
8788 \renewcommand*{\Glsstrfullformat}[2]{%
8789 \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
8790 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
8791 \glstrparen{\glfirstlongemfont{\glaccesslong{##1}}}%
8792 }%
8793 \renewcommand*{\Glsstrfullplformat}[2]{%
8794 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
8795 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
8796 \glstrparen{\glfirstlongemfont{\glaccesslongpl{##1}}}%
8797 }%
8798 }

```

em-long-em-desc

```

8799 \newabbreviationstyle{short-em-long-em-desc}%
8800 {%
8801 \renewcommand*{\CustomAbbreviationFields}{%
8802 name={\glstrshortlongdescname},%
8803 sort={\glstrshortlongdescsort},%
8804 first={\protect\glfirstabbrvemfont{\the\glsshorttok}%
8805 \protect\glstrfullsep{\the\glslabeltok}%
8806 \glstrparen{\protect\glfirstlongemfont{\the\glslongtok}}},%
8807 firstplural={\protect\glfirstabbrvemfont{\the\glsshortpltok}%
8808 \protect\glstrfullsep{\the\glslabeltok}%
8809 \glstrparen{\protect\glfirstlongemfont{\the\glslongpltok}}},%
8810 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8811 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8812 }%

```

Unset the regular attribute if it has been set.

```

8813 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8814 \glshasattribute{\the\glslabeltok}{regular}%
8815 {%
8816 \glissetattribute{\the\glslabeltok}{regular}{false}%
8817 }%
8818 {}%
8819 }%
8820 }%
8821 {%
8822 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8823 }

```

short-em

```

8824 \newabbreviationstyle{short-em}%
8825 {%
8826   \renewcommand*{\CustomAbbreviationFields}{%
8827     name={\glxtrshortnolongname},
8828     sort={\the\glsshorttok},
8829     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8830     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8831     text={\protect\glsabbrvemfont{\the\glsshorttok}},
8832     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8833     description={\the\glslongtok}}%
8834   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8835     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8836 }%
8837 {%
8838   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8839   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8840   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8841   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8842   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8843   \renewcommand*{\glxtrinlinefullformat}[2]{%
8844     \protect\glsfirstabbrvemfont{\glsaccesssshort{##1}}%
8845     \ifglxtrininsertinside##2\fi}%
8846   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8847   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8848 }%
8849   \renewcommand*{\glxtrinlinefullplformat}[2]{%
8850     \protect\glsfirstabbrvemfont{\glsaccesssshortpl{##1}}%
8851     \ifglxtrininsertinside##2\fi}%
8852   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8853   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8854 }%
8855   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8856     \protect\glsfirstabbrvemfont{\Glsaccesssshort{##1}}%
8857     \ifglxtrininsertinside##2\fi}%
8858   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8859   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8860 }%
8861   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8862     \protect\glsfirstabbrvemfont{\Glsaccesssshortpl{##1}}%
8863     \ifglxtrininsertinside##2\fi}%
8864   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8865   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8866 }%

```

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.

```

8867   \renewcommand*{\glxtrfullformat}[2]{%

```

```

8868 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8869 \ifglxtrinsertinside\else##2\fi
8870 }%
8871 \renewcommand*{\glxtrfullplformat}[2]{%
8872 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8873 \ifglxtrinsertinside\else##2\fi
8874 }%
8875 \renewcommand*{\Glsxtrfullformat}[2]{%
8876 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8877 \ifglxtrinsertinside\else##2\fi
8878 }%
8879 \renewcommand*{\Glsxtrfullplformat}[2]{%
8880 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8881 \ifglxtrinsertinside\else##2\fi
8882 }%
8883 }

```

short-em-nolong

```

8884 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8885 \newabbreviationstyle{short-em-desc}%
8886 {%
8887 \renewcommand*{\CustomAbbreviationFields}{%
8888 name={\glxtrshortdescname},
8889 sort={\the\glsshorttok},
8890 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8891 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8892 text={\protect\glsabbrvemfont{\the\glsshorttok}},
8893 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8894 description={\the\glslongtok}}%
8895 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8896 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8897 }%
8898 {%
8899 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8900 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8901 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8902 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8903 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8904 \renewcommand*{\glxtrinlinefullformat}[2]{%
8905 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8906 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8907 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8908 }%
8909 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8910 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8911 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

8912 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8913 }%
8914 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8915 \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8916 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
8917 \glstrparen{\glfirstlongdefaultfont{\glsaccesslong{##1}}}%
8918 }%
8919 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8920 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8921 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
8922 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8923 }%

```

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.

```

8924 \renewcommand*{\glstrfullformat}[2]{%
8925 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8926 \ifglxtrinsertinside\else##2\fi
8927 }%
8928 \renewcommand*{\glstrfullplformat}[2]{%
8929 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8930 \ifglxtrinsertinside\else##2\fi
8931 }%
8932 \renewcommand*{\Glsxtrfullformat}[2]{%
8933 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8934 \ifglxtrinsertinside\else##2\fi
8935 }%
8936 \renewcommand*{\Glsxtrfullplformat}[2]{%
8937 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8938 \ifglxtrinsertinside\else##2\fi
8939 }%
8940 }

```

-em-nolong-desc

```

8941 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8942 \newabbreviationstyle{nolong-short-em}%
8943 {%
8944 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8945 }%
8946 {%
8947 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8948 \renewcommand*{\glxtrinlinefullformat}[2]{%
8949 \protect\glfirstlongdefaultfont{\glsaccesslong{##1}%
8950 \ifglxtrinsertinside##2\fi}%
8951 \ifglxtrinsertinside\else##2\fi\glstrfullsep{##1}%
8952 \glstrparen{\glfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

8953 }%
8954 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8955   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8956     \ifglxtrininsertinside##2\fi}%
8957   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8958   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8959 }%
8960 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8961   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8962     \ifglxtrininsertinside##2\fi}%
8963   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8964   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8965 }%
8966 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8967   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8968     \ifglxtrininsertinside##2\fi}%
8969   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8970   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8971 }%
8972 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8973 \newabbreviationstyle{long-noshort-em}%
8974 {%
8975   \renewcommand*{\CustomAbbreviationFields}{%
8976     name={\glxtrlongnoshortname},
8977     sort={\the\glsshorttok},
8978     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8979     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8980     text={\protect\glslongdefaultfont{\the\glslongtok}},
8981     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8982     description={\the\glslongtok}%
8983   }%
8984   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8985     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8986 }%
8987 {%
8988   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8989   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8990   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8991   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8992   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8993 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8994   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
8995   \ifglxtrininsertinside \else##2\fi
8996 }%
8997 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8998   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%

```

```

8999 \ifglxtrinsertinside \else##2\fi
9000 }%
9001 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9002 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9003 \ifglxtrinsertinside \else##2\fi
9004 }%
9005 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9006 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9007 \ifglxtrinsertinside \else##2\fi
9008 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9009 \renewcommand*{\glxtrinlinefullformat}[2]{%
9010 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9011 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9012 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9013 }%
9014 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9015 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9016 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9017 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9018 }%
9019 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9020 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9021 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9022 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9023 }%
9024 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9025 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9026 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9027 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9028 }%

```

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.

```

9029 \renewcommand*{\glxtrfullformat}[2]{%
9030 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9031 \ifglxtrinsertinside\else##2\fi
9032 }%
9033 \renewcommand*{\glxtrfullplformat}[2]{%
9034 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9035 \ifglxtrinsertinside\else##2\fi
9036 }%
9037 \renewcommand*{\Glsxtrfullformat}[2]{%
9038 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9039 \ifglxtrinsertinside\else##2\fi
9040 }%
9041 \renewcommand*{\Glsxtrfullplformat}[2]{%
9042 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9043 \ifglxtrinsertinside\else##2\fi

```

```

9044 }%
9045 }

```

long-em Backward compatibility:

```

9046 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

9047 \newabbreviationstyle{long-em-noshort-em}%
9048 {%
9049   \renewcommand*{\CustomAbbreviationFields}{%
9050     name={\glxtrlongnoshortname},
9051     sort={\the\glsshorttok},
9052     first={\protect\glslongemfont{\the\glslongtok}},
9053     firstplural={\protect\glslongemfont{\the\glslongpltok}},
9054     text={\protect\glslongemfont{\the\glslongtok}},
9055     plural={\protect\glslongemfont{\the\glslongpltok}},%
9056     description={\protect\glslongemfont{\the\glslongtok}}%
9057   }%
9058   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9059     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9060 }%
9061 {%
9062   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9063   \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
9064   \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
9065   \renewcommand*{\glfirstlongfont}[1]{\glfirstlongemfont{##1}}%
9066   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9067 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9068   \glslongemfont{\glaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9069   \ifglxtrininsertinside \else##2\fi
9070 }%
9071 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9072   \glslongemfont{\glaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9073   \ifglxtrininsertinside \else##2\fi
9074 }%
9075 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9076   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9077   \ifglxtrininsertinside \else##2\fi
9078 }%
9079 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9080   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9081   \ifglxtrininsertinside \else##2\fi
9082 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9083 \renewcommand*{\glxtrinlinefullformat}[2]{%
9084   \glslongemfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9085   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

9086 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9087 }%
9088 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9089 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9090 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9091 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9092 }%
9093 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9094 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9095 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9096 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9097 }%
9098 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9099 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9100 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9101 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9102 }%

```

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.

```

9103 \renewcommand*{\glsxtrfullformat}[2]{%
9104 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9105 \ifglsxtrinsertinside\else##2\fi
9106 }%
9107 \renewcommand*{\glsxtrfullplformat}[2]{%
9108 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9109 \ifglsxtrinsertinside\else##2\fi
9110 }%
9111 \renewcommand*{\Glsxtrfullformat}[2]{%
9112 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9113 \ifglsxtrinsertinside\else##2\fi
9114 }%
9115 \renewcommand*{\Glsxtrfullplformat}[2]{%
9116 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9117 \ifglsxtrinsertinside\else##2\fi
9118 }%
9119 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9120 \newabbreviationstyle{long-em-noshort-em-noreg}%
9121 {%
9122 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9123 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9124 \glshasattribute{\the\glslabeltok}{regular}%
9125 {%
9126 \glssetattribute{\the\glslabeltok}{regular}{false}%
9127 }%
9128 }%

```

```

9129 }%
9130 }%
9131 {%
9132   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9133 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9134 \newabbreviationstyle{long-noshort-em-desc}%
9135 {%
9136   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9137 }%
9138 {%
9139   \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9140   \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
9141   \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
9142   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9143   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9144 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9145   \glslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9146   \ifglsxtrininsertinside \else##2\fi
9147 }%
9148 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9149   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9150   \ifglsxtrininsertinside \else##2\fi
9151 }%
9152 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9153   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9154   \ifglsxtrininsertinside \else##2\fi
9155 }%
9156 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9157   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9158   \ifglsxtrininsertinside \else##2\fi
9159 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9160 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9161   \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9162   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9163   \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshort{##1}}}%
9164 }%
9165 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9166   \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9167   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9168   \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshortpl{##1}}}%
9169 }%
9170 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9171   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%

```

```

9172     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9173     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9174 }%
9175 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9176     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9177     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9178     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9179 }%

```

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.

```

9180 \renewcommand*{\glxtrfullformat}[2]{%
9181     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9182     \ifglxtrinsertinside\else##2\fi
9183 }%
9184 \renewcommand*{\glxtrfullplformat}[2]{%
9185     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9186     \ifglxtrinsertinside\else##2\fi
9187 }%
9188 \renewcommand*{\Glsxtrfullformat}[2]{%
9189     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9190     \ifglxtrinsertinside\else##2\fi
9191 }%
9192 \renewcommand*{\Glsxtrfullplformat}[2]{%
9193     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9194     \ifglxtrinsertinside\else##2\fi
9195 }%
9196 }

```

long-desc-em Backward compatibility:

```

9197 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like `\glssshort`. The long form is emphasized.

```

9198 \newabbreviationstyle{long-em-noshort-em-desc}%
9199 {%
9200 \renewcommand*{\CustomAbbreviationFields}{%
9201     name={\glxtrlongnoshortdescname},
9202     sort={\the\glslongtok},
9203     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9204     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9205     text={\glslongemfont{\the\glslongtok}},
9206     plural={\glslongemfont{\the\glslongpltok}}%
9207 }%
9208 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9209     \glsssetattribute{\the\glslabeltok}{regular}{true}}%
9210 }%
9211 {%
9212 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%

```

```

9213 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
9214 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
9215 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
9216 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9217 \renewcommand*\glsxtrsubsequentfmt[2]{%
9218   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9219   \ifglsxtrininsertinside \else##2\fi
9220 }%
9221 \renewcommand*\glsxtrsubsequentplfmt[2]{%
9222   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9223   \ifglsxtrininsertinside \else##2\fi
9224 }%
9225 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9226   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9227   \ifglsxtrininsertinside \else##2\fi
9228 }%
9229 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9230   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9231   \ifglsxtrininsertinside \else##2\fi
9232 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9233 \renewcommand*\glsxtrinlinefullformat[2]{%
9234   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9235   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9236   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9237 }%
9238 \renewcommand*\glsxtrinlinefullplformat[2]{%
9239   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9240   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9241   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9242 }%
9243 \renewcommand*\Glsxtrinlinefullformat[2]{%
9244   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9245   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9246   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9247 }%
9248 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9249   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9250   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9251   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9252 }%

```

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.

```

9253 \renewcommand*\glsxtrfullformat[2]{%
9254   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9255   \ifglsxtrininsertinside\else##2\fi
9256 }%

```

```

9257 \renewcommand*{\glxtrfullplformat}[2]{%
9258   \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9259   \ifglxtrinsertinside\else##2\fi
9260 }%
9261 \renewcommand*{\Glsxtrfullformat}[2]{%
9262   \glsfirstlongemfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9263   \ifglxtrinsertinside\else##2\fi
9264 }%
9265 \renewcommand*{\Glsxtrfullplformat}[2]{%
9266   \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9267   \ifglxtrinsertinside\else##2\fi
9268 }%
9269 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9270 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9271 {%
9272   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
   Unset the regular attribute if it has been set.
9273   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9274     \glshasattribute{\the\glslabeltok}{regular}%
9275     {%
9276       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9277     }%
9278   }%
9279 }%
9280 }%
9281 {%
9282   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9283 }

```

ort-em-footnote

```

9284 \newabbreviationstyle{short-em-footnote}%
9285 {%
9286   \renewcommand*{\CustomAbbreviationFields}{%
9287     name={\glxtrfootnotename},
9288     sort={\the\glssshorttok},
9289     description={\the\glslongtok},%
9290     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}%
9291       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9292       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9293     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}%
9294       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9295       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9296     plural={\protect\glsabbrvemfont{\the\glssshortpltok}}}%
   Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute
   if it has been set.
9297   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

9298 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9299 \glssetattribute{\the\glslabeltok}{regular}%
9300 {%
9301 \glssetattribute{\the\glslabeltok}{regular}{false}%
9302 }%
9303 {}%
9304 }%
9305 }%
9306 {%
9307 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
9308 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9309 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9310 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9311 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9312 \renewcommand*{\glsxtrfullformat}[2]{%
9313 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9314 \ifglsxtrininsertinside\else##2\fi
9315 \protect\glsxtrabbrvfootnote{##1}%
9316 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9317 }%
9318 \renewcommand*{\glsxtrfullplformat}[2]{%
9319 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9320 \ifglsxtrininsertinside\else##2\fi
9321 \protect\glsxtrabbrvfootnote{##1}%
9322 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9323 }%
9324 \renewcommand*{\Glsxtrfullformat}[2]{%
9325 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9326 \ifglsxtrininsertinside\else##2\fi
9327 \protect\glsxtrabbrvfootnote{##1}%
9328 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9329 }%
9330 \renewcommand*{\Glsxtrfullplformat}[2]{%
9331 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9332 \ifglsxtrininsertinside\else##2\fi
9333 \protect\glsxtrabbrvfootnote{##1}%
9334 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9335 }%

```

The first use full form and the inline full form use the short (long) style.

```

9336 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9337 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9338 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9339 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9340 }%
9341 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9342 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9343 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

9344 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9345 }%
9346 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9347 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9348 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9349 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9350 }%
9351 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9352 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9353 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9354 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9355 }%
9356 }

```

footnote-em Backward compatibility:

```

9357 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9358 \newabbreviationstyle{short-em-postfootnote}%
9359 {%
9360 \renewcommand*{\CustomAbbreviationFields}{%
9361 name={\glxtrfootnotename},
9362 sort={\the\glssshorttok},
9363 description={\the\glslongtok},%
9364 first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},%
9365 firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},%
9366 plural={\protect\glsabbrvemfont{\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.

```

9367 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9368 \csdef{glxtrpostlink\glscategorylabel}{%
9369 \glxtrifwasfirstuse
9370 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9371 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9372 {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9373 }%
9374 {}%
9375 }%
9376 \glshasattribute{\the\glslabeltok}{regular}%
9377 {%
9378 \glissetattribute{\the\glslabeltok}{regular}{false}%
9379 }%
9380 {}%
9381 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9382 \renewcommand*\glxtrsetupfulldefs}{%
9383 \let\glxtrifwasfirstuse\@secondoftwo
9384 }%
9385 }%
9386 {%
9387 \renewcommand*\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9388 \renewcommand*\glxabbrvfont[1]{\glxabbrvemfont{##1}}%
9389 \renewcommand*\glxfirstabbrvfont[1]{\glxfirstabbrvemfont{##1}}%
9390 \renewcommand*\glxfirstlongfont[1]{\glxfirstlongfootnotefont{##1}}%
9391 \renewcommand*\glxlongfont[1]{\glxlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9392 \renewcommand*\glxtrfullformat}[2]{%
9393 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9394 \ifglxtrininsertinside\else##2\fi
9395 }%
9396 \renewcommand*\glxtrfullplformat}[2]{%
9397 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9398 \ifglxtrininsertinside\else##2\fi
9399 }%
9400 \renewcommand*\Glxtrfullformat}[2]{%
9401 \glxfirstabbrvemfont{\Glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9402 \ifglxtrininsertinside\else##2\fi
9403 }%
9404 \renewcommand*\Glxtrfullplformat}[2]{%
9405 \glxfirstabbrvemfont{\Glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9406 \ifglxtrininsertinside\else##2\fi
9407 }%

```

The first use full form and the inline full form use the short (long) style.

```

9408 \renewcommand*\glxtrininlinefullformat}[2]{%
9409 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9410 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9411 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslong{##1}}}%
9412 }%
9413 \renewcommand*\glxtrininlinefullplformat}[2]{%
9414 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9415 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9416 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslongpl{##1}}}%
9417 }%
9418 \renewcommand*\Glxtrininlinefullformat}[2]{%
9419 \glxfirstabbrvemfont{\Glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9420 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9421 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslong{##1}}}%
9422 }%
9423 \renewcommand*\Glxtrininlinefullplformat}[2]{%
9424 \glxfirstabbrvemfont{\Glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9425 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

9426 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9427 }%
9428 }

```

postfootnote-em Backward compatibility:

```

9429 \@glsxtr@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:

glsxtruserfield Default is the useri field.

```

9430 \newcommand*{\glsxtruserfield}{useri}

```

glsxtruserparen 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.

```

9431 \ifdef\glscurrentfieldvalue
9432 {
9433   \newcommand*{\glsxtruserparen}[2]{%
9434     \glsxtrfullsep{#2}%
9435     \glsxtrparen
9436       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9437   }
9438 }
9439 {
9440   \newcommand*{\glsxtruserparen}[2]{%
9441     \glsxtrfullsep{#2}%
9442     \glsxtrparen
9443       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9444   }
9445 }

```

Font used for short form:

lsabbrvuserfont

```

9446 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

```

Font used for short form on first use:

stabbrvuserfont

```

9447 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

```

Font used for long form:

glslonguserfont

```

9448 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

```

Font used for long form on first use:

rstlonguserfont

```
9449 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
9450 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
9451 \newabbreviationstyle{long-short-user}%
9452 {%
9453   \renewcommand*{\CustomAbbreviationFields}{%
9454     name={\glsxtrlongshortname},
9455     sort={\the\glsshorttok},
9456     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9457       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9458     {\the\glslabeltok}},%
9459     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9460       \protect\glsxtruserparen
9461       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9462     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9463     description={\protect\glslonguserfont{\the\glslongtok}}}%
9464   }
```

Unset the regular attribute if it has been set.

```
9464   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9465     \glsattribute{\the\glslabeltok}{regular}%
9466     {%
9467       \glssetattribute{\the\glslabeltok}{regular}{false}%
9468     }%
9469   }%
9470 }%
9471 }%
9472 }
```

In case the user wants to mix and match font styles, these are redefined here.

```
9473 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9474 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9475 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9476 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9477 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9478 \renewcommand*{\glsxtrfullformat}[2]{%
9479   \glsfirstlonguserfont{\glsaccesslong{##1}}\ifglsxtrininsertinside##2\fi}%
9480 \ifglsxtrininsertinside\else##2\fi
9481 \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9482 }%
9483 \renewcommand*{\glsxtrfullplformat}[2]{%
9484   \glsfirstlonguserfont{\glsaccesslongpl{##1}}\ifglsxtrininsertinside##2\fi}%
9485 \ifglsxtrininsertinside\else##2\fi
```

```

9486 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9487 }%
9488 \renewcommand*{\Glsxtrfullformat}[2]{%
9489 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9490 \ifglxtrinsertinside\else##2\fi
9491 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9492 }%
9493 \renewcommand*{\Glsxtrfullplformat}[2]{%
9494 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9495 \ifglxtrinsertinside\else##2\fi
9496 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9497 }%
9498 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9499 \newabbreviationstyle{long-postshort-user}%
9500 {%
9501 \renewcommand*{\CustomAbbreviationFields}{%
9502 name={\glxtrlongshortname},
9503 sort={\the\glsshorttok},
9504 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9505 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9506 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9507 description={\protect\glslonguserfont{\the\glslongtok}}}%
9508 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9509 \csdef{glxtrpostlink\glscategorylabel}{%
9510 \glxtrifwasfirstuse
9511 {%
9512 \glxtruserparen
9513 {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
9514 {\glslabel}%
9515 }%
9516 }%
9517 }%
9518 \glshasattribute{\the\glslabeltok}{regular}%
9519 {%
9520 \glssetattribute{\the\glslabeltok}{regular}{false}%
9521 }%
9522 }%
9523 }%
9524 }%
9525 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9526 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9527 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9528 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9529 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9530 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9531 \renewcommand*{\glxtrfullformat}[2]{%
9532   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9533   \ifglxtrinsertinside\else##2\fi
9534 }%
9535 \renewcommand*{\glxtrfullplformat}[2]{%
9536   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9537   \ifglxtrinsertinside\else##2\fi
9538 }%
9539 \renewcommand*{\Glsxtrfullformat}[2]{%
9540   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9541   \ifglxtrinsertinside\else##2\fi
9542 }%
9543 \renewcommand*{\Glsxtrfullplformat}[2]{%
9544   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9545   \ifglxtrinsertinside\else##2\fi
9546 }%

```

In-line format:

```

9547 \renewcommand*{\glxtrinlinefullformat}[2]{%
9548   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9549   \ifglxtrinsertinside\else##2\fi
9550   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9551 }%
9552 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9553   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9554   \ifglxtrinsertinside\else##2\fi
9555   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9556 }%
9557 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9558   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9559   \ifglxtrinsertinside\else##2\fi
9560   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9561 }%
9562 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9563   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9564   \ifglxtrinsertinside\else##2\fi
9565   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9566 }%
9567 }

```

ortuserdesname

```

9568 \newcommand*{\glxtrlongshortuserdesname}{%
9569   \protect\glslonguserfont{\the\glslongtok}%
9570   \protect\glxtruserparen
9571   {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
9572 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9573 \newabbreviationstyle{long-postshort-user-desc}%
9574 {%
9575   \renewcommand*{\CustomAbbreviationFields}{%
9576     name={\glxtrlongshortuserdescname},
9577     sort={\the\glslongtok},
9578     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9579     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9580     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9581     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
9582   }%
9583   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9584     \csdef{glxtrpostlink\glscategorylabel}{%
9585       \glxtrifwasfirstuse
9586       {%
9587         \glxtruserparen
9588         {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
9589         {\glslabel}%
9590       }%
9591     }%
9592   }%
9593   \glshasattribute{\the\glslabeltok}{regular}%
9594   {%
9595     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9596   }%
9597   {}%
9598 }%
9599 }%
9600 {%
9601   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9602 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9603 \newabbreviationstyle{short-postlong-user}%
9604 {%
9605   \renewcommand*{\CustomAbbreviationFields}{%
9606     name={\glxtrshortlongname},
9607     sort={\the\glsshorttok},
9608     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9609     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9610     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9611     description={\protect\glslonguserfont{\the\glslongtok}}}%
9612   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9613     \csdef{glxtrpostlink\glscategorylabel}{%
9614       \glxtrifwasfirstuse
9615       {%
9616         \glxtruserparen
9617         {\glsfirstlonguserfont{\glstrylong{\glslabel}}}%
9618         {\glslabel}%

```

```

9619     }%
9620     {}%
9621     }%
9622     \glshasattribute{\the\glslabeltok}{regular}%
9623     {%
9624         \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9625     }%
9626     {}%
9627 }%
9628 }%
9629 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9630 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9631 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9632 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9633 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9634 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9635 \renewcommand*{\glsxtrfullformat}[2]{%
9636     \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9637     \ifglsxtrininsertinside\else##2\fi
9638 }%
9639 \renewcommand*{\glsxtrfullplformat}[2]{%
9640     \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9641     \ifglsxtrininsertinside\else##2\fi
9642 }%
9643 \renewcommand*{\Glsxtrfullformat}[2]{%
9644     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9645     \ifglsxtrininsertinside\else##2\fi
9646 }%
9647 \renewcommand*{\Glsxtrfullplformat}[2]{%
9648     \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9649     \ifglsxtrininsertinside\else##2\fi
9650 }%

```

In-line format:

```

9651 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9652     \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9653     \ifglsxtrininsertinside\else##2\fi
9654     \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9655 }%
9656 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9657     \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9658     \ifglsxtrininsertinside\else##2\fi
9659     \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9660 }%
9661 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9662     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```

```

9663 \ifglxtrinsertinside\else##2\fi
9664 \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9665 }%
9666 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9667 \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9668 \ifglxtrinsertinside\else##2\fi
9669 \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9670 }%
9671 }

```

onguserdescname

```

9672 \newcommand*{\glxtrshortlonguserdescname}{%
9673 \protect\glssabbrvuserfont{\the\glssshorttok}%
9674 \protect\glxtruserparen
9675 {\protect\glslonguserfont{\the\glslongpltok}}}%
9676 {\the\glslabeltok}%
9677 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

9678 \newabbreviationstyle{short-postlong-user-desc}%
9679 {%
9680 \renewcommand*{\CustomAbbreviationFields}{%
9681 name={\glxtrshortlonguserdescname},
9682 sort={\the\glssshorttok},
9683 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9684 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9685 text={\protect\glssabbrvuserfont{\the\glssshorttok}},%
9686 plural={\protect\glssabbrvuserfont{\the\glssshortpltok}}}%
9687 }%
9688 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9689 \csdef{glxtrpostlink\glscategorylabel}{%
9690 \glxtrifwasfirstuse
9691 {%
9692 \glxtruserparen
9693 {\glsfirstlonguserfont{\glssentrylong{\glslabel}}}%
9694 {\glslabel}%
9695 }%
9696 {}}%
9697 }%
9698 \glshasattribute{\the\glslabeltok}{regular}%
9699 {%
9700 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9701 }%
9702 {}%
9703 }%
9704 }%
9705 {%
9706 \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9707 }

```

short-user-desc

```
9708 \newabbreviationstyle{long-short-user-desc}%
9709 {%
9710   \renewcommand*{\CustomAbbreviationFields}{%
9711     name={\glxstrlongshortuserdescname},
9712     sort={\glxstrlongshortdescsort},%

9713     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9714       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9715       {\the\glslabeltok}},%
9716     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9717       \protect\glxtruserparen
9718       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9719     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9720     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9721   }%

   Unset the regular attribute if it has been set.
9722   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9723     \glshasattribute{\the\glslabeltok}{regular}%
9724     {%
9725       \glissetattribute{\the\glslabeltok}{regular}{false}%
9726     }%
9727   }%
9728 }%
9729 }%
9730 {%
9731   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9732 }
```

short-long-user

```
9733 \newabbreviationstyle{short-long-user}%
9734 {%

   \glslonguserfont is used in the description since \glsdesc doesn't set the style.
9735   \renewcommand*{\CustomAbbreviationFields}{%
9736     name={\glxstrshortlongname},
9737     sort={\the\glsshorttok},
9738     description={\protect\glslonguserfont{\the\glslongtok}},%
9739     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9740       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9741       {\the\glslabeltok}},%
9742     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9743       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
9744       {\the\glslabeltok}},%

9745     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

   Unset the regular attribute if it has been set.
9746   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```



```

9747 \glshasattribute{\the\glslabeltok}{regular}%
9748 {%
9749 \glsetattribute{\the\glslabeltok}{regular}{false}%
9750 }%
9751 {}%
9752 }%
9753 }%
9754 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9755 \renewcommand*\abbrvpluralsuffix{\glxtrusersuffix}%
9756 \renewcommand*\glsabbrvfont[1]{\glsabbrvuserfont{##1}}%
9757 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvuserfont{##1}}%
9758 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonguserfont{##1}}%
9759 \renewcommand*\glslongfont[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9760 \renewcommand*\glxtrfullformat[2]{%
9761 \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9762 \ifglxtrinsertinside\else##2\fi
9763 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9764 }%
9765 \renewcommand*\glxtrfullplformat[2]{%
9766 \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9767 \ifglxtrinsertinside\else##2\fi
9768 \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9769 }%
9770 \renewcommand*\Glsxtrfullformat[2]{%
9771 \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9772 \ifglxtrinsertinside\else##2\fi
9773 \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9774 }%
9775 \renewcommand*\Glsxtrfullplformat[2]{%
9776 \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9777 \ifglxtrinsertinside\else##2\fi
9778 \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9779 }%
9780 }

```

-long-user-desc

```

9781 \newabbreviationstyle{short-long-user-desc}%
9782 {%
9783 \renewcommand*\CustomAbbreviationFields{%
9784 name={\glxtrshortlonguserdescname},
9785 sort={\glxtrshortlongdescsort},%
9786 first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9787 \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9788 {\the\glslabeltok}},%
9789 firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%

```

```

9790 \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9791 {\the\glslabeltok}},%
9792 text={\protect\glsabbrvfont{\the\glsshorttok}},%
9793 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9794 }%

```

Unset the regular attribute if it has been set.

```

9795 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9796 \glshasattribute{\the\glslabeltok}{regular}%
9797 {%
9798 \glissetattribute{\the\glslabeltok}{regular}{false}%
9799 }%
9800 {}%
9801 }%
9802 }%
9803 {%
9804 \GlsXtrUseAbbrStyleFmts{short-long-user}%
9805 }

```

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 `\glxtrlong` 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.

```

9806 \newrobustcmd*{\glxtrifhyphenstart}[3]{%
9807 \ifx\glsinsert#1\relax
9808 \expandafter\@glxtrifhyphenstart#1\relax\relax
9809 \@end@glxtrifhyphenstart{#2}{#3}%
9810 \else
9811 \@glxtrifhyphenstart#1\relax\relax\@end@glxtrifhyphenstart{#2}{#3}%
9812 \fi
9813 }

```

`trifhyphenstart`

```

9814 \def\@glxtrifhyphenstart#1#2\@end@glxtrifhyphenstart#3#4{%
9815 \ifx-#1\relax#3\else #4\fi
9816 }

```

`rlonghyphenshort`

```
\glxtrlonghyphenshort{<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.

9817 \newcommand*{\glxtrlonghyphenshort}[4]{%

Grouping is needed to localise the redefinitions.

9818 {%

If *<insert>* starts with a hyphen, redefine \glxtrwordsep to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to \glxtrwordsep if *<insert>* doesn't start with a hyphen.

9819 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}}%

9820 \glsfirstlonghyphenfont{#2\ifglxtrininsertinside{#4}\fi}%

9821 \ifglxtrininsertinside\else{#4}\fi

9822 \glxtrfullsep{#1}%

9823 \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrininsertinside{#4}\fi}%

9824 \ifglxtrininsertinside\else{#4}\fi}%

9825 }%

9826 }

abbrvhyphenfont

9827 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%

abbrvhyphenfont

9828 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%

slonghyphenfont

9829 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%

tlonghyphenfont

9830 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%

The default short form suffix:

xtrhyphensuffix

9831 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}

en-short-hyphen Designed for use with the markwords attribute.

9832 \newabbreviationstyle{long-hyphen-short-hyphen}%

9833 {%

9834 \renewcommand*{\CustomAbbreviationFields}{%

9835 name={\glxtrlongshortname},

9836 sort={\the\glsshorttok},

9837 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%

9838 \protect\glxtrfullsep{\the\glslabeltok}}%

9839 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%

9840 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%

9841 \protect\glxtrfullsep{\the\glslabeltok}}%

9842 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%

9843 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%

9844 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

Unset the regular attribute if it has been set.

```

9845 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9846   \glshasattribute{\the\glslabeltok}{regular}%
9847   {%
9848     \glissetattribute{\the\glslabeltok}{regular}{false}%
9849   }%
9850   {}%
9851 }%
9852 }%
9853 {%
9854 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9855 \renewcommand*{\glssabrvfont}[1]{\glssabrvhyphenfont{##1}}%
9856 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvhyphenfont{##1}}%
9857 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonghyphenfont{##1}}%
9858 \renewcommand*{\glsslongfont}[1]{\glsslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9859 \renewcommand*{\glsxtrfullformat}[2]{%
9860   \glsxtrlonghyphenshort{##1}{\glssaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9861 }%
9862 \renewcommand*{\glsxtrfullplformat}[2]{%
9863   \glsxtrlonghyphenshort{##1}{\glssaccesslongpl{##1}}%
9864   {\glssaccessshortpl{##1}}{##2}%
9865 }%
9866 \renewcommand*{\Glsxtrfullformat}[2]{%
9867   \glsxtrlonghyphenshort{##1}{\glssaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9868 }%
9869 \renewcommand*{\Glsxtrfullplformat}[2]{%
9870   \glsxtrlonghyphenshort{##1}{\glssaccesslongpl{##1}}%
9871   {\glssaccessshortpl{##1}}{##2}%
9872 }%
9873 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9874 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9875 {%
9876   \renewcommand*{\CustomAbbreviationFields}{%
9877     name={\glsxtrlongshortdescname},
9878     sort={\glsxtrlongshortdescsort},
9879     first={\protect\glssfirstlonghyphenfont{\the\glsslongtok}}%
9880     \protect\glsxtrfullsep{\the\glslabeltok}%
9881     \glsxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshorttok}}},%
9882     firstplural={\protect\glssfirstlonghyphenfont{\the\glsslongpltok}}%
9883     \protect\glsxtrfullsep{\the\glslabeltok}%
9884     \glsxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshortpltok}}},%
9885     text={\protect\glssabrvhyphenfont{\the\glssshorttok}},%
9886     plural={\protect\glssabrvhyphenfont{\the\glssshortpltok}}}%
9887   }%

```

Unset the regular attribute if it has been set.

