

`glossaries-extra.sty v1.26: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2018-01-05

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	105
1.3.8	Support for bib2gls	133
1.4	Link Counting	140
1.5	Integration with glossaries-accsupp	142
1.6	Categories	153
1.7	Abbreviations	178
1.7.1	Abbreviation Styles Setup	197
1.7.2	Predefined Styles (Default Font)	200
1.7.3	Predefined Styles (Small Capitals)	217
1.7.4	Predefined Styles (Fake Small Capitals)	231
1.7.5	Predefined Styles (Emphasized)	245
1.7.6	Predefined Styles (User Parentheses Hook)	267
1.7.7	Predefined Styles (Hyphen)	276
1.7.8	Predefined Styles (No Short on First Use)	290
1.8	Using Entries in Headings	293
1.9	Multi-Lingual Support	312
2	Style Adjustments (glossaries-extra-stylemods.sty)	314
2.1	Package Initialisation	314
2.2	List-Like Styles	315
2.3	Longtable Styles	318
2.4	Long Ragged Styles	320
2.5	Supertabular Styles	322
2.6	Super Ragged Styles	324
2.7	Inline Style	326
2.8	Tree Styles	326
2.9	Multicolumn Styles	343

3 bookindex style (glossary-bookindex.sty)	349
3.1 Package Initialisation and Options	349
Glossary	355
Change History	356
Index	372

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2018/01/05 v1.26 (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\@glsl@label{\glstoklabel{#1}}%
87 \let\glslabel\@glsl@label
88 \glswriteentry{#1}%
89 {%
90 \ifdefempty{\@glxtr@thevalue}%
91 {%
92 \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
93 \else
94 \let\theHglentrycounter\@glxtr@theHvalue
95 \fi
96 \glxtr@saveentrycounter
97 \let\@do@wrglossary\@glxtr@dorecord
98 }%
99 {%
100 \let\theHglentrycounter\@glxtr@thevalue
101 \let\theHglentrycounter\@glxtr@theHvalue
102 \let\@do@wrglossary\@glxtr@dorecordnodefer
103 }%
104 \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
105 \glxtr@do@wrglossary{#1}%
106 \else
107 \@glxtrwrglossmark
108 \@do@wrglossary
109 \fi
110 }%
111 \fi
112 \endgroup
113 }

```

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

```

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

```

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

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

```

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

```


122 \beginngroup

Save the label in case it's needed.

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

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

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

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

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

Check if thevalue has been set.

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

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

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

Save the entry counter.

```
142     \glsxtr@saveentrycounter
```

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

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

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

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

No need to escape special characters.

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

```

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

```

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

```

160 \newcommand*\@glsxtr@dorecord{%
161   \global\let\@glsrecordlocref\theHglentrycounter
162   \let\@glsxtr@orgprefix\@glo@counterprefix
163   \ifx\theHglentrycounter\theHglentrycounter
164     \def\@glo@counterprefix{}%
165   \else
166     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
167       {\theHglentrycounter}{\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 `\theHglentrycounter` 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\theHglentrycounter\theHglentrycounter
179     \protected@write\@auxout{{\string\glsxtr@record
180       {\@gls@label}{\@gls@counter}{\@glsnumberformat}}%
181       {\theHglentrycounter}}}%
182   \else
183     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
184       {\theHglentrycounter}{\theHglentrycounter}}%
185     }%
186     \@do@gls@getcounterprefix
187     \protected@write\@auxout{{\string\glsxtr@record
188       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}}%
189       {\theHglentrycounter}}}%
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}

```

`bbreviationsdef` Set by abbreviations option.

```

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

```

`bbreviationsdef`

```

394 \newcommand*{\@glsxtr@doabbreviationsdef}{%
395 \ifpackageloaded{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   \ifdef\printsymbols
471     {%
472       \newcommand*{\newsym}{\gl{s}xtrnewsymbol}%
473     }{}%
474   \ifdef\printnumbers
475     {%
476       \newcommand*{\newnum}{\gl{s}xtrnewnumber}%
477     }{}%
478   \let\GlsXtrDefineOtherShortcuts\relax
479 }
```

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

@setupshortcuts Command used to set the shortcuts option.

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

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

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

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

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

```

516      \glsacrshortcutstrue

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

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

    Leave none and false as last option.

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

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

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

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

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

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

xtr@redefstyles
548 \newcommand*{\@glxtr@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` `\glstrifemptyglossary{<type>}{<true>}{<false>}`

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

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

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       \gl s@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}

```

`\glstrfmt`

```

746 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

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

```

`\@glstrfmt` Unstarred form.

```

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

```

`\s@glstrfmt` Starred form.

```

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

```

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

```

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

```

`\@glstrfmt` Actual inner working.

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

`\glsxtrentryfmt` No link or indexing.

```

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

```

`@glsxtrentryfmt`

```

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

```

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

```

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

```

trfieldlistgadd Similarly but uses \listcsgadd.

```

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

trfieldlistead Similarly but uses \listcseadd.

```

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

trfieldlistxadd Similarly but uses \listcsxadd.

```

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

Now provide commands to iterate over these lists.

fielddolistloop

```

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

fieldforlistloop

```

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

List element tests:

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

```

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

trfieldxifinlist Expands item.

```

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

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

```

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

```

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

```

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

```

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

```

`\lsxtrifhasfield` Unstarred version adds grouping.