```

9888 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9889   \glshasattribute{\the\glslabeltok}{regular}%
9890   {%
9891     \glissetattribute{\the\glslabeltok}{regular}{false}%
9892   }%
9893   {}%
9894 }%
9895 }%
9896 {%
9897   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9898 }

```

onghyphennoshort

```
\glsxtrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```

9899 \newcommand*{\glsxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```
9900 {%
```

If *<insert>* starts with a hyphen, redefine `\glsxtrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glsxtrwordsep` if *<insert>* doesn't start with a hyphen.

```

9901 \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9902 \glsfirstlonghyphenfont{#2\ifglsxtrininsertinside{#3}\fi}%
9903 \ifglsxtrininsertinside\else{#3}\fi
9904 }%
9905 }

```

short-desc-noreg

This version doesn't show the short form (except explicitly with `\glsxtrshort`). Since `\glsxtrshort` 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.

```

9906 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
9907 {%
9908   \renewcommand*{\CustomAbbreviationFields}{%
9909     name={\glsxtrlongnoshortdescname},
9910     sort={\expandonce\glxtrorglong},
9911     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9912     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9913     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
9914   }%

```

Unset the regular attribute if it has been set.

```

9915 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9916   \glshasattribute{\the\glslabeltok}{regular}%
9917   {%
9918     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9919 }%
9920 {}%
9921 }%
9922 }%
9923 {%
9924 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9925 \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
9926 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
9927 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
9928 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
9929 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9930 \renewcommand*\glxtrsubsequentfmt[2]{%
9931   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9932 }%
9933 \renewcommand*\glxtrsubsequentplfmt[2]{%
9934   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9935 }%
9936 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9937   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9938 }%
9939 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9940   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9941 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9942 \renewcommand*\glxtrinlinefullformat[2]{%
9943   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9944   \glxtrfullsep{##1}%
9945   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9946 }%
9947 \renewcommand*\glxtrinlinefullplformat[2]{%
9948   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9949   \glxtrfullsep{##1}%
9950   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9951 }%
9952 \renewcommand*\Glsxtrinlinefullformat[2]{%
9953   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9954   \glxtrfullsep{##1}%
9955   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9956 }%
9957 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9958   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9959   \glxtrfullsep{##1}%
9960   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9961 }%

```

The first use full form only displays the long form.

```

9962 \renewcommand*{\glxtrfullformat}[2]{%
9963   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9964 }%
9965 \renewcommand*{\glxtrfullplformat}[2]{%
9966   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9967 }%
9968 \renewcommand*{\Glsxtrfullformat}[2]{%
9969   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9970 }%
9971 \renewcommand*{\Glsxtrfullplformat}[2]{%
9972   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9973 }%
9974 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

9975 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9976 {%
9977   \renewcommand*{\CustomAbbreviationFields}{%
9978     name={\glxtrlongnoshortname},
9979     sort={\the\glsshorttok},
9980     first={\protect\glxtrfirstlonghyphenfont{\the\glslongtok}},%
9981     firstplural={\protect\glxtrfirstlonghyphenfont{\the\glslongpltok}},%
9982     text={\protect\glxtrlonghyphenfont{\the\glslongtok}},%
9983     plural={\protect\glxtrlonghyphenfont{\the\glslongpltok}},%
9984     description={\the\glslongtok}%
9985   }%

```

Unset the regular attribute if it has been set.

```

9986 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9987   \glshasattribute{\the\glslabeltok}{regular}%
9988   {%
9989     \glissetattribute{\the\glslabeltok}{regular}{false}%
9990   }%
9991   {}}%
9992 }%
9993 }%
9994 {%
9995   \GlsXtrUseAbbrStyleFmts{long-desc}%
9996 }

```

glxtrlonghyphen `\glxtrlonghyphen{<long>}{<label>}{<insert>}`

Used by long-hyphen-postshort-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

9997 \newcommand*{\glxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

9998 {%
9999   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{ }%
10000   \glsfirstlonghyphenfont{#1}%
10001 }%
10002 }

```

posthyphenshort `\glxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```

10003 \newcommand*{\glxtrposthyphenshort}[2]{%
10004   {%
10005     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{ }%
10006     \ifglxtrininsertinside{\glsfirstlonghyphenfont{#2}}\else{#2}\fi
10007     \glxtrfullsep{#1}%
10008     \glxtrparen
10009     {\glsfirstabbrvhyphenfont{\glstryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
10010     \ifglxtrininsertinside\else{#2}\fi
10011   }%
10012 }%
10013 }

```

hyphensubsequent `\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

10014 \newcommand*{\glxtrposthyphensubsequent}[2]{%
10015   \glsabrvfont{\ifglxtrininsertinside {#2}\fi}%
10016   \ifglxtrininsertinside \else{#2}\fi
10017 }

```

postshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10018 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
10019 {%
10020   \renewcommand*{\CustomAbbreviationFields}{%
10021     name={\glxtrlongshortname},
10022     sort={\the\glsshorttok},
10023     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10024     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10025     plural={\protect\glsabrvhyphenfont{\the\glsshortpltok}},%
10026     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10027   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```



```

10028 \csdef{glxstrpostlink\glscategorylabel}{%
10029 \glxstrifwasfirstuse
10030 {%
10031 \glxstrposthyphenshort{\glslabel}{\glsininsert}%
10032 }%
10033 {%

```

Put the insertion into the post-link:

```

10034 \glxstrposthyphensubsequent{\glslabel}{\glsininsert}%
10035 }%
10036 }%
10037 \glshasattribute{\the\glslabeltok}{regular}%
10038 {%
10039 \glissetattribute{\the\glslabeltok}{regular}{false}%
10040 }%
10041 {}%
10042 }%
10043 }%
10044 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10045 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10046 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10047 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10048 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10049 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10050 \renewcommand*{\glxtrssubsequentfmt}[2]{%
10051 \glsabbrvfont{\glssaccessshort{##1}}%
10052 }%
10053 \renewcommand*{\glxtrssubsequentplfmt}[2]{%
10054 \glsabbrvfont{\glssaccessshortpl{##1}}%
10055 }%
10056 \renewcommand*{\Glsxtrssubsequentfmt}[2]{%
10057 \glsabbrvfont{\Glsaccessshort{##1}}%
10058 }%
10059 \renewcommand*{\Glsxtrssubsequentplfmt}[2]{%
10060 \glsabbrvfont{\Glsaccessshortpl{##1}}%
10061 }%

```

First use full form:

```

10062 \renewcommand*{\glxtrfullformat}[2]{%
10063 \glxtrlonghyphen{\glssaccesslong{##1}}{##1}{##2}%
10064 }%
10065 \renewcommand*{\glxtrfullplformat}[2]{%
10066 \glxtrlonghyphen{\glssaccesslongpl{##1}}{##1}{##2}%
10067 }%
10068 \renewcommand*{\Glsxtrfullformat}[2]{%
10069 \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
10070 }%

```

```

10071 \renewcommand*{\Glsxtrfullplformat}[2]{%
10072   \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
10073 }%

```

In-line format.

```

10074 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10075   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
10076     \ifglsxtrininsertinside{##2}\fi}%
10077   \ifglsxtrininsertinside \else{##2}\fi
10078 }%
10079 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10080   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10081     \ifglsxtrininsertinside{##2}\fi}%
10082   \ifglsxtrininsertinside \else{##2}\fi
10083 }%
10084 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10085   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
10086     \ifglsxtrininsertinside{##2}\fi}%
10087   \ifglsxtrininsertinside \else{##2}\fi
10088 }%
10089 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10090   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10091     \ifglsxtrininsertinside{##2}\fi}%
10092   \ifglsxtrininsertinside \else{##2}\fi
10093 }%
10094 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10095 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10096 {%
10097   \renewcommand*{\CustomAbbreviationFields}{%
10098     name={\glsxtrlongshortdescname},
10099     sort={\glsxtrlongshortdescsort},%
10100     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10101     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10102     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10103     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10104 }%
10105 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10106   \csdef{glsxtrpostlink\glscategorylabel}{%
10107     \glsxtrifwasfirstuse
10108     {%
10109       \glsxtrposthyphenshort{\glslabel}{\glsinsert}%
10110     }%
10111     {%

```

Put the insertion into the post-link:

```

10112       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10113     }%
10114   }%

```

```

10115 \glshasattribute{\the\glslabeltok}{regular}%
10116 {%
10117 \glissetattribute{\the\glslabeltok}{regular}{false}%
10118 }%
10119 {}%
10120 }%
10121 }%
10122 {%
10123 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10124 }

```

`\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.

```

10125 \newcommand*{\glxtrshorthyphenlong}[4]{%

```

Grouping is needed to localise the redefinitions.

```

10126 {%

```

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.)

```

10127 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{%
10128 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
10129 \ifglxtrininsertinside\else{#4}\fi
10130 \glxtrfullsep{#1}%
10131 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
10132 \ifglxtrininsertinside\else{#4}\fi}%
10133 }%
10134 }

```

`hen-long-hyphen` Designed for use with the `markwords` attribute.

```

10135 \newabbreviationstyle{short-hyphen-long-hyphen}%
10136 {%
10137 \renewcommand*{\CustomAbbreviationFields}{%
10138 name={\glxtrshortlongname},
10139 sort={\the\glsshorttok},
10140 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10141 \protect\glxtrfullsep{\the\glslabeltok}%
10142 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10143 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10144 \protect\glxtrfullsep{\the\glslabeltok}%
10145 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10146 plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}}},%
10147 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10148 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10149   \glshasattribute{\the\glslabeltok}{regular}%
10150   {%
10151     \glsselattribute{\the\glslabeltok}{regular}{false}%
10152   }%
10153   {}%
10154 }%
10155 }%
10156 {%
10157 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
10158 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10159 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10160 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10161 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10162 \renewcommand*{\glsxtrfullformat}[2]{%
10163   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10164 }%
10165 \renewcommand*{\glsxtrfullplformat}[2]{%
10166   \glsxtrshorthyphenlong{##1}%
10167   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10168 }%
10169 \renewcommand*{\Glsxtrfullformat}[2]{%
10170   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10171 }%
10172 \renewcommand*{\Glsxtrfullplformat}[2]{%
10173   \glsxtrshorthyphenlong{##1}%
10174   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10175 }%
10176 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10177 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10178 {%
10179   \renewcommand*{\CustomAbbreviationFields}{%
10180     name={\glsxtrshortlongdescname},
10181     sort={\glsxtrshortlongdescsort},
10182     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
10183     \protect\glsxtrfullsep{\the\glslabeltok}%
10184     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10185     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
10186     \protect\glsxtrfullsep{\the\glslabeltok}%
10187     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10188     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10189     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10190 }%

```

Unset the regular attribute if it has been set.

```

10191 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10192   \glshasattribute{\the\glslabeltok}{regular}%
10193   {%
10194     \glissetattribute{\the\glslabeltok}{regular}{false}%
10195   }%
10196   {}%
10197 }%
10198 }%
10199 {%
10200   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10201 }

```

`\glstrshorthyphen` `\glstrshorthyphen{<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.

```

10202 \newcommand*{\glstrshorthyphen}[3]{%
    Grouping is needed to localise the redefinitions.
10203   {%
10204     \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}{%
10205       \glsfirstabbrvhyphenfont{#1}%
10206     }%
10207 }

```

`\glstrposthyphenlong` `\glstrposthyphenlong{<label>}{<insert>}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glstrshorthyphenlong` but omits the `<short>` part. This always uses the singular long form.

```

10208 \newcommand*{\glstrposthyphenlong}[2]{%
10209   {%
10210     \glstrifhyphenstart{#2}{\def\glstrwordsep{-}}{%
10211       \ifglstrinsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10212       \glstrfullsep{#1}%
10213       \glstrparen
10214       {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglstrinsertinside{#2}\fi}%
10215       \ifglstrinsertinside\else{#2}\fi
10216     }%
10217   }%
10218 }

```

`postlong-hyphen` Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10219 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10220 {%
10221   \renewcommand*{\CustomAbbreviationFields}{%
10222     name={\glxtrshortlongname},
10223     sort={\the\glsshorttok},
10224     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10225     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10226     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10227     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10228 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10229   \csdef{glxtrpostlink\glscategorylabel}{%
10230     \glxtrifwasfirstuse
10231     {%
10232       \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10233     }%
10234   }%

```

Put the insertion into the post-link:

```

10235     \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10236   }%
10237 }%
10238 \glshasattribute{\the\glslabeltok}{regular}%
10239 {%
10240   \glssetattribute{\the\glslabeltok}{regular}{false}%
10241 }%
10242 {}%
10243 }%
10244 }%
10245 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10246 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10247 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10248 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10249 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10250 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10251 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10252   \glsabbrvfont{\glsaccessshort{##1}}%
10253 }%
10254 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10255   \glsabbrvfont{\glsaccessshortpl{##1}}%
10256 }%
10257 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10258   \glsabbrvfont{\Glsaccessshort{##1}}%
10259 }%
10260 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10261   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10262 }%

```

First use full form:

```

10263 \renewcommand*{\glxstrfullformat}[2]{%
10264   \glxstrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10265 }%
10266 \renewcommand*{\glxstrfullplformat}[2]{%
10267   \glxstrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10268 }%
10269 \renewcommand*{\Glsxstrfullformat}[2]{%
10270   \glxstrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10271 }%
10272 \renewcommand*{\Glsxstrfullplformat}[2]{%
10273   \glxstrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10274 }%

```

In-line format. Commands like `\glxstrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10275 \renewcommand*{\glxstrinlinefullformat}[2]{%
10276   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
10277   \ifglxstrinsertinside{##2}\fi}%
10278   \ifglxstrinsertinside \else{##2}\fi
10279 }%
10280 \renewcommand*{\glxstrinlinefullplformat}[2]{%
10281   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
10282   \ifglxstrinsertinside{##2}\fi}%
10283   \ifglxstrinsertinside \else{##2}\fi
10284 }%
10285 \renewcommand*{\Glsxstrinlinefullformat}[2]{%
10286   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10287   \ifglxstrinsertinside{##2}\fi}%
10288   \ifglxstrinsertinside \else{##2}\fi
10289 }%
10290 \renewcommand*{\Glsxstrinlinefullplformat}[2]{%
10291   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10292   \ifglxstrinsertinside{##2}\fi}%
10293   \ifglxstrinsertinside \else{##2}\fi
10294 }%
10295 }

```

`ong-hyphen-desc` Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

10296 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10297 {%
10298   \renewcommand*{\CustomAbbreviationFields}{%
10299     name={\glxstrshortlongdescname},
10300     sort={\glxstrshortlongdescsort},%
10301     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10302     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10303     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10304     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10305 }%

```

```

10306 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10307 \csdef{glxtrpostlink\glscategorylabel}{%
10308 \glxtrifwasfirstuse
10309 {%
10310 \glxtrposthyphenlong{\glslabel}{\glinsert}%
10311 }%
10312 {%
    Put the insertion into the post-link:
10313 \glxtrposthyphensubsequent{\glslabel}{\glinsert}%
10314 }%
10315 }%
10316 \glshasattribute{\the\glslabeltok}{regular}%
10317 {%
10318 \glissetattribute{\the\glslabeltok}{regular}{false}%
10319 }%
10320 {}%
10321 }%
10322 }%
10323 {%
10324 \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10325 }

```

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`

```
10326 \newcommand*{\glabbrvonlyfont}{\glabbrvdefaultfont}%
```

`stabbrvonlyfont`

```
10327 \newcommand*{\glfirstabbrvonlyfont}{\glabbrvonlyfont}%
```

`glslongonlyfont`

```
10328 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

`rstlongonlyfont`

```
10329 \newcommand*{\glfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

`lsxtronlysuffix`

```
10330 \newcommand*{\glxtronlysuffix}{\glxtrabbrvpluralsuffix}
```

`\glxtronlyname` The default name format for this style.

```

10331 \newcommand*{\glxtronlyname}{%
10332 \protect\glabbrvonlyfont{\the\glsshorttok}%
10333 }

```


only-short-only

```
10334 \newabbreviationstyle{long-only-short-only}%
10335 {%
10336   \renewcommand*{\CustomAbbreviationFields}{%
10337     name={\glstronlname},
10338     sort={\the\glsshorttok},
10339     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10340     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10341     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10342     description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
10343   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10344     \glshasattribute{\the\glslabeltok}{regular}%
10345     {%
10346       \glissetattribute{\the\glslabeltok}{regular}{false}%
10347     }%
10348   }%
10349 }%
10350 }%
10351 {%
10352   \renewcommand*{\abbrvpluralsuffix}{\protect\glstronlxsuffix}%
10353   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
10354   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10355   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10356   \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
10357   \renewcommand*{\glsxtrfullformat}[2]{%
10358     \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10359     \ifglsxtrininsertinside\else##2\fi
10360   }%
10361   \renewcommand*{\glsxtrfullplformat}[2]{%
10362     \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10363     \ifglsxtrininsertinside\else##2\fi
10364   }%
10365   \renewcommand*{\Glsxtrfullformat}[2]{%
10366     \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10367     \ifglsxtrininsertinside\else##2\fi
10368   }%
10369   \renewcommand*{\Glsxtrfullplformat}[2]{%
10370     \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10371     \ifglsxtrininsertinside\else##2\fi
10372   }%
```

The inline full form does show the short form.

```
10373   \renewcommand*{\glsxtrinlinefullformat}[2]{%
10374     \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10375     \ifglsxtrininsertinside\else##2\fi
10376     \glsxtrfullsep{##1}%
```

```

10377 \glstrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10378 }%
10379 \renewcommand*{\glstrinlinefullplformat}[2]{%
10380 \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
10381 \ifglstrinsertinside\else##2\fi
10382 \glstrfullsep{##1}%
10383 \glstrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10384 }%
10385 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10386 \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglstrinsertinside##2\fi}%
10387 \ifglstrinsertinside\else##2\fi
10388 \glstrfullsep{##1}%
10389 \glstrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10390 }%
10391 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10392 \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglstrinsertinside##2\fi}%
10393 \ifglstrinsertinside\else##2\fi
10394 \glstrfullsep{##1}%
10395 \glstrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10396 }%
10397 }

```

xtronlydescsort

```

10398 \newcommand*{\glstronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

10399 \newcommand*{\glstronlydescname}{%
10400 \protect\glslongfont{\the\glslongtok}}%
10401 }

```

short-only-desc

```

10402 \newabbreviationstyle{long-only-short-only-desc}%
10403 {%
10404 \renewcommand*{\CustomAbbreviationFields}{%
10405 name={\glstronlydescname},
10406 sort={\glstronlydescsort},%
10407 first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10408 firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10409 text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
10410 plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
10411 }%

```

Unset the regular attribute if it has been set.

```

10412 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10413 \glshasattribute{\the\glslabeltok}{regular}%
10414 {%
10415 \glissetattribute{\the\glslabeltok}{regular}{false}%
10416 }%
10417 {}%

```

```

10418 }%
10419 }%
10420 {%
10421 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10422 }

```

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 `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref's \texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glstrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to `false`, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads `textcase`, so the label can be protected from case change with `textcase's \NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
10423 \let\@glstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

10424 \renewcommand*{\markright}[1]{%
10425 \glstrmarkhook
10426 \@glstr@org@markright{\@glstrinmark#1\@glstrnotinmark}%
10427 \glstrrestoremarkhook
10428 }

```

`\markboth` Save original definition:

```
10429 \let\@glstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```

10430 \renewcommand*{\markboth}[2]{%
10431 \glstrmarkhook

```

```

10432 \@glsxtr@org@markboth
10433   {\@glsxtrinmark#1\@glsxtrnotinmark}%
10434   {\@glsxtrinmark#2\@glsxtrnotinmark}%
10435 \glsxtrrestoremarkhook
10436 }

```

Also do this for \@starttoc

\@starttoc Save original definition:

```

10437 \let\@glsxtr@org@@starttoc\@starttoc

```

Redefine:

```

10438 \renewcommand*{\@starttoc}[1]{%
10439   \glsxtrmarkhook
10440   \@glsxtrinmark
10441   \@glsxtr@org@@starttoc{#1}%
10442   \@glsxtrnotinmark
10443   \glsxtrrestoremarkhook
10444 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

10445 \newcommand*{\glsxtrRevertMarks}{%
10446   \let\markright\@glsxtr@org@markright
10447   \let\markboth\@glsxtr@org@markboth
10448   \let\@starttoc\@glsxtr@org@@starttoc
10449 }

```

\glsxtrifinmark

```

10450 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

10451 \newrobustcmd*{\@glsxtrinmark}{%
10452   \let\glsxtrifinmark\@firstoftwo
10453 }

```

glsxtrnotinmark

```

10454 \newrobustcmd*{\@glsxtrnotinmark}{%
10455   \let\glsxtrifinmark\@secondoftwo
10456 }

```

eorpdforheading

```

10457 \ifdef\texorpdfstring
10458 {
10459   \newcommand*{\glsxtrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10460 }
10461 {
10462   \newcommand*{\glsxtrtitleorpdforheading}[3]{#1}
10463 }

```

`\glxtrmarkhook` Hook used in new definition of `\markboth` and `\markright` to make some changes to apply to the marks:

```
10464 \newcommand*{\glxtrmarkhook}{%
```

Save current definitions:

```
10465 \let\@glxtr@org@MakeUppercase\MakeUppercase
10466 \let\@glxtr@org@glxtrtitleorpdforheading\glxtrtitleorpdforheading
10467 \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
10468 \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
10469 \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10470 \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
10471 \let\@glxtr@org@glxtrtitlename\glxtrtitlename
10472 \let\@glxtr@org@Glsxtrtitlename\Glsxtrtitlename
10473 \let\@glxtr@org@glxtrtitletext\glxtrtitletext
10474 \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
10475 \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
10476 \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10477 \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
10478 \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10479 \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
10480 \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10481 \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
10482 \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl
10483 \let\@glxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
10484 \let\@glxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10485 \let\@glxtr@org@glxtrtitlefull\glxtrtitlefull
10486 \let\@glxtr@org@glxtrtitlefullpl\glxtrtitlefullpl
10487 \let\@glxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
10488 \let\@glxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

New definitions

```
10489 \let\glxtrifinmark\@firstoftwo
10490 \let\MakeUppercase\MakeTextUppercase
10491 \let\glxtrtitleorpdforheading\@thirdofthree
10492 \let\glxtrtitleshort\glxtrheadshort
10493 \let\glxtrtitleshortpl\glxtrheadshortpl
10494 \let\Glsxtrtitleshort\Glsxtrheadshort
10495 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
10496 \let\glxtrtitlename\glxtrheadname
10497 \let\Glsxtrtitlename\Glsxtrheadname
10498 \let\glxtrtitletext\glxtrheadtext
10499 \let\Glsxtrtitletext\Glsxtrheadtext
10500 \let\glxtrtitleplural\glxtrheadplural
10501 \let\Glsxtrtitleplural\Glsxtrheadplural
10502 \let\glxtrtitlefirst\glxtrheadfirst
10503 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10504 \let\glxtrtitlefirstplural\glxtrheadfirstplural
10505 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10506 \let\glxtrtitlelong\glxtrheadlong
10507 \let\glxtrtitlelongpl\glxtrheadlongpl
```

```

10508 \let\Glsxtrtitlelong\Glsxtrheadlong
10509 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10510 \let\glxstrtitlefull\glxstrheadfull
10511 \let\glxstrtitlefullpl\glxstrheadfullpl
10512 \let\Glsxtrtitlefull\Glsxtrheadfull
10513 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10514 }

```

`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.)

```

10515 \newcommand*{\glxstrrestoremarkhook}{%
10516 \let\glxstrifinmark\@secondoftwo
10517 \let\MakeUppercase\@glxstr@org@MakeUppercase
10518 \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
10519 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
10520 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
10521 \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
10522 \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
10523 \let\glxstrtitlename\@glxstr@org@glxstrtitlename
10524 \let\Glsxtrtitlename\@glxstr@org@Glsxtrtitlename
10525 \let\glxstrtitletext\@glxstr@org@glxstrtitletext
10526 \let\Glsxstrtitletext\@glxstr@org@Glsxstrtitletext
10527 \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
10528 \let\Glsxstrtitleplural\@glxstr@org@Glsxstrtitleplural
10529 \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
10530 \let\Glsxstrtitlefirst\@glxstr@org@Glsxstrtitlefirst
10531 \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
10532 \let\Glsxstrtitlefirstplural\@glxstr@org@Glsxstrtitlefirstplural
10533 \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
10534 \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
10535 \let\Glsxstrtitlelong\@glxstr@org@Glsxstrtitlelong
10536 \let\Glsxstrtitlelongpl\@glxstr@org@Glsxstrtitlelongpl
10537 \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
10538 \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
10539 \let\Glsxstrtitlefull\@glxstr@org@Glsxstrtitlefull
10540 \let\Glsxstrtitlefullpl\@glxstr@org@Glsxstrtitlefullpl
10541 }

```

Instead of using one document-wide conditional, use `headuc` attribute to determine whether or not to use the all upper case form.

`glxstrheadshort` Command used to display short form in the page header.

```

10542 \newcommand*{\glxstrheadshort}[1]{%
10543 \protect\NoCaseChange
10544 {%
10545 \gl@ifattribute{#1}{headuc}{true}%
10546 {%

```

```

10547 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10548 }%
10549 {%
10550 \glxtrshort[noindex,hyper=false]{#1}[]%
10551 }%
10552 }%
10553 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

10554 \newrobustcmd*{\lsxtrtitleshort}[1]{%
10555 \glxtrshort[noindex,hyper=false]{#1}[]%
10556 }

```

sxtrheadshortpl Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10557 \newcommand*{\glxtrheadshortpl}[1]{%
10558 \protect\NoCaseChange
10559 {%
10560 \gl@ifattribute{#1}{headuc}{true}%
10561 {%
10562 \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10563 }%
10564 {%
10565 \glxtrshortpl[noindex,hyper=false]{#1}[]%
10566 }%
10567 }%
10568 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

10569 \newrobustcmd*{\lsxtrtitleshortpl}[1]{%
10570 \glxtrshortpl[noindex,hyper=false]{#1}[]%
10571 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

10572 \newcommand*{\Glsxtrheadshort}[1]{%
10573 \protect\NoCaseChange
10574 {%
10575 \gl@ifattribute{#1}{headuc}{true}%
10576 {%
10577 \GLSxtrshort[noindex,hyper=false]{#1}[]%
10578 }%
10579 {%
10580 \Glsxtrshort[noindex,hyper=false]{#1}[]%
10581 }%
10582 }%
10583 }

```

`\glxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10584 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10585   \Glsxtrshort[noindex,hyper=false]{#1}[]%
10586 }
```

`\glxtrheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```
10587 \newcommand*{\Glsxtrheadshortpl}[1]{%
10588   \protect\NoCaseChange
10589   {%
10590     \glsifattribute{#1}{headuc}{true}%
10591     {%
10592       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10593     }%
10594     {%
10595       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10596     }%
10597   }%
10598 }
```

`\glxtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10599 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10600   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10601 }
```

`\glxtrheadname` As above but for the name value.

```
10602 \newcommand*{\glxtrheadname}[1]{%
10603   \protect\NoCaseChange
10604   {%
10605     \glsifattribute{#1}{headuc}{true}%
10606     {%
10607       \GLSname[noindex,hyper=false]{#1}[]%
10608     }%
10609     {%
10610       \glsname[noindex,hyper=false]{#1}[]%
10611     }%
10612   }%
10613 }
```

`\glxtrtitlename` Command to display name value in section title and table of contents.

```
10614 \newrobustcmd*{\glxtrtitlename}[1]{%
10615   \glsname[noindex,hyper=false]{#1}[]%
10616 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10617 \newcommand*{\Glsxtrheadname}[1]{%
```



```

10618 \protect\NoCaseChange
10619 {%
10620   \glsifattribute{#1}{headuc}{true}%
10621   {%
10622     \GLSname[noindex,hyper=false]{#1}[]%
10623   }%
10624   {%
10625     \Glsname[noindex,hyper=false]{#1}[]%
10626   }%
10627 }%
10628 }

```

Glsxtrtitlename Command to display name value in section title and table of contents with the first letter changed to upper case.

```

10629 %\changes{1.21}{2017-11-03}{new}
10630 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10631   \GLSname[noindex,hyper=false]{#1}[]%
10632 }

```

\glsxtrheadtext As above but for the text value.

```

10633 \newcommand*{\glsxtrheadtext}[1]{%
10634   \protect\NoCaseChange
10635   {%
10636     \glsifattribute{#1}{headuc}{true}%
10637     {%
10638       \GLStext[noindex,hyper=false]{#1}[]%
10639     }%
10640     {%
10641       \glstext[noindex,hyper=false]{#1}[]%
10642     }%
10643   }%
10644 }

```

glsxtrtitletext Command to display text value in section title and table of contents.

```

10645 \newrobustcmd*{\glsxtrtitletext}[1]{%
10646   \glstext[noindex,hyper=false]{#1}[]%
10647 }

```

\Glsxtrheadtext First letter converted to upper case

```

10648 \newcommand*{\Glsxtrheadtext}[1]{%
10649   \protect\NoCaseChange
10650   {%
10651     \glsifattribute{#1}{headuc}{true}%
10652     {%
10653       \GLStext[noindex,hyper=false]{#1}[]%
10654     }%
10655     {%
10656       \Glsxtrheadtext[noindex,hyper=false]{#1}[]%
10657     }%

```

```
10658 }%
10659 }
```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```
10660 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10661   \Gls{text[noindex,hyper=false]}{#1}[]%
10662 }
```

lsxtrheadplural As above but for the plural value.

```
10663 \newcommand*{\Glsxtrheadplural}[1]{%
10664   \protect\NoCaseChange
10665   {%
10666     \glsifattribute{#1}{headuc}{true}%
10667     {%
10668       \GLSplural[noindex,hyper=false]{#1}[]%
10669     }%
10670     {%
10671       \glsplural[noindex,hyper=false]{#1}[]%
10672     }%
10673   }%
10674 }
```

sxtrtitleplural Command to display plural value in section title and table of contents.

```
10675 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10676   \glsplural[noindex,hyper=false]{#1}[]%
10677 }
```

lsxtrheadplural Convert first letter to upper case.

```
10678 \newcommand*{\Glsxtrheadplural}[1]{%
10679   \protect\NoCaseChange
10680   {%
10681     \glsifattribute{#1}{headuc}{true}%
10682     {%
10683       \GLSplural[noindex,hyper=false]{#1}[]%
10684     }%
10685     {%
10686       \Glsplural[noindex,hyper=false]{#1}[]%
10687     }%
10688   }%
10689 }
```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
10690 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10691   \Glsplural[noindex,hyper=false]{#1}[]%
10692 }
```

glsxtrheadfirst As above but for the first value.

```
10693 \newcommand*{\glsxtrheadfirst}[1]{%
10694   \protect\NoCaseChange
10695   {%
10696     \glsifattribute{#1}{headuc}{true}%
10697     {%
10698       \GLSfirst[noindex,hyper=false]{#1}[]%
10699     }%
10700     {%
10701       \glsfirst[noindex,hyper=false]{#1}[]%
10702     }%
10703   }%
10704 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
10705 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10706   \glsfirst[noindex,hyper=false]{#1}[]%
10707 }
```

Glsxtrheadfirst First letter converted to upper case

```
10708 \newcommand*{\Glsxtrheadfirst}[1]{%
10709   \protect\NoCaseChange
10710   {%
10711     \glsifattribute{#1}{headuc}{true}%
10712     {%
10713       \GLSfirst[noindex,hyper=false]{#1}[]%
10714     }%
10715     {%
10716       \Glsfirst[noindex,hyper=false]{#1}[]%
10717     }%
10718   }%
10719 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10720 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10721   \Glsfirst[noindex,hyper=false]{#1}[]%
10722 }
```

headfirstplural As above but for the firstplural value.

```
10723 \newcommand*{\glsxtrheadfirstplural}[1]{%
10724   \protect\NoCaseChange
10725   {%
10726     \glsifattribute{#1}{headuc}{true}%
10727     {%
10728       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10729     }%
10730     {%
10731       \Glsfirstplural[noindex,hyper=false]{#1}[]%
10732     }%
10733   }%
10734 }
```

```

10731 \glsfirstplural[noindex,hyper=false]{#1}[]%
10732 }%
10733 }%
10734 }

```

titlefirstplural Command to display firstplural value in section title and table of contents.

```

10735 \newrobustcmd*{\glsxtrtitlefirstplural}[1]{%
10736 \glsfirstplural[noindex,hyper=false]{#1}[]%
10737 }

```

headfirstplural First letter converted to upper case

```

10738 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10739 \protect\NoCaseChange
10740 {%
10741 \glsifattribute{#1}{headuc}{true}%
10742 {%
10743 \GLSfirstplural[noindex,hyper=false]{#1}[]%
10744 }%
10745 {%
10746 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10747 }%
10748 }%
10749 }

```

titlefirstplural Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10750 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10751 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10752 }

```

\glsxtrheadlong Command used to display long form in the page header.

```

10753 \newcommand*{\glsxtrheadlong}[1]{%
10754 \protect\NoCaseChange
10755 {%
10756 \glsifattribute{#1}{headuc}{true}%
10757 {%
10758 \GLSxtrlong[noindex,hyper=false]{#1}[]%
10759 }%
10760 {%
10761 \glsxtrlong[noindex,hyper=false]{#1}[]%
10762 }%
10763 }%
10764 }

```

glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents.

```

10765 \newrobustcmd*{\glsxtrtitlelong}[1]{%
10766 \glsxtrlong[noindex,hyper=false]{#1}[]%
10767 }

```

`lsxtrheadlongpl` Command used to display plural long form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrlongpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10768 \newcommand*{\glxtrheadlongpl}[1]{%
10769   \protect\NoCaseChange
10770   {%
10771     \gl@ifattribute{#1}{headuc}{true}%
10772     {%
10773       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10774     }%
10775     {%
10776       \glxtrlongpl[noindex,hyper=false]{#1}[]%
10777     }%
10778   }%
10779 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

10780 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
10781   \glxtrlongpl[noindex,hyper=false]{#1}[]%
10782 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

10783 \newcommand*{\Glsxtrheadlong}[1]{%
10784   \protect\NoCaseChange
10785   {%
10786     \gl@ifattribute{#1}{headuc}{true}%
10787     {%
10788       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10789     }%
10790     {%
10791       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10792     }%
10793   }%
10794 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10795 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10796   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10797 }
```

`lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

10798 \newcommand*{\Glsxtrheadlongpl}[1]{%
10799   \protect\NoCaseChange
10800   {%
10801     \gl@ifattribute{#1}{headuc}{true}%
10802     {%
10803       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10804     }%
10805     {%
10806       \glxtrlongpl[noindex,hyper=false]{#1}[]%
10807     }%
10808   }%
10809 }
```

```

10802   {%
10803     \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10804   }%
10805   {%
10806     \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10807   }%
10808 }%
10809 }

```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10810 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10811   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10812 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

10813 \newcommand*{\glsxtrheadfull}[1]{%
10814   \protect\NoCaseChange
10815   {%
10816     \glsifattribute{#1}{headuc}{true}%
10817     {%
10818       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10819     }%
10820     {%
10821       \glsxtrfull[noindex,hyper=false]{#1}[]%
10822     }%
10823   }%
10824 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

10825 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10826   \glsxtrfull[noindex,hyper=false]{#1}[]%
10827 }

```

lsxtrheadfullpl 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.