```

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

```

`\lsxtrifhasfield` Starred version omits grouping.

```

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

```

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

Just uses `\ifcsundef`.

```

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

```

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

```

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

```

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

```

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

```

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

```
853 \newcommand*{\glstrdeffield}[2]{\csdef{glo@\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@glstdetoklabel{#1}@#2}{#3}}%
883 }
```

XtrIfFieldEqStr

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

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

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

lossarypreamble

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

lossarypreamble

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



```

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

```

1.3 Modifications to Commands Provided by glossaries

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

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

`ewglossaryentry`

```

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

```

`ewglossaryentry` Starred version.

```

927 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
928   \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*{\glsxtrpostlongdescription}{\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\glsxtr@s@newignoredglossary\glsxtr@org@newignoredglossary
964 }

```

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

```

965 \newcommand*{\glsxtr@org@newignoredglossary}[1]{%
966 \ifcsdef{glolist@#1}
967 {%
968 \glsxtrundefaction{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 }
```

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

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

`\glsseealsoformat` The format used by `\glstruseseealso`. 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\glstruseseealsoformat\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\@glx@xref{#2}%
1227       \@onelevel@sanitize\@glx@xref
1228       \@gl@checkmkidxchars\@glx@xref
1229       \glsglossary{\csname glo@#1@type\endcsname}{%
1230         (indexentry
1231           :tkey (\csname glo@#1@index\endcsname)
1232           :xref (\string"\@glx@xref\string")
1233           :attr \string"seealso\string"
1234         )
1235       }%
1236     }%
1237   }
1238 }
1239 {

```

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

```

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

```

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

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

```

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

```

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

```

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

```

Add to the key mappings.

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

Set the default value.

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

Assign the field values.

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

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

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

Provide user-level commands to access the values.

`\glstralias`

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

`trseealsolabels`

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

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

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

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

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

```

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

```

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

`\glstralias`

```

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

```

`trseealsolabels`

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

1316 \newcommand*{\glxtraddallcrossrefs}{%
1317   \forallglossaries{\@glo@type}%
1318   {%
1319     \forallgsentries[\@glo@type]{\@glo@label}%
1320     {%
1321       \ifglused{\@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\@glstetoklabel{#1}@see}%
1328   \ifdefvoid\@glo@see
1329   {%
1330     {%
1331       \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1332     }%
1333     \letcs{\@glo@see}{glo\@glstetoklabel{#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     \ifglused{\@glxtr@label}}}%
1347   {%
1348     \gladd[format=glxtrunusedformat]{\@glxtr@label}%
1349     \glunset{\@glxtr@label}%
1350     \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1351   }%
1352 }%
1353 }

```

xtrunusformat

```
1354 \newcommand*{\glxtrunusformat}[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\endcename}%
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   \glsdefglossaryentry{##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.

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

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

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

`\etglossarystyle`

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

Only set the current style if it exists.

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

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

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

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

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

Similarly for `\glstdescwidth`:

\glsdescwidth

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

and for \glspagelistwidth:

lspagelistwidth

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

aryentrynumbers Has the nonumberlist option been used?

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

```

```

1603     }{}%
1604   }%
1605 \fi

```

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

```

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

```

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

`\ePreLocationTag`

```

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

```

`\glxtrpreloctag`

```

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

```

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

```

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

```

```

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

```

glxtrpreloctag

```

1643 \newcommand*{\glxtrpreloctag}{%

```

@glxtr@pagetag

```

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

```

glxtr@pagetag

```

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

```

lsxtrpostloctag

```

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

```

lsxtrpostloctag

```

1653 \newcommand*{\glxtrpostloctag}{%

```

lsxtr@preloctag

```

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

```

glxtr@doloctag

```

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

```

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

```

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

```

```

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

```

1.3.4 Entry Formatting, Hyperlinks and Indexing

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

```

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

```

`sxtrregularfont` Font used for regular entries.

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

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

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

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

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

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

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

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



```

1766 \glxtrinitwrglossbeforetrue
1767 }%
1768 }

```

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

```

1769 \newif\ifglxtrinitwrglossbefore
1770 \glxtrinitwrglossbeforetrue

```

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 \glxtrinitwrglossbeforetrue
1775 \or
1776 \glxtrinitwrglossbeforefalse
1777 \fi
1778 }

```

```

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

```

```

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

```

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

```

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

```

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

```

1783 \newcommand*{\glxtrinithyperoutside}{%
1784 \glsifattribute{\glslabel}{hyperoutside}{false}%
1785 {%
1786 \glxtr@hyperoutsidedefalse
1787 }%
1788 {%
1789 \glxtr@hyperoutsidettrue
1790 }%
1791 }

```

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

```

1792 \newcommand*{\glxtr@inc@linkcount}{}

```

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

```

1793 \newcommand*{\glslinkpresetkeys}{}

```

`\@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`.

```

1794 \def\@gls@link[#1]#2#3{%

```

```

1795 \leavevmode
1796 \edef\glslabel{\glsdetoklabel{#2}}%
1797 \def\@gls@link@opts{#1}%
1798 \let\@gls@link@label\glslabel
1799 \let\@gls@numberformat\@gls@defaultnumberformat
1800 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1801 \edef\@gls@type{\csname glo@\glslabel @type\endcsname}%
1802 \let\org@ifKV@glslink@hyper\ifKV@glslink@hyper

  Initialise thevalue and theHvalue (v1.19).
1803 \def\@gls@tr@thevalue{}%
1804 \def\@gls@tr@theHvalue{\@gls@tr@thevalue}%

  Initialise when indexing should occur (new to v1.14).
1805 \gls@tr@init@wrgloss

  Initialise whether \hyperlink should be outside \gls@text@format (new to v1.21).
1806 \gls@tr@init@hyper@outside

  Note that the default link options may override \gls@tr@init@wrgloss.
1807 \@gls@set@default@glslink@opts

  Increment link counter if enabled (new to v1.26).
1808 \gls@tr@inc@linkcount

  As the original definition.
1809 \do@gls@disable@hyper@in@list
1810 \do@gls@link@check@first@hyper

  User hook before options are set (new to v1.26):
1811 \glslink@presetkeys

  Set options.
1812 \setkeys{glslink}{#1}%

  User hook after options are set:
1813 \glslink@postsetkeys

  Check thevalue and theHvalue before saving (v1.19).
1814 \ifdefempty{\@gls@tr@thevalue}%
1815 {%
1816   \@gls@save@entry@counter
1817 }%
1818 {%
1819   \let\the@gl@entry@counter\@gls@tr@thevalue
1820   \def\the@H@gl@entry@counter{\@gls@tr@theHvalue}%
1821 }%
1822 \@gls@set@sort{\glslabel}%

  Check textformat attribute (new to v1.21).
1823 \gls@has@attribute{\glslabel}{textformat}%
1824 {%
1825   \edef\@gls@tr@attrval{\gls@get@attribute{\glslabel}{textformat}}%
1826   \ifcsdef{\@gls@tr@attrval}%

```

```

1827     {%
1828         \letcs{\@glsxtr@textformat}{\@glsxtr@attrval}%
1829     }%
1830     {%
1831         \GlossariesExtraWarning{Unknown control sequence name
1832         '\@glsxtr@attrval' supplied in textformat attribute
1833         for entry '\glslabel'. Reverting to default \string\glstextformat}%
1834         \let\@glsxtr@textformat\glstextformat
1835     }%
1836 }%
1837 {%
1838     \let\@glsxtr@textformat\glstextformat
1839 }%

```

Do write if it should occur before the link text:

```

1840 \ifglsxtrinitwrglossbefore
1841 \do@wrglossary{#2}%
1842 \fi

```

Do the link text:

```

1843 \ifKV@glslink@hyper
1844 \ifglsxtr@hyperoutside
1845     \@glslink{\glolinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
1846 \else
1847     \@glsxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
1848 \fi
1849 \else
1850 \ifglsxtr@hyperoutside
1851     \glsdonohyperlink{\glolinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
1852 \else
1853     \@glsxtr@textformat{\glsdonohyperlink{\glolinkprefix\glslabel}{#3}}%
1854 \fi
1855 \fi

```

Do write if it should occur after the link text:

```

1856 \ifglsxtrinitwrglossbefore
1857 \else
1858     \do@wrglossary{#2}%
1859 \fi

```

As the original definition:

```

1860 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
1861 }

```

```

1862 \define@key{glossadd}{thevalue}{\def\@glsxtr@thevalue{#1}}

```

```

1863 \define@key{glossadd}{theHvalue}{\def\@glsxtr@theHvalue{#1}}

```

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

```

1864 \renewrobustcmd*{\glsadd}[2][\@glsxtr@record]{%
1865     \glsxtrifinmark

```

```

1866 {}%
1867 {%
1868   \@gls@adjustmode
1869   \@glsxtr@record{#1}{#2}{glossadd}%
1870   \glsdoifexists{#2}%
1871   {%
1872     \let\@glsnumberformat\@glsxtr@defaultnumberformat
1873     \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
1874     \def\@glsxtr@thevalue{}%
1875     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
1876     \setkeys{glossadd}{#1}%
1877     \ifdefempty{\@glsxtr@thevalue}%
1878     {%
1879       \@gls@saveentrycounter
1880     }%
1881     {%
1882       \let\theglentrycounter\@glsxtr@thevalue
1883       \def\theHglentrycounter{\@glsxtr@theHvalue}%
1884     }%

```

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

```

1885   \@gls@setsort{#2}%
1886   \@do@wrglossary{#2}%
1887 }%
1888 }%
1889 }

```

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

```

1890 \newcommand*{\@glsxtr@field@linkdefs}{%
1891   \let\glsxtrifwasfirstuse\@secondoftwo
1892   \let\glsifplural\@secondoftwo
1893   \let\glscapscase\@firstofthree
1894   \let\glsinsert\@empty
1895 }

```

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

`assignfieldfont`

```

1896 \newcommand*{\glsxtrassignfieldfont}[1]{%
1897   \ifglentryexists{#1}%
1898   {%
1899     \ifglshasshort{#1}%
1900     {%
1901       \glssetabbrvfmt{\glscategory{#1}}%
1902       \glsifregular{#1}%
1903       {\let\@gls@field@font\glsxtrregularfont}%
1904       {\let\@gls@field@font\@firstofone}%
1905     }%
1906     {%