```

10828 \newcommand*{\glsxtrheadfullpl}[1]{%
10829   \protect\NoCaseChange
10830   {%
10831     \glsifattribute{#1}{headuc}{true}%
10832     {%
10833       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10834     }%
10835     {%
10836       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10837     }%
10838   }%
10839 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

10840 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10841   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10842 }

```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

10843 \newcommand*{\Glsxtrheadfull}[1]{%
10844   \protect\NoCaseChange
10845   {%
10846     \glsifattribute{#1}{headuc}{true}%
10847     {%
10848       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10849     }%
10850     {%
10851       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10852     }%
10853   }%
10854 }

```

`Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10855 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10856   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10857 }

```

`lGlsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```

10858 \newcommand*{\Glsxtrheadfullpl}[1]{%
10859   \protect\NoCaseChange
10860   {%
10861     \glsifattribute{#1}{headuc}{true}%
10862     {%
10863       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10864     }%
10865     {%
10866       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10867     }%
10868   }%
10869 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10870 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10871   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10872 }

```

`\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.

```
10873 \ifdef\texorpdfstring
10874 {
10875   \newcommand*\glsfmtshort}[1]{%
10876     \texorpdfstring
10877       {\glsxtrtitleshort{#1}}%
10878       {\glsentryshort{#1}}%
10879   }
10880 }
10881 {
10882   \newcommand*\glsfmtshort}[1]{%
10883     \glsxtrtitleshort{#1}}
10884 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
10885 \ifdef\texorpdfstring
10886 {
10887   \newcommand*\glsfmtshortpl}[1]{%
10888     \texorpdfstring
10889       {\glsxtrtitleshortpl{#1}}%
10890       {\glsentryshortpl{#1}}%
10891   }
10892 }
10893 {
10894   \newcommand*\glsfmtshortpl}[1]{%
10895     \glsxtrtitleshortpl{#1}}
10896 }
```

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).

```
10897 \ifdef\texorpdfstring
10898 {
10899   \newcommand*\Glsfmtshort}[1]{%
10900     \texorpdfstring
10901       {\Glsxtrtitleshort{#1}}%
10902       {\glsentryshort{#1}}%
10903   }
10904 }
10905 {
10906   \newcommand*\Glsfmtshort}[1]{%
10907     \Glsxtrtitleshort{#1}}
10908 }
```

`\Glsfmtshortpl` Plural form (first letter uppercase).


```

10909 \ifdef\texorpdfstring
10910 {
10911   \newcommand*\Glsfmtshortpl}[1]{%
10912     \texorpdfstring
10913     {\Glsxtrtitleshortpl{#1}}%
10914     {\glsentryshortpl{#1}}%
10915   }
10916 }
10917 {
10918   \newcommand*\Glsfmtshortpl}[1]{%
10919     \Glsxtrtitleshortpl{#1}}
10920 }

```

`\glsfmtname` As above but for the name value.

```

10921 \ifdef\texorpdfstring
10922 {
10923   \newcommand*\glsfmtname}[1]{%
10924     \texorpdfstring
10925     {\glsxtrtitlename{#1}}%
10926     {\glsentryname{#1}}%
10927   }
10928 }
10929 {
10930   \newcommand*\glsfmtname}[1]{%
10931     \glsxtrtitlename{#1}}
10932 }

```

`\Glsfmtname` First letter converted to upper case.

```

10933 \ifdef\texorpdfstring
10934 {
10935   \newcommand*\Glsfmtname}[1]{%
10936     \texorpdfstring
10937     {\Glsxtrtitlename{#1}}%
10938     {\glsentryname{#1}}%
10939   }
10940 }
10941 {
10942   \newcommand*\Glsfmtname}[1]{%
10943     \Glsxtrtitlename{#1}}
10944 }

```

`\glsfmttext` As above but for the text value.

```

10945 \ifdef\texorpdfstring
10946 {
10947   \newcommand*\glsfmttext}[1]{%
10948     \texorpdfstring
10949     {\glsxtrtitletext{#1}}%
10950     {\glsentrytext{#1}}%
10951   }

```

```

10952 }
10953 {
10954   \newcommand*{\glsfmttext}[1]{%
10955     \glsxtrtitletext{#1}}
10956 }

```

`\Glsfmttext` First letter converted to upper case.

```

10957 \ifdef\texorpdfstring
10958 {
10959   \newcommand*{\Glsfmttext}[1]{%
10960     \texorpdfstring
10961     {\Glsxtrtitletext{#1}}%
10962     {\glsentrytext{#1}}%
10963   }
10964 }
10965 {
10966   \newcommand*{\Glsfmttext}[1]{%
10967     \Glsxtrtitletext{#1}}
10968 }

```

`\glsfmtplural` As above but for the plural value.

```

10969 \ifdef\texorpdfstring
10970 {
10971   \newcommand*{\glsfmtplural}[1]{%
10972     \texorpdfstring
10973     {\glsxtrtitleplural{#1}}%
10974     {\glsentryplural{#1}}%
10975   }
10976 }
10977 {
10978   \newcommand*{\glsfmtplural}[1]{%
10979     \glsxtrtitleplural{#1}}
10980 }

```

`\Glsfmtplural` First letter converted to upper case.

```

10981 \ifdef\texorpdfstring
10982 {
10983   \newcommand*{\Glsfmtplural}[1]{%
10984     \texorpdfstring
10985     {\Glsxtrtitleplural{#1}}%
10986     {\glsentryplural{#1}}%
10987   }
10988 }
10989 {
10990   \newcommand*{\Glsfmtplural}[1]{%
10991     \Glsxtrtitleplural{#1}}
10992 }

```

`\glsfmtfirst` As above but for the first value.

```

10993 \ifdef\teorpdfstring
10994 {
10995   \newcommand*\glsfmtfirst}[1]{%
10996     \teorpdfstring
10997     {\glsxtrtitlefirst{#1}}%
10998     {\glsentryfirst{#1}}%
10999   }
11000 }
11001 {
11002   \newcommand*\glsfmtfirst}[1]{%
11003     \glsxtrtitlefirst{#1}}
11004 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

11005 \ifdef\teorpdfstring
11006 {
11007   \newcommand*\Glsfmtfirst}[1]{%
11008     \teorpdfstring
11009     {\Glsxtrtitlefirst{#1}}%
11010     {\glsentryfirst{#1}}%
11011   }
11012 }
11013 {
11014   \newcommand*\Glsfmtfirst}[1]{%
11015     \Glsxtrtitlefirst{#1}}
11016 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

11017 \ifdef\teorpdfstring
11018 {
11019   \newcommand*\glsfmtfirstpl}[1]{%
11020     \teorpdfstring
11021     {\glsxtrtitlefirstplural{#1}}%
11022     {\glsentryfirstplural{#1}}%
11023   }
11024 }
11025 {
11026   \newcommand*\glsfmtfirstpl}[1]{%
11027     \glsxtrtitlefirstplural{#1}}
11028 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

11029 \ifdef\teorpdfstring
11030 {
11031   \newcommand*\Glsfmtfirstpl}[1]{%
11032     \teorpdfstring
11033     {\Glsxtrtitlefirstplural{#1}}%
11034     {\glsentryfirstplural{#1}}%
11035   }

```

```

11036 }
11037 {
11038   \newcommand*{\Glsfmtfirstpl}[1]{%
11039     \Glsxtrtitlefirstplural{#1}}
11040 }

```

`\glsfmtlong` As above but for the long value.

```

11041 \ifdef\teorpdfstring
11042 {
11043   \newcommand*{\glsfmtlong}[1]{%
11044     \teorpdfstring
11045     {\Glsxtrtitlelong{#1}}%
11046     {\Glsentrylong{#1}}%
11047   }
11048 }
11049 {
11050   \newcommand*{\glsfmtlong}[1]{%
11051     \Glsxtrtitlelong{#1}}
11052 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11053 \ifdef\teorpdfstring
11054 {
11055   \newcommand*{\Glsfmtlong}[1]{%
11056     \teorpdfstring
11057     {\Glsxtrtitlelong{#1}}%
11058     {\Glsentrylong{#1}}%
11059   }
11060 }
11061 {
11062   \newcommand*{\Glsfmtlong}[1]{%
11063     \Glsxtrtitlelong{#1}}
11064 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11065 \ifdef\teorpdfstring
11066 {
11067   \newcommand*{\glsfmtlongpl}[1]{%
11068     \teorpdfstring
11069     {\Glsxtrtitlelongpl{#1}}%
11070     {\Glsentrylongpl{#1}}%
11071   }
11072 }
11073 {
11074   \newcommand*{\glsfmtlongpl}[1]{%
11075     \Glsxtrtitlelongpl{#1}}
11076 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11077 \ifdef\teorpdfstring
11078 {
11079   \newcommand*{\Glsfmtlongpl}[1]{%
11080     \teorpdfstring
11081     {\Glsxtrtitlelongpl{#1}}%
11082     {\glsentrylongpl{#1}}%
11083   }
11084 }
11085 {
11086   \newcommand*{\Glsfmtlongpl}[1]{%
11087     \Glsxtrtitlelongpl{#1}}
11088 }

```

`\glsfmtfull` In-line full format.

```

11089 \ifdef\teorpdfstring
11090 {
11091   \newcommand*{\glsfmtfull}[1]{%
11092     \teorpdfstring
11093     {\glsxtrtitlefull{#1}}%
11094     {\glsxtrinlinefullformat{#1}{}}%
11095   }
11096 }
11097 {
11098   \newcommand*{\glsfmtfull}[1]{%
11099     \glsxtrtitlefull{#1}}
11100 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11101 \ifdef\teorpdfstring
11102 {
11103   \newcommand*{\Glsfmtfull}[1]{%
11104     \teorpdfstring
11105     {\Glsxtrtitlefull{#1}}%
11106     {\Glsxtrinlinefullformat{#1}{}}%
11107   }
11108 }
11109 {
11110   \newcommand*{\Glsfmtfull}[1]{%
11111     \Glsxtrtitlefull{#1}}
11112 }

```

`\glsfmtfullpl` In-line full plural format.

```

11113 \ifdef\teorpdfstring
11114 {
11115   \newcommand*{\glsfmtfullpl}[1]{%
11116     \teorpdfstring
11117     {\glsxtrtitlefullpl{#1}}%
11118     {\glsxtrinlinefullplformat{#1}{}}%
11119   }

```

```

11120 }
11121 {
11122   \newcommand*{\Glsfmtfullpl}[1]{%
11123     \Glsxtrtitlefullpl{#1}}
11124 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11125 \ifdef\teorpdfstring
11126 {
11127   \newcommand*{\Glsfmtfullpl}[1]{%
11128     \teorpdfstring
11129     {\Glsxtrtitlefullpl{#1}}%
11130     {\Glsxtrinlinefullplformat{#1}{}}%
11131   }
11132 }
11133 {
11134   \newcommand*{\Glsfmtfullpl}[1]{%
11135     \Glsxtrtitlefullpl{#1}}
11136 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

11137 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11138   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
11139 }

```

`sariesExtraLang`

```

11140 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11141   \ProvidesFile{glossariesxtr-#1.ldf}%
11142 }

```

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.)

```

11143 \@ifpackageloaded{tracklang}
11144 {%
11145   \AnyTrackedLanguages
11146   {%
11147     \ForEachTrackedDialect{\this@dialect}{%
11148       \IfTrackedLanguageFileExists{\this@dialect}%
11149       {glossariesxtr-}% prefix
11150       {.ldf}%
11151     }

```

```

11152         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11153     }%
11154     {%
11155     }%
11156 }%
11157 }%
11158 {}%
11159 }
11160 {}

    Load glossaries-extra-stylemods if required.
11161 \@glstr@redefstyles
    and set the style:
11162 \@glstr@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:

```
11163 \NeedsTeXFormat{LaTeX2e}
11164 \ProvidesPackage{glossaries-extra-stylemods}[2017/11/24 v1.25 (NLCT)]
```

Provide package options to automatically load required predefined styles. The simplest method is to just test for the existence of the file `glossary-<option>.sty`. Packages can't be loaded whilst the options are being processed, so save the list in `\@glxtr@loadstyles`.

`sxtr@loadstyles`

```
11165 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
11166 \DeclareOption{all}{%
11167   \appto\@glxtr@loadstyles{%
11168     \RequirePackage{glossary-inline}%
11169     \RequirePackage{glossary-list}%
11170     \RequirePackage{glossary-tree}%
11171     \RequirePackage{glossary-mcols}%
11172     \RequirePackage{glossary-long}%
11173     \RequirePackage{glossary-longragged}%
11174     \RequirePackage{glossary-longbooktabs}%
11175     \RequirePackage{glossary-super}%
11176     \RequirePackage{glossary-superragged}%
11177     \RequirePackage{glossary-bookindex}%
11178   }
11179 }

11180 \DeclareOption*{%
11181   \IfFileExists{glossary-\CurrentOption.sty}
11182   {\eappto\@glxtr@loadstyles{%
11183     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
11184   }%
11185   {%
11186     \PackageError{glossaries-extra-styles}%

```



```

11187     {Unknown option ‘\CurrentOption’}{}%
11188   }%
11189 }

```

Process the package options:

```
11190 \ProcessOptions
```

Load the required packages:

```
11191 \@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`.

```
11192 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

11193 \providecommand{\renewglossarystyle}[2]{%
11194   \ifcsundef{@glsstyle@#1}%
11195   {%
11196     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
11197   }%
11198   {%
11199     \csdef{@glsstyle@#1}{#2}%
11200   }%
11201 }

```

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.

```

11202 \ifdef{\@glsstyle@listdotted}
11203 {%
11204   \renewglossarystyle{listdotted}{%
11205     \setglossarystyle{list}%
11206     \renewcommand*{\glossentry}[2]{%
11207       \item[]\makebox[\glslistdottedwidth][l]{%
11208         \glsentryitem{##1}%
11209         \glstarget{##1}{\glossentryname{##1}}%
11210         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11211         \glossentrydesc{##1}\glspostdescription}%
11212     \renewcommand*{\subglossentry}[3]{%
11213       \item[]\makebox[\glslistdottedwidth][l]{%
11214         \glssubentryitem{##2}%
11215         \glstarget{##2}{\glossentryname{##2}}%
11216         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11217         \glossentrydesc{##2}\glspostdescription}%
11218   }

```

```
11219 }
11220 {%
```

Assume the style isn't required if it hasn't already been defined.

```
11221 }
```

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.

```
11222 \ifdef{\@glsstyle@list}
11223 {%
```

listprelocation Space before number list for top-level entries.

```
11224 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11225 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11226 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11227 \renewglossarystyle{list}{%
11228 \renewenvironment{theglossary}%
11229 {\begin{description}}{\end{description}}%
11230 \renewcommand*\{glossaryheader}{}%
11231 \renewcommand*\{glsgroupheading}[1]{}%
11232 \renewcommand*\{glossentry}[2]{%
11233 \item[\glsentryitem{##1}]%
11234 \glstarget{##1}{\glossentryname{##1}}]
11235 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11236 \renewcommand*\{subglossentry}[3]{%
11237 \glssubentryitem{##2}%
11238 \glstarget{##2}{\strut}\space
11239 \glossentrydesc{##2}\glspostdescription
11240 \glslistchildprelocation ##3\glslistchildpostlocation}%
11241 \renewcommand*\{glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11242 }
11243 }
11244 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11245 \ifdef{\@glsstyle@altlist}
11246 {%
11247 \renewglossarystyle{altlist}{%
11248 \setglossarystyle{list}%
11249 \renewcommand*\{glossentry}[2]{%
11250 \item[\glsentryitem{##1}]%
```

```

11251      \glstarget{##1}{\glossentryname{##1}}}%
11252      \mbox{}\par\nobreak\@afterheading
11253      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11254 \renewcommand{\subglossentry}[3]{%
11255     \par
11256     \glssubentryitem{##2}%
11257     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11258     \glslistchildprelocation ##3}%
11259 }
11260 }
11261 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11262 \ifdef{\@glsstyle@listgroup}
11263 {%
11264   \renewglossarystyle{listgroup}{%
11265     \setglossarystyle{list}%
11266     \renewcommand*\glsgroupheading[1]{%
11267       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11268       \mbox{}\par\nobreak\@afterheading
11269     }%
11270   }
11271 }
11272 {}

```

Similarly for listhypergroup.

```

11273 \ifdef{\@glsstyle@listhypergroup}
11274 {%
11275   \renewglossarystyle{listhypergroup}{%
11276     \setglossarystyle{list}%
11277     \renewcommand*\glossaryheader{%
11278       \glslistnavigationitem{\glsnavigation}}%
11279     \renewcommand*\glsgroupheading[1]{%
11280       \item[\glslistgroupheaderfmt
11281         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11282       \mbox{}\par\nobreak\@afterheading
11283     }%
11284   }
11285 }
11286 {}

```

Similarly for altlistgroup.

```

11287 \ifdef{\@glsstyle@altlistgroup}
11288 {%
11289   \renewglossarystyle{altlistgroup}{%
11290     \setglossarystyle{altlist}%
11291     \renewcommand*\glsgroupheading[1]{%
11292       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11293       \mbox{}\par\nobreak\@afterheading
11294     }%
11295   }

```

```

11296 }
11297 {}

    Similarly for altlisthypergroup.
11298 \ifdef{\@glsstyle@altlisthypergroup}
11299 {%
11300   \renewglossarystyle{altlisthypergroup}{%
11301     \setglossarystyle{altlist}%
11302     \renewcommand*\{glossaryheader\}{%
11303       \glslistnavigationitem{\glsnavigation}}%
11304     \renewcommand*\{glsgroupheading\}[1]{%
11305       \item[\glslistgroupheaderfmt
11306         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11307       \mbox{}\par\nobreak\@afterheading
11308     }%
11309   }
11310 }
11311 {}

```

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`.

```

11312 \ifcsdef{@glsstyle@long}
11313 {%
11314   \renewglossarystyle{long}{%
11315     \renewenvironment{theglossary}%
11316       {\begin{longtable}{lp{\glsdescwidth}}}%
11317       {\end{longtable}}%
11318     \renewcommand*\{glossaryheader\}{}%
11319     \renewcommand*\{glsgroupheading\}[1]{}%
11320     \renewcommand{\glossentry}[2]{%
11321       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11322       \glossentrydesc{##1}\glspostdescription
11323       \glstrprelocation ##2\tabularnewline
11324     }%
11325     \renewcommand{\subglossentry}[3]{%
11326       &
11327       \glssubentryitem{##2}%
11328       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11329       \glstrprelocation ##3\tabularnewline
11330     }%
11331     \ifglsnogroupskip
11332       \renewcommand*\{glsgroupskip\}{}%
11333     \else
11334       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11335     \fi
11336   }

```

```
11337 }
11338 {}
```

Three column style:

```
11339 \ifcsdef{@glsstyle@long3col}
11340 {%
11341   \renewglossarystyle{long3col}{%
11342     \renewenvironment{theglossary}%
11343       {\begin{longtable}{lp{@glsdescwidth}p{@glspagelistwidth}}}%
11344       {\end{longtable}}}%
11345     \renewcommand*{\glossaryheader}{}%
11346     \renewcommand*{\glsgroupheading}[1]{}%
11347     \renewcommand{\glossentry}[2]{%
11348       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11349       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11350     }%
11351     \renewcommand{\subglossentry}[3]{%
11352       &
11353       \glssubentryitem{##2}%
11354       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11355       ##3\tabularnewline
11356     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11357   \ifglsnogroupskip
11358     \renewcommand*{\glsgroupskip}{}%
11359   \else
11360     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11361   \fi
11362 }
11363 }
11364 {}
```

Four column style:

```
11365 \ifcsdef{@glsstyle@long4col}
11366 {%
11367   \renewglossarystyle{long4col}{%
11368     \renewenvironment{theglossary}%
11369       {\begin{longtable}{llll}}%
11370       {\end{longtable}}}%
11371     \renewcommand*{\glossaryheader}{}%
11372     \renewcommand*{\glsgroupheading}[1]{}%
11373     \renewcommand{\glossentry}[2]{%
11374       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11375       \glossentrydesc{##1}\glspostdescription &
11376       \glossentrysymbol{##1} &
11377       ##2\tabularnewline
11378     }%
11379     \renewcommand{\subglossentry}[3]{%
11380       &
```

```

11381     \glssubentryitem{##2}%
11382     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11383     \glossentrysymbol{##2} & ##3\tabularnewline
11384 }%

11385 \ifglsnogroupskip
11386   \renewcommand*{\glsgroupskip}{}%
11387 \else
11388   \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11389 \fi
11390 }
11391 }
11392 {}

```

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`.

```

11393 \ifcsdef{@glstyle@longragged}
11394 {%
11395   \renewglossarystyle{longragged}{%
11396     \renewenvironment{theglossary}%
11397       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
11398       {\end{longtable}}%
11399     \renewcommand*{\glossaryheader}{}%
11400     \renewcommand*{\glsgroupheading}[1]{}%
11401     \renewcommand{\glossentry}[2]{%
11402       \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11403       \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11404       \tabularnewline
11405     }%
11406     \renewcommand{\subglossentry}[3]{%
11407       &
11408       \glssubentryitem{##2}%
11409       \glstarget{##2}{\strut}\glossentrydesc{##2}%
11410       \glspostdescription\glxtrprelocation ##3%
11411       \tabularnewline
11412     }%
11413     \ifglsnogroupskip
11414       \renewcommand*{\glsgroupskip}{}%
11415     \else
11416       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11417     \fi
11418   }
11419 }

```

11420 {}

Three and four column styles don't use `\glstrprelocation` since the number list is in its own column.

```
11421 \ifcsdef{@glsstyle@longragged3col}
11422 {%
11423   \renewglossarystyle{longragged3col}{%
11424     \renewenvironment{theglossary}%
11425       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11426         >{\raggedright}p{\glspagelistwidth}}}%
11427       {\end{longtable}}}%
11428     \renewcommand*{\glossaryheader}{}%
11429     \renewcommand*{\glsgroupheading}[1]{}%
11430     \renewcommand{\glossentry}[2]{%
11431       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11432       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11433     }%
11434     \renewcommand{\subglossentry}[3]{%
11435       &
11436       \glssubentryitem{##2}%
11437       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11438       ##3\tabularnewline
11439     }%
11440     \ifglsgroupskip
11441       \renewcommand*{\glsgroupskip}{}%
11442     \else
11443       \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11444     \fi
11445   }
11446 }
11447 {}
```

Four column style:

```
11448 \ifcsdef{@glsstyle@altlongragged4col}
11449 {%
11450   \renewglossarystyle{altlongragged4col}{%
11451     \renewenvironment{theglossary}%
11452       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11453         >{\raggedright}p{\glspagelistwidth}}}%
11454       {\end{longtable}}}%
11455     \renewcommand*{\glossaryheader}{}%
11456     \renewcommand*{\glsgroupheading}[1]{}%
11457     \renewcommand{\glossentry}[2]{%
11458       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11459       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11460       ##2\tabularnewline
11461     }%
11462     \renewcommand{\subglossentry}[3]{%
11463       &
```

```

11464     \glssubentryitem{##2}%
11465     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11466     \glossentrysymbol{##2} & ##3\tabularnewline
11467 }%

11468 \ifglsnogroupskip
11469     \renewcommand*{\glsgroupskip}{}%
11470 \else
11471     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11472 \fi
11473 }
11474 }
11475 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glstrprelocation`.

```

11476 \ifcsdef{@glstyle@super}
11477 {%
11478     \renewglossarystyle{super}{%
11479         \renewenvironment{theglossary}%
11480             {\tablehead{}\tabletail}%
11481             \begin{supertabular}{lp{\glsdescwidth}}%
11482             {\end{supertabular}}%
11483         \renewcommand*{\glossaryheader}{}%
11484         \renewcommand*{\glsgroupheading}[1]{}%
11485         \renewcommand{\glossentry}[2]{%
11486             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11487             \glossentrydesc{##1}\glspostdescription
11488             \glstrprelocation ##2\tabularnewline
11489         }%
11490         \renewcommand{\subglossentry}[3]{%
11491             &
11492             \glssubentryitem{##2}%
11493             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11494             \glstrprelocation ##3\tabularnewline
11495         }%
11496         \ifglsnogroupskip
11497             \renewcommand*{\glsgroupskip}{}%
11498         \else
11499             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11500         \fi
11501     }
11502 }
11503 {}

```

Three column style:

```

11504 \ifcsdef{@glstyle@super3col}

```



```

11505 {%
11506   \renewglossarystyle{super3col}{%
11507     \renewenvironment{theglossary}%
11508       {\tablehead{}\tabletail{}}%
11509       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
11510       {\end{supertabular}}}%
11511   \renewcommand*{\glossaryheader}{}%
11512   \renewcommand*{\glsgroupheading}[1]{}%
11513   \renewcommand{\glossentry}[2]{%
11514     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11515     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11516   }%
11517   \renewcommand{\subglossentry}[3]{%
11518     &
11519     \glssubentryitem{##2}%
11520     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11521     ##3\tabularnewline
11522   }%

11523   \ifglsgnogroupskip
11524     \renewcommand*{\glsgroupskip}{}%
11525   \else
11526     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11527   \fi
11528 }
11529 }
11530 {}

```

Four column styles:

```

11531 \ifcsdef{@glstyle@super4col}
11532 {%
11533   \renewglossarystyle{super4col}{%
11534     \renewenvironment{theglossary}%
11535       {\tablehead{}\tabletail{}}%
11536       \begin{supertabular}{llll}}{%
11537       \end{supertabular}}}%
11538   \renewcommand*{\glossaryheader}{}%
11539   \renewcommand*{\glsgroupheading}[1]{}%
11540   \renewcommand{\glossentry}[2]{%
11541     \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11542     \glossentrydesc{##1}\glspostdescription &
11543     \glossentrysymbol{##1} & ##2\tabularnewline
11544   }%
11545   \renewcommand{\subglossentry}[3]{%
11546     &
11547     \glssubentryitem{##2}%
11548     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11549     \glossentrysymbol{##2} & ##3\tabularnewline
11550   }%

```

```

11551 \ifglsgroupskip
11552 \renewcommand*{\glsgroupskip}{}%
11553 \else
11554 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11555 \fi
11556 }
11557 }
11558 {}

```

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`.

```

11559 \ifcsdef{@glstyle@superragged}
11560 {%
11561 \renewglossarystyle{superragged}{%
11562 \renewenvironment{theglossary}%
11563 {\tablehead{}\tabletail{}}%
11564 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11565 {\end{supertabular}}%
11566 \renewcommand*{\glossaryheader}{}%
11567 \renewcommand*{\glsgroupheading}[1]{}%
11568 \renewcommand{\glossentry}[2]{%
11569 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11570 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11571 \tabularnewline
11572 }%
11573 \renewcommand{\subglossentry}[3]{%
11574 &
11575 \glssubentryitem{##2}%
11576 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11577 \glstrprelocation ##3%
11578 \tabularnewline
11579 }%
11580 \ifglsgroupskip
11581 \renewcommand*{\glsgroupskip}{}%
11582 \else
11583 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11584 \fi
11585 }
11586 }
11587 {}

```

Three column style:

```

11588 \ifcsdef{@glstyle@superragged3col}
11589 {%
11590 \renewglossarystyle{superragged3col}{%

```

```

11591 \renewenvironment{theglossary}%
11592   {\tablehead{}\tabletail{}}%
11593   \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
11594     >{\raggedright}p{\glspagelistwidth}}}%
11595   {\end{supertabular}}%
11596 \renewcommand*{\glossaryheader}{}%
11597 \renewcommand*{\glsgroupheading}[1]{}%
11598 \renewcommand{\glossentry}[2]{%
11599   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11600   \glossentrydesc{##1}\glspostdescription &
11601   ##2\tabularnewline
11602 }%
11603 \renewcommand{\subglossentry}[3]{%
11604   &
11605   \glssubentryitem{##2}%
11606   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11607   ##3\tabularnewline
11608 }%
11609 \ifglsnogroupskip
11610   \renewcommand*{\glsgroupskip}{}%
11611 \else
11612   \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
11613 \fi
11614 }
11615 }
11616 {}

```

Four columns:

```

11617 \ifcsdef{@glsstyle@altsuperragged4col}
11618 {%
11619   \renewglossarystyle{altsuperragged4col}{%
11620     \renewenvironment{theglossary}%
11621       {\tablehead{}\tabletail{}}%
11622       \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
11623         >{\raggedright}p{\glspagelistwidth}}}%
11624       {\end{supertabular}}%
11625     \renewcommand*{\glossaryheader}{}%
11626     \renewcommand{\glossentry}[2]{%
11627       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11628       \glossentrydesc{##1}\glspostdescription &
11629       \glossentrysymbol{##1} & ##2\tabularnewline
11630     }%
11631     \renewcommand{\subglossentry}[3]{%
11632       &
11633       \glssubentryitem{##2}%
11634       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11635       \glossentrysymbol{##2} & ##3\tabularnewline
11636     }%

```

```

11637 \ifglsnogroupskip
11638 \renewcommand*{\glsgroupskip}{}%
11639 \else
11640 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11641 \fi
11642 }
11643 }
11644 {}

```

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`.

```

11645 \ifdef{\@glsstyle@inline}
11646 {%
11647 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11648 \renewcommand*{\glsinlinedescformat}[3]{%
11649 \space#1\glxtrpostdescription}
11650 \renewcommand*{\glsinlinesubdescformat}[3]{%
11651 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11652 }
11653 {}

```

2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11654 \ifdef{\@glsstyle@index}
11655 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11656 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11657 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

```

```

11658 \renewglossarystyle{index}{%
11659 \renewenvironment{theglossary}%
11660 {\setlength{\parindent}{0pt}%
11661 \setlength{\parskip}{0pt plus 0.3pt}%
11662 \let\item\glstreeitem
11663 \let\subitem\glstreesubitem

```

```

11664 \let\subsubitem\glstreesubsubitem
11665 }%
11666 {\par}%
11667 \renewcommand*\glossaryheader{}%
11668 \renewcommand*\glsgroupheading}[1]{%
11669 \renewcommand*\glossentry}[2]{%
11670 \item\glstentryitem{##1}%
11671 \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11672 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}%
11673 \glstreepredesc \glossentrydesc{##1}\glspostdescription
11674 \glstreeprelocation ##2%
11675 }%
11676 \renewcommand{\subglossentry}[3]{%
11677 \ifcase##1\relax
11678 \item
11679 \or
11680 \subitem
11681 \glssubentryitem{##2}%
11682 \else
11683 \subsubitem
11684 \fi
11685 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11686 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}%
11687 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11688 \glstreechildprelocation ##3%
11689 }%
11690 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else\indexspace\fi}%
11691 }
11692 }
11693 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11694 \ifdef{\@glsstyle@indexgroup}
11695 {%
11696 \renewglossarystyle{indexgroup}{%
11697 \setglossarystyle{index}%
11698 \renewcommand*\glsgroupheading}[1]{%
11699 \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
11700 \nopagebreak\indexspace
11701 \nobreak\@afterheading
11702 }%
11703 }
11704 }
11705 {}

```

Similarly for indexhypergroup.

```

11706 \ifdef{\@glsstyle@indexhypergroup}
11707 {%
11708 \renewglossarystyle{indexhypergroup}{%
11709 \setglossarystyle{index}%

```

```

11710 \renewcommand*{\glossaryheader}{%
11711 \item\glstreenavigationfmt{\glsnavigation}%
11712 \nobreak\@afterheading\indexspace}%
11713 \renewcommand*{\glsgroupheading}[1]{%
11714 \item\glstreegroupheaderfmt
11715 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
11716 \nopagebreak\indexspace
11717 \nobreak\@afterheading}%
11718 }%
11719 }
11720 {}

```

Adjust tree style to remove hard coded space before number list.

```

11721 \ifdef{\@glsstyle@tree}
11722 {%
11723 \renewglossarystyle{tree}{%
11724 \renewenvironment{theglossary}%
11725 {\setlength{\parindent}{0pt}%
11726 \setlength{\parskip}{0pt plus 0.3pt}}%
11727 {}%
11728 \renewcommand*{\glossaryheader}{}%
11729 \renewcommand*{\glsgroupheading}[1]{}%
11730 \renewcommand{\glossentry}[2]{%
11731 \hangindent0pt\relax
11732 \parindent0pt\relax
11733 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11734 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11735 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11736 \glstreeprelocation##2\par
11737 }%
11738 \renewcommand{\subglossentry}[3]{%
11739 \hangindent##1\glstreeindent\relax
11740 \parindent##1\glstreeindent\relax
11741 \ifnum##1=1\relax
11742 \glssubentryitem{##2}%
11743 \fi
11744 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11745 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11746 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11747 \glstreechildprelocation ##3\par
11748 }%
11749 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11750 }%
11751 }
11752 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11753 \ifdef{\@glsstyle@treegroup}
11754 {%
11755 \renewglossarystyle{treegroup}{%

```

```

11756 \setglossarystyle{tree}%
11757 \renewcommand{\glsgroupheading}[1]{\par
11758 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11759 \nopagebreak\indexspace\nobreak\@afterheading}%
11760 }
11761 }
11762 {}

```

Similarly for treehypergroup

```

11763 \ifdef{\@glsstyle@treehypergroup}
11764 {%
11765 \renewglossarystyle{treehypergroup}{%
11766 \setglossarystyle{tree}%
11767 \renewcommand*\glossaryheader{%
11768 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11769 \nobreak\@afterheading\indexspace}%
11770 \renewcommand*\glsgroupheading}[1]{%
11771 \par\noindent
11772 \glstreegroupheaderfmt
11773 {\glsnavigationhypertarget{##1}{\glsgrouptitle{##1}}}\par
11774 \nopagebreak\indexspace\nobreak\@afterheading}%
11775 }
11776 }
11777 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11778 \ifdef{\@glsstyle@treenoname}
11779 {%
11780 \renewglossarystyle{treenoname}{%
11781 \renewenvironment{theglossary}%
11782 {\setlength{\parindent}{0pt}%
11783 \setlength{\parskip}{0pt plus 0.3pt}}%
11784 {}%
11785 \renewcommand*\glossaryheader{}%
11786 \renewcommand*\glsgroupheading}[1]{}%
11787 \renewcommand{\glossentry}[2]{%
11788 \hangindent0pt\relax
11789 \parindent0pt\relax
11790 \glstentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11791 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11792 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11793 \glstreeprelocation##2\par
11794 }%
11795 \renewcommand{\subglossentry}[3]{%
11796 \hangindent##1\glstreeindent\relax
11797 \parindent##1\glstreeindent\relax
11798 \ifnum##1=1\relax
11799 \glssubentryitem{##2}%
11800 \fi
11801 \glstarget{##2}{\strut}%

```

```

11802     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11803 }%
11804 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11805 }
11806 }
11807 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11808 \ifdef{\@glsstyle@treenonamegroup}
11809 {%
11810   \renewglossarystyle{treenonamegroup}{%
11811     \setglossarystyle{treenoname}%
11812     \renewcommand{\glsgroupheading}[1]{\par
11813       \noindent\glstreegroupheaderfmt
11814       {\glsggetgrouptitle{##1}}}%
11815     \nopagebreak\indexspace\nobreak\@afterheading
11816   }%
11817 }
11818 }
11819 {}

```

Similarly for `treenonamehypergroup`

```

11820 \ifdef{\@glsstyle@treenonamehypergroup}
11821 {%
11822   \renewglossarystyle{treenonamehypergroup}{%
11823     \setglossarystyle{treenoname}%
11824     \renewcommand*{\glossaryheader}{%
11825       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11826       \nobreak\@afterheading\indexspace}%
11827     \renewcommand*{\glsgroupheading}[1]{%
11828       \par\noindent
11829       \glstreegroupheaderfmt
11830       {\glsnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
11831     \nopagebreak\indexspace\nobreak\@afterheading}%
11832   }
11833 }
11834 {}

```

The `alttree` style is redefined to make it easier to made minor adjustments.

```

11835 \ifdef{\@glsstyle@alttree}
11836 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtralttreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

11837 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```



```

11838   {%
11839     \let\par\glxstrAltTreePar
11840     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11841     \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11842   }%
11843 }

```

trAltTreeIndent Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11844 \newlength\glxstrAltTreeIndent

```

lsxtrAltTreePar Multi-paragraph descriptions need to keep the hanging indent.

```

11845 \newcommand{\glxstrAltTreePar}{%
11846   @@par
11847   \glxstrAltTreeSetHangIndent
11848   \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11849 }

```

symbolDescLocation `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11850 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11851   \glxstralttreeSymbolDescLocation{#2}{#3}%
11852 }

```

trtreetopindent The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

11853 \newlength\glxstrtreetopindent

```

lxstralttreeInit User-level initialisation for the alttree style.

```

11854 \newcommand*{\glxstralttreeInit}{%
11855   \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
11856   \glxstrAltTreeIndent=\parindent
11857 }

```

\glsglissetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\gdef`.

```

11858 \newcommand*{\glsglissetwidest}[2][0]{%
11859   \csgdef{@glswidestname\romannumeral#1}{#2}%
11860 }

```

\eglssetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

11861 \newcommand*{\eglssetwidest}[2][0]{%
11862   \protected@csedef{@glswidestname\romannumeral#1}{#2}%
11863 }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```
11864 \newcommand*\xglissetwidest}[2][0]{%
11865   \protected@csxdef{#1}{#2}%
11866 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
11867 \newcommand*\glsupdatewidest}[2][0]{%
11868   \ifcsundef{#1}%
11869     {\csdef{#1}{#2}}%
11870     {%
11871       \settowidth{#1}{\csuse{#1}}%
11872       \settowidth{#2}%
11873       \ifdim#2>#1%
11874         \csdef{#1}{#2}%
11875       \fi
11876     }%
11877 }
```

`glsupdatewidest` As above but global definition.

```
11878 \newcommand*\glsupdatewidest}[2][0]{%
11879   \ifcsundef{#1}%
11880     {\csgdef{#1}{#2}}%
11881     {%
11882       \settowidth{#1}{\csuse{#1}}%
11883       \settowidth{#2}%
11884       \ifdim#2>#1%
11885         \csgdef{#1}{#2}%
11886       \fi
11887     }%
11888 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
11889 \newcommand*\eglsupdatewidest}[2][0]{%
11890   \ifcsundef{#1}%
11891     {\protected@csedef{#1}{#2}}%
11892     {%
11893       \settowidth{#1}{\csuse{#1}}%
11894       \settowidth{#2}%
11895       \ifdim#2>#1%
11896         \protected@csedef{#1}{#2}%
11897       \fi
11898     }%
11899 }
```

`glsupdatewidest` As above but global.

```
11900 \newcommand*\xglsupdatewidest}[2][0]{%
11901   \ifcsundef{#1}%
11902     {\protected@csxdef{#1}{#2}}%
11903     {%
```

```

11904      \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
11905      \settowidth{\dimen@ii}{#2}%
11906      \ifdim\dimen@ii>\dimen@
11907      \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
11908      \fi
11909  }%
11910  }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

11911  \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

11912  \newcommand*{\glswidestsubname}[1]{%
11913    \ifcsundef{@glswidestname\romannumeral#1}%
11914    {\@glswidestname}%
11915    {\csuse{@glswidestname\romannumeral#1}}%
11916  }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidesttoplevelname`.

```

11917  \let\glswidestTopLevelName\glswidesttoplevelname

```

glswidestUsedTopLevelName Like `\glswidesttoplevelname` but has an additional check that the entry has been used. Only useful if the glossaries occur at the end of the document, in which case this command should go at the start of the glossary. Alternatively, place at the end of the document and save for the next run.

```

11918  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
11919    \dimen@=0pt\relax
11920    \glstmplen=0pt\relax
11921    \foralllglossaries[#1]{\@gls@type}%
11922    {%
11923      \forglsentries[\@gls@type]{\@glo@label}%
11924      {%
11925        \ifglused{\@glo@label}%
11926        {%
11927          \ifglshasparent{\@glo@label}%
11928          {}%
11929          {%
11930            \settowidth{\dimen@}%
11931            {\glstreenamfmt{\glstryname{\@glo@label}}}%
11932            \ifdim\dimen@>\glstmplen
11933            \glstmplen=\dimen@
11934            \eglissetwidest{\glstryname{\@glo@label}}%
11935            \fi
11936          }%
11937        }%
11938      }%
11939    }%
11940  }%
11941  }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11942 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11943   \dimen@=0pt\relax
11944   \gls@tmplen=0pt\relax
11945   \forallglossaries[#1]{\@gls@type}%
11946   {%
11947     \forallgsentries[\@gls@type]{\@glo@label}%
11948     {%
11949       \ifglsused{\@glo@label}%
11950       {%
11951         \settowidth{\dimen@}%
11952           {\glstreenamefmt{\glssentryname{\@glo@label}}}%
11953         \ifdim\dimen@>\gls@tmplen
11954           \gls@tmplen=\dimen@
11955           \eglssetwidest{\glssentryname{\@glo@label}}%
11956         \fi
11957       }%
11958     }%
11959   }%
11960 }%
11961 }

```

`ndWidestAnyName` Like the above but doesn't check if the entry has been used.

```

11962 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
11963   \dimen@=0pt\relax
11964   \gls@tmplen=0pt\relax
11965   \forallglossaries[#1]{\@gls@type}%
11966   {%
11967     \forallgsentries[\@gls@type]{\@glo@label}%
11968     {%
11969       \settowidth{\dimen@}%
11970         {\glstreenamefmt{\glssentryname{\@glo@label}}}%
11971       \ifdim\dimen@>\gls@tmplen
11972         \gls@tmplen=\dimen@
11973         \eglssetwidest{\glssentryname{\@glo@label}}%
11974       \fi
11975     }%
11976   }%
11977 }

```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

11978 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
11979   \dimen@=0pt\relax
11980   \dimen@i=0pt\relax
11981   \dimen@ii=0pt\relax
11982   \forallglossaries[#1]{\@gls@type}%
11983   {%

```

```

11984 \forglsentries[\@gls@type]{\@glo@label}%
11985 {%
11986 \ifglsused{\@glo@label}%
11987 {%
11988 \ifglsashasparent{\@glo@label}%
11989 {%
11990 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
11991 \ifglsashasparent{\@glo@parent}%
11992 {%
11993 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
11994 \ifglsashasparent{\@glo@parent}%
11995 {}%
11996 {%
11997 \settowidth{\gls@tmplen}%
11998 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11999 \ifdim\gls@tmplen>\dimen@ii
12000 \dimen@ii=\gls@tmplen
12001 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
12002 \fi
12003 }%
12004 }%
12005 {%
12006 \settowidth{\gls@tmplen}%
12007 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12008 \ifdim\gls@tmplen>\dimen@i
12009 \dimen@i=\gls@tmplen
12010 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
12011 \fi
12012 }%
12013 }%
12014 {%
12015 \settowidth{\gls@tmplen}%
12016 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12017 \ifdim\gls@tmplen>\dimen@
12018 \dimen@=\gls@tmplen
12019 \eglssetwidest{\glsentryname{\@glo@label}}%
12020 \fi
12021 }%
12022 }%
12023 {}%
12024 }%
12025 }%
12026 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

12027 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
12028 \dimen@=0pt\relax
12029 \dimen@i=0pt\relax
12030 \dimen@ii=0pt\relax

```

```

12031 \forall glossaries[#1]{\@gls@type}%
12032 {%
12033   \forall sentries[\@gls@type]{\@glo@label}%
12034   {%
12035     \ifglshasparent{\@glo@label}%
12036     {%
12037       \edef\@glo@parent{\csuse{glo@\glsetoklabel{\@glo@label}@parent}}%
12038       \ifglshasparent{\@glo@parent}%
12039       {%
12040         \edef\@glo@parent{\csuse{glo@\glsetoklabel{\@glo@parent}@parent}}%
12041         \ifglshasparent{\@glo@parent}%
12042         {}%
12043         {%
12044           \settowidth{\gls@tmplen}%
12045             {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12046           \ifdim\gls@tmplen>\dimen@ii
12047             \dimen@ii=\gls@tmplen
12048             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
12049           \fi
12050         }%
12051       }%
12052     {%
12053       \settowidth{\gls@tmplen}%
12054         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12055       \ifdim\gls@tmplen>\dimen@i
12056         \dimen@i=\gls@tmplen
12057         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
12058       \fi
12059     }%
12060   }%
12061   {%
12062     \settowidth{\gls@tmplen}%
12063       {\glstreenamfmt{\glstentryname{\@glo@label}}}%
12064     \ifdim\gls@tmplen>\dimen@
12065       \dimen@=\gls@tmplen
12066       \eglssetwidest{\glstentryname{\@glo@label}}%
12067     \fi
12068   }%
12069 }%
12070 }%
12071 }

```

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.

```

12072 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
12073   \dimen@=0pt\relax
12074   \gls@tmplen=0pt\relax
12075   #2=0pt\relax
12076   \forall glossaries[#1]{\@gls@type}%

```

```

12077  {%
12078    \forglsentries[\@gls@type]{\@glo@label}%
12079    {%
12080      \ifglsused{\@glo@label}%
12081      {%
12082        \settowidth{\dimen@}%
12083          {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12084        \ifdim\dimen@>\gls@tmplen
12085          \gls@tmplen=\dimen@
12086          \eglssetwidest{\glsentryname{\@glo@label}}%
12087        \fi
12088        \settowidth{\dimen@}%
12089          {\glsentrysymbol{\@glo@label}}%
12090        \ifdim\dimen@>#2\relax
12091          #2=\dimen@
12092        \fi
12093      }%
12094    }%
12095  }%
12096 }%
12097 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

12098 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
12099   \dimen@=0pt\relax
12100   \gls@tmplen=0pt\relax
12101   #2=0pt\relax
12102   \forallglossaries[#1]{\@gls@type}%
12103   {%
12104     \forglsentries[\@gls@type]{\@glo@label}%
12105     {%
12106       \settowidth{\dimen@}%
12107         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12108       \ifdim\dimen@>\gls@tmplen
12109         \gls@tmplen=\dimen@
12110         \eglssetwidest{\glsentryname{\@glo@label}}%
12111       \fi
12112       \settowidth{\dimen@}%
12113         {\glsentrysymbol{\@glo@label}}%
12114       \ifdim\dimen@>#2\relax
12115         #2=\dimen@
12116       \fi
12117     }%
12118   }%
12119 }

```

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.

```

12120 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
12121   \dimen@=0pt\relax
12122   \gls@tmplen=0pt\relax
12123   #2=0pt\relax
12124   #3=0pt\relax
12125   \foralllglossaries[#1]{\@gls@type}%
12126   {%
12127     \forglsentries[\@gls@type]{\@glo@label}%
12128     {%
12129       \ifglsused{\@glo@label}%
12130       {%
12131         \settowidth{\dimen@}%
12132         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12133         \ifdim\dimen@>\gls@tmplen
12134           \gls@tmplen=\dimen@
12135           \eglssetwidest{\glsentryname{\@glo@label}}%
12136         \fi
12137         \settowidth{\dimen@}%
12138         {\glsentrysymbol{\@glo@label}}%
12139         \ifdim\dimen@>#2\relax
12140           #2=\dimen@
12141         \fi
12142         \settowidth{\dimen@}%
12143         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12144         \ifdim\dimen@>#3\relax
12145           #3=\dimen@
12146         \fi
12147       }%
12148     }%
12149   }%
12150 }%
12151 }

```

eSymbolLocation Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

12152 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
12153   \dimen@=0pt\relax
12154   \gls@tmplen=0pt\relax
12155   #2=0pt\relax
12156   #3=0pt\relax
12157   \foralllglossaries[#1]{\@gls@type}%
12158   {%
12159     \forglsentries[\@gls@type]{\@glo@label}%
12160     {%
12161       \settowidth{\dimen@}%
12162       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12163       \ifdim\dimen@>\gls@tmplen
12164         \gls@tmplen=\dimen@
12165       \eglssetwidest{\glsentryname{\@glo@label}}%

```



```

12166      \fi
12167      \settowidth{\dimen@}%
12168      {\glstrysymbol{\@glo@label}}}%
12169      \ifdim\dimen@>#2\relax
12170      #2=\dimen@
12171      \fi
12172      \settowidth{\dimen@}%
12173      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12174      \ifdim\dimen@>#3\relax
12175      #3=\dimen@
12176      \fi
12177    }%
12178  }%
12179 }

```

AnyNameLocation Like the `\glstryFindWidestUsedAnyNameSymbolLocation` but doesn't measure the symbol. The length of the widest location list is stored in the second argument, which should be a length register.

```

12180 \newrobustcmd*{\glstryFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
12181   \dimen@=0pt\relax
12182   \glstry@tmplen=0pt\relax
12183   #2=0pt\relax
12184   \forallglossaries[#1]{\@gls@type}%
12185   {%
12186     \forallglsentries[\@gls@type]{\@glo@label}%
12187     {%
12188       \ifglstryused{\@glo@label}%
12189       {%
12190         \settowidth{\dimen@}%
12191         {\glstrynamefmt{\glstryname{\@glo@label}}}%
12192         \ifdim\dimen@>\glstry@tmplen
12193         \glstry@tmplen=\dimen@
12194         \eglstrysetwidest{\glstryname{\@glo@label}}%
12195       \fi
12196       \settowidth{\dimen@}%
12197       {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12198       \ifdim\dimen@>#2\relax
12199       #2=\dimen@
12200     \fi
12201   }%
12202   {}}%
12203 }%
12204 }%
12205 }

```

AnyNameLocation Like the `\glstryFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

12206 \newrobustcmd*{\glstryFindWidestAnyNameLocation}[2][\@glo@types]{%
12207   \dimen@=0pt\relax
12208   \glstry@tmplen=0pt\relax

```

```

12209      #2=Opt\relax
12210      \forallglossaries[#1]{\@gls@type}%
12211      {%
12212        \forglsentries[\@gls@type]{\@glo@label}%
12213        {%
12214          \settowidth{\dimen@}%
12215            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12216          \ifdim\dimen@>\gls@tmplen
12217            \gls@tmplen=\dimen@
12218            \eglssetwidest{\glsentryname{\@glo@label}}%
12219          \fi
12220          \settowidth{\dimen@}%
12221            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12222          \ifdim\dimen@>#2\relax
12223            #2=\dimen@
12224          \fi
12225        }%
12226      }%
12227    }

```

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`.

```

12228    \newcommand*{\glsxtrComputeTreeIndent}[1]{%
12229      \glstreeindent=\glsxtrtreetopindent\relax
12230    }

```

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.

```

12231    \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
12232      \ifcsundef{@glswidestname\romannumeral#1}%
12233      {%
12234        \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
12235      }%
12236      {%
12237        \settowidth{#3}{\glstreenamefmt{%
12238          \csname @glswidestname\romannumeral#1\endcsname\space}}%
12239      }%
12240    }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

12241    \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
12242 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
12243 \renewglossarystyle{alttree}{%
12244   \renewenvironment{theglossary}%
12245     {%
12246       \glxtralttreeInit
12247       \def\@gls@prevlevel{-1}%
12248       \mbox{}\par}%
12249     {\par}%
12250   \renewcommand*{\glossaryheader}{}%
12251   \renewcommand*{\glsgroupheading}[1]{}%
12252   \renewcommand{\glossentry}[2]{%
12253     \ifnum\@gls@prevlevel=0\relax
12254     \else
12255       \glxtrComputeTreeIndent{##1}%
12256     \fi
12257     \parindent\glstreeindent
12258     \glxtrAltTreeSetHangIndent
12259     \makebox[Opt][r]%
12260     {%
12261       \glstreenamebox{\glstreeindent}%
12262       {%
12263         \glssentryitem{##1}%
12264         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
12265       }%
12266     }%
12267     \glxtralttreeSymbolDescLocation{##1}{##2}%
12268     \def\@gls@prevlevel{0}%
12269   }
12270   \renewcommand{\subglossentry}[3]{%
12271     \ifnum##1=1\relax
12272       \glssubentryitem{##2}%
12273     \fi
12274     \ifnum\@gls@prevlevel=##1\relax
12275     \else
12276       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12277       \ifnum\@gls@prevlevel<##1\relax
12278         \setlength\glstreeindent\gls@tmplen
12279         \addtolength\glstreeindent\parindent
12280         \parindent\glstreeindent
12281       \else
12282         \ifnum\@gls@prevlevel=0\relax
12283           \glxtrComputeTreeIndent{##2}%
12284         \else
12285           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12286         \fi
12287         \addtolength\parindent{-\glstreeindent}%

```

```

12288         \setlength\glstreeindent\parindent
12289     \fi
12290 \fi
12291 \glxtrAltTreeSetSubHangIndent{##1}%
12292 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
12293     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12294 \glxtralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12295 \def\@gls@prevlevel{##1}%
12296 }%
12297 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12298 }
12299 }%
12300 {%
12301 }

```

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.

```

12302 \ifdef{\@glsstyle@almtreegroup}
12303 {%
12304     \renewglossarystyle{almtreegroup}{%
12305         \setglossarystyle{almtree}%
12306         \renewcommand{\glsgroupheading}[1]{\par
12307             \def\@gls@prevlevel{-1}%
12308             \hangindentOpt\relax
12309             \parindentOpt\relax
12310             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%
12311         \nopagebreak\indexspace\nopagebreak
12312     }%
12313 }%
12314 }%
12315 {%
12316 }

```

Similarly for `almtreehypergroup`.

```

12317 \ifdef{\@glsstyle@almtreehypergroup}
12318 {%
12319     \renewglossarystyle{almtreehypergroup}{%
12320         \setglossarystyle{almtree}%
12321         \renewcommand*\{\glossaryheader}{%
12322             \par
12323             \def\@gls@prevlevel{-1}%
12324             \hangindentOpt\relax
12325             \parindentOpt\relax
12326             \glstreenavigationfmt{\glsnavigation}\par\indexspace
12327         }%
12328         \renewcommand*\{\glsgroupheading}[1]{%
12329             \par
12330             \def\@gls@prevlevel{-1}%
12331             \hangindentOpt\relax
12332             \parindentOpt\relax

```

```

12333 \glstreegroupheaderfmt
12334     {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12335     \nopagebreak\indexspace\nopagebreak
12336 }%
12337 }
12338 }%
12339 {%
12340 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12341 \ifdef{\@glsstyle@mcolindexgroup}
12342 {%
12343   \renewglossarystyle{mcolindexgroup}{%
12344     \setglossarystyle{mcolindex}%
12345     \renewcommand*{\glsgroupheading}[1]{%
12346       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12347       \nopagebreak\indexspace\nobreak\@afterheading
12348     }%
12349   }
12350 }%
12351 {%
12352 }

```

Similarly for mcolindexhypergroup.

```

12353 \ifdef{\@glsstyle@mcolindexhypergroup}
12354 {%
12355   \renewglossarystyle{mcolindexhypergroup}{%
12356     \setglossarystyle{mcolindex}%
12357     \renewcommand*{\glossaryheader}{%
12358       \item\glstreenavigationfmt{\glsnavigation}%
12359       \indexspace
12360     }%
12361     \renewcommand*{\glsgroupheading}[1]{%
12362       \item\glstreegroupheaderfmt
12363         {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12364       \nopagebreak\indexspace\nobreak\@afterheading
12365     }%
12366   }
12367 }%
12368 {%
12369 }

```

Similarly for mcolindexspannav.

```

12370 \ifdef{\@glsstyle@mcolindexspannav}
12371 {%
12372   \renewglossarystyle{mcolindexspannav}{%
12373     \setglossarystyle{index}%

```

```

12374 \renewenvironment{theglossary}%
12375 {%
12376 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12377 \setlength{\parindent}{0pt}%
12378 \setlength{\parskip}{0pt plus 0.3pt}%
12379 \let\item\glstreeitem}%
12380 {\end{multicols}}}%
12381 \renewcommand*{\glsgroupheading}[1]{%
12382 \item\glstreegroupheaderfmt
12383 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12384 \nopagebreak\indexspace\nobreak\@afterheading
12385 }%
12386 }
12387 }%
12388 {%
12389 }

```

Similarly for mcoltreegroup.

```

12390 \ifdef{\@glstyle@mcoltreegroup}
12391 {%
12392 \renewglossarystyle{mcoltreegroup}{%
12393 \setglossarystyle{mcoltree}%
12394 \renewcommand{\glsgroupheading}[1]{\par
12395 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12396 \nopagebreak\indexspace\nobreak\@afterheading
12397 }%
12398 }
12399 }%
12400 {%
12401 }

```

Similarly for mcoltreehypergroup.

```

12402 \ifdef{\@glstyle@mcoltreehypergroup}
12403 {%
12404 \renewglossarystyle{mcoltreehypergroup}{%
12405 \setglossarystyle{mcoltree}%
12406 \renewcommand*{\glossaryheader}{%
12407 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12408 }%
12409 \renewcommand*{\glsgroupheading}[1]{%
12410 \par\noindent
12411 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12412 \nopagebreak\indexspace\nobreak\@afterheading
12413 }%
12414 }
12415 }%
12416 {%
12417 }

```

Similarly for mcoltreespannav.

```

12418 \ifdef{\@glstyle@mcoltreespannav}

```

```

12419 {%
12420   \renewglossarystyle{mcoltreespannav}{%
12421     \setglossarystyle{tree}%
12422     \renewenvironment{theglossary}%
12423     {%
12424       \begin{multicols}{\glsmcols}%
12425       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12426       \setlength{\parindent}{0pt}%
12427       \setlength{\parskip}{0pt plus 0.3pt}%
12428     }%
12429     {\end{multicols}}}%
12430   \renewcommand*{\glsgroupheading}[1]{%
12431     \par\noindent
12432     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12433     \nopagebreak\indexspace\nobreak\@afterheading
12434   }%
12435 }
12436 }%
12437 {%
12438 }

```

Similarly for mcoltreenonamegroup.

```

12439 \ifdef{\@glsstyle@mcoltreenonamegroup}
12440 {%
12441   \renewglossarystyle{mcoltreenonamegroup}{%
12442     \setglossarystyle{mcoltreenoname}%
12443     \renewcommand*{\glsgroupheading}[1]{\par
12444       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12445     \nopagebreak\indexspace\nobreak\@afterheading
12446   }%
12447 }
12448 }%
12449 {%
12450 }

```

Similarly for mcoltreenonamehypergroup.

```

12451 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12452 {%
12453   \renewglossarystyle{mcoltreenonamehypergroup}{%
12454     \setglossarystyle{mcoltreenoname}%
12455     \renewcommand*{\glossaryheader}{%
12456       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12457     \renewcommand*{\glsgroupheading}[1]{%
12458       \par\noindent
12459       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12460       \nopagebreak\indexspace\nobreak\@afterheading}%
12461   }
12462 }%
12463 {%
12464 }

```

Similarly for mcoltreenonamespannav.

```

12465 \ifdef{\@glsstyle@mcoltreenonamespannav}
12466 {%
12467   \renewglossarystyle{mcoltreenonamespannav}{%
12468     \setglossarystyle{treenoname}%
12469     \renewenvironment{theglossary}%
12470     {%
12471       \begin{multicols}{\glsmcols}%
12472       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12473       \setlength{\parindent}{0pt}%
12474       \setlength{\parskip}{0pt plus 0.3pt}%
12475     }%
12476     {\end{multicols}}}%
12477   \renewcommand*{\glsgroupheading}[1]{%
12478     \par\noindent
12479     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12480     \nopagebreak\indexspace\nobreak\@afterheading}%
12481   }
12482 }%
12483 {%
12484 }

```

mcolalmtree needs adjusting so that it uses \glsxtralmtreeInit This doesn't use \mbox{\par which would unbalance the top of the columns.

```

12485 \ifdef{\@glsstyle@mcolalmtree}
12486 {%
12487   \renewglossarystyle{mcolalmtree}{%
12488     \setglossarystyle{almtree}%
12489     \renewenvironment{theglossary}%
12490     {%
12491       \glsxtralmtreeInit
12492       \def\@gls@prevlevel{-1}%
12493       \begin{multicols}{\glsmcols}%
12494     }%
12495     {\par\end{multicols}}}%
12496   }
12497 }%
12498 {%
12499 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12500 \ifdef{\@glsstyle@mcolalmtreegroup}
12501 {%
12502   \renewglossarystyle{mcolalmtreegroup}{%
12503     \setglossarystyle{mcolalmtree}%
12504     \renewcommand{\glsgroupheading}[1]{\par
12505       \def\@gls@prevlevel{-1}%
12506       \hangindent0pt\relax
12507       \parindent0pt\relax
12508       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```



```

12509      \nopagebreak\indexspace\nopagebreak
12510    }%
12511  }
12512 }%
12513 {%
12514 }

```

Similarly for mcolalttreehypergroup.

```

12515 \ifdef{\@glsstyle@mcolalttreehypergroup}
12516 {%
12517   \renewglossarystyle{mcolalttreehypergroup}{%
12518     \setglossarystyle{mcolalttree}%
12519     \renewcommand*{\glossaryheader}{%
12520       \par
12521       \def\@gls@prevlevel{-1}%
12522       \hangindent0pt\relax
12523       \parindent0pt\relax
12524       \glstreenavigationfmt{\glsnavigation}%
12525       \par\indexspace
12526     }%
12527     \renewcommand*{\glsgroupheading}[1]{%
12528       \par
12529       \def\@gls@prevlevel{-1}%
12530       \hangindent0pt\relax
12531       \parindent0pt\relax
12532       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12533       \nopagebreak\indexspace\nopagebreak
12534     }%
12535   }
12536 }%
12537 {%
12538 }

```

Similarly for mcolalttreespannav.

```

12539 \ifdef{\@glsstyle@mcolalttreespannav}
12540 {%
12541   \renewglossarystyle{mcolalttreespannav}{%
12542     \setglossarystyle{alttree}%
12543     \renewenvironment{theglossary}%
12544     {%
12545       \glsxtralttreeInit
12546       \def\@gls@prevlevel{-1}%
12547       \begin{multicols}{\glsncols}%
12548         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12549     }%
12550     {\par\end{multicols}}%
12551     \renewcommand*{\glsgroupheading}[1]{%
12552       \par
12553       \def\@gls@prevlevel{-1}%
12554       \hangindent0pt\relax

```

```

12555     \parindent0pt\relax
12556     \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgroupptitle{##1}}}%
12557     \nopagebreak\indexspace\nopagebreak
12558   }%
12559 }
12560}%
12561{%
12562}

Reset the default style
12563 \ifx\@glossary@default@style\relax
12564 \else
12565   \setglossarystyle{\@glxtr@current@style}
12566 \fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
12567 \NeedsTeXFormat{LaTeX2e}
12568 \ProvidesPackage{glossary-bookindex}[2017/11/24 v1.25 (NLCT)]
```

Load required packages.

```
12569 \RequirePackage{multicol}
12570 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12571 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12572 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`bookindexsubname` Format used for sub entries.

```
12573 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`glstrprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12574 \providecommand*{\glstrprelocation}{\space}
```

`indexprelocation` Separator used before location list for top-level entries. Version 1.22 has removed the `\ifglsnopostdot` check since this style doesn't display the description.

```
12575 \newcommand*{\glstrbookindexprelocation}[1]{%
12576   \glstrifhasfield{location}{#1}%
12577   {,\glstrprelocation}%
12578   {\glstrprelocation}%
12579 }
```

`glstrsubprelocation` Separator used before location list for sub-entries.

```
12580 \newcommand*{\glstrbookindexsubprelocation}[1]{%
12581   \glstrbookindexprelocation{#1}%
12582 }
```

`glstrparentchildsep` Separator used between top-level parent and child entry.

```
12583 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`glstrparentsubchildsep` Separator used between sub-level parent and child entry.

```
12584 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
12585 \newcommand{\glxstrbookindexbetween}[2]{}

indexsubbetween Between two level 1 entries identified by the labels in the arguments.
12586 \newcommand{\glxstrbookindexsubbetween}[2]{}

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.
12587 \newcommand{\glxstrbookindexsubsubbetween}[2]{}

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.
12588 \newcommand{\glxstrbookindexatendgroup}[1]{}

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.
12589 \newcommand{\glxstrbookindexsubatendgroup}[1]{}

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.
12590 \newcommand{\glxstrbookindexsubsubatendgroup}[1]{}

kindexgroupskip Group separator.
12591 \newcommand{\glxstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}

Format group title.

dexformatheader Group separator.
12592 \newcommand*{\glxstrbookindexformatheader}[1]{%
12593 \par{\centering\glstreegroupheaderfmt{#1}\par}%
12594 }

bookindexbookmark Book mark group heading if supported.
12595 \ifdef\pdfbookmark
12596 {%
12597 \newcommand*{\glxstrbookindexbookmark}[2]{%
12598 \ifdefstring{\@@glossarysec}{chapter}%
12599 {\pdfbookmark[1]{#1}{#2}}%
12600 {\pdfbookmark[2]{#1}{#2}}%
12601 }
12602 }
12603 {%
12604 \newcommand*{\glxstrbookindexbookmark}[2]{}
12605 }

kindexcolspread
12606 \newcommand*{\glxstrbookindexcolspread}{}

dexmulticolenv
12607 \newcommand*{\glxstrbookindexmulticolenv}{multicols}
```

Define the style.

```

12608 \newglossarystyle{bookindex}{%
12609   \setglossarystyle{index}%
12610   \renewenvironment{theglossary}%
12611   {%
12612     \ifdefempty{glxstrbookindexcolspread}
12613     {%
12614       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
12615       {\glxstrbookindexcols}%
12616     }%
12617     {%
12618       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
12619       {\glxstrbookindexcols}[\glxstrbookindexcolspread]%
12620     }%
12621     \setlength{\parindent}{0pt}%
12622     \setlength{\parskip}{0pt plus 0.3pt}%
12623     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12624     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12625     \let\@glxstr@bookindex@between\@gobble
12626     \let\@glxstr@bookindex@subbetween\@gobble
12627     \let\@glxstr@bookindex@subsubbetween\@gobble
12628     \let\@glxstr@bookindex@atendgroup\relax
12629     \let\@glxstr@bookindex@subatendgroup\relax
12630     \let\@glxstr@bookindex@subsubatendgroup\relax
12631     \let\@glxstr@bookindex@groupskip\relax
12632   }%
12633   {%

```

Do end group hooks.

```

12634     \@glxstr@bookindex@subsubatendgroup
12635     \@glxstr@bookindex@subatendgroup
12636     \@glxstr@bookindex@atendgroup

```

End multicol environment.

```

12637     \expandafter\end\expandafter{\glxstrbookindexmulticolseenv}%
12638   }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12639   \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12640   \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12641     \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12642     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12643     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12644     \let\@glxstr@bookindex@subbetween\@gobble
12645     \let\@glxstr@bookindex@subsubbetween\@gobble
12646     \edef\@glxstr@bookindex@between{%

```

```

12647     \noexpand\glxstrbookindexbetween{##1}%
12648 }%
12649 \edef\@glxstr@bookindex@atendgroup{%
12650     \noexpand\glxstrbookindexatendgroup{##1}%
12651 }%
12652 \let\@glxstr@bookindex@subatendgroup\relax
12653 \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12654 \glstreeitem
12655 \glstryitem{##1}%
12656 \glstarget{##1}{\glxstrbookindexname{##1}}%
12657 \glxstrbookindexprelocation{##1}##2%
12658 }%
12659 \renewcommand{\subglossentry}[3]{%
12660 \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12661 \glstreeitem
12662 \or

```

Level 1.

```

12663 \@glxstr@bookindex@sep
12664 \@glxstr@bookindex@subbetween{##2}%
12665 \let\@glxstr@bookindex@sep\relax

```

Update separators.

```

12666 \let\@glxstr@bookindex@subsubbetween\@gobble
12667 \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
12668 \edef\@glxstr@bookindex@subbetween{%
12669     \noexpand\glxstrbookindexsubbetween{##2}%
12670 }%
12671 \edef\@glxstr@bookindex@atsubendgroup{%
12672     \noexpand\glxstrbookindexatsubendgroup{##1}%
12673 }%

```

Start sub-item.

```

12674 \glstreesubitem
12675 \glssubentryitem{##2}%
12676 \else

```

All other levels.

```

12677 \@glxstr@bookindex@subsep
12678 \@glxstr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12679 \let\@glxstr@bookindex@subsep\relax
12680 \edef\@glxstr@bookindex@subsubbetween{%
12681     \noexpand\glxstrbookindexsubsubbetween{##2}%
12682 }%
12683 \edef\@glxstr@bookindex@atsubsubendgroup{%
12684     \noexpand\glxstrbookindexatsubsubendgroup{##1}%
12685 }%

```

Start sub-sub-item.

```
12686 \glstreesubsubitem
12687 \fi
```

Format entry.

```
12688 \glstarget{##2}{\glxtrbookindexsubname{##2}}%
12689 \glxtrbookindexsubprelocation{##2}##3%
12690 }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12691 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12692 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12693 \@glxtr@bookindex@subsubatendgroup
12694 \@glxtr@bookindex@subatendgroup
12695 \@glxtr@bookindex@atendgroup
12696 \@glxtr@bookindexgroupskip
```

Update separators.

```
12697 \let\@glxtr@bookindexgroupskip\glxtrbookindexgroupskip
12698 \let\@glxtr@bookindex@between\@gobble
12699 \let\@glxtr@bookindex@atendgroup\relax
12700 \let\@glxtr@bookindex@subatendgroup\relax
12701 \let\@glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12702 \glxtrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12703 \glxtrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12704 \glxtrbookindexformatheader{\thisgrptitle}%
12705 \nopagebreak\indexspace\nopagebreak\@afterheading
12706 }%
12707 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glxtrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`bookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
12708 \newcommand{\glxtrbookindexthepage}{%
12709 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
12710 }
```

bookindexmarkentry Writes entry information to the .aux file. The argument is the entry label.

```
12711 \newcommand*\glxtrbookindexmarkentry}[1]{%
12712   \protected@write\@auxout
12713   {\let\glxtrbookindexthepage\relax}%
12714   {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
12715 }
```

etbookindexmark

```
12716 \newcommand*\glxtr@setbookindexmark}[2]{%
12717   \ifcsundef\glxtr@idxfirstmark@#1}%
12718   {\csgdef\glxtr@idxfirstmark@#1}{#2}}%
12719   {}%
12720   \csgdef\glxtr@idxlastmark@#1}{#2}%
12721 }
```

indexfirstmarkfmt

```
12722 \newcommand*\glxtrbookindexfirstmarkfmt}[1]{%
12723   \glxtr@entryname{#1}%
12724 }
```

indexfirstmark

```
12725 \newcommand*\glxtrbookindexfirstmark{%
12726   \letcs\glxtr@label\glxtr@idxfirstmark@\glxtrbookindexthepage}%
12727   \ifdef\glxtr@label
12728   {\glxtrbookindexfirstmarkfmt\glxtr@label}}%
12729   {}%
12730 }
```

indexlastmarkfmt

```
12731 \newcommand*\glxtrbookindexlastmarkfmt}[1]{%
12732   \glxtr@entryname{#1}%
12733 }
```

okindexlastmark

```
12734 \newcommand*\glxtrbookindexlastmark{%
12735   \letcs\glxtr@label\glxtr@idxlastmark@\glxtrbookindexthepage}%
12736   \ifdef\glxtr@label
12737   {\glxtrbookindexlastmarkfmt\glxtr@label}}%
12738   {}%
12739 }
```


Glossary

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **First use flag** & **First use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

`makeindex` An indexing application.