```

```

1907      \glsifnotregular{#1}%
1908      {\let\@gls@field@font\@firstofone}%
1909      {\let\@gls@field@font\glsxtrregularfont}%
1910      }%
1911  }%
1912  {%
1913      \let\@gls@field@font\@gobble
1914  }%
1915 }

```

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

```

1916 \def\@gls@text@#1#2[#3]{%
1917   \glsxtrassignfieldfont{#2}%
1918   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1919 }

```

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

```

1920 \def\@GLStext@#1#2[#3]{%
1921   \glsxtrassignfieldfont{#2}%
1922   \@gls@field@link{\let\gls@caps@case\@thirdofthree}{#1}{#2}%
1923   {\@gls@field@font{\GLS@accesstext{#2}\mfirstucMakeUppercase{#3}}}%
1924 }

```

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

```

1925 \def\@Gls@text@#1#2[#3]{%
1926   \glsxtrassignfieldfont{#2}%
1927   \@gls@field@link{\let\gls@caps@case\@secondofthree}{#1}{#2}%
1928   {\@gls@field@font{\Gls@accesstext{#2}#3}}%
1929 }

```

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

`\@gls@checknohyperfirst`

```

1930 \newcommand*\@gls@checknohyperfirst{[1]{%
1931   \glsifattribute{#1}{nohyperfirst}{true}{\KV@gls@link@hyperfalse}{}}%
1932 }

```

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

```

1933 \def\@glsfirst@#1#2[#3]{%
1934   \glsxtrassignfieldfont{#2}%
1935   \@gls@field@link
1936   [\let\glsxtrifwasfirstuse\@firstoftwo
1937   \glsxtrchecknohyperfirst{#2}%
1938   ]{#1}{#2}%
1939   {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1940 }

```

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

```
1941 \def\@Glsfirst@#1#2[#3]{%
1942   \glstrassignfieldfont{#2}%
      Ensure that \Glsfirst honours the nohyperfirst attribute.
1943   \@gls@field@link
1944   [\let\glstrifwasfirstuse\@firstoftwo
1945   \let\glscapscase\@secondofthree
1946   \glstrchecknohyperfirst{#2}%
1947   ]%
1948   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1949 }
```

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

```
1950 \def\@GLSfirst@#1#2[#3]{%
1951   \glstrassignfieldfont{#2}%
      Ensure that \GLSfirst honours the nohyperfirst attribute.
1952   \@gls@field@link
1953   [\let\glstrifwasfirstuse\@firstoftwo
1954   \let\glscapscase\@thirdofthree
1955   \glstrchecknohyperfirst{#2}%
1956   ]%
1957   {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1958 }
```

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

```
1959 \def\@glsplural@#1#2[#3]{%
1960   \glstrassignfieldfont{#2}%
1961   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1962   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1963 }
```

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

```
1964 \def\@Glsplural@#1#2[#3]{%
1965   \glstrassignfieldfont{#2}%
1966   \@gls@field@link
1967   [\let\glsifplural\@firstoftwo
1968   \let\glscapscase\@secondofthree
1969   ]%
1970   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1971 }
```

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

```
1972 \def\@GLSplural@#1#2[#3]{%
1973   \glstrassignfieldfont{#2}%
1974   \@gls@field@link
1975   [\let\glsifplural\@firstoftwo
1976   \let\glscapscase\@thirdofthree
```

```

1977 ]%
1978 {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1979 }

```

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

```

1980 \def\@glsfirstplural@#1#2[#3]{%
1981   \glstrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
1982   \@gls@field@link
1983   [\let\glstrifwasfirstuse\@firstoftwo
1984     \let\glsifplural\@firstoftwo
1985     \glstrchecknohyperfirst{#2}%
1986   ]%
1987   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1988 }

```

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

```

1989 \def\@Glsfirstplural@#1#2[#3]{%
1990   \glstrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
1991   \@gls@field@link
1992   [\let\glstrifwasfirstuse\@firstoftwo
1993     \let\glsifplural\@firstoftwo
1994     \let\glscapscase\@secondofthree
1995     \glstrchecknohyperfirst{#2}%
1996   ]%
1997   {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1998 }

```

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

```

1999 \def\@GLSfirstplural@#1#2[#3]{%
2000   \glstrassignfieldfont{#2}%
      Ensure that \glsfirstplural honours the nohyperfirst attribute.
2001   \@gls@field@link
2002   [\let\glstrifwasfirstuse\@firstoftwo
2003     \let\glsifplural\@firstoftwo
2004     \let\glscapscase\@thirdofthree
2005     \glstrchecknohyperfirst{#2}%
2006   ]%
2007   {#1}{#2}%
2008   {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2009 }

```

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

```

2010 \def\@glsname@#1#2[#3]{%
2011   \glstrassignfieldfont{#2}%
2012   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2013 }

```

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

```
2014 \def\@Glsname@#1#2[#3]{%
2015   \glstrassignfieldfont{#2}%
2016   \@gls@field@link
2017   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2018   {\@gls@field@font{\Glsaccessname{#2}#3}}%
2019 }
```

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

```
2020 \def\@GLSname@#1#2[#3]{%
2021   \glstrassignfieldfont{#2}%
2022   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2023   {#1}{#2}%
2024   {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2025 }
```

\@glsdesc@

```
2026 \def\@glsdesc@#1#2[#3]{%
2027   \glstrassignfieldfont{#2}%
2028   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2029 }
```