`xindy` An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)		\@Glsymbol@: added redefinition	65
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	65
\Glsfmtshort: new	304	\@Glstext@: added redefinition	61
\glsfmtshort: new	304	\@Glsuseri@: added redefinition	66
\Glsfmtshortpl: new	304	\@Glsuserii@: added redefinition	66
\glsfmtshortpl: new	304	\@Glsuseriii@: added redefinition	66
short: switched inline full form to short		\@Glsuseriv@: added redefinition	66
(long)	207	\@Glsuserv@: added redefinition	67
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	67
\@ACRlong: added redefinition	70	\@acrlong: added redefinition	70
\@ACRlongpl: added redefinition	71	\@acrlongpl: added redefinition	71
\@ACRshort: added redefinition	68	\@acrshort: added redefinition	67
\@ACRshortpl: added redefinition	69	\@acrshortpl: added redefinition	68
\@Acrlong: added redefinition	70	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	71	argument	54
\@Acrshort: added redefinition	68	\@glsdescplural@: added redefinition	64
\@Acrshortpl: added redefinition	69	\@glsfirst@: added redefinition	61
\@GLSdesc@: added redefinition	64	\@glsfirstplural@: added redefinition	62
\@GLSdescplural@: added redefinition	64	\@glsplural@: added redefinition	62
\@GLSfirst@: added redefinition	61	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	63	redefinition	65
\@GLSname@: added redefinition	63	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	62	new	121
\@GLSSymbol@: added redefinition	65	\@glsxtr@field@linkdefs: new	60
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	177
redefinition	65	\@print@glossary: added redefinition	117
\@GLStext@: added redefinition	60	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	66	\abbrvdefaultfont	182
\@GLSuserii@: added redefinition	66	\glsaccessdesc: new	144
\@GLSuseriii@: added redefinition	66	\glsaccessdescplural: new	145
\@GLSuseriv@: added redefinition	67	\glsaccessfirst: new	142
\@GLSuserv@: added redefinition	67	\glsaccessfirstplural: new	142
\@GLSuservi@: added redefinition	67	\Glsaccesslong: new	146
\@Glsdesc@: added redefinition	64	\glsaccesslong: new	146
\@Glsdescplural@: added redefinition	64	\glsaccessname: new	140
\@Glsfirst@: added redefinition	61	\glsaccessplural: new	141
\@Glsfirstplural@: added redefinition	63	\Glsaccessshort: new	145
\@Glsname@: added redefinition	63	\glsaccessshort: new	145
\@Glsplural@: added redefinition	62	\Glsaccessshortpl: new	146

\glsaccesssshortpl: new	146	\cGLSpl: new	95
\glsaccessssymbol: new	143	\cGLSpl@: new	95
\glsaccessssymbolplural: new	143	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	141	new	90
\glstentryfmt: added check for short ..	54	\cGLS: new	94
\glslongpltok: new	177	\cGLSformat: new	95
\glsshortpltok: new	176	\cGLSpl: new	95
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	95
name in \setkeys	178	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	15
for plural	173	\glsenableentrycount: new	90
\GLSxtrlongpl: new	191	\glsfirstabbrvdefaultfont: new ..	182
\Glsxtrlongpl: new	191	\glsfirstlongdefaultfont: new ...	182
\glsxtrlongpl: new	190	\Glsfmtfirst: new	307
\glsxtrNoGlossaryWarning: new	20	\glsfmtfirst: new	306
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	307
new	173	\glsfmtfirstpl: new	307
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	306
new	173	\glsfmtplural: new	306
\glsxtrpostlinkendsentence: new ..	173	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	190	\Glsxtrtitleshort	304
\Glsxtrshortpl: new	189	renamed from \Glstentryfmtshort ..	304
\glsxtrshortpl: new	189	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	304
\glslabeltok	202	renamed from \glstentryfmtshort ..	304
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	200	\Glsxtrtitleshortpl	304
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	304
redefinition of \acronymtype	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	304
\Glsxtrshort	304	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	304
\glsxtrshort	304	\Glsfmttext: new	306
\Glsfmtshortpl: changed to use		\glsfmttext: new	305
\glsxtrshortpl	304	\glshasattribute: new	152
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	151
\glsxtrshortpl	304	\glsxtremsuffix: new	243
\glsxtrifemptyglossary: new	25	\GlsXtrEnableEntryCounting: new ..	89
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	92
argument	155	\glsxtrscfont: new	215
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	215
argument	155	\glsxtrsmfont: new	229
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	229
default type to \acronymtype	103	short-em: new	250
\newterm: fixed name argument	154	short-em-desc: new	252
0.5 (2015-12-07)		short-em-footnote: new	261
\@cGLS: new	94	short-em-long: new	247
\@cGLS@: new	95	short-em-long-desc: new	248

short-em-postfootnote: new	263	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	225	attribute	295
short-sc-postfootnote: new	227	\Glsxtrheadtext: now uses headuc	
short-sm: new	233	attribute	297
short-sm-desc: new	234	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	240	attribute	297
short-sm-long: new	231	short-em-footnote: switch off regular	
short-sm-long-desc: new	232	attribute if set	261
short-sm-postfootnote: new	241	short-long: switch off regular attribute	
long-noshort-em: new	254	if set	201
long-noshort-em-desc: new	258	short-long-desc: switch off regular	
long-noshort-sm: new	236	attribute if set	202
long-noshort-sm-desc: new	238	short-sc-footnote: switch off regular	
long-short-em: new	244	attribute if set	226
long-short-em-desc: new	245	short-sm-footnote: switch off regular	
long-short-sm: new	230	attribute if set	240
long-short-sm-desc: new	231	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	199
\Glsaccessstext: new	141	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	200
\@glxtr@doaccsupp: new	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	294	attribute if set	217
\Glsaccessdesc: new	144	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	145	set	203
\Glsaccessfirst: new	142	postfootnote: switch off regular	
\Glsaccessfirstplural: new	142	attribute if set	205
\Glsaccessname: new	140	0.5.2 (2015-12-08)	
\Glsaccessplural: new	141	\@GLSdesc@: added accessibility support	64
\Glsaccesssymbol: new	143	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	144	support	64
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	299	support	61
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	299	support	63
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	63
headuc attribute	300	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	62
headuc attribute	299	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	65
attribute	298	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	65
attribute	298	\@GLStext@: added accessibility support	60
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	64
attribute	295	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	64
attribute	294	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	61
attribute	296	\@GLSfirstplural@: added accessibility	
		support	63

\@Glsname@: add accessibility support ..	63	\GLSaccesssymbolplural: new ..	144, 149
\@Glsplural@: added accessibility support	62	\GLSaccessstext: new	141, 148
\@Glsymbol@: added accessibility support	65	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	65	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	61	\glsxtrabbrvfmt to here	54
\@glsdesc@: added accessibility support	64	\GlsXtrEnableInitialTagging: new	169
\@glsdescplural@: added accessibility support	64	\glsxtrfieldtitlecase: new	156
\@glsfirst@: added accessibility support	61	\GlsXtrFormatLocationList: new ...	52
\@glsfirstplural@: added accessibility support	62	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	63	new	180
\@Glsplural@: added accessibility support	62	\glsxtrtagfont: new	171
\@Glsymbol@: added accessibility support	65	\KV@printgloss@nonumberlist: added	53
\@Glsymbolplural@: added accessibility support	65	\mfu@checkword@do: added	170
\@Glstext@: added accessibility support	60	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	195
new	171	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	171	\@glsxtr@autoindex@at: new	167
\@glsxtr@tag: new	171	\@glsxtr@autoindex@encap: new ...	167
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	168
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	167
\@ifpackageloaded	140	\@glsxtr@autoindex@setname: new ..	165
removed \glsxtrabbrvfmt	192	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added	51	General: removed	
\Glossentrydesc: added	169	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	161	120
\Glossentrysymbol: added	169	\glsdescwidth: added	51
\glossentrysymbol: added	169	\glspagelistwidth: added	51
\GLSaccessdesc: new	144, 149	\glsxtrdoautoindexname: new	165
\GLSaccessdescplural: new ...	145, 149	\glsxtrpostnamehook: new	162
\GLSaccessfirst: new	142, 148	\if@glsxtr@format@override: new ..	164
\GLSaccessfirstplural: new ..	143, 148	\ProvidesGlossariesExtraLang: new	310
\GLSaccesslong: new	147, 150	\RequireGlossariesExtraLang: new	310
\GLSaccesslongpl: new	147, 150	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new	147	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new	147	new	96
\GLSaccessname: new	140, 147	\@GLSxtr@p@acrlong@: new	83
\GLSaccessplural: new	141, 148	\@GLSxtr@p@acrlongpl@: new	83
\GLSaccessshort: new	146, 150	\@GLSxtr@p@acrshort@: new	83
\GLSaccessshortpl: new	146, 150	\@GLSxtr@p@acrshortpl@: new	83
\GLSaccesssymbol: new	143, 149	\@GLSxtr@p@long@: new	82
		\@GLSxtr@p@longpl@: new	83
		\@GLSxtr@p@plural@: new	81
		\@GLSxtr@p@short@: new	82
		\@GLSxtr@p@shortpl@: new	82
		\@GLSxtr@p@text@: new	81
		\@GlsXtrEnableOnTheFly: new	47
		\@Glsxtr: new	48
		\@Glsxtr@p@acrlong@: new	83
		\@Glsxtr@p@acrlongpl@: new	83

\@Glsxtr@p@acrshort@: new	83	\glsenableentryunitcount: new	98
\@Glsxtr@p@acrshortpl@: new	83	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	82	entry's existence	152
\@Glsxtr@p@longpl@: new	82	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	81	entry's existence	152
\@Glsxtr@p@short@: new	81	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	82	check	172
\@Glsxtr@p@text@: new	81	\Glsxtr: new	48
\@Glsxtrpl: new	48	\glxtr: new	47
\@alt@glshyp@opt: new	77	\glxtrcat: new	47
\@glscalt@hyp@opt: new	77	\glxtrdowrglossaryhook: new	77
\@glscalt@hyp@opt@char: new	77	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	77	new	101
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	46
new	97	\Glsxtrpl: new	48
\@glsc@local@increment@currunitcount:		\glxtrpl: new	48
new	97	\glxtrpostlocalreset: new	89
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	89
new	74	\glxtrpostreset: new	89
\@glsxtr: new	47	\glxtrpostunset: new	89
\@glsxtr@addunitcounter: new	96	\glxtrprotectlinks: new	80
\@glsxtr@currunitcount: new	98	\GlsXtrSetAltModifier: new	77
\@glsxtr@ifunitcounter: new	96	\GlsXtrSetDefaultGlsOpts: new	76
\@glsxtr@p@acrlong@: new	83	\glxtrstarflywarn: new	47
\@glsxtr@p@acrlongpl@: new	83	\GlsXtrWarning: new	49
\@glsxtr@p@acrshort@: new	83	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	83	disables \setacronymstyle	103
\@glsxtr@p@long@: new	82	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	82	\@glsxtr@autoindexcrossrefs: new .	15
\@glsxtr@p@plural@: new	81	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	81	new	111
\@glsxtr@p@shortpl@: new	82	\@glsxtr@idx@entrynumberlist: new	113
\@glsxtr@p@text@: new	81	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	98	new	112
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	101	new	113
\@glsxtr@unitcountlist: new	96	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	48	new	112
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	104
empty see value if not set and added		\makeglossaries: new	105
'see' to field key map	39	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	46	\glxtrdiscardperiod: added check	
\cGlsformat: added	95	for first use	173
\cGlsformat: added	95	short-desc: fixed typo in	
\cGlsplformat: added	96	\glxtrinlinefullformat and	
\cGlsplformat: added	95	added missing second argument ...	209
\glstdisablehyper: added	79	1.02 (2016-04-25)	
\glstdohyperlink: added	78	\@glsxtr@current@style: new	50
\glstdonohyperlink: added	80	\Glsfmtfull: new	309

\glsfmtfull: new	309	\@GLSdescplural@: set abbreviation and regular format	64
\Glsfmtfullpl: new	310	\@GLSfirst@: set abbreviation format ..	61
\glsfmtfullpl: new	309	\@GLSfirstplural@: set abbreviation and regular format	63
\Glsfmtlong: new	308	\@GLSname@: set abbreviation and regular format	63
\glsfmtlong: new	308	\@GLSplural@: set abbreviation and regular format	62
\Glsfmtlongpl: new	308	\@GLSsymbol@: set regular format	65
\glsfmtlongpl: new	308	\@GLSsymbolplural@: set regular format	65
\Glsxtrheadfull: new	303	\@GLStext@: set abbreviation and regular format	60
\glsxtrheadfull: new	302	\@GLSuseri@: set regular format	66
\Glsxtrheadfullpl: new	303	\@GLSuserii@: set regular format	66
\glsxtrheadfullpl: new	302	\@GLSuseriii@: set regular format	66
\Glsxtrheadlong: new	301	\@GLSuseriv@: set regular format	67
\glsxtrheadlong: new	300	\@GLSuservi@: set regular format	67
\Glsxtrheadlongpl: new	301	\@GLSdesc@: set abbreviation and regular format	64
\glsxtrheadlongpl: new	301	\@GLSdescplural@: set abbreviation and regular format	64
\Glsxtrtitlefull: new	303	\@GLSfirst@: set abbreviation and regular format	61
\glsxtrtitlefull: new	302	\@GLSfirstplural@: set abbreviation and regular format	63
\Glsxtrtitlefullpl: new	303	\@GLSname@: set abbreviation and regular format	63
\glsxtrtitlefullpl: new	303	\@GLSplural@: set abbreviation and regular format	62
\Glsxtrtitlelong: new	301	\@GLSsymbol@: set regular format	65
\glsxtrtitlelong: new	300	\@GLSsymbolplural@: set regular format	65
\Glsxtrtitlelongpl: new	302	\@GLStext@: set abbreviation and regular format	61
\glsxtrtitlelongpl: new	301	\@GLSuseri@: set regular format	66
\ifglxtrinsertinside: new	198	\@GLSuserii@: set regular format	66
postfootnote: added redef of \glxtrsetupfulldefs	205	\@GLSuseriii@: set regular format	66
stylemods: new	21	\@GLSuseriv@: set regular format	66
1.03 (2016-04-27)		\@GLSuservi@: set regular format	67
\@GLSfirstplural@: bug fix: misspelt cs name	63	\@GLSdesc@: set abbreviation and regular format	64
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLSdescplural@: set abbreviation and regular format	64
\@GLSfirstplural@: bug fix: misspelt cs name	63	\@GLSfirst@: set abbreviation and regular format	61
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLSfirstplural@: set abbreviation and regular format	63
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	62	\@GLSname@: set abbreviation and regular format	63
\glsxtrtitlelongpl: bug fix: changed \glxtrlong to \glxtrlongpl ..	301	\@GLSplural@: set abbreviation and regular format	62
\glsxtrtitleshortpl: bug fix: changed \glxtrshort to \glxtrshortpl	295	\@GLSsymbol@: set regular format	65
1.04 (2015-04-30)		\@GLSsymbolplural@: set regular format	65
short-em-footnote: renamed from “footnote-em”	261	\@GLStext@: set abbreviation and regular format	61
1.04 (2016-05-02)		\@GLSuseri@: set regular format	66
\@glxtrpostloctag: new	53	\@GLSuserii@: set regular format	66
\@GLSdesc@: set abbreviation and regular format	64	\@GLSuseriii@: set regular format	66
		\@GLSuseriv@: set regular format	66
		\@GLSuservi@: set regular format	67
		\@GLSdesc@: set abbreviation and regular format	64
		\@GLSdescplural@: set abbreviation and regular format	64
		\@GLSfirst@: set abbreviation and regular format	61

\@glsfirstplural@: set abbreviation and regular format	62	\glsxtruserparen: new	265
\@glsname@: set abbreviation and regular format	63	\glsxtrusersuffix: new	266
\@glsplural@: set abbreviation and regular format	62	\GlsXtrWarnDeprecatedAbbrStyle: new	197
\@glsymbol@: set regular format	65	short-em-long-em: new	249
\@glsymbolplural@: set regular format	65	short-em-long-em-desc: new	250
\@glstext@: set abbreviation and regular format	60	short-em-nolong: new	252
\@glsxtr@deprecated@abbrstyle: new	197	short-em-nolong-desc: new	253
\@glsxtr@do@style: new	21	short-em-postfootnote: renamed from “postfootnote-em”	263
\@glsxtr@doloctag: new	53	short-footnote: new	205
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	113	short-long-user: new	272
\@glsxtr@pagetag: new	53	short-long-user-desc: new	273
\@glsxtr@pagetag: new	53	short-nolong: new	208
\@glsxtr@preloctag: new	53	short-nolong-desc: new	210
\@glsxtr@postloctag: new	53	short-postfootnote: new	207
\@glsxtr@preloctag: new	52, 53	short-sc-footnote: renamed from “footnote-sc”	225
\glossentrydesc: added glossdescfont attribute check	157	short-sc-nolong: new	220
\Glossentryname: added glossnamefont attribute check	161	short-sc-nolong-desc: new	221
\glossentryname: added glossnamefont attribute check	158	short-sc-postfootnote: renamed from “postfootnote-sc”	227
moved post name hook inside condition	160	short-sm-footnote: renamed from “footnote-sm”	240
\glsabbrvmfont: new	243	short-sm-nolong: new	234
\glsabbrvuserfont: new	265	short-sm-nolong-desc: new	236
\glsfirstabbrvmfont: new	243	short-sm-postfootnote: renamed from “postfootnote-sm”	241
\glsfirstabbrvuserfont: new	265	\letabbreviationstyle: new	197
\glsfirstlongemfont: new	243	\newabbreviationstyle: bug fix: corrected test for existence	196
\glsfirstlonguserfont: new	266	long-em-noshort-em: new	256
\glsifnotregularcategory: new ...	153	long-em-noshort-em-desc: new ...	259
\glslongdefaultfont: new	182	long-em-short-em: new	245
\glslongemfont: new	244	long-em-short-em-desc: new	246
\glslongfont: new	182	long-noshort: new	214
\glslonguserfont: new	265	long-noshort-desc: new	213
\glsxtrassignfieldfont: new	60	long-noshort-em: renamed from “long-em”	254
\GlsXtrEnablePreLocationTag: new .	52	long-noshort-em-desc: renamed from “long-desc-em”	258
\glsxtrfirstscfont: new	215	long-noshort-sc: renamed from “long-sc”	222
\glsxtrfirstsmfont: new	229	long-noshort-sc-desc: renamed from “long-desc-sc”	224
\glsxtrlongshortdescsort: new ...	200	long-noshort-sm: renamed from “long-sm”	236
\glsxtrpostnamehook: added category check	162	long-noshort-sm-desc: renamed from \long-desc-sm	238
\glsxtrregularfont: new	54		
\glsxtruserfield: new	265		

long-short-user: new	266	docdef option changed to choice	14
long-short-user-desc: new	272	\glstr@usesee: new	39
\renewabbreviationstyle: new	196	\glstrusesee: new	39
style: new	21	\glstruseseeformat: new	39
1.05 (2016-06-10)		\if@glstrdocdefrestricted: new ..	14
\eglssetwidest: new	329	1.07 (2016-08-15)	
\glFindWidestAnyName: new	332	\@@glstrp: new	84
\glFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	337	nohyperfirst attribute	62
\glFindWidestAnyNameSymbol: new	335	\@GLSfirstplural@: added check for	
\glFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	63
new	336	\@GLSxtrp: new	85
\glFindWidestLevelTwo: new	333	\@Glsfirst@: added check for	
\glFindWidestUsedAnyName: new ..	332	nohyperfirst attribute	61
\glFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	337	nohyperfirst attribute	63
\glFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	84
new	334	\@gls@preglossaryhook: added	
\glFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	172
new	335	\@glsfirst@: added check for	
\glFindWidestUsedLevelTwo: new .	332	nohyperfirst attribute	61
\glFindWidestUsedTopLevelName:		\@Glsfirstplural@: added check for	
new	331	nohyperfirst attribute	62
\glsfirstlongfootnotefont: new ..	203	\@glstrinmark: new	292
\glsgetwidestname: new	331	\@glstrnotinmark: new	292
\glsgetwidestsubname: new	331	\@glstrp: new	84
\glslongfootnotefont: new	203	\@glstrp@opt: new	83
\glstrAltTreeIndent: new	329	\glossxtrsetpopts: new	84
\glstralttreeInit: new	329	\glsps: new	86
\glstrAltTreePar: new	329	\glspt: new	86
\glstrAltTreeSetHangIndent: new	338	\glstr@entry@p: new	85
\glstrAltTreeSetSubHangIndent:		\glstrabbrvfootnote: new	203
new	339	\glstrchecknohyperfirst: new	61
\glstralttreeSubSymbolDescLocation:		\glstrfieldtitlecasecs: new	156
new	329	\glstrifinmark: new	292
\glstralttreeSymbolDescLocation:		\GLSxtrp: new	87
new	328	\Glsxtrp: new	86
\glstrComputeTreeIndent: new ...	338	\glstrp: new	85
\glstrComputeTreeSubIndent: new	338	\glstrsetpopts: new	84
\glstrtreetopindent: new	329	short-long-desc: added text key	202
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form	248	plural key	202
\xglsetwidest: new	330	long-short-desc: added missing text	
1.06 (2016-06-18)		key	200
\@glsdoifexistsorwarn: new	14	fixed misspelling of \glsabbrvfont .	200
\@glstr@docdefval: new	14	footnote: changed first forms to use	
\@glstr@usesee: new	39	\glsfirstlongfootnotefont ...	203
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	25	from first keys	205

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	206	1.10 (2016-12-17)	\@GLSp1@: fixed bug caused by typo in command name	56
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse	104	1.11 (2017-01-19)	\@glsxtr@do@redef@for@gl@sentries: new	6
1.08 (2016-12-13)			\@glsxtr@noidx@do: new	131
\@glsxtr@record: new	8		\@glsxtr@redef@for@gl@sentries: new ..	6
\@GLS@: added \@glsxtr@record	55		\@glsxtr@shortcutsval: new	19
\@GLSp1@: added \@glsxtr@record ...	56		\@glsxtr@unsrt@getgroup@title: new	130
\@Gls@: added \@glsxtr@record	55		\@print@noidx@glossary: added redefinition	115
\@Glspl@: added \@glsxtr@record ...	55		\glsxtr@addloc@listfield: added group key	12
\@gls@: added \@glsxtr@record	55		added location key	12
\@gls@@link@: added \@glsxtr@record	56		\glsxtr@fields: new	123
\@gls@field@link: added \@glsxtr@record	54		\glsxtr@linkprefix: new	123
\@gls@saveentrycounter: new	25		\glsxtr@org@new@ignored@glossary: new	34
\@glsdisp: added \@glsxtr@record ..	56		\glsxtr@s@new@ignored@glossary: new	35
\@glspl@: added \@glsxtr@record ...	55		\glsxtr@shortcutsval: new	123
\@glsxtr@dorecord: new	10		\glsxtr@texencoding: new	123
\@glsxtr@err@undefaction: new	6		\glsxtr@writefields: new	123
\@glsxtr@record: new	7		\GlsXtrLoadResources: new	122
\@glsxtr@warn@onexists@do: new ...	6		\glsxtr@resourcefile: changed extension to .glstex	122
\@glsxtr@warn@undefaction: new	6		\new@ignored@glossary: added starred version	34
\@print@unsrt@glossary: new	127	1.12 (2017-02-03)		
General: added record package option ...	12		\@glsxtr@recordcounter: new	10
\glsadd: added \@glsxtr@record	59		\@gls@preglossaryhook: check for definition	171
\glsdoifexists: now defines \glslabel	37		\@glsxtr@counter@recordhook: new ..	125
\glsxtr@do@wrglossary: new	25		\@glsxtr@display@loc: new	115
\glsxtr@addloc@listfield: new	11		\@glsxtr@docounter@record: new ...	125
\glsxtr@indexonly@saveentrycounter: new	11		\@glsxtr@longnew@glossaryentry: new	33
\glsxtr@record: new	125		\@glsxtr@noop@recordcounter: new ..	11
\glsxtr@resource: new	123		\@glsxtr@op@recordcounter: new ...	11
\glsxtr@saveentrycounter: new	25		\@glsxtr@provide@storagekey: new ..	26
\glsxtr@setup@record: new	11		\@glsxtr@s@longnew@glossaryentry: new	33
\glsxtr@assignfieldfont: added check for existence	60		\@glsxtrentryfmt: new	28
\glsxtr@resourcefile: new	122		\@glsxtr@index@aliased: new	75
\printunsrt@glossaries: new	127		\@glsxtr@set@alias@noindex: new	75
\printunsrt@glossary: new	127		\@new@glossaryentry@posthook: added check for alias key	43
1.09 (2016-12-16)			\@no@glsxtr@index@aliased: new	75
\@glsxtr@gettype: new	111		\@printunsrt@glossary: new	127
\@glsxtr@mixed@assign@sortkey: new	111			
\@print@glossary: redefined to save options	110			
\glsxtr@make@glossaries: new	111			

General: added target key to printgloss		\GlsXtrLoadResources: removed	
family	110	restriction on only one per document	122
\apptoglossarypreamble: new	32	\glxtrlocrangefmt: new	117
\csGlsXtrLetField: new	31	\glxtrpostlongdescription: new ..	34
\eGlsXtrSetField: new	32	\glxtrprovidestoragekey: new	26
\gGlsXtrSetField: new	31	\GlsXtrRecordCounter: new	125
\glsdohyperlink: added check for alias		\glxtrresourcecount: new	122
field	79	\glxtrresourcefile: added catcode	
\glsnoidxdisplayloc: added		change for @	122
redefinition	115	\glxtrsetaliasnoindex: new	75
\glissettoctitle: added patch	35	\GlsXtrSetField: new	31
\glxtr@counterrecord: new	125	\glxtrsetfieldifexists: new	31
\glxtr@langtag: new	123	\glxtrunsrtdo: new	130
\glxtr@newabbreviation: new	178	\Glsxtrusefield: new	30
\glxtr@org@newignoredglossary:		\glxtrusefield: new	30
Added check for existence	34	short-postlong-user: new	269
\glxtr@pluralsuffixes: new	123	short-postlong-user-desc: new ...	271
\glxtr@provideignoredglossary:		\longnewglossaryentry: added starred	
new	36	version	33
\glxtr@s@newignoredglossary:		long-postshort-user: new	267
Added check for existence	35	long-postshort-user-desc: new ...	268
\glxtr@s@provideignoredglossary:		postdot: new	15
new	36	\pretoglossarypreamble: new	32
\glxtrabbrvpluralsuffix: new ...	182	\print@noop@unsrtglossaryunit:	
\glxtralias: new	43	new	130
\glxtrcopytoglossary: new	37	\print@op@unsrtglossaryunit: new	129
\glxtrdeffield: new	31	\printunsrtglossary: added starred	
\glxtrdisplayendloc: new	116	form	127
\glxtrdisplayendlochook: new ...	116	\printunsrtglossaryhandler: new .	129
\glxtrdisplaysingleloc: new	116	\printunsrtglossaryunit: new	11
\glxtrdisplaystartloc: new	116	\printunsrtglossaryunitsetup: new	129
\glxtrdeffield: new	31	\provideignoredglossary: new	36
\glxtrentryfmt: new	28	\s@glxtr@provide@storagekey: new	26
\glxtrfielddolistloop: new	29	\s@printunsrtglossary: new	127
\glxtrfieldforlistloop: new	29	\xGlsXtrSetField: new	31
\glxtrfieldifinlist: new	29	1.13 (2017-02-07)	
\glxtrfieldlistadd: new	28	\@glsdisp: removed	
\glxtrfieldlistead: new	29	\@glxtr@org@glsdisp	56
\glxtrfieldlistgadd: new	29	\glxtrsetaliasnoindex: switched to	
\glxtrfieldlistxadd: new	29	\providecommand	75
\glxtrfieldxifinlist: new	29	1.14 (2017-04-18)	
\glxtrfmt: new	27	\@gls@link: added redefinition	57
\GlsXtrFmtDefaultOptions: new	27	\@gls@noidx@getgrouptitle: new ..	113
\GlsXtrFmtField: new	27	\@gls@removespaces: new	117
\glxtrifkeydefined: new	26	\@glxtr@do@automake@err: new ...	124
\glxtrindexaliased: new	76	\@glxtr@org@gloautosee: new	23
\GlsXtrLetField: new	31	\@glxtr@record: added third arg	7
\GlsXtrLetFieldToField: new	31	\@glxtr@recordsee: new	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue	59	\@glo@autosee: added redefinition	24
added \glsadd option thevalue	59	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	79	bug	113
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	91	check for seealso field	44
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@	99	instead of \csname	131
\glsnavigation: new	114	\@glsxtr@dorecordnodefer: new	10
\glsxtr@org@getgrouptitle: new ..	114	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new	7	misspelt command	128
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake	124	new	129
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	116	\@gls@setupsort@none	13
\glsxtrgetgrouptitle: new	114	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new	56	redefinition	24
\glsxtrlocationhyperlink: new ...	117	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new	114	\providecommand lines	123
\glsxtrsupphypernumber: new	117	\@glsxtrautoindex: new	165
\ifglsxtrwrglossbefore: new	57	\@glsxtrautoindexentry: new	165
1.15 (2017-05-10)		\@glsxtrautoindexsort: new	166
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new	40
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new	43
short-em-long-em: fixed spelling of		\@glsxtrseelist: new	40
\glsabbrvfont	249	\@glsxtruseseealso: new	40
short-long: fixed spelling of		\@glsxtruseseealsoformat: new	40
\glsabbrvfont	201	\@seealso: new	40
short-long-user: fixed spelling of		autoseeindex: new	15
\glsabbrvfont	272	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	177
\glsabbrvfont	269	\@glsxtr@markwordseps: new	177
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	271	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	112
\glsabbrvfont	246	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	267	\glsxtrundeftag	113
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	269	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	112
\glsabbrvfont	199	\@glsxtrifhyphenstart: new	274
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	266	in the abbreviation styles	198
footnote: fixed spelling of		\glsabbrvhyphenfont: new	275
\glsabbrvfont	203	\glsabbrvonlyfont: new	288
postfootnote: fixed spelling of		\glsabbrvscfont: new	215
\glsabbrvfont	205	\glsabbrvsmfont: new	229

\glsabbrvuserfont: initialised to default font	265	short-long-user-desc: corrected first forms	273
\glsfirstabbrvhyphenfont: new ...	275	short-nolong-desc-noreg: new ...	210
\glsfirstabbrvonlyfont: new	288	short-nolong-noreg: new	208
\glsfirstabbrvscfont: new	215	long-em-noshort-em-desc-noreg: new	261
\glsfirstabbrvsmfont: new	229	long-em-noshort-em-noreg: new ...	257
\glsfirstlonghyphenfont: new ...	275	long-hyphen-noshort-desc-noreg: new	277
\glsfirstlongonlyfont: new	288	long-hyphen-postshort-hyphen: new	280
\glslonghyphenfont: new	275	long-hyphen-postshort-hyphen-desc: new	282
\glslongonlyfont: new	288	long-hyphen-short-hyphen: new ...	275
\glslonguserfont: initialised to default font	265	long-hyphen-short-hyphen-desc: new	276
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong	178	long-noshort-desc-noreg: new ...	213
\GlsXtrDefineAcShortcuts: new	18	long-noshort-noreg: new	214
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside	192	long-only-short-only: new	289
\glsxtrhyphensuffix: new	275	long-only-short-only-desc: new ..	290
\glsxtrifhyphenstart: new	274	long-short-user-desc: corrected first forms	272
\glsxtrlonghyphen: new	279	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	277	stylemods: changed default value to "default"	21
\glsxtrlonghyphenshort: new	274	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	200	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new	290	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref	10
\glsxtronlydescsort: new	290	\@glsxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new	288	\@glsxtr@record@setting: new	12
\glsxtrparen: new	180	\@glsxtr@record@setting@alsoindex: new	12
\glsxtrposthyphenlong: new	285	\@glsxtrifhasfield: new	30
\glsxtrposthyphenshort: new	280	General: added \glslink option theHvalue	57
\glsxtrposthyphensubsequent: new	280	added \glslink option thevalue ...	57
\glsxtrshortdescname: new	209	\glsxtr@writefields: removed double-quotes around \jobname ..	124
\glsxtrshorthyphen: new	285	\glsxtrdoautoindexname: changed format test	165
\glsxtrshorthyphenlong: new	283	\glsxtrhyperlink: new	79
\glsxtrshortlongdescname: new ...	202	\glsxtrifhasfield: new	30
\glsxtrshortlongdescsort: new ...	202	\GlsXtrSetDefaultNumberFormat: new	7
\Glsxtrsubsequentfmt: new	195	\s@glsxtrifhasfield: new	30
\glsxtrsubsequentfmt: new	194		
\Glsxtrsubsequentplfmt: new	195		
\glsxtrsubsequentplfmt: new	194		
\glsxtrword: new	177		
\glsxtrwordsep: new	177		
short-hyphen-long-hyphen: new ...	283		
short-hyphen-long-hyphen-desc: new	284		
short-hyphen-postlong-hyphen: new	285		
short-hyphen-postlong-hyphen-desc: new	287		

headings	343	\glxstrbookindexbetween: new	348
redefined		\glxstrbookindexbookmark: new ...	348
mcoltreenonamehypergroup to		\glxstrbookindexcols: new	347
discourage breaks after group		\glxstrbookindexcolspread: new ..	348
headings	343	\glxstrbookindexfirstmark: new ..	352
redefined mcoltreenonamespannav to		\glxstrbookindexfirstmarkfmt: new	352
discourage breaks after group		\glxstrbookindexformatheader: new	348
headings	344	\glxstrbookindexgroupskip: new ..	348
redefined mcoltreepspannav to		\glxstrbookindexlastmark: new ...	352
discourage breaks after group		\glxstrbookindexlastmarkfmt: new	352
headings	342	\glxstrbookindexmarkentry: new ..	352
redefined treegroup to discourage		\glxstrbookindexname: new	347
breaks after group headings	326	\glxstrbookindexparentchildsep:	
redefined treehypergroup to		new	347
discourage breaks after group		\glxstrbookindexparentschildsep:	
headings	327	new	347
redefined treenonamegroup to		\glxstrbookindexprelocation: new	347
discourage breaks after group		\glxstrbookindexsubatendgroup:	
headings	328	new	348
redefined treenonamehypergroup to		\glxstrbookindexsubbetween: new ..	348
discourage breaks after group		\glxstrbookindexsubname: new	347
headings	328	\glxstrbookindexsubprelocation:	
debug: new	22	new	347
\gglsssetwidest: new	329	\glxstrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new	348
existence	79	\glxstrbookindexsubsubbetween:	
changed to use \def rather than \let .	79	new	348
\glsenablehyper: changed to use \def		\glxstrbookindexthepage: new	351
rather than \let	80	\glxstrdetoklocation: new	135
\Glsfmtname: new	305	\glxtrenablerecordcount: new ...	135
\glsfmtname: new	305	\glxstrglossentry: new	125
\glshex: new	132	\glxstrgroupfield: new	130
\glslstchildpostlocation: new ..	314	\Glsxtrheadname: new	296
\glslstchildprelocation: new ...	314	\glxtrheadname: new	296
\glslstprelocation: new	314	\GlsXtrIfFieldEqStr: new	32
\glsnahyperlink: patched	78	\glxstriflabelinlist: new	129
\glssseeitemformat: new	39	\glxstrifrecordtrigger: new	135
\glsshows target: new	23	\glxstrindexseealso: added check	
\glstreechildprelocation: new ...	324	that the entry exists	41
\glstreeprelocation: new	324	\glxstrinithyperoutside: new	57
\glstriggerrecordformat: new	136	\GlsXtrLocationRecordCount: new ..	135
\glseuseabbrvfont: new	192	\glxstrnewgl: new	132, 133
\glseuselongfont: new	192	\glxstrnewGLSlike: new	134
\glxstr@do@alsoindex@wrglossary:		\glxstrnewglsl: new	134
new	8	\glxstrnewrgls: new	134
\glxstr@org@@do@wrglossary: new ..	25	\glxstrnewrGLSlike: new	134
\glxstr@org@dohyperlink: new	77	\glxstrnewrglslike: new	134
\glxstr@setbookindexmark: new ...	352	\glxstrprelocation: new	313, 347
\glxstrbookindexatendgroup: new ..	348	\GlsXtrRecordCount: new	134

\glxtrrecordtriggervalue: new ..	135	\glssseeitemformat: switched check	
\glxtrresourcefile: now disables		from regular to short	39
record key	122	\glxtr@setaccessdisplay: new ...	162
\glxtrresourceinit: new	132	\glxtr@writefields: provide	
\GlsXtrSetRecordCountAttribute:		\glxtr@record in aux file	123
new	135	\glxtractivatenopost: new	109
\glxtrtitlename: new	296	\glxtrbookindexprelocation:	
\glxtrtitleorpdforheading: new .	292	removed check for no post dot	347
\GlsXtrTotalRecordCount: new	134	\glxtrglossentryother: new	126
\glxtrwrglossmark: new	22	\glxtrnopostpunc: new	110
short-em: new	251	1.23 (2017-11-12)	
short-sc: corrected first letter		\@@glxtrfmt: added check for indexing	28
uppercasing	219	added grouping	27
short-sm: corrected first letter		new	27
uppercasing	234	\@glxtr@nopostpunc@postdesc: new	110
\ifglxtr@hyperoutside: new	57	\@glxtr@restore@postpunc: new ..	110
all: new	312	\@glxtrentryfmt: fixed missing label	
nolong-short: new	210	argument	28
nolong-short-em: new	253	\@glxtrfmt: new	27
nolong-short-noreg: new	211	\eglsupdatewidest: new	330
nolong-short-sc: new	221	\gglupdatewidest: new	330
nolong-short-sm: new	236	\glsupdatewidest: new	330
nopostdot: new	15	\GlsXtrDefineAbbreviationShortcuts:	
\printunstrtglossaryentryprocesshook:		changed \newabbr definition to use	
new	129	\providecommand	17
\printunstrtglossarypredoglossary:		\GlsXtrDefineAcShortcuts: changed	
new	129	\newabbr definition to use	
\rGLS: new	138	\providecommand	18
\rGls: new	137	\glxtrfmtdisplay: new	28
\rgls: new	137	\glxtrifcustomdiscardperiod: new	173
\rGLSformat: new	140	\GlsXtrIfFieldUndef: new	30
\rGlsformat: new	139	\glxtrrestorepostpunc: new	110
\rglsformat: new	139	\s@glxtrfmt: new	27
\rGLSpl: new	139	\s@glxtrfmt: new	27
\rGlspl: new	138	\xglupdatewidest: new	330
\rglspl: new	137	1.24 (2017-11-14)	
\rGLSplformat: new	140	\glsadd: added \@gls@setsort	60
\rglsplformat: new	139	\glxtrforcsvfield: new	29
\rglsplformat: new	139	\glxtrlocalsetgrouptitle: new ..	114
\s@glxtrifhasfield: switched from		1.25 (2017-11-14)	
\ifdef to \ifundef	30	\glxtrbookindexmulticolseenv: new	348
1.22 (2017-11-08)		1.25 (2017-11-24)	
\@glxtr@nopostpunc: new	110	\glsextrapostnamehook: new	162
\@glxtr@orgprintglossary: changed		\glxtrfootnotename: new	203
explicit \let for \nopostdesc to		\glxtrlongnoshortdescname: new .	212
\glxtractivatenopost	109	\glxtrlongnoshortname: new	214
\@glxtrglossentryother: new	126	\glxtrlongshortname: new	198
\glossentrynameother: new	163	\glxtrlongshortuserdescname: new	268
		\glxtronlyname: new	288

\glstrpostlinkAddDescOnFirstUse:	\glstrshortlongname: new 200
changed to use \glstrparen 173	\glstrshortlonguserdescname: new 271
\glstrpostlinkAddSymbolOnFirstUse:	\glstrshortnolongname: new 207
changed to use \glstrparen 173	

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\.	15, 16, 173, 324
\@	122
\@cGLS@	91, 100
\@cGLSpl@	91, 100
\@cGls@	91, 99
\@cGlspl@	91, 100
\@cgls@	91, 99
\@cglspl@	91, 99
\@do@wrglossary	8, 9, 106
\@do@wrglossary	11, 13, 25, 60, 75
\@glo@assign@sortkey	111
\@glo@list	6
\@glo@type	127
\@glossarysec	348
\@gls@expand@field	26
\@glslocalreset	89
\@glslocalunset	89
\@glsreset	89
\@glsunset	89
\@glsxtr@autoindex@escspch	167, 168
\@glsxtr@checkspch	166–169
\@glsxtr@disabledflycommand	49
\@glsxtr@org@postdescription	110
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 125
\@glsxtrfmt	27
\@glsxtrp	84, 85
\@glsxtrpostloctag	52
\@glsxtrpreloctag	52
\@glsxtrwrglossmark	8, 11, 23, 25, 41, 45, 106
\@newglossaryentry@defcounters	90
\@newglossaryentry@defunitcounters	98
\@par	329
\@ACRlong	81
\@ACRlongpl	81
\@ACRshort	81
\@ACRshortpl	81
\@Acrlong	81
\@Acrlongpl	81
\@Acrshort	81
\@Acrshortpl	81
\@GLS@	80, 94, 95, 138
\@GLSdesc@	64
\@GLSpl@	80, 94, 95, 139
\@GLSplural@	81
\@GLSsymbol@	65
\@GLStext@	81
\@GLSxtr@full	184
\@GLSxtr@fullpl	185
\@GLSxtr@p@acrlong@	81
\@GLSxtr@p@acrlongpl@	81
\@GLSxtr@p@acrshort@	81
\@GLSxtr@p@acrshortpl@	81
\@GLSxtr@p@long@	80
\@GLSxtr@p@longpl@	80
\@GLSxtr@p@plural@	80
\@GLSxtr@p@short@	80
\@GLSxtr@p@shortpl@	80
\@GLSxtr@p@text@	80
\@GLSxtrlong	80, 188
\@GLSxtrlongpl	80, 191, 192
\@GLSxtrp	88
\@GLSxtrshort	80, 187
\@GLSxtrshortpl	80, 190
\@Gls@	80, 93, 94, 138
\@Gls@acentryname	102
\@Gls@entry@field	72, 87, 163
\@Gls@entryname	102
\@GlsXtrEnableOnTheFly	46, 47
\@Glspl@	80, 93, 94, 138
\@Glsplural@	81
\@Glstext@	81
\@Glsxtr	48, 49

\@Glsxtr@full	183	\@firstofone	60, 128, 157, 158, 164, 170
\@Glsxtr@fullpl	185	\@firstofthree	56,
\@Glsxtr@p@acrlong@	81		60, 68–71, 77, 183, 184, 186, 187, 189, 191
\@Glsxtr@p@acrlongpl@	81	\@firstoftwo	61–65, 69, 71, 74, 77, 104, 162,
\@Glsxtr@p@acrshort@	81		163, 174, 175, 183–185, 189–192, 292, 293
\@Glsxtr@p@acrshortpl@	81	\@for	6, 21, 30, 44, 90,
\@Glsxtr@p@long@	80		102, 105, 108, 115, 128, 135, 156, 162, 170
\@Glsxtr@p@longpl@	80	\@glo@alias	41–43
\@Glsxtr@p@plural@	80	\@glo@assign@sortkey	108
\@Glsxtr@p@short@	80	\@glo@autosee	23
\@Glsxtr@p@shortpl@	80	\@glo@autoseehook	42
\@Glsxtr@p@text@	80	\@glo@category	96
\@Glsxtrlong	80, 188	\@glo@check@sortallowed	108
\@Glsxtrlongpl	80, 191	\@glo@counterprefix	10, 117
\@Glsxtrp	87	\@glo@countunit	96
\@Glsxtrpl	48, 49	\@glo@default@sorttype	108
\@Glsxtrshort	80, 186	\@glo@desc	33, 34
\@Glsxtrshortpl	80, 189	\@glo@descplural	33, 34
\@acrlong	81	\@glo@group	12
\@acrlongpl	81	\@glo@label	
\@acrshort	81		11, 12, 26, 39, 42–44, 72, 79, 331–338
\@acrshortpl	81	\@glo@location	12
\@afterheading		\@glo@loclist	11
	315, 316, 325–328, 341–344, 351	\@glo@name	165
\@alt@glshyp@opt	77	\@glo@no@assign@sortkey	111
\@auxout	10, 11,	\@glo@parent	333, 334
	45, 53, 92, 100, 105, 106, 118, 122–125, 352	\@glo@see	39, 40, 42–44
\@bibgl@restoreat	122	\@glo@seealso	41, 42
\@cGLS	94	\@glo@sort	165
\@cGLS@	91, 94, 100	\@glo@sorttype	108, 115
\@cGLSpl	95	\@glo@text	56
\@cGLSpl@	91, 95, 100	\@glo@thislettergrp	131
\@cGls@	91, 99	\@glo@thisvalue	265
\@cGlspl@	91, 100	\@glo@tmp	26, 40, 72
\@cgls@	91, 99	\@glo@type	44, 78, 102, 105,
\@cglspl@	91, 99		108, 109, 111, 114, 115, 118, 121, 122, 128
\@disable@onlypremakeg	106	\@glo@types	154, 331–337
\@do@auxoutstuff	118	\@glossary@default@style	50, 109, 346
\@do@gl@getcounterprefix	10	\@glossarystyle	109
\@do@gl@see	42, 43	\@gl@	80, 93, 94, 137
\@do@newglossaryentry	102, 103, 180	\@gl@@link	56
\@do@seeglossary	13, 23, 45, 106	\@gl@ReturnAfterFi	117
\@do@wrglossary	58, 59, 136	\@gl@actualchar	166
\@empty	60, 68–71, 110, 166, 167, 183–192	\@gl@adjustmode	59
\@end@glxtr@addunused	44	\@gl@alt@hyp@opt	77
\@end@glxtr@gettype	108, 111	\@gl@alt@hyp@opt@char	77
\@end@glxtr@usese	39	\@gl@alt@hyp@opt@keys	77
\@end@glxtr@ifhyphenstart	274	\@gl@automake	108
\@endfortrue	29, 162, 195	\@gl@between	114, 115

\@gls@checkedmkidx	166–169	\@gls@noidxloclist@sep	112
\@gls@checkmkidxchars	41, 165	\@gls@noref@warn	106, 115
\@gls@codepage	118	\@gls@org@glsnoidxdisplayloc ..	112, 113
\@gls@counter	9, 10, 57, 59, 75, 136	\@gls@org@glsseeformat	112, 113
\@gls@currentlettergroup ...	115, 128, 131	\@gls@preglossaryhook	109, 170
\@gls@declareoption	5	\@gls@prevlevel	339, 340, 344, 345
\@gls@default@longpl	178, 179	\@gls@quotechar	166
\@gls@doautomake	108, 124	\@gls@reference	45, 105, 106
\@gls@doautomake@err	124	\@gls@saveentrycounter .	13, 25, 58, 59, 136
\@gls@encapchar	166	\@gls@see@noindex	24, 122
\@gls@entry@count	91, 92	\@gls@setdefault@glslink@opts	9, 28, 58, 76
\@gls@entry@field	26, 30, 42, 72, 85–88, 91, 126, 127	\@gls@setsort	58, 60
\@gls@entry@unitcount	100	\@gls@setupsort@none	13
\@gls@field@font	60–67	\@gls@short	179
\@gls@field@link	60–67, 72, 73	\@gls@shortpl	176, 179, 180
\@gls@getcounterprefix	10	\@gls@sort	131
\@gls@getgrouptitle	114, 128	\@gls@thisval	162, 163
\@gls@grptitle	78, 115	\@gls@tmp	115
\@gls@hyp@opt	72, 73, 77, 94, 95, 133, 137–139, 182–191	\@gls@tmpb	168, 169
\@gls@hyp@opt@cs	77	\@gls@type	106, 108, 195, 331–338
\@gls@ifinlist	129	\@gls@write@entrycounts	91
\@gls@increment@currcount	91	\@gls@write@entryunitcounts	100
\@gls@increment@currunitcount	99	\@gls@write@entryunitcounts@do	101
\@gls@keymap ..	11, 12, 26, 39, 42, 72, 123, 162	\@gls@xref	11, 41
\@gls@label ...	8–10, 45, 76, 77, 106, 125, 195	\@gls@abbrv@current@abbreviation	178, 192
\@gls@levelchar	166	\@gls@acronymlists	102
\@gls@link	27, 55, 56, 68–72, 183–192	\@gls@doifexistsorwarn	14, 158, 159, 161, 163
\@gls@link@checkfirsthyper	56, 104	\@gls@entry	92, 100, 101
\@gls@link@label	57, 136	\@gls@link	59, 78–80
\@gls@link@nocheckfirsthyper	55, 68–71, 183–192	\@gls@nextpages	109
\@gls@link@opts	57	\@gls@nonextpages	109
\@gls@list	114, 115	\@gls@numberformat	9, 10, 57, 59, 75, 136, 162, 165
\@gls@local@increment@currcount	91	\@gls@order	105
\@gls@local@increment@currunitcount	99	\@gls@pl@	80, 93, 94, 137
\@gls@location	131, 132	\@gls@plural@	81
\@gls@loclist	112, 113, 131, 132	\@gls@punc@token	175
\@gls@long	178	\@gls@recordlocref	10
\@gls@longpl	176, 178–180	\@gls@showtarget	79
\@gls@map	162	\@gls@style@altlist	314
\@gls@nohyperlist	34, 36	\@gls@style@altlistgroup	315
\@gls@noidx@do	115	\@gls@style@altlisthypergroup	316
\@gls@noidx@getgrouptitle	128	\@gls@style@alttree	328
\@gls@noidx@nosanitizesort	107	\@gls@style@alttreegroup	340
\@gls@noidx@sanitizesort	107	\@gls@style@alttreehypergroup	340
\@gls@noidxloclist@finalsep	112	\@gls@style@index	324
\@gls@noidxloclist@prev	112	\@gls@style@indexgroup	325
		\@gls@style@indexhypergroup	325

<code>\@glsstyle@inline</code>	324	<code>\@glsxtr@autoseeindexfalse</code>	13
<code>\@glsstyle@list</code>	314	<code>\@glsxtr@autoseeindextrue</code>	15
<code>\@glsstyle@listdotted</code>	313	<code>\@glsxtr@bookindex@atendgroup</code> ..	349–351
<code>\@glsstyle@listgroup</code>	315	<code>\@glsxtr@bookindex@atsubendgroup</code> ..	350
<code>\@glsstyle@listhypergroup</code>	315	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	350
<code>\@glsstyle@mcolalttree</code>	344	<code>\@glsxtr@bookindex@between</code>	349, 351
<code>\@glsstyle@mcolalttreegroup</code>	344	<code>\@glsxtr@bookindex@sep</code>	349, 350
<code>\@glsstyle@mcolalttreehypergroup</code> ..	345	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcolalttreespannav</code>	345	349–351
<code>\@glsstyle@mcolindexgroup</code>	341	<code>\@glsxtr@bookindex@subbetween</code> ..	349, 350
<code>\@glsstyle@mcolindexhypergroup</code>	341	<code>\@glsxtr@bookindex@subsep</code>	349, 350
<code>\@glsstyle@mcolindexspannav</code>	341	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcoltreegroup</code>	342	349–351
<code>\@glsstyle@mcoltreehypergroup</code>	342	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcoltreenamegroup</code>	343	349, 350
<code>\@glsstyle@mcoltreenamehypergroup</code>	343	<code>\@glsxtr@bookindexgroupskip</code> ...	349, 351
<code>\@glsstyle@mcoltreenamespannav</code> ..	344	<code>\@glsxtr@cat</code>	90, 102, 135, 170
<code>\@glsstyle@mcoltreespannav</code>	342	<code>\@glsxtr@checkgroup</code>	128
<code>\@glsstyle@tree</code>	326	<code>\@glsxtr@counterrecordhook</code>	10, 11
<code>\@glsstyle@treegroup</code>	326	<code>\@glsxtr@csname</code>	97, 99
<code>\@glsstyle@treehypergroup</code>	327	<code>\@glsxtr@current@style</code>	50, 346
<code>\@glsstyle@treename</code>	327	<code>\@glsxtr@currentunitcount</code>	97, 99
<code>\@glsstyle@treenamegroup</code>	328	<code>\@glsxtr@currunitcount</code>	98, 100
<code>\@glsstyle@treenamehypergroup</code> ...	328	<code>\@glsxtr@declareoption</code>	5, 15, 17, 20
<code>\@glstarget</code>	80, 111	<code>\@glsxtr@defaultnoglossarywarning</code> ..	20
<code>\@glstext@</code>	81	<code>\@glsxtr@defaultnumberformat</code>	
<code>\@glswidestname</code>	331, 338	7, 9, 57, 59, 75, 162, 165
<code>\@glsxtr</code>	47, 49	<code>\@glsxtr@defpostpunc</code>	15, 16, 23
<code>\@glsxtr@do@wrglossary</code>	106	<code>\@glsxtr@deprecated@abbrstyle</code>	
<code>\@glsxtr@abbreviationsdef</code>	17, 24	224, 225, 227,
<code>\@glsxtr@accessdisplay</code>	162–164	229, 238, 240, 241, 243, 256, 259, 263, 265
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@disabledflycommand</code>	49
.....	170, 171	<code>\@glsxtr@display@loc</code>	115
<code>\@glsxtr@addunitcounter</code>	96	<code>\@glsxtr@do@windex</code>	76
<code>\@glsxtr@addunused</code>	44	<code>\@glsxtr@do@gl:disablehyperinlist</code> ..	74
<code>\@glsxtr@addunusedxrefs</code>	44	<code>\@glsxtr@do@record@wrglossary</code>	8, 13
<code>\@glsxtr@attrval</code>		<code>\@glsxtr@do@redef@for@gl:sentries</code>	7
.....	58, 157, 158, 160, 161, 163, 165	<code>\@glsxtr@do@style</code>	21, 311
<code>\@glsxtr@autoindex@at</code>	165–167	<code>\@glsxtr@do@titlecaps@warn</code>	
<code>\@glsxtr@autoindex@doextra@esc</code>	165	157–160, 163, 170, 171
<code>\@glsxtr@autoindex@encap</code>	165–167	<code>\@glsxtr@doabbreviationsdef</code>	17
<code>\@glsxtr@autoindex@esc</code>	166, 168, 169	<code>\@glsxtr@doaccsupp</code>	20, 22
<code>\@glsxtr@autoindex@escat</code>	166, 167	<code>\@glsxtr@docdefval</code>	14, 46
<code>\@glsxtr@autoindex@escencap</code> ...	166, 167	<code>\@glsxtr@docounterrecord</code>	11
<code>\@glsxtr@autoindex@esclevel</code> ...	166, 167	<code>\@glsxtr@doglossary</code>	128
<code>\@glsxtr@autoindex@escquote</code> ...	166, 168	<code>\@glsxtr@doiflabelinlist</code>	129
<code>\@glsxtr@autoindex@level</code>	166, 167	<code>\@glsxtr@doloctag</code>	52, 53
<code>\@glsxtr@autoindex@setname</code>	165	<code>\@glsxtr@dorecord</code>	8, 9
<code>\@glsxtr@autoindexcrossrefs</code> 13, 14, 39, 42		<code>\@glsxtr@dorecordnodefer</code>	8, 9

\@glsxtr@dosee@alsoindex@glossary ..	13	\@glsxtr@nopostpunc@postdesc	110
\@glsxtr@doseeglossary	13, 23	\@glsxtr@notfoundinlist	175
\@glsxtr@dostylewarn	195	\@glsxtr@op@recordcounter	13
\@glsxtr@enabletagging	169	\@glsxtr@optlist	49
\@glsxtr@end@	47	\@glsxtr@org@starttoc	292
\@glsxtr@endescspch	166–169	\@glsxtr@org@GLS@	55
\@glsxtr@entrycount@org@localreset ..	91	\@glsxtr@org@GLSpl@	56
\@glsxtr@entrycount@org@localunset ..	91	\@glsxtr@org@Gls@	55
\@glsxtr@entrycount@org@reset	91	\@glsxtr@org@Glspl@	55
\@glsxtr@entrycount@org@unset	91	\@glsxtr@org@Glsxtrtitlefirst ..	293, 294
\@glsxtr@entryunitcount@org@localreset	99	\@glsxtr@org@Glsxtrtitlefirstplural	293, 294
\@glsxtr@entryunitcount@org@localunset	99	\@glsxtr@org@Glsxtrtitlefull ..	293, 294
\@glsxtr@entryunitcount@org@reset ..	99	\@glsxtr@org@Glsxtrtitlefullpl ..	293, 294
\@glsxtr@entryunitcount@org@unset ..	99	\@glsxtr@org@Glsxtrtitlelong ..	293, 294
\@glsxtr@err@undefaction	7, 13	\@glsxtr@org@Glsxtrtitlelongpl ..	293, 294
\@glsxtr@field@linkdefs	55	\@glsxtr@org@Glsxtrtitlename ..	293, 294
\@glsxtr@format@overridefalse	164	\@glsxtr@org@Glsxtrtitleplural ..	293, 294
\@glsxtr@format@overridetrue ..	164, 165	\@glsxtr@org@Glsxtrtitleshort ..	293, 294
\@glsxtr@foundinlist	175	\@glsxtr@org@Glsxtrtitleshortpl ..	293, 294
\@glsxtr@full	182	\@glsxtr@org@Glsxtrtitletext ..	293, 294
\@glsxtr@fullpl	184	\@glsxtr@org@MakeUppercase	293, 294
\@glsxtr@gettype	108	\@glsxtr@org@checkfirsthyper ...	74, 104
\@glsxtr@glossdescfont	157, 158	\@glsxtr@org@delimN	52, 53
\@glsxtr@glossnamefont	158–164	\@glsxtr@org@delimR	52, 53
\@glsxtr@gobbleto@endescspch	168	\@glsxtr@org@doseeglossary	23, 106
\@glsxtr@groupheading	128, 131	\@glsxtr@org@gloautosee	24
\@glsxtr@idx@displaynumberlist	107	\@glsxtr@org@gls@	55
\@glsxtr@idx@entrynumberlist	107	\@glsxtr@org@glsdohypertarget	111
\@glsxtr@ifcsstart	47	\@glsxtr@org@glsignore	52, 53
\@glsxtr@ifpunctoken	175	\@glsxtr@org@glspl@	55
\@glsxtr@ifunitcounter	96	\@glsxtr@org@glsxtrtitlefirst ..	293, 294
\@glsxtr@insert@dots	177	\@glsxtr@org@glsxtrtitlefirstplural	293, 294
\@glsxtr@insert@dots@next	177	\@glsxtr@org@glsxtrtitlefull ..	293, 294
\@glsxtr@insertdots	179	\@glsxtr@org@glsxtrtitlefullpl ..	293, 294
\@glsxtr@label	30, 44, 156	\@glsxtr@org@glsxtrtitlelong ..	293, 294
\@glsxtr@loadstyles	312, 313	\@glsxtr@org@glsxtrtitlelongpl ..	293, 294
\@glsxtr@longnewglossaryentry	33	\@glsxtr@org@glsxtrtitlename ..	293, 294
\@glsxtr@mark@wordseps	177	\@glsxtr@org@glsxtrtitleorpdforheading	293, 294
\@glsxtr@mark@wordseps@next	178	\@glsxtr@org@glsxtrtitleplural ..	293, 294
\@glsxtr@markwordseps	178–180	\@glsxtr@org@glsxtrtitleshort ..	293, 294
\@glsxtr@mixed@assign@sortkey	108	\@glsxtr@org@glsxtrtitleshortpl ..	293, 294
\@glsxtr@noidx@displaynumberlist ..	107	\@glsxtr@org@glsxtrtitletext ..	293, 294
\@glsxtr@noidx@do	130	\@glsxtr@org@makeglossaries	105
\@glsxtr@noidx@entrynumberlist	107	\@glsxtr@org@markboth	291, 292
\@glsxtr@noidx@numberlistloop	107	\@glsxtr@org@markright	291, 292
\@glsxtr@noop@recordcounter	10, 13	\@glsxtr@org@newacronymstyle ..	103, 104
\@glsxtr@nopostpunc	109		

<code>\@glxtr@org@postdescription</code> ..	110, 171	<code>\@glxtr@thevalue</code>	8, 9, 57–59, 136
<code>\@glxtr@org@see@noindex</code>	122	<code>\@glxtr@thisloctag</code>	52, 53
<code>\@glxtr@org@setacronymstyle</code> ..	103, 104	<code>\@glxtr@titlelabel</code>	113, 114, 130
<code>\@glxtr@org@theHvalue</code>	8, 9	<code>\@glxtr@tmp</code>	21, 116
<code>\@glxtr@orgprefix</code>	10	<code>\@glxtr@type</code>	156
<code>\@glxtr@orgprintglossary</code>	49, 110	<code>\@glxtr@unitcountlist</code>	96
<code>\@glxtr@orgwarndep</code>	176	<code>\@glxtr@unsrt@getgrouptitle</code>	128
<code>\@glxtr@p@acrlong@</code>	81	<code>\@glxtr@usesee</code>	39
<code>\@glxtr@p@acrlongpl@</code>	81	<code>\@glxtr@warn@onexistsordo</code>	7, 13
<code>\@glxtr@p@acrshort@</code>	81	<code>\@glxtr@warn@undefaction</code>	7, 13
<code>\@glxtr@p@acrshortpl@</code>	81	<code>\@glxtr@docdeffalse</code>	45
<code>\@glxtr@p@long@</code>	80	<code>\@glxtr@entryfmt</code>	28
<code>\@glxtr@p@longpl@</code>	80	<code>\@glxtr@fmt</code>	27
<code>\@glxtr@p@plural@</code>	80	<code>\@glxtr@glossentry</code>	125
<code>\@glxtr@p@short@</code>	80	<code>\@glxtr@glossentryother</code>	126
<code>\@glxtr@p@shortpl@</code>	80	<code>\@glxtr@hypernameprefix</code>	111, 130
<code>\@glxtr@p@text@</code>	80	<code>\@glxtr@trifhasfield</code>	29, 30
<code>\@glxtr@pagestag</code>	52	<code>\@glxtr@trifhyphenstart</code>	274
<code>\@glxtr@pagetag</code>	52	<code>\@glxtr@indexaliased</code>	75
<code>\@glxtr@prevunitcount</code>	98	<code>\@glxtr@indexcrossrefsfalse</code>	14
<code>\@glxtr@printglossopts</code>	49, 108, 110	<code>\@glxtr@indexcrossrefstrue</code>	15
<code>\@glxtr@printunsrtglossaryskipentry</code>	128, 129	<code>\@glxtr@inmark</code>	291, 292
<code>\@glxtr@provide@addstoragekey</code>	27	<code>\@glxtr@long</code>	80, 187
<code>\@glxtr@provide@storagekey</code>	26	<code>\@glxtr@longpl</code>	80, 190, 191
<code>\@glxtr@record</code>	13, 54–56, 59	<code>\@glxtr@newgls</code>	133, 134
<code>\@glxtr@record@setting</code>	8, 9, 12, 41, 45, 46, 105	<code>\@glxtr@newgls@inner</code>	132, 133
<code>\@glxtr@record@setting@alsoindex</code>	8, 9, 41, 105	<code>\@glxtr@newgls@innercsname</code>	133
<code>\@glxtr@record@setting@off</code>	45	<code>\@glxtr@notinmark</code>	291, 292
<code>\@glxtr@record@setting@only</code>	105	<code>\@glxtr@p</code>	86
<code>\@glxtr@recordsee</code>	13, 23, 41	<code>\@glxtr@p@opt</code>	84
<code>\@glxtr@redef@forglsentries</code>	7, 24	<code>\@glxtr@pl</code>	48, 49
<code>\@glxtr@redefstyles</code>	21, 311	<code>\@glxtr@postloctag</code>	51, 52, 54
<code>\@glxtr@reg@glosslist</code>	105–108, 111	<code>\@glxtr@preloctag</code>	51, 52, 54
<code>\@glxtr@restore@postpunc</code>	110	<code>\@glxtr@setaliasnoindex</code>	75, 76
<code>\@glxtr@rglstrigger@record</code> ...	137–139	<code>\@glxtr@short</code>	80, 186
<code>\@glxtr@s@longnewglossaryentry</code>	33	<code>\@glxtr@shortpl</code>	80, 189
<code>\@glxtr@savepreloctag</code>	52, 53	<code>\@glxtr@undeftag</code>	6, 25
<code>\@glxtr@setentrycountunsetattr</code>	90	<code>\@glxtr@wrglossmark</code>	22
<code>\@glxtr@setentryunitcountunsetattr</code>	101	<code>\@gobble</code> ..	7, 13, 15, 60, 129, 130, 177, 349–351
<code>\@glxtr@setupshortcuts</code>	19, 20, 24	<code>\@gobbletwo</code>	176
<code>\@glxtr@shortcutsval</code>	19, 124	<code>\@ifnextchar</code>	77
<code>\@glxtr@swaptwo</code>	176	<code>\@ifpackageloaded</code>	5, 16, 124, 140, 156, 158, 161, 162, 164, 310
<code>\@glxtr@tag</code>	170	<code>\@ifstar</code> ..	26, 27, 30, 33, 34, 36, 46, 77, 127, 169
<code>\@glxtr@taggingcs</code>	170	<code>\@ifundefined</code>	310
<code>\@glxtr@textformat</code>	58, 59	<code>\@ignored@glossaries</code>	34–36
<code>\@glxtr@theHvalue</code>	8, 9, 57–59, 136	<code>\@input</code>	122
		<code>\@input@</code>	118
		<code>\@istfilename</code>	105