\@Glsdesc@ First letter uppercase version.

```
2030 \def\@Glsdesc@#1#2[#3]{%
2031   \glstrassignfieldfont{#2}%
2032   \@gls@field@link
2033   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2034   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2035 }
```

\@GLSdesc@ All uppercase version.

```
2036 \def\@GLSdesc@#1#2[#3]{%
2037   \glstrassignfieldfont{#2}%
2038   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2039   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2040 }
```

@glsdescplural@ No case-changing version.

```
2041 \def\@glsdescplural@#1#2[#3]{%
2042   \glstrassignfieldfont{#2}%
2043   \@gls@field@link
2044   [\let\glscapscase\@secondoftwo
2045   \let\glsifplural\@firstoftwo
2046   ]{#1}{#2}{\@gls@field@font{\glsaccessdescplural{#2}#3}}%
2047 }
```

@Glsdescplural@ First letter uppercase version.

```
2048 \def\@Glsdescplural@#1#2[#3]{%
```



```

2049 \glstrassignfieldfont{#2}%
2050 \@gls@field@link
2051 [\let\glscapscase\@secondoftwo
2052 \let\glsifplural\@firstoftwo
2053 ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2054 }

```

@GLSdescplural@ All uppercase version.

```

2055 \def\@GLSdesc@#1#2[#3]{%
2056 \glstrassignfieldfont{#2}%
2057 \@gls@field@link
2058 [\let\glscapscase\@thirdoftwo
2059 \let\glsifplural\@firstoftwo
2060 ]%
2061 {#1}{#2}%
2062 {\@gls@field@font{\Glsaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2063 }

```

\@glssymbol@

```

2064 \def\@glssymbol@#1#2[#3]{%
2065 \glstrassignfieldfont{#2}%
2066 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2067 }

```

\@GLssymbol@ First letter uppercase version.

```

2068 \def\@GLssymbol@#1#2[#3]{%
2069 \glstrassignfieldfont{#2}%
2070 \@gls@field@link
2071 [\let\glscapscase\@secondoftwo]%
2072 {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2073 }

```

\@GLSsymbol@ All uppercase version.

```

2074 \def\@GLSsymbol@#1#2[#3]{%
2075 \glstrassignfieldfont{#2}%
2076 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2077 {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2078 }

```

lssymbolplural@ No case-changing version.

```

2079 \def\@glssymbolplural@#1#2[#3]{%
2080 \glstrassignfieldfont{#2}%
2081 \@gls@field@link
2082 [\let\glscapscase\@secondoftwo
2083 \let\glsifplural\@firstoftwo
2084 ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2085 }

```

lssymbolplural@ First letter uppercase version.

```
2086 \def\@Glsymbolplural@#1#2[#3]{%
2087   \glstrassignfieldfont{#2}%
2088   \@gls@field@link
2089   [\let\glscapscase\@secondoftwo
2090   \let\glsifplural\@firstoftwo
2091   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2092 }
```

LSsymbolplural@ All uppercase version.

```
2093 \def\@GLSymbol@#1#2[#3]{%
2094   \glstrassignfieldfont{#2}%
2095   \@gls@field@link
2096   [\let\glscapscase\@thirdoftwo
2097   \let\glsifplural\@firstoftwo
2098   ]%
2099   {#1}{#2}%
2100   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2101 }
```

\@Glsuseri@ First letter uppercase version.

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

\@GLSuseri@ All uppercase version.

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

\@Glsuserii@ First letter uppercase version.

```
2113 \def\@Glsuserii@#1#2[#3]{%
2114   \glstrassignfieldfont{#2}%
2115   \@gls@field@link
2116   [\let\glscapscase\@secondoftwo]%
2117   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2118 }
```

\@GLSuserii@ All uppercase version.

```
2119 \def\@GLSuserii@#1#2[#3]{%
2120   \glstrassignfieldfont{#2}%
2121   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2122   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glsentryuserii{#2}#3}}}%
2123 }
```

\@Glsuseriii@ First letter uppercase version.

```
2124 \def\@Glsuseriii@#1#2[#3]{%
2125   \glstrassignfieldfont{#2}%
2126   \@gls@field@link
2127   [\let\glscapscase\@secondoftwo]%
2128   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2129 }
```

\@GLSuseriii@ All uppercase version.

```
2130 \def\@GLSuseriii@#1#2[#3]{%
2131   \glstrassignfieldfont{#2}%
2132   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2133   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2134 }
```

\@Glsuseriv@ First letter uppercase version.

```
2135 \def\@Glsuseriv@#1#2[#3]{%
2136   \glstrassignfieldfont{#2}%
2137   \@gls@field@link
2138   [\let\glscapscase\@secondoftwo]%
2139   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2140 }
```

\@GLSuseriv@ All uppercase version.

```
2141 \def\@GLSuseriv@#1#2[#3]{%
2142   \glstrassignfieldfont{#2}%
2143   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2144   {#1}{#2}%
2145   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2146 }
```

\@Glsuserv@ First letter uppercase version.

```
2147 \def\@Glsuserv@#1#2[#3]{%
2148   \glstrassignfieldfont{#2}%
2149   \@gls@field@link
2150   [\let\glscapscase\@secondoftwo]%
2151   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}%
2152 }
```