\Aclp	18	\AtEndDocument	43, 91, 100, 118
\aclp	18		
\ACP	18	B	
\Acp	18	babel package	165, 167, 174
\acp	18	\begin	15, 16, 115, 119, 120, 128, 314, 316–323, 342–345, 349
\ACRfullfmt	103	\begingroup	8, 9, 27, 28, 75, 126, 127
\Acrfullfmt	103	\bgroup	33, 109
\acrfullfmt	103	bib2gls	3, 28, 130, 132, 136, 137
\ACRfullplfmt	103		
\Acrfullplfmt	103	C	
\acrfullplfmt	103	\catcode	122
\acronymentry	102	category attributes:	
\acronymfont	68–71, 83, 103, 104	apospplural	179
\acronymname	16	discardperiod	173
\acronymsort	102	entrycount	89, 90, 92, 101
\acronymtype	17, 102, 103	firstuc	160
\acrpluralsuffix	102, 124	glossdesc	156
\ACS	18	glossdescfont	157
\Acs	18	glossname	158
\acs	18	glossnamefont	158, 161
\ACSP	18	headuc	294
\Acsp	18	indexname	165
\acsp	18	indexonlyfirst	76
\actualchar	168	insertdots	179
\addtolength	339	markshortwords	179
\advance	92, 101, 123, 133	markwords	178, 179, 274, 275, 283
\AF	17	nohyper	74
\Af	17	nohyperfirst	61–63
\af	17	noshortplural	179
\AFP	17	regular	54, 95, 198–203, 205, 208, 210, 211, 213, 214, 217, 218, 220, 221, 223, 225–227, 231–235, 238–240, 242, 245–251, 253, 255, 257, 259–261, 263, 266, 272, 274, 276, 277, 279, 284, 289, 290
\Afp	17	textformat	58
\afp	17	\cdot	22
\AL	17	\centering	348
\Al	17	\cGLS	17, 18, 90, 101
\alp	17	\cGls	17, 18, 90, 101
\AnyTrackedLanguages	310	\cglS	17, 18, 90, 101
\appto	11, 12, 21, 26, 39–43, 72, 76, 90, 98, 128, 130, 164, 174, 177, 178, 312	\cGLSformat	94
\arabic	351	\cGlsformat	93
\AS	17	\cglSformat	93, 95
\As	17	\cGLSpl	17, 18, 90, 101
\as	17	\cGlspl	17, 18, 90, 101
\ASP	17	\cglSpl	17, 18, 90, 101
\Asp	17	\cGLSplformat	94
\asp	17	\cGlsplformat	93
\AtBeginDocument	22, 25, 50, 51, 124	\cglSplformat	93, 95

\changes	16, 297	\define@key	
\char	113	.	11, 12, 16, 21, 26, 41, 57, 59, 72, 111, 176
\columnwidth	50, 51	\DefineAcronymSynonyms	19
\count@	92, 100, 101	\delimN	52, 53
\cs	16	\delimR	52, 53
\csappto	32	\detokenize	47
\csdef	26, 31, 32, 72, 73, 91, 96, 97, 99, 151, 195–197, 205, 227, 242, 263, 267, 269, 271, 281, 282, 286, 288, 313, 330	\dimen@	104, 330–338
\cseappto	37	\dimen@i	332–334
\csedef	31, 97, 114	\dimen@ii	330–334
\csgdef	31, 34–36, 45, 52, 91, 97, 99, 100, 329, 330, 352	\dimexpr	50, 51, 329
\cslet	31, 33, 34, 109	\disable@keys	17, 25, 45, 122
\csletcs	31, 197	\do	6, 21, 30, 44, 90, 102, 105, 108, 115, 128, 135, 156, 162, 170
\csname	6, 26, 35, 41, 45, 50, 53, 56, 57, 59, 68– 73, 75, 84, 97, 106, 114, 118, 121, 122, 128, 133–136, 156, 176, 183–192, 198, 338	\do@glsl@link@checkfirsthyper	
\cspreto	32		27, 55, 56, 58, 68–71, 183–192
\csuse	28, 35, 43, 52, 72, 73, 85–87, 96–100, 108, 113, 115, 116, 125, 126, 129–131, 151, 162, 172, 173, 196, 197, 330, 331, 333, 334	\do@glsl:disablehyperinlist	58, 75
\csxdef	39, 42, 97, 100, 114	doc package	168
\currentglossary	109, 126, 127, 351	\dolistcsloop	29
\CurrentOption	22, 312, 313	\DTLifinlist	106, 107, 111
\CurrentTrackedLanguageTag	124	\DTLifint	113
\CurrentTrackedTag	311		
\CustomAbbreviationFields	180, 198, 200–203, 205, 207, 209, 212, 214– 220, 222, 226, 227, 230–234, 237, 240, 241, 244, 245, 247–252, 254, 256, 259, 261, 263, 266, 267, 269, 271–273, 275– 277, 279, 280, 282–284, 286, 287, 289, 290	E	
D		\eappto	11, 21, 34–36, 128, 131, 165, 312
\DeclareAcronymList	102	\edef	6, 8–10, 34–37, 40, 42, 43, 45, 57–59, 74, 75, 78, 79, 96, 97, 99, 105, 106, 111, 113, 116– 118, 122, 126, 127, 136, 157, 158, 160– 163, 166, 168, 169, 176, 333, 334, 349, 350
\DeclareOption	5, 312	\eglssetwidest	331–338
\DeclareOptionX	5, 22	\egroup	33, 34, 109
\def	9–13, 23, 25, 27, 33, 35, 39, 41, 42, 44, 47–49, 51, 53, 55– 71, 77, 79–83, 93–95, 102, 106, 108–112, 114–117, 128, 130, 131, 133, 136–139, 166–171, 175–179, 183–192, 195, 274, 275, 277, 280, 283, 285, 339, 340, 344, 345	\else	8–11, 14, 15, 17, 19, 20, 23, 28, 32, 45, 47, 51, 53, 56, 59, 75, 76, 92, 104, 105, 107, 109–111, 113, 116, 117, 119–121, 123, 136, 164–166, 168, 169, 175, 177–179, 186–192, 194, 195, 199, 201, 202, 204–213, 216, 218–246, 248–264, 266–268, 270, 271, 273–275, 277, 280, 282, 283, 285, 287, 289, 290, 314, 316–326, 328, 339, 340, 346, 348, 350
\defglstryfmt	34–36	\emph	243, 244
\define@boolkey	14, 15, 57, 74	\empty	115–117
\define@choicekey		\encapchar	168
	7, 12, 14, 15, 19, 20, 22, 57, 110	\end	. 115, 120, 128, 314, 316–323, 342–345, 349
		\end@glstr@display@loc	115, 116
		\endscname	6, 26, 35, 41, 45, 50, 53, 56, 57, 59, 68– 73, 75, 84, 97, 106, 114, 118, 121, 122, 128, 133–136, 156, 176, 183–192, 198, 338
		\endgroup	8, 10, 28, 75, 126, 127

<code>\ensuremath</code>	22	<code>\global</code>	10, 33, 34, 109, 131
entry categories:		<code>\glolinkprefix</code>	59, 79, 124
abbreviation	192	glossaries package	13, 23, 24, 39–41, 43, 108, 313
general	151, 153	glossaries-accsupp package	20, 22, 140
index	154	glossaries-extra package	2
<code>\epreto</code>	165	glossaries-extra-stylemods package .	20, 172, 311
<code>\equal</code>	121	glossaries-stylemods package	347
etoolbox package	5	glossaries.sty package	34
<code>\expandafter</code>	22,	<code>\GlossariesExtraWarning</code>	6,
26, 27, 30, 39, 40, 44, 47–49, 72, 73,		15, 32, 33, 47, 49, 58, 104, 106, 116, 120,	
77, 84, 95, 96, 106–108, 111, 114, 116,		122, 128, 157, 158, 160, 161, 163, 170, 197	
117, 128, 131, 133, 140, 156, 159, 160,		<code>\GlossariesExtraWarningNoLine</code>	15, 92, 101
162, 164, 165, 168, 175, 178–180, 274, 349		<code>\GlossariesWarning</code>	52, 107, 109, 112, 113, 195
<code>\expandonce</code>	102, 131, 166, 200, 277	<code>\GlossariesWarningNoLine</code>	106, 118
F			
<code>\fi</code>	7–11, 13–15, 17, 19,	glossary styles:	
20, 22–24, 28, 32, 39, 41–43, 45–47, 50–		altlist	314
52, 54, 56–59, 75, 76, 92, 100, 101, 104,		altlistgroup	315
107–111, 113, 116–124, 136, 164–167,		altlisthypergroup	316
169, 175, 177, 178, 180, 186–192, 194,		alttree	328, 329, 339
195, 199, 201, 202, 204–213, 216, 218–		alttreegroup	340
246, 248–264, 266–268, 270, 271, 273–		alttreehypergroup	340
275, 277, 280, 282, 283, 285, 287, 289,		index	324
290, 314, 316–328, 330–340, 346, 348, 351		indexgroup	325
first use	353	indexhypergroup	325
flag	353	inline	324
text	353	list	314
<code>\firstacronymfont</code>	103, 104	listdotted	313
fontspec package	124	listdottedstyle	314
<code>\footnote</code>	203	listgroup	315
<code>\forall glossaries</code>		listhypergroup	315
.....	44, 127, 154, 156, 331, 332, 334–338	mcolalttree	344
<code>\forall gloss entries</code>	92, 101	mcolalttreegroup	344
<code>\ForEachTrackedDialect</code>	310	mcolalttreehypergroup	345
<code>\forall gloss entries</code>	6, 44, 154, 156, 331–338	mcolalttreespannav	345
<code>\forall listcsloop</code>	29, 101, 115	mcolindexgroup	341
<code>\forall listloop</code>	112, 171	mcolindexhypergroup	341
<code>\futurelet</code>	175	mcolindexspannav	341
G			
<code>\gdef</code>	52, 53, 167, 168	mcoltreegroup	342
<code>\Genacrfullformat</code>	103	mcoltreehypergroup	342
<code>\genacrfullformat</code>	103	mcoltreenonamegroup	343
<code>\GenericAcronymFields</code>	103	mcoltreenonamehypergroup	343
<code>\Genplacrfullformat</code>	103	mcoltreenonamespannav	344
<code>\genplacrfullformat</code>	103	mcoltreespannav	342
<code>\glo@grabfirst</code>	131	sublistdotted	314
<code>\glo@name</code>	159, 160, 164	tree	326
<code>\gloaliaslabel</code>	79	treegroup	326
		treehypergroup	327
		treenoname	327
		treenonamegroup	328

treenonamehypergroup	328	214, 216, 217, 219, 220, 223, 224, 226,
glossary-bookindex package	313	228, 230, 232, 233, 235, 237, 238, 240,
glossary-hypernav package	78	242, 244, 246, 248, 249, 251, 252, 254,
glossary-long package	318	256, 258, 260, 262, 264, 266, 267, 270,
glossary-longbooktabs package	318	272–274, 276, 278, 280, 281, 284, 286, 289
\glossaryentrynumbers ...	53, 109, 131, 132	\glsabbrvhyphenfont
\glossaryheader	115, 275, 276, 280–284, 286, 287
128, 314–323, 325–328, 339–343, 345, 349		\glsabbrvonlyfont
\glossaryname	108	288–290
\glossarypostamble	115, 128, 130	\glsabbrvscfont .
\glossarypreamble	115, 127	215–220, 223, 224, 226–228
\glossarysection	115, 121, 127, 130	\glsabbrvsmfont
\glossarytitle ...	35, 108, 109, 115, 121, 127 229–233, 235, 237, 238, 240, 242
\glossarytoctitle ...	35, 108, 115, 121, 127	\glsabbrvuserfont
\glossentry	109,	265–273
132, 313, 314, 316–323, 325–327, 339, 349		\GLSaccessdesc
\glossentrydesc	313–323, 325–329	64
\glossentryname		\GLSaccessdesc
.... 126, 313–323, 325–327, 339, 340, 347		64, 157, 169
\glossentrynameother	127	\glsaccessdesc
\glossentrysymbol 317–321, 323, 325–327, 329		64, 157, 173
\glossxtrsetpopts	172	\GLSaccessdescplural
\GLS	90, 101, 135	64
\Gls	48, 90, 101, 135	\Glsaccessdescplural
\gls	32, 48, 49, 90, 101, 107, 119, 135	64
\gls@assign@desc	33, 34	\GLSaccessfirst
\gls@assign@field	11, 12, 26, 72	62
\gls@checkseeallowed	46, 106	\Glsaccessfirst
\gls@codepage	118	61
\gls@defdocnewglossaryentry	90, 98	\glsaccessfirst
\gls@defglossaryentry	33, 34, 47–49	61
\gls@dotocitle	109	\GLSaccessfirstplural
\gls@glossary	41	63
\gls@grplabel	78	\Glsaccessfirstplural
\gls@level	131	63
\gls@noidxglossary	106	\glsaccessfirstplural
\gls@org@glossaryentryfield	109	62
\gls@org@glossarysubentryfield	109	\Glsaccesslong
\gls@save@numberlist	51, 53, 54	70, 181, 188,
\gls@set@xr@key	41	199, 208, 212, 213, 216, 222–225, 230,
\gls@tmplen	331–340	236–239, 244, 246, 254–258, 260, 267,
\gls@type	106	268, 276, 278, 279, 281, 282, 284, 289, 290
\glsabbrvdefaultfont ...	182, 199, 201,	\glsaccesslong
204, 206, 207, 209, 212, 265, 275, 278, 288	 70, 181, 187, 188, 199, 201, 202,
\glsabbrvmfont		204, 206, 207, 209, 211–213, 216, 218,
..... 243–252, 254, 256, 258, 260–264		219, 221–227, 229, 230, 232–241, 243,
\glsabbrvfont		244, 246, 248, 250–264, 266, 268, 270,
..... 81, 82, 103, 182, 186, 187, 189,		271, 273, 276, 278, 279, 281, 282, 284, 289
190, 192, 194, 195, 198–207, 209, 212,		\Glsaccesslongpl
		71, 181,
		191, 199, 208, 212, 213, 216, 222, 223,
		225, 230, 236–239, 245, 246, 254–260,
		267, 268, 276, 278, 279, 282, 284, 289, 290
		\glsaccesslongpl
	 71, 181, 191, 192, 199, 201,
		202, 204–207, 209–213, 216, 218–227,
		229, 230, 232, 234–241, 243, 244, 246,
		248, 250, 251, 253–266, 268, 270, 271,
		273, 276, 278, 279, 281, 282, 284, 289, 290
		\GLSaccessname
		63
		\Glsaccessname
		63
		\glsaccessname
		40, 63
		\GLSaccessplural
		62
		\Glsaccessplural
		62

<code>\glsaccessplural</code>	62	<code>\glsdefaulttype</code> .	6, 16, 32, 108, 119, 127, 129
<code>\Glsaccessshort</code>	68, 186, 195, 201, 204, 206, 209, 211, 218, 219, 221, 226– 229, 232, 234, 235, 241–243, 248, 250, 251, 253, 262–264, 270, 273, 281, 286, 287	<code>\glsdescriptionaccessdisplay</code> ..	144, 157
<code>\glsaccessshort</code> ..	68, 181, 186, 187, 194, 199, 201, 204, 206–211, 213, 216, 218– 223, 225–230, 232–244, 246, 248, 249, 251–255, 257–260, 262, 264, 266–268, 270, 273, 276, 278, 281, 284, 286, 287, 290	<code>\glsdescriptionpluralaccessdisplay</code>	145
<code>\Glsaccessshortpl</code>	69, 190, 195, 202, 204–206, 210, 211, 218, 220, 221, 227–229, 232, 234, 235, 241, 243, 248, 250, 251, 253, 262–264, 270, 271, 273, 281, 286, 287, 290	<code>\glsdescwidth</code>	316–323
<code>\glsaccessshortpl</code>	69, 181, 189, 190, 194, 199, 201, 204, 206–211, 213, 216, 218– 223, 225–232, 234–239, 241–246, 248, 250–255, 257–260, 262, 264, 267, 268, 270, 273, 276, 278, 281, 284, 286, 287, 290	<code>\glsdetoklabel</code>	8, 9, 28–34, 37–40, 44–46, 57, 59, 75, 79, 91, 96–101, 106, 109, 112, 113, 126, 127, 131, 134–136, 156, 159, 160, 164, 333, 334
<code>\GLSaccesssymbol</code>	65	<code>\glsdisplaynumberlist</code>	107, 111
<code>\Glsaccesssymbol</code>	65, 169	<code>\glsdohyperlink</code>	77, 78, 80
<code>\glsaccesssymbol</code>	65, 169, 173	<code>\glsdohypertarget</code>	80, 111
<code>\GLSaccesssymbolplural</code>	65	<code>\glsdoifexists</code>	14, 23, 31, 37, 39–41, 54, 56, 59, 68–71, 79, 106, 112, 113, 126, 127, 183–192
<code>\Glsaccesssymbolplural</code>	65	<code>\glsdoifexistsordo</code>	27, 28, 56
<code>\glsaccesssymbolplural</code>	65	<code>\glsdoifexistsorwarn</code>	14, 156, 157, 169
<code>\GLSaccessstext</code>	61	<code>\glsdoifnoexists</code>	33
<code>\Glsaccessstext</code>	61	<code>\glsdonohyperlink</code>	59, 79, 80
<code>\glsaccessstext</code>	40, 60	<code>\glsdosanitizesort</code>	107
<code>\glsacrshortcutstrue</code>	19, 20	<code>\glsenableentrycount</code>	90, 92, 100
<code>\glsacspacemax</code>	104	<code>\glsenableentryunitcount</code>	91, 101
<code>\glsadd</code>	28, 44, 119	<code>\glsentrycounter</code>	117
<code>\glsadd options</code>		<code>\glsentrycurrcount</code>	91, 92, 98
<code>theHvalue</code>	9	<code>\Glsentrydesc</code>	144, 149, 158
<code>thevalue</code>	9	<code>\glsentrydesc</code>	144, 149, 158
<code>\glsaddstoragekey</code>	43, 151	<code>\Glsentrydescplural</code>	145, 149
<code>\glsbackslash</code>	47	<code>\glsentrydescplural</code>	145, 149
<code>\glscapscase</code>	56, 60–71, 73, 183–194	<code>\Glsentryfirst</code>	95, 139, 142, 148
<code>\glscategory</code>	54, 60, 74, 81, 82, 152, 153, 156–158, 160–163, 169, 172, 173, 183–187, 189, 190	<code>\glsentryfirst</code>	95, 139, 142, 148, 307
<code>\glscategorylabel</code>	74, 176, 178, 179, 205, 227, 242, 263, 267, 269, 271, 281, 282, 286, 288	<code>\Glsentryfirstplural</code>	96, 139, 143, 148
<code>\glsclsebrace</code>	41, 120, 121	<code>\glsentryfirstplural</code>	95, 139, 142, 143, 148, 307
<code>\glscurrententrylabel</code>	51–53, 109, 117, 126–129, 171, 172	<code>\glsentryfmt</code>	34–36
<code>\glscurrentfieldvalue</code> ..	27, 28, 30, 32, 265	<code>\Glsentryfull</code>	103
<code>\glscustomtext</code> ..	55, 56, 68–71, 183–192, 194	<code>\glsentryfull</code>	103
		<code>\Glsentryfullpl</code>	103
		<code>\glsentryfullpl</code>	103
		<code>\glsentryitem</code>	126, 127, 313, 314, 316–323, 325–327, 339, 350
		<code>\Glsentrylong</code>	82, 83, 95, 139, 147, 150
		<code>\glsentrylong</code> ..	82, 83, 95, 139, 146, 147, 150, 205, 228, 242, 263, 269, 271, 285, 308
		<code>\Glsentrylongpl</code>	82, 83, 96, 147, 150
		<code>\glsentrylongpl</code>	82, 83, 95, 147, 150, 308, 309
		<code>\Glsentrylongplural</code>	139
		<code>\glsentrylongplural</code>	139
		<code>\Glsentryname</code>	140, 147, 159–162
		<code>\glsentryname</code>	125, 126, 140, 147, 165, 305, 331–338, 352

<code>\glsentrynumberlist</code>	107, 113, 336–338	<code>\glsfirstabbrvscfont</code>	215–229
<code>\Glsentryplural</code>	141, 148	<code>\glsfirstabbrvsmfont</code>	230–243
<code>\glsentryplural</code>	141, 142, 148, 306	<code>\glsfirstabbrvuserfont</code>	266–273
<code>\glsentryprevcount</code>	91, 92, 98	<code>\glsfirstaccessdisplay</code>	142
<code>\glsentryprevmaxcount</code>	99	<code>\glsfirstlongdefaultfont</code>	
<code>\glsentryprevtotalcount</code>	98		199, 201, 207, 209, 212, 215–225, 230–
<code>\Glsentryshort</code>	81, 83, 145, 150		240, 244, 245, 247, 248, 251–255, 258, 259
<code>\glsentryshort</code>	81–83,	<code>\glsfirstlongemfont</code>	
	104, 145, 146, 149, 150, 267, 269, 280, 304	 245–247, 249, 250, 256, 257, 259–261
<code>\Glsentryshortpl</code>	82, 83, 146, 150	<code>\glsfirstlongfont</code>	181, 199–202,
<code>\glsentryshortpl</code> ..	82, 83, 146, 150, 304, 305		204, 206–214, 216, 218, 219, 221, 223,
<code>\Glsentrysymbol</code>	143, 149		224, 226, 228, 230, 232, 233, 235, 237,
<code>\glsentrysymbol</code>	143, 148, 149, 335–337		238, 240, 242, 244, 246, 248, 249, 251,
<code>\Glsentrysymbolplural</code>	144, 149		252, 254, 256, 258, 260, 262, 264, 266,
<code>\glsentrysymbolplural</code>	144, 149		267, 270, 273, 276, 278, 281, 284, 286, 289
<code>\Glsentrytext</code>	141, 148	<code>\glsfirstlongfootnotefont</code>	
<code>\glsentrytext</code>	79, 141, 148, 305, 306	 203–206, 226–229, 240–243, 261–265
<code>\glsentrytype</code>	126, 127	<code>\glsfirstlonghyphenfont</code>	275–286
<code>\Glsentryuseri</code>	66	<code>\glsfirstlongonlyfont</code>	289, 290
<code>\glsentryuseri</code>	66	<code>\glsfirstlonguserfont</code>	266–274
<code>\Glsentryuserii</code>	66	<code>\GLSfirstplural</code>	299, 300
<code>\glsentryuserii</code>	66	<code>\Glsfirstplural</code>	300
<code>\Glsentryuseriii</code>	66	<code>\glsfirstplural</code>	300
<code>\glsentryuseriii</code>	66	<code>\glsfirstpluralaccessdisplay</code> ..	142, 143
<code>\Glsentryuseriv</code>	67	<code>\glsforeachincategory</code>	195
<code>\glsentryuseriv</code>	67	<code>\glsgenentryfmt</code>	54
<code>\Glsentryuserv</code>	67	<code>\glsgetattribute</code>	58, 78, 92,
<code>\glsentryuserv</code>	67		96–98, 117, 136, 157, 158, 160, 161, 163, 165
<code>\Glsentryuservi</code>	67	<code>\glsgetcategoryattribute</code>	152
<code>\glsentryuservi</code>	67	<code>\glsgetgrouptitle</code> 315, 316, 325–328, 340–346	
<code>\glsextrapostnamehook</code>	162	<code>\glsgetwidestname</code>	329
<code>\glsfieldfetch</code>	79	<code>\glsgroupheading</code>	
<code>\glsfieldxdef</code>	156	 131, 314–323, 325–328, 339–345, 351
<code>\glsfindwidesttoplevelname</code>	331	<code>\glsgroupskip</code> 131, 314, 316–326, 328, 340, 351	
<code>\GLSfirst</code>	299	<code>\glshasattribute</code>	
<code>\Glsfirst</code>	299	 58, 78, 92, 97, 99, 101, 117, 136,
<code>\glsfirst</code>	299		157, 158, 160, 161, 163, 165, 199–203,
<code>\glsfirstabbrvdefaultfont</code>			205, 208, 210, 211, 214, 215, 217, 218,
	182, 199, 201, 204, 206, 207, 209, 212, 278		226, 228, 230–233, 240, 242, 244–247,
<code>\glsfirstabbrvmfont</code>	244–264		249, 250, 257, 261–263, 266, 267, 269–
<code>\glsfirstabbrvfont</code>	103,		274, 276, 277, 279, 281, 283–286, 288–290
	181, 199–213, 216, 217, 219, 221, 223,	<code>\glshascategoryattribute</code>	152
	224, 226, 228, 230, 232, 233, 235, 237,	<code>\glshyperlink</code>	79
	238, 240, 242, 244, 246, 248, 249, 251,	<code>\glshypernavsep</code>	115
	252, 254, 256, 258, 260, 262, 264, 266,	<code>\glshypernumber</code>	117, 164
	267, 270, 273, 276, 278, 281, 284, 286, 289	<code>\glsifattribute</code> ...	56, 57, 61, 74, 76, 85,
<code>\glsfirstabbrvhyphenfont</code>			154, 157–160, 163, 164, 171, 174, 294–303
 275, 276, 280, 281, 283–287	<code>\glsifcategory</code>	154
<code>\glsfirstabbrvonlyfont</code>	289, 290		

<code>\glsifcategoryattribute</code>	<code>\glslongemfont</code> ..
..... 74, 152, 153, 178, 179	243, 246, 249, 256, 259, 260
<code>\glsifnotregular</code>	<code>\glslongfont</code>
60	82, 83, 182, 187, 188,
<code>\glsifnotregularcategory</code>	191, 192, 199–202, 204, 206, 207, 209,
153	212, 214, 216, 218, 219, 221, 223, 224,
<code>\glsifplural</code> 56, 60, 62–65, 68–71, 174, 183–193	226, 228, 230, 232, 233, 235, 237, 238,
<code>\glsifregular</code>	240, 242, 244, 246, 248, 249, 251, 252,
54, 60, 95, 96, 139	254, 256, 258, 260, 262, 264, 266, 267,
<code>\glsifregularcategory</code>	270, 273, 276, 278, 281, 284, 286, 289, 290
153	<code>\glslongfootnotefont</code>
<code>\glsifusetranslator</code>	203, 204, 206, 226, 228, 240, 242, 262, 264
35	<code>\glslonghyphenfont</code> .
<code>\glsignore</code>	275–281, 283, 284, 286
52, 53	<code>\glslongonlyfont</code>
<code>\glsinlinedescformat</code>	288, 289
324	<code>\glslongpltok</code>
<code>\glsinlinesubdescformat</code> 180, 199–203, 212, 214–218, 222,
324	226, 230, 231, 233, 237, 240, 244–250,
<code>\glsinsert</code>	254, 256, 259, 261, 266, 267, 269, 271,
56,	272, 274–277, 279, 280, 282–284, 289, 290
60, 68–71, 183–194, 274, 281, 282, 286, 288	<code>\glslongpluralaccessdisplay</code>
<code>\glskeylisttok</code>	147
102, 103, 178, 180	<code>\glslongtok</code>
<code>\glslabel</code> 102, 178, 180, 199–203, 205, 207, 209,
8,	212, 214–220, 222, 226, 227, 230, 231,
9, 27, 37, 40, 54, 56–59, 74, 75, 78, 79,	233, 235, 237, 240, 242, 244–252, 254,
104, 136, 172, 173, 192–194, 205, 228,	256, 259, 261, 263, 266–269, 271–273,
242, 263, 267, 269, 271, 281, 282, 286, 288	275–277, 279, 280, 282–284, 286, 289, 290
<code>\glslabeltok</code>	<code>\glslonguserfont</code>
102, 178,	266–273
180, 199–203, 205, 207–212, 214–220,	<code>\glsmcols</code>
222, 226, 228, 230–233, 235, 237, 240,	342–345
242, 244–252, 254, 256, 257, 259, 261–	<code>\GLSname</code>
263, 266–277, 279, 281, 283–286, 288–290	296, 297
<code>\glsletentryfield</code>	<code>\Glsname</code>
166	297
<code>\glslink</code>	<code>\glsname</code>
103	296
<code>\glslink options</code>	<code>\glsnameaccessdisplay</code>
counter	140, 159, 161
9	<code>\glsnamefont</code>
format	158–163
164	<code>\glsnavhyperlink</code>
hyper	115
291	<code>\glsnavhyperlinkname</code>
<code>\glslinkhyperoutside</code>	78
57	<code>\glsnavhypertarget</code>
<code>\glslinknoindex</code> 315, 316, 326–328, 341–346
8, 74, 291	<code>\glsnavigation</code> ..
<code>\glslinktheHvalue</code>	315, 316, 326–328, 340–345
58	<code>\glsnextpages</code>
<code>\glslinkthevalue</code>	109
58, 132	<code>\glsnoidxdisplayloc</code>
<code>\glslinkwrgloss</code>	112, 113
8, 56, 57	<code>\glsnoidxdisplaylocclsthandler</code>
<code>\glslinkcheckfirsthyperhook</code>	112
74	<code>\glsnoidxloclist</code>
<code>\glslinkpostsetkeys</code>	113, 131, 132
58, 136	<code>\glsnoidxnumberlistloophandler</code>
<code>\glslinkvar</code>	112
77	<code>\glsnonnextpages</code>
<code>\glslistchildpostlocation</code>	109
314	<code>\glsnonnumberlistfalse</code>
<code>\glslistchildprelocation</code>	51
314, 315	<code>\glsnonnumberlisttrue</code>
<code>\glslistdottedwidth</code>	51
313	<code>\glsnopostdotfalse</code>
<code>\glslistgroupheaderfmt</code>	110
315, 316	<code>\glsnopostdottrue</code>
<code>\glslistnavigationitem</code>	110
315, 316	<code>\glsnumberlistloop</code>
<code>\glslistprelocation</code>	107
314, 315	<code>\glsnumlistlastsep</code>
<code>\glslocalunset</code>	112
56, 136	<code>\glsnumlistsep</code>
<code>\glslongaccessdisplay</code>	112
146, 147	
<code>\glslongdefaultfont</code>	
182, 199, 201,	
203, 207, 209, 212, 216, 218, 219, 221–	
225, 230, 232, 233, 235, 237–239, 244,	
248, 251, 252, 254, 255, 258, 265, 275, 288	

<code>\glsopenbrace</code>	40, 120, 121	<code>\glstarget</code>	126, 127, 313–323, 325–327, 339, 340, 350, 351
<code>\glsorder</code>	105	<code>\GLStext</code>	297
<code>\glspagelistwidth</code>	317, 319, 321, 323	<code>\Glstext</code>	297, 298
<code>\glspar</code>	130	<code>\glstext</code>	297
<code>\GLSpl</code>	90, 101, 135	<code>\glstextaccessdisplay</code>	141
<code>\Glspl</code>	49, 90, 101, 135	<code>\glstextformat</code>	56, 58
<code>\glspl</code>	48, 90, 101, 135	<code>\glstextup</code>	215
<code>\GLSplural</code>	298	<code>\glstreechildpredesc</code>	325, 326
<code>\Glsplural</code>	298	<code>\glstreechildprelocation</code> ...	325, 326, 328
<code>\glsplural</code>	298	<code>\glstreegroupheaderfmt</code>	325–328, 340–346, 348
<code>\glspluralaccessdisplay</code>	141, 142	<code>\glstreeindent</code>	326, 327, 338–340
<code>\glspluralsuffix</code>	124, 178, 182	<code>\glstreeitem</code>	324, 342, 350
<code>\glspostdescription</code>	15, 16, 110, 171, 313–323, 325–329	<code>\glstreenamebox</code>	339, 340
<code>\glspostinline</code>	324	<code>\glstreenamefmt</code>	325–327, 329, 331–340
<code>\glspostlinkhook</code> .	55, 56, 68–72, 84, 183–192	<code>\glstreenavigationfmt</code> ..	326–328, 340–345
<code>\glsprestandardsort</code>	107	<code>\glstreepredesc</code>	325–327
<code>\glsresetentrylist</code>	115, 128	<code>\glstreeprelocation</code>	324–327, 329
<code>\glssee</code>	41, 43	<code>\glstreesubitem</code>	324, 350
<code>\glsseeformat</code>	39, 40, 45, 106, 112, 113	<code>\glstreesubsubitem</code>	325, 351
<code>\glsseelist</code>	40	<code>\GlstrLetField</code>	31
<code>\glssetabbrvfmt</code> .	54, 60, 81, 82, 156–158, 160, 161, 163, 169, 183–187, 189, 190, 192	<code>\glstype</code>	56, 57, 68–72, 136, 183–192
<code>\glssetattribute</code> .	199–203, 205, 207–212, 214–220, 222, 226, 228, 230–233, 235, 237, 240, 242, 244–247, 249–252, 254, 256, 257, 259, 261–263, 266, 267, 269– 274, 276, 277, 279, 281, 283–286, 288–290	<code>\glunset</code>	44, 56, 93, 94, 136
<code>\glssetcategoryattribute</code>	90, 102, 104, 135, 152, 153, 155, 170	<code>\glswrite</code>	40, 105
<code>\glssetnoexpandfield</code>	11, 12	<code>\glswriteentry</code>	8, 9
<code>\glssettoctitle</code>	109	<code>\Glsxtr</code>	49
<code>\glsshortaccessdisplay</code>	145, 146	<code>\glxtr</code>	49
<code>\glsshortpltok</code>	180, 199–203, 205, 207, 209, 215, 217– 220, 226, 227, 230, 231, 233, 235, 240, 242, 244–252, 261, 263, 266, 267, 269, 271–276, 280, 282–284, 286, 287, 289, 290	<code>\glxtr@@do@wrglossary</code>	8, 9, 11, 13
<code>\glsshortpluralaccessdisplay</code>	146	<code>\glxtr@addloclistfield</code>	13
<code>\glsshorttok</code>	102, 178–180, 198– 203, 205, 207, 209, 214–220, 222, 226, 227, 230–233, 235, 237, 240–242, 244, 245, 247–252, 254, 256, 261, 263, 266– 269, 271–276, 279, 280, 282–284, 286–290	<code>\glxtr@addunused</code>	44
<code>\glssubentryitem</code>	126, 127, 313–323, 325–327, 339, 350	<code>\glxtr@applyabbrvfmt</code>	192
<code>\glssymbolaccessdisplay</code>	143	<code>\glxtr@applyabbrvstyle</code>	176, 178, 195
<code>\glssymbolpluralaccessdisplay</code> .	143, 144	<code>\glxtr@counterrecord</code>	125
		<code>\glxtr@do@alsoindex@wrglossary</code>	13
		<code>\glxtr@dooption</code>	5, 15, 16, 22, 24
		<code>\glxtr@fields</code>	123
		<code>\glxtr@headentry@p</code>	85, 86
		<code>\glxtr@hyperoutsidefalse</code>	57
		<code>\glxtr@hyperoutsidettrue</code>	57
		<code>\glxtr@ifnextpunc</code>	175
		<code>\glxtr@ifpunctoken</code>	175
		<code>\glxtr@indexonly@saveentrycounter</code>	13, 25
		<code>\glxtr@keylist</code>	47–49
		<code>\glxtr@label</code>	352
		<code>\glxtr@langtag</code>	124
		<code>\glxtr@linkprefix</code>	123, 124
		<code>\glxtr@makeglossaries</code>	105