\@GLSuserv@ All uppercase version.

```
2153 \def\@GLSuserv@#1#2[#3]{%
2154   \glstrassignfieldfont{#2}%
2155   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2156   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2157 }
```

\@Glsuservi@ First letter uppercase version.

```
2158 \def\@Glsuservi@#1#2[#3]{%
```

```

2159 \glstrassignfieldfont{#2}%
2160 \@gls@field@link
2161 [\let\glscapscase\@secondoftwo]%
2162 {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
2163 }

```

\@GLSuservi@ All uppercase version.

```

2164 \def\@GLSuservi@#1#2[#3]{%
2165 \glstrassignfieldfont{#2}%
2166 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2167 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2168 }

```

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

\@acrshort No case change.

```

2169 \def\@acrshort#1#2[#3]{%
2170 \glsoifexists{#2}%
2171 {%
2172 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2173 \let\glstrifwasfirstuse\@secondoftwo
2174 \let\glsifplural\@secondoftwo
2175 \let\glscapscase\@firstofthree
2176 \let\glsinsert\@empty
2177 \def\glscustomtext{%
2178 \acronymfont{\Glsaccesssshort{#2}}#3%
2179 }%
2180 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2181 }%
2182 \glspostlinkhook
2183 }

```

\@Acrshort First letter uppercase.

```

2184 \def\@Acrshort#1#2[#3]{%
2185 \glsoifexists{#2}%
2186 {%
2187 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2188 \let\glstrifwasfirstuse\@secondoftwo
2189 \let\glsifplural\@secondoftwo
2190 \let\glscapscase\@secondofthree
2191 \let\glsinsert\@empty
2192 \def\glscustomtext{%
2193 \Glsaccesssshort{#2}}#3%
2194 }%
2195 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2196 }%
2197 \glspostlinkhook
2198 }

```

\@ACRshort All uppercase.

```
2199 \def\@ACRshort#1#2[#3]{%
2200   \glsdoifexists{#2}%
2201   {%
2202     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2203     \let\glxtrifwasfirstuse\@secondoftwo
2204     \let\gl@sifplural\@secondoftwo
2205     \let\glscapscase\@thirdofthree
2206     \let\gl$insert\@empty
2207     \def\glscustomtext{%
2208       \mfirstucMakeUppercase{\acronymfont{\gl@saccesssshort{#2}}#3}%
2209     }%
2210     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2211   }%
2212   \glspostlinkhook
2213 }
```

\@acrshortpl No case change.

```
2214 \def\@acrshortpl#1#2[#3]{%
2215   \glsdoifexists{#2}%
2216   {%
2217     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2218     \let\glxtrifwasfirstuse\@secondoftwo
2219     \let\gl@sifplural\@firstoftwo
2220     \let\glscapscase\@firstofthree
2221     \let\gl$insert\@empty
2222     \def\glscustomtext{%
2223       \acronymfont{\gl@saccesssshortpl{#2}}#3%
2224     }%
2225     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2226   }%
2227   \glspostlinkhook
2228 }
```

\@Acrshortpl First letter uppercase.

```
2229 \def\@Acrshortpl#1#2[#3]{%
2230   \glsdoifexists{#2}%
2231   {%
2232     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2233     \let\glxtrifwasfirstuse\@secondoftwo
2234     \let\gl@sifplural\@firstoftwo
2235     \let\glscapscase\@secondofthree
2236     \let\gl$insert\@empty
2237     \def\glscustomtext{%
2238       \acronymfont{\Glsaccesssshortpl{#2}}#3%
2239     }%
2240     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2241   }%
2242   \glspostlinkhook
```

2243 }

\@ACRshortpl All uppercase.

```
2244 \def\@ACRshortpl#1#2[#3]{%
2245   \glsdoifexists{#2}%
2246   {%
2247     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2248     \let\glxtrifwasfirstuse\@secondoftwo
2249     \let\glsifplural\@firstoftwo
2250     \let\glscapscase\@thirdofthree
2251     \let\glsinsert\@empty
2252     \def\glscustomtext{%
2253       \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2254     }%
2255     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2256   }%
2257   \glspostlinkhook
2258 }
```

\@acrlong No case change.

```
2259 \def\@acrlong#1#2[#3]{%
2260   \glsdoifexists{#2}%
2261   {%
2262     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2263     \let\glxtrifwasfirstuse\@secondoftwo
2264     \let\glsifplural\@secondoftwo
2265     \let\glscapscase\@firstofthree
2266     \let\glsinsert\@empty
2267     \def\glscustomtext{%
2268       \acronymfont{\glsaccesslong{#2}}#3%
2269     }%
2270     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2271   }%
2272   \glspostlinkhook
2273 }
```

\@Acrlong First letter uppercase.

```
2274 \def\@Acrlong#1#2[#3]{%
2275   \glsdoifexists{#2}%
2276   {%
2277     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2278     \let\glxtrifwasfirstuse\@secondoftwo
2279     \let\glsifplural\@secondoftwo
2280     \let\glscapscase\@secondofthree
2281     \let\glsinsert\@empty
2282     \def\glscustomtext{%
2283       \acronymfont{\Glsaccesslong{#2}}#3%
2284     }%
2285     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2286   }
```

```

2286 }%
2287 \glspostlinkhook
2288 }

```

\@ACRlong All uppercase.

```

2289 \def\@ACRlong#1#2[#3]{%
2290 \glsdoidexists{#2}%
2291 {%
2292 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2293 \let\glxtrifwasfirstuse\@secondoftwo
2294 \let\gl@sifplural\@secondoftwo
2295 \let\glscapscase\@thirdofthree
2296 \let\gl$insert\@empty
2297 \def\glscustomtext{%
2298 \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2299 }%
2300 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2301 }%
2302 \glspostlinkhook
2303 }

```

\@acrlongpl No case change.

```

2304 \def\@acrlongpl#1#2[#3]{%
2305 \glsdoidexists{#2}%
2306 {%
2307 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2308 \let\glxtrifwasfirstuse\@secondoftwo
2309 \let\gl@sifplural\@firstoftwo
2310 \let\glscapscase\@firstofthree
2311 \let\gl$insert\@empty
2312 \def\glscustomtext{%
2313 \acronymfont{\gl@saccesslongpl{#2}}#3%
2314 }%
2315 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2316 }%
2317 \glspostlinkhook
2318 }

```

\@Acrlongpl First letter uppercase.

```

2319 \def\@Acrlongpl#1#2[#3]{%
2320 \glsdoidexists{#2}%
2321 {%
2322 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2323 \let\glxtrifwasfirstuse\@secondoftwo
2324 \let\gl@sifplural\@firstoftwo
2325 \let\glscapscase\@secondofthree
2326 \let\gl$insert\@empty
2327 \def\glscustomtext{%
2328 \acronymfont{\Glsaccesslongpl{#2}}#3%

```

```

2329 }%
2330 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2331 }%
2332 \glspostlinkhook
2333 }

```

\@ACRlongpl All uppercase.

```

2334 \def\@ACRlongpl#1#2[#3]{%
2335 \glsdoifexists{#2}%
2336 {%
2337 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2338 \let\glstrifwasfirstuse\@secondoftwo
2339 \let\glsifplural\@firstoftwo
2340 \let\glscapscase\@thirdofthree
2341 \let\glsinsert\@empty
2342 \def\glscustomtext{%
2343 \mfirstucMakeUppercase{\acronymfont{\glsaccesslongpl{#2}}#3}%
2344 }%
2345 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2346 }%
2347 \glspostlinkhook
2348 }

```

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

\@glsaddkey

```

2349 \renewcommand*{\@glsaddkey}[7]{%
2350 \key@ifundefined{glossentry}{#1}%
2351 {%
2352 \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2353 \appto\@gls@keymap{,{#1}{#1}}%
2354 \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2355 \appto\@newglossaryentryposthook{%
2356 \letcs{\@glo@tmp}{@glo@#1}%
2357 \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2358 }%
2359 \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2360 \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2361 \ifcsdef{@gls@user@#1@}%
2362 {%
2363 \PackageError{glossaries}%
2364 {Can't define '\string#5' as helper command
2365 '\expandafter\string\csname @gls@user@#1@\endcsname' already
2366 exists}%
2367 }%
2368 }%
2369 {%

```



```

2370 \expandafter\newcommand\expandafter*\expandafter
2371 {\csname @gls@user@#1\endcsname}[2][\%
2372 \new@ifnextchar[
2373 {\csuse{@gls@user@#1@}{##1}{##2}}%
2374 {\csuse{@gls@user@#1@}{##1}{##2}[]}}%
2375 \csdef{@gls@user@#1@}##1##2[##3]{%
2376 \@gls@field@link{##1}{##2}{#3{##2}##3}%
2377 }%
2378 \newrobustcmd*{#5}{%
2379 \expandafter\@gls@hyp@opt\csname @gls@user@#1\endcsname}%
2380 }%

```

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

```

2381 \ifcsdef{@Gls@user@#1@}%
2382 {%
2383 \PackageError{glossaries}%
2384 {Can't define '\string#6' as helper command
2385 '\expandafter\string\csname @Gls@user@#1\endcsname' already
2386 exists}%
2387 }%
2388 }%
2389 {%
2390 \expandafter\newcommand\expandafter*\expandafter
2391 {\csname @Gls@user@#1\endcsname}[2][\%
2392 \new@ifnextchar[
2393 {\csuse{@Gls@user@#1@}{##1}{##2}}%
2394 {\csuse{@Gls@user@#1@}{##1}{##2}[]}}%
2395 \csdef{@Gls@user@#1@}##1##2[##3]{%
2396 \@gls@field@link[\let\glscaps@case\@secondofthree]%
2397 {##1}{##2}{#4{##2}##3}%
2398 }%
2399 \newrobustcmd*{#6}{%
2400 \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2401 }%

```

Finally the all caps version (modified):

```

2402 \ifcsdef{@GLS@user@#1@}%
2403 {%
2404 \PackageError{glossaries}%
2405 {Can't define '\string#7' as helper command
2406 '\expandafter\string\csname @GLS@user@#1\endcsname' already
2407 exists}%
2408 }%
2409 }%
2410 {%
2411 \expandafter\newcommand\expandafter*\expandafter
2412 {\csname @GLS@user@#1\endcsname}[2][\%
2413 \new@ifnextchar[
2414 {\csuse{@GLS@user@#1@}{##1}{##2}}%
2415 {\csuse{@GLS@user@#1@}{##1}{##2}[]}}%

```

```

2416 \csdef{@GLS@user@#1@}##1##2[##3]{%
2417 \@gls@field@link[\let\glscapscase\@thirdofthree]%
2418 {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}}%
2419 }%
2420 \newrobustcmd*{#7}{%
2421 \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2422 }%
2423 }%
2424 {%
2425 \PackageError{glossaries-extra}{Key ‘#1’ already exists}{}%
2426 }%
2427 }

```

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

```

2428 \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.