<code>\glxtr@newabbreviation</code>	103, 178	<code>\glxtrbookindexbetween</code>	350
<code>\glxtr@next</code>	175	<code>\glxtrbookindexbookmark</code>	351
<code>\glxtr@org@do@wrglossary</code>	25	<code>\glxtrbookindexcols</code>	349
<code>\glxtr@org@dohyperlink</code>	78	<code>\glxtrbookindexcolspread</code>	349
<code>\glxtr@org@getgrouptitle</code>	114	<code>\glxtrbookindexfirstmarkfmt</code>	352
<code>\glxtr@org@newignoredglossary</code>	34	<code>\glxtrbookindexformatheader</code>	351
<code>\glxtr@org@makenoidxglossaries</code>	45	<code>\glxtrbookindexgroupskip</code>	351
<code>\glxtr@pluralsuffixes</code>	123, 124	<code>\glxtrbookindexlastmarkfmt</code>	352
<code>\glxtr@process</code>	128, 129	<code>\glxtrbookindexmulticolseenv</code>	349
<code>\glxtr@provideignoredglossary</code>	36	<code>\glxtrbookindexname</code>	347, 350
<code>\glxtr@punctlist</code>	174, 175	<code>\glxtrbookindexparentchildsep</code>	347, 349
<code>\glxtr@record</code>	10, 123	<code>\glxtrbookindexparentsubchildsep</code>	349, 350
<code>\glxtr@recordsee</code>	11	<code>\glxtrbookindexprelocation</code> ...	347, 350
<code>\glxtr@resource</code>	122, 123	<code>\glxtrbookindexsubbetween</code>	350
<code>\glxtr@s@newignoredglossary</code>	34	<code>\glxtrbookindexsubname</code>	351
<code>\glxtr@s@provideignoredglossary</code> ...	36	<code>\glxtrbookindexsubprelocation</code> ...	351
<code>\glxtr@saveentrycounter</code>	8, 9, 11, 75	<code>\glxtrbookindexsubsubbetween</code>	350
<code>\glxtr@setaccessdisplay</code>	163	<code>\glxtrbookindexthepage</code>	352
<code>\glxtr@setbookindexmark</code>	352	<code>\glxtrcat</code>	47–49
<code>\glxtr@setup@record</code>	12, 13, 24, 25	<code>\glxtrchecknohyperfirst</code>	61–63
<code>\glxtr@shortcutsval</code>	123, 124	<code>\glxtrComputeTreeIndent</code>	339
<code>\glxtr@texencoding</code>	124	<code>\glxtrComputeTreeSubIndent</code>	339
<code>\glxtr@usesee</code>	39	<code>\GlsXtrdefaultsubsequentfmt</code> ...	195, 196
<code>\glxtr@warnonexistsordo</code>	7, 13, 38	<code>\glxtrdefaultsubsequentfmt</code> ...	194, 196
<code>\glxtr@writefields</code>	122	<code>\GlsXtrdefaultsubsequentplfmt</code> .	195, 196
<code>\glxtrabbrvfootnote</code>	203–205, 226–228, 240–242, 261–263	<code>\glxtrdefaultsubsequentplfmt</code> .	195, 196
<code>\glxtrabbrvpluralsuffix</code>	124, 182, 199, 201, 204, 206, 207, 209, 212, 215, 229, 243, 266, 275, 278, 281, 286, 288	<code>\GlsXtrDefineAbbreviationShortcuts</code>	19, 20
<code>\glxtrabbrvtype</code>	16, 17, 180	<code>\GlsXtrDefineAcShortcuts</code>	19, 20
<code>\glxtractivateno</code>	109	<code>\GlsXtrDefineOtherShortcuts</code>	19, 20
<code>\glxtraddallcrossrefs</code>	43	<code>\glxtrdetoklocation</code>	135
<code>\glxtralias</code>	75	<code>\glxtrdiscardperiod</code>	172
<code>\glxtrAltTreeIndent</code>	329	<code>\glxtrdisplayendloc</code>	116
<code>\glxtralttreeInit</code>	339, 344, 345	<code>\glxtrdisplayendlochook</code>	116
<code>\glxtrAltTreePar</code>	329	<code>\glxtrdisplayingleloc</code>	116
<code>\glxtrAltTreeSetHangIndent</code> ...	329, 339	<code>\glxtrdisplaystartloc</code>	116
<code>\glxtrAltTreeSetSubHangIndent</code> ...	340	<code>\glxtrdoautoindexname</code>	76, 77, 162
<code>\glxtralttreeSubSymbolDescLocation</code> ..	340	<code>\glxtrdopostpunc</code>	205, 228, 242, 263
<code>\glxtralttreeSymbolDescLocation</code> ..	329, 339	<code>\glxtrdowrglossaryhook</code>	76
<code>\glxtrassignfieldfont</code>	60–67	<code>\glxtremsuffix</code>	244, 246, 248, 249, 251, 252, 254, 256, 258, 259, 262, 264
<code>\glxtrautoindex</code>	165	<code>\GlsXtrEnableEntryCounting</code>	101
<code>\glxtrautoindexassignsort</code> ...	165, 166	<code>\GlsXtrEnableEntryUnitCounting</code>	90
<code>\glxtrautoindexentry</code>	165	<code>\GlsXtrEnableOnTheFly</code>	47, 49
<code>\glxtrbookindexatendgroup</code>	350	<code>\glxtrendfor</code>	29
<code>\glxtrbookindexatsubendgroup</code>	350	<code>\glxtrfieldlistgadd</code>	125
<code>\glxtrbookindexatsubsubendgroup</code> ..	350	<code>\glxtrfieldtitlecase</code>	157–160, 164
		<code>\glxtrfieldtitlecasecs</code>	156

<code>\glxtrfieldxifinlist</code>	129	<code>\glxtrgroupfield</code>	131
<code>\glxtrfirstscfont</code>	215	<code>\Glsxtrheadfirst</code>	293
<code>\glxtrfirstsmfont</code>	229	<code>\glxtrheadfirst</code>	293
<code>\GlsXtrFmtDefaultOptions</code>	27, 28	<code>\Glsxtrheadfirstplural</code>	293
<code>\glxtrfmtdisplay</code>	27, 28	<code>\glxtrheadfirstplural</code>	293
<code>\GlsXtrFmtField</code>	27, 28	<code>\Glsxtrheadfull</code>	294
<code>\glxtrfootnotename</code>		<code>\glxtrheadfull</code>	294
..... 203, 205, 226, 227, 240, 241, 261, 263		<code>\Glsxtrheadfullpl</code>	294
<code>\GlsXtrFormatLocationList</code> 51, 54, 336–338		<code>\glxtrheadfullpl</code>	294
<code>\GLSxtrfull</code>	17, 18, 302, 303	<code>\Glsxtrheadlong</code>	294
<code>\Glsxtrfull</code>	17, 18, 303	<code>\glxtrheadlong</code>	293
<code>\glxtrfull</code>	17, 18, 302	<code>\Glsxtrheadlongpl</code>	294
<code>\Glsxtrfullformat</code>		<code>\glxtrheadlongpl</code>	293
..... 181, 194, 196, 197, 199, 201, 204,		<code>\Glsxtrheadname</code>	293
206, 208, 210, 213, 216, 218, 220, 221,		<code>\glxtrheadname</code>	125, 126, 293
224–226, 228, 230, 232, 234, 235, 238,		<code>\Glsxtrheadplural</code>	293
239, 241, 242, 244, 246, 248, 250, 252,		<code>\glxtrheadplural</code>	293
253, 255, 257, 259, 261, 262, 264, 267,		<code>\Glsxtrheadshort</code>	293
268, 270, 273, 276, 279, 281, 284, 287, 289		<code>\glxtrheadshort</code>	293
<code>\glxtrfullformat</code>		<code>\Glsxtrheadshortpl</code>	293
..... 181, 194, 196, 197, 199, 201,		<code>\glxtrheadshortpl</code>	293
204, 206, 208, 210, 213, 216, 218, 220,		<code>\Glsxtrheadtext</code>	293
221, 224–226, 228, 230, 232, 234, 235,		<code>\glxtrheadtext</code>	293
238–240, 242, 244, 246, 248, 249, 251,		<code>\glxtrhyperlink</code>	79, 117
253, 255, 257, 259, 260, 262, 264, 266,		<code>\glxtrhyphensuffix</code>	276, 284
268, 270, 273, 276, 279, 281, 284, 287, 289		<code>\glxtrifcounttrigger</code>	93, 94
<code>\GLSxtrfullpl</code>	17, 18, 302, 303	<code>\glxtrifcustomdiscardperiod</code>	172
<code>\Glsxtrfullpl</code>	17, 18, 303	<code>\glxtrifemptyglossary</code>	115, 121, 128
<code>\glxtrfullpl</code>	17, 18, 302, 303	<code>\glxtrifhasfield</code>	32, 75, 347
<code>\Glsxtrfullplformat</code>		<code>\glxtrifhyphenstart</code> 275, 277, 280, 283, 285	
. 181, 194, 196, 197, 199, 202, 204, 206,		<code>\glxtrifindexing</code>	76
208, 210, 213, 216, 218, 220, 221, 224,		<code>\glxtrifinmark</code>	59, 85–88, 292–294
225, 227, 228, 230, 232, 234, 236, 238,		<code>\glxtrifnextpunc</code>	175, 176
240, 241, 243, 245, 246, 248, 250, 252,		<code>\glxtrifperiod</code>	172, 174
253, 255, 257, 259, 261, 262, 264, 267,		<code>\glxtrifrecordtrigger</code>	137–139
268, 270, 273, 276, 279, 282, 284, 287, 289		<code>\glxtrifwasfirstuse</code>	
<code>\glxtrfullplformat</code> 60–63, 68–71, 74, 104,	
..... 193, 194, 196, 197, 199, 201,		173, 174, 183, 186–192, 205, 228, 242,	
204, 206, 208, 210, 213, 216, 218, 220,		263, 264, 267, 269, 271, 281, 282, 286, 288	
221, 224–226, 228, 230, 232, 234, 235,		<code>\glxtrindexaliased</code>	75
238–240, 242, 244, 246, 248, 250, 252,		<code>\glxtrindexseealso</code>	42, 43
253, 255, 257, 259, 261, 262, 264, 266,		<code>\glxtrinithyperoutside</code>	58
268, 270, 273, 276, 279, 281, 284, 287, 289		<code>\glxtrinitwrgloss</code>	58, 136
<code>\glxtrfullsep</code>		<code>\glxtrinitwrglossbeforefalse</code> ...	56, 57
. 181, 199–202, 204–211, 213, 215–223,		<code>\glxtrinitwrglossbeforetrue</code>	57
225, 227, 229–239, 241, 243–260, 262–		<code>\Glsxtrinlinefullformat</code> 181, 183, 196,	
265, 275, 276, 278, 280, 283–285, 289, 290		197, 204, 206, 207, 209, 211, 213, 219,	
<code>\glxtrgenabbrvfmt</code>	54	221–223, 225, 227, 229, 234–237, 239,	
<code>\glxtrgetgroupitle</code>	115, 351		

241, 243, 251, 253–255, 257, 258, 260, 263, 264, 268, 270, 278, 282, 287, 290, 309	\GlsXtrNoGlsWarningEmptyMain 121
\glsxtrinlinefullformat	\GlsXtrNoGlsWarningEmptyNotMain . . . 121
181, 183, 184, 196, 197, 204, 206, 207, 209, 211, 213, 219, 221– 223, 225, 227, 229, 233, 235–237, 239, 241, 243, 251–253, 255, 256, 258, 260, 262, 264, 268, 270, 278, 282, 287, 289, 309	\GlsXtrNoGlsWarningEmptyStart 121
\Glsxtrinlinefullplformat 182, 185, 196, 197, 205, 206, 208, 210, 211, 213, 220– 223, 225, 227, 229, 234–236, 238, 239, 241, 243, 251, 253–255, 257, 259, 260, 263, 264, 268, 271, 278, 282, 287, 290, 310	\GlsXtrNoGlsWarningHead 121
\glsxtrinlinefullplformat	\GlsXtrNoGlsWarningMisMatch 122
181, 182, 184, 185, 196, 197, 204, 206, 207, 209, 211, 213, 219, 221– 223, 225, 227, 229, 234–237, 239, 241, 243, 251, 252, 254, 255, 257, 258, 260, 262, 264, 268, 270, 278, 282, 287, 290, 309	\GlsXtrNoGlsWarningNoOut 122
\glsxtrinsertinsidefalse 198	\GlsXtrNoGlsWarningTail 122
\glsxtrlocationhyperlink 117	\glsxtrnopostpunc 109
\glsxtrlocrangefmt 116	\glsxtronlydescname 290
\GLSxtrlong 17, 18, 300, 301	\glsxtronlydescsort 290
\Glsxtrlong 17, 18, 301	\glsxtronlyname 289
\glsxtrlong 17, 18, 300	\glsxtronlysuffix 289
\glsxtrlonghyphen 281, 282	\glsxtrorg@ifKV@glslink@hyper 55
\glsxtrlonghyphennohshort 278, 279	\glsxtrorglong 178, 200, 277
\glsxtrlonghyphenshort 276	\glsxtrorgshort 178, 200
\glsxtrlongnohshortdescname . 212, 259, 277	\GLSxtrp 85
\glsxtrlongnohshortname	\Glsxtrp 85
214, 222, 237, 254, 256, 279	\glsxtrp 84, 86
\GLSxtrlongpl 17, 18, 301, 302	\glsxtrparen 173, 181, 199–202, 204–211, 213, 215–223, 225, 227, 229–239, 241, 243–255, 257–260, 262–265, 275, 276, 278, 280, 283–285, 290
\Glsxtrlongpl 17, 18, 302	\Glsxtrpl 49
\glsxtrlongpl 17, 18, 301	\glxtrpl 49
\glsxtrlongshortdescname	\glxtrpostdescription . 110, 155, 171, 324
200, 216, 231, 245, 247, 276, 282	\glxtrposthyphenlong 286, 288
\glsxtrlongshortdescsort	\glxtrposthyphenshort 281, 282
200, 216, 231, 245, 247, 272, 276, 282	\glxtrposthyphensubsequent
\glsxtrlongshortname	281, 282, 286, 288
198, 215, 230, 244, 245, 266, 267, 275, 280	\glxtrpostlink 172
\glsxtrlongshortuserdescname . . 269, 272	\glxtrpostlinkendsentence 172
\glsxtrmarkhook 291, 292	\glxtrpostlinkhook 172
\glsxtrnewabbrevpresetkeyhook 179	\glxtrpostlocalreset 89, 91, 99
\glsxtrnewnumber 18	\glxtrpostlocalunset 89, 91, 99
\glsxtrnewsymbol 18	\glxtrpostlongdescription 33
\glsxtrNoGlossaryWarning 20, 118	\glxtrpostnamehook 159–162, 164
\GlsXtrNoGlsWarningAutoMake 121	\GlsXtrPostNewAbbreviation
\GlsXtrNoGlsWarningBuildInfo 122	180, 196, 197, 199–203, 205, 207–212, 214, 215, 217–220, 222, 226, 227, 230–233, 235, 237, 240, 242, 244–247, 249–252, 254, 256, 257, 259, 261, 263, 266, 267, 269, 271, 272, 274, 276, 277, 279, 280, 282, 284–286, 288–290
\GlsXtrNoGlsWarningCheckFile 121	\glxtrpostreset 89, 91, 99
	\glxtrpostunset 89, 91, 99
	\glxtrprelocation
	314, 316, 318, 320, 322, 324, 347
	\glxtrprotectlinks 78–80
	\GlsXtrRecordCounter 11

\glxtrrecordtriggervalue	136	\glxtrsubsequentplfmt	
\glxtrregularfont	54, 60 193, 196, 212, 223, 224,	
\glxtrresourcecount	123	237, 239, 254, 256, 258, 260, 278, 281, 286	
\glxtrresourcefile	123	\glxtrsupplocationurl	117
\glxtrresourceinit	122	\glxtrtagfont	171
\glxtrrestoremarkhook	291, 292	\Glsxtrtitlefirst	293, 294, 307
\glxtrrestorepostpunc	110	\glxtrtitlefirst	293, 294, 307
\glxtrscfont	215	\Glsxtrtitlefirstplural	293, 294, 307, 308
\glxtrscsuffix		\glxtrtitlefirstplural	293, 294, 307
.... 216, 217, 219, 220, 223, 224, 226, 228		\Glsxtrtitlefull	293, 294, 309
\GlsXtrSetActualChar	168	\glxtrtitlefull	293, 294, 309
\glxtrsetaliasnoindex	13, 75	\Glsxtrtitlefullpl	293, 294, 310
\GlsXtrSetEncapChar	168	\glxtrtitlefullpl	293, 294, 309, 310
\GlsXtrSetEscChar	168	\Glsxtrtitlelong	293, 294, 308
\glxtrsetfieldifexists	31, 32	\glxtrtitlelong	293, 294, 308
\GlsXtrSetLevelChar	168	\Glsxtrtitlelongpl	293, 294, 309
\glxtrsetpopts	84	\glxtrtitlelongpl	293, 294, 308
\glxtrsetupfulldefs		\Glsxtrtitlename	293, 294, 305
..... 183–185, 205, 228, 242, 264		\glxtrtitlename	293, 294, 305
\GLSxtrshort	17, 18, 88, 295	\glxtrtitleorpdforheading	
\Glsxtrshort	17, 18, 295, 296 23, 125–127, 293, 294	
\glxtrshort	17, 18, 295	\Glsxtrtitleplural	293, 294, 306
\glxtrshortdescname ...	209, 220, 234, 252	\glxtrtitleplural	293, 294, 306
\glxtrshorthyphen	287	\Glsxtrtitleshort	293, 294, 304
\glxtrshorthyphenlong	284	\glxtrtitleshort	293, 294, 304
\glxtrshortlongdescname		\Glsxtrtitleshortpl	293, 294, 305
..... 202, 218, 232, 248, 250, 284, 287		\glxtrtitleshortpl	293, 294, 304
\glxtrshortlongdescsort		\Glsxtrtitletext	293, 294, 306
.... 202, 218, 232, 248, 250, 273, 284, 287		\glxtrtitletext	293, 294, 305, 306
\glxtrshortlongname		\GlsXtrTotalRecordCount	135
201, 217, 231, 247, 249, 269, 272, 283, 286		\glxtrtreetopindent	329, 338
\glxtrshortlonguserdescname ..	271, 273	\glxtrundefaction ..	7, 13, 25, 34, 35, 37, 38
\glxtrshortnolongname .	207, 219, 233, 251	\glxtrundeftag	25, 112, 113
\GLSxtrshortpl	17, 18, 295, 296	\glxtrunsrtdo	129, 130
\Glsxtrshortpl	17, 18, 296	\GlsXtrUseAbbrStyleFmts	
\glxtrshortpl	17, 18, 295 200, 202, 209–211,	
\glxtrsmfont	229	214, 215, 217, 219, 222, 231, 233, 236,	
\glxtrsmsuffix		245, 247, 249, 250, 253, 258, 261, 269,	
.... 230, 232, 233, 235, 237, 238, 240, 242		271, 272, 274, 277–279, 283, 285, 288, 291	
\Glsxtrsubsequentfmt		\GlsXtrUseAbbrStyleSetup	
..... 193, 196, 212, 223, 224,	 208, 210, 211, 213,	
237, 239, 255, 256, 258, 260, 278, 281, 286		214, 222, 224, 236, 238, 253, 257, 258, 261	
\glxtrsubsequentfmt		\glxtruserfield	265
..... 193, 196, 212, 223, 224,		\glxtruserparen	266–274
237, 238, 254, 256, 258, 260, 278, 281, 286		\glxtrusersuffix	266, 267, 270, 273
\Glsxtrsubsequentplfmt		\glxtruseseealsoformat	40
..... 193, 196, 212, 223, 224,		\glxtruseseeformat	39
237, 239, 255, 256, 258, 260, 278, 281, 286		\GlsXtrWarnDeprecatedAbbrStyle	176, 197
		\GlsXtrWarning	47–49

<code>\glxtrword</code>	178	<code>\ifglstryexists</code>	8,
<code>\glxtrwordsep</code> ..	178, 275, 277, 280, 283, 285		37, 38, 47–49, 51, 60, 131, 152, 171, 172
<code>\glxtrwrglossmark</code>	22	<code>\ifglsfiedeq</code>	151
H			
<code>\hangindent</code> .	326, 327, 329, 338–340, 344, 345	<code>\ifglshasfield</code>	27, 28, 265
<code>\hbox</code>	313	<code>\ifglshaslong</code>	95, 96, 139
<code>\hfill</code>	313	<code>\ifglshasparent</code> .	126–128, 131, 331, 333, 334
<code>\href</code>	78	<code>\ifglshasshort</code>	40, 54, 60
<code>\hsize</code>	50, 51	<code>\ifglshassymbol</code>	173, 325–327, 329
<code>\hss</code>	313	<code>\ifglsindeonlyfirst</code>	76
<code>\hyperlink</code>	79	<code>\ifglsnogroupskip</code> 314, 316–326, 328, 340, 348	
<code>\hyperpage</code>	164	<code>\ifglssonumberlist</code>	53
<code>\hyperref</code>	78, 117	<code>\ifglsnopostdot</code>	15, 110
<code>hyperref</code> package	80, 164, 291, 304	<code>\ifglssanitizesort</code>	107
I			
<code>\if</code>	47	<code>\ifglssubentrycounter</code>	32
<code>\if@glxtr@autoseeindex</code>	23, 24, 39, 42	<code>\ifglssused</code>	44,
<code>\if@glxtr@format@override</code>	165		74, 76, 92, 101, 104, 192, 331–333, 335–337
<code>\if@glxtr@docdefrestricted</code>	45	<code>\ifglsxindy</code>	118–120
<code>\if@glxtr@indexcrossrefs</code>	14, 43	<code>\ifglxtr@hyperoutside</code>	59
<code>\ifblank</code>	26, 47–49, 105	<code>\ifglxtr@trinitwrglossbefore</code> ...	57–59, 136
<code>\ifcase</code> ..	7, 12, 19, 20, 22, 46, 57, 110, 325, 350	<code>\ifglxtr@insertinside</code> ..	186–192, 194,
<code>\ifcsdef</code> 25, 32, 34–37, 58, 72, 73, 84–88, 96,			195, 199, 201, 202, 204–213, 216, 218–
108, 114, 115, 126, 130, 133–135, 157,			246, 248–264, 266–268, 270, 271, 273,
158, 160–163, 173, 176, 192, 196, 316–323			275, 277, 280, 282, 283, 285, 287, 289, 290
<code>\ifcsstring</code>	25, 152, 195	<code>\ifHy@hyperindex</code>	164
<code>\ifcsundef</code>	30, 32, 34–36,	<code>\ifinlistcs</code>	29, 45
45, 50, 52, 80, 91, 96–100, 113, 114, 118,		<code>\ifinner</code>	23
130, 152, 195–198, 313, 330, 331, 338, 352		<code>\ifKV@glslink@hyper</code>	55, 57, 59
<code>\ifcsvoid</code>	43, 151	<code>\ifKV@glslink@local</code>	56, 136
<code>\ifdef</code> 13, 18, 24, 28, 37, 38, 40, 41, 50, 51, 74,		<code>\ifKV@glslink@noindex</code> ..	8, 9, 11, 28, 75, 76
78, 79, 85–87, 108, 112, 113, 124, 133,		<code>\ifmmode</code>	23
154, 155, 168, 171, 265, 292, 304–310,		<code>\ifnum</code>	14,
313–316, 324–328, 340–345, 348, 351, 352			92, 100, 101, 113, 123, 136, 326, 327, 339
<code>\ifdefempty</code>	7–9, 30,	<code>\ifstreempty</code>	126, 133
34–36, 39, 40, 58, 59, 90, 102, 105, 108,		<code>\ifstrequal</code>	16, 21
116, 128, 131, 135, 136, 170, 177, 192, 349		<code>\ifthenelse</code>	121
<code>\ifdefequal</code>	45, 121, 131, 162	<code>\IfTrackedLanguageFileExists</code>	310
<code>\ifdefstring</code>	6, 32, 165, 170, 348	<code>\ifundef</code>	30, 105, 170, 171
<code>\ifdefvoid</code> 39, 42–44, 79, 96, 113, 117, 131, 132		<code>\ifx</code> ...	8–10, 41, 50, 51, 105, 109, 116, 117,
<code>\ifdim</code>	50, 51, 104, 330–338		124, 165, 166, 168, 175, 177–179, 274, 346
<code>\IfFileExists</code>	21, 118, 121, 122, 124, 312	<code>\immediate</code>	92, 100, 118, 124
<code>\ifglossaryexists</code>	38	<code>\index</code>	165
<code>\ifglssacronym</code>	17, 121	<code>\indexspace</code> .	314, 325–328, 340–346, 348, 351
<code>\ifglssacrshortcuts</code>	19	<code>\input</code>	310
<code>\ifglssautomake</code>	108, 121, 124	<code>\inputencodingname</code>	124
<code>\ifglstrycounter</code>	32	<code>\istfilename</code>	105
J			
		<code>\item</code>	119, 120, 313–316, 324–326, 341, 342
		<code>\jobname</code>	118–124

K	
<code>\key@ifundefined</code>	11, 12, 26, 72, 128, 131
<code>\KV@glslink@hyperfalse</code>	61, 74, 79, 80
<code>\KV@glslink@hypertrue</code>	80
<code>\KV@glslink@noindexfalse</code>	74, 75
<code>\KV@glslink@noindextrue</code>	75, 80
L	
<code>\LaTeX</code>	119, 120
<code>\leaders</code>	313
<code>\leavevmode</code>	34, 57
<code>\let</code>	5, 7–13, 15, 17–19, 23–25, 27, 29, 33, 45, 46, 49, 50, 52, 53, 55–71, 73–78, 80, 81, 84, 90, 91, 99–106, 108–115, 122, 124, 128, 129, 131, 136, 157–167, 170, 171, 175–179, 183–192, 194–196, 205, 228, 242, 264, 291–294, 324, 325, 329, 331, 342, 349–352
<code>\letabbreviationstyle</code> . .	205, 207, 208, 210, 213, 214, 220, 221, 234, 236, 252, 253
<code>\letcs</code>	26, 30, 39, 40, 44, 58, 72, 112–114, 130, 131, 157–164, 352
<code>\levelchar</code>	168
<code>\listadd</code>	96
<code>\listbreak</code>	170
<code>\listcsadd</code>	28
<code>\listcseadd</code>	29, 97
<code>\listcsgadd</code>	29, 45
<code>\listcsxadd</code>	29, 97
<code>\loadglsentries</code>	46, 119
<code>\long</code>	33
M	
<code>\MakeAcronymsAbbreviations</code>	104
<code>\makeatletter</code>	118, 122, 167
<code>\makeatother</code>	167
<code>\makebox</code>	313, 339, 340
<code>\makefirstuc</code>	171
<code>makeglossaries</code>	111
<code>\makeglossaries</code>	105, 118, 120, 121, 125
<code>\makeglossary</code>	105
<code>makeindex</code>	353
<code>makeindex</code>	104
<code>\makenoidxglossaries</code>	120
<code>\MakeTextUppercase</code>	293
<code>\MakeUppercase</code>	293, 294
<code>\marginpar</code>	23
<code>\markboth</code>	292
<code>\markright</code>	292
<code>\maxdimen</code>	50, 51
<code>\mbox</code>	315, 316, 339
<code>\medskip</code>	121, 130
<code>\MessageBreak</code> 46, 49, 92, 101, 105, 108, 109, 195	
<code>mfistuc package</code>	170
<code>\mfistucMakeUppercase</code>	61–71, 73, 82, 83, 85, 88, 95, 103, 140–150, 159, 160, 164, 184, 185, 187, 188, 190, 192–194
<code>\mfu@checkword@arg</code>	170, 171
<code>\mfu@checkword@do</code>	171
N	
<code>\NeedsTeXFormat</code>	5, 312, 347
<code>\new@glossaryentry</code>	46, 108
<code>\new@ifnextchar</code>	27, 72, 73, 94, 95, 133, 137–139, 174, 182–191
<code>\newabbr</code>	17, 18
<code>\newabbreviation</code>	17, 18
<code>\newabbreviationhook</code>	180
<code>\newabbreviationstyle</code> 198, 200–203, 205, 207–220, 222, 224, 226, 227, 230–234, 236, 238, 240, 241, 244–254, 256–259, 261, 263, 266, 267, 269, 271–273, 275– 277, 279, 280, 282–284, 286, 287, 289, 290	
<code>\newacronym</code>	102, 103
<code>\newacronymhook</code>	102
<code>\newacronymstyle</code>	103, 104
<code>\newcommand</code>	5–8, 10–12, 14–40, 42, 44–50, 52–54, 56, 57, 60, 61, 72, 73, 75–77, 79, 80, 83–92, 94– 104, 108–114, 116–127, 129–156, 162– 178, 180–192, 194–198, 200, 202, 203, 207, 209, 212, 214, 215, 229, 243, 244, 265, 266, 268, 271, 275, 277, 279, 280, 283, 285, 288, 290, 292–310, 312, 314, 324, 328–331, 338, 339, 347, 348, 351, 352
<code>\newcount</code>	14, 122, 132
<code>\newentry</code>	18
<code>\newglossary</code>	16, 105
<code>\newglossaryentry</code>	18, 46, 90, 98, 102, 154, 155, 180
<code>\newglossaryentry options</code>	
<code>alias</code>	15, 38, 41–44
<code>desc</code>	144, 149
<code>descplural</code>	145, 149
<code>first</code>	78, 142, 148, 198, 299, 300, 306, 353
<code>firstplural</code> 142, 143, 148, 198, 299, 300, 307, 353	
<code>group</code>	130, 131
<code>loclist</code>	28
<code>long</code>	147, 150, 308

xindy	40, 41	\ProvidesFile	310
\PackageError	6, 11, 21, 24, 46, 49, 50, 72, 73, 75, 84, 85, 90, 91, 98, 100, 101, 103, 105, 107, 108, 115, 124, 129, 130, 133, 195–198, 312, 313	\ProvidesPackage	5, 312, 347
\PackageWarning	15	Q	
\PackageWarningNoLine	15	\quotechar	168
\pageref	32	R	
\par .	121, 122, 315, 316, 325–329, 339–345, 348	\raggedright	318, 319, 322, 323, 349
\parindent	324, 326, 327, 329, 339, 340, 342–346, 349	\relax	7, 10, 12–14, 17–20, 22, 24, 27, 46, 49–51, 53, 56, 57, 77, 84, 91, 92, 100, 105, 106, 109, 112, 113, 116, 122–124, 131, 136, 166–168, 171, 173, 177–179, 274, 325–327, 331–340, 344–346, 349–352
\parskip	324, 326, 327, 342–344, 349	resize package	229
\PassOptionsToPackage	5, 21	\renewcommand	6, 7, 13–17, 19–22, 24, 33–38, 40, 45, 46, 49–54, 56, 72, 74, 76, 78–80, 84, 89–92, 95, 96, 98– 111, 113–115, 118, 129, 130, 135, 154, 156–159, 161, 169–172, 181, 182, 196– 264, 266–292, 313–328, 339–345, 349–351
\pdfbookmark	348	\renewenvironment	314, 316–324, 326, 327, 339, 342–345, 349
\preglossarypreamble	32	\renewglossarystyle	313–328, 339–345
\preto	75	\renewrobustcmd	59, 79
\print@noop@unsrtglossaryunit ...	11, 13	\RequireGlossariesExtraLang	311
\print@op@unsrtglossaryunit	13	\RequirePackage	5, 20–22, 312, 347
\printabbreviations	17	\reserved@a	175
\printglossaries	106, 120	\reserved@b	175
\printglossary	17, 106, 120	\reserved@d	175
\printglossary options		\RestoreAcronyms	103, 104
nonumberlist	53	\rGLS	135
type	108	\rGLs	135
\printnoidxglossaries	120	\rgls	135
\printnoidxglossary	106, 107, 120	\rGLSformat	138
\printnumbers	18, 155	\rGLsformat	138
\printsymbols	18, 155	\rglsformat	137, 140
\printunsrtglossary	127	\rGLSpl	135
\printunsrtglossaryentryprocesshook	128	\rGLspl	135
\printunsrtglossaryhandler	129	\rGLSplformat	139
\printunsrtglossarypredoglossary ..	128	\rGLsplformat	138
\printunsrtglossaryskipentry ..	128, 129	\rglsplformat	137, 140
\printunsrtglossaryunit	13, 130	\romannumeral	329–331, 338
\printunsrtglossaryunitsetup	129	S	
\ProcessOptions	313	\s@@glstrfmt	27
\ProcessOptionsX	22	\s@glshyp@opt	77
\protect	85–88, 147– 150, 178, 181, 198–205, 207–209, 211– 220, 222–228, 230–242, 244–264, 266– 269, 271–280, 282–284, 286–290, 294–303	\s@glstr@enabletagging	169, 170
\protected@csedef	32, 329, 330	\s@glstrfmt	27
\protected@csxdef	31, 330, 331	\s@glstrifhasfield	30
\protected@edef	50, 78, 102, 113, 114, 129–131, 165, 180		
\protected@write	10, 11, 45, 53, 105, 106, 122–125, 352		
\providecommand	16–18, 26, 40, 53, 73, 75, 92, 100, 105, 118, 123, 313, 347		