```

2429 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2430 \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.

```

2431 \ifglsused{\glslabel}%
2432 {\let\glsxtrifwasfirstuse\@secondoftwo}
2433 {\let\glsxtrifwasfirstuse\@firstoftwo}%

```

Store the category label for convenience.

```

2434 \edef\glscategorylabel{\glscategory{\glslabel}}%
2435 \ifglsused{\glslabel}%
2436 {%
2437 \glsifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2438 {\KV@glslink@hyperfalse}}}%
2439 }%
2440 {%
2441 \glsifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2442 {\KV@glslink@hyperfalse}}}%
2443 }%
2444 \glslinkcheckfirsthyperhook
2445 }

```

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.

```

2446 \ifdef\do@gl:disablehyperinlist
2447 {%
2448 \let\@glsxtr@do@gl:disablehyperinlist\do@gl:disablehyperinlist
2449 \renewcommand*{\do@gl:disablehyperinlist}{%

```

```

2450 \glstr@do@gl:disablehyperinlist
2451 \glstrifattribute{\glslabel}{nohyper}{true}{\KV@glslink@hyperfalse}{}%
2452 }
2453 }
2454 {}

```

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

```

2455 \define@boolkey{glslink}{noindex}[true]{}
2456 \KV@glslink@noindexfalse

```

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

`lt@glslink@opts`

```

2457 \ifdef\@glstr@setdefault@glslink@opts
2458 {
2459 \renewcommand*{\@glstr@setdefault@glslink@opts}{%
2460 \KV@glslink@noindexfalse
2461 \@glstrsetaliasnoindex
2462 }
2463 }
2464 {

```

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

```

2465 \newcommand*{\@glstr@setdefault@glslink@opts}{%
2466 \KV@glslink@noindexfalse
2467 \@glstrsetaliasnoindex
2468 }
2469 \preto\do@gl:disablehyperinlist{\@glstr@setdefault@glslink@opts}
2470 }

```

`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.)

```

2471 \providecommand*{\glstrsetaliasnoindex}{%
2472 \KV@glslink@noindextrue
2473 }

```

`setaliasnoindex`

```

2474 \newcommand*{\@glstrsetaliasnoindex}{%
2475 \glstrifhasfield{alias}{\glslabel}%
2476 {%
2477 \let\glstrindexaliased\@glstrindexaliased
2478 \glstrsetaliasnoindex
2479 \let\glstrindexaliased\@no@glstrindexaliased
2480 }%
2481 {}%
2482 }

```

xtrindexaliased

```
2483 \newcommand{\@glxtrindexaliased}{%
2484   \ifKV@glslink@noindex
2485   \else
2486     \begingroup
2487     \let\@glslnumberformat\@glxtr@defaultnumberformat
2488     \edef\@glsl@counter{\csname glo@\glsdetoklabel{\glslabel}@counter\endcsname}%
2489     \glxtr@saveentrycounter
2490     \@@do@wrglossary{\glxtralias{\glslabel}}%
2491     \endgroup
2492   \fi
2493 }
```

xtrindexaliased

```
2494 \newcommand{\@no@glxtrindexaliased}{%
2495   \PackageError{glossaries-extra}{\string\glxtrindexaliased\space
2496     not permitted outside definition of \string\glxtrsetaliasnoindex}%
2497   {}%
2498 }
```

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

```
2499 \let\glxtrindexaliased\@no@glxtrindexaliased
```

tDefaultGlsOpts Set the default options for \glslink etc.

```
2500 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2501   \renewcommand*{\@glsl@setdefault@glslink@opts}{%
2502     \setkeys{glslink}{#1}%
2503     \@glxtrsetaliasnoindex
2504   }%
2505 }
```

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

```
2506 \newcommand*{\glxtrifindexing}[2]{%
2507   \ifKV@glslink@noindex #2\else #1\fi
2508 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2509 \renewcommand*{\glswriteentry}[2]{%
2510   \glxtrifindexing
2511   {%
2512     \ifglslindexonlyfirst
2513       \ifglslused{#1}
2514       {\glxtrdoautoindexname{#1}{dualindex}}%
2515       {#2}%
2516     \else
2517       \glslifattribute{#1}{indexonlyfirst}{true}%
2518       {\ifglslused{#1}
2519         {\glxtrdoautoindexname{#1}{dualindex}}%

```

```

2520      {\#2}}%
2521      {\#2}%
2522      \fi
2523    }%
2524    {}%
2525  }

```

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

```

2526 \appto@@do@@wrglossary{\@glxtr@do@@wrindex
2527   \glxtrdowrglossaryhook{\@gls@label}}%
2528 }

```

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

Similarly for the “noidx” version:

`s@noidxglossary`

```

2529 \appto\gls@noidxglossary{\@glxtr@do@@wrindex
2530   \glxtrdowrglossaryhook{\@gls@label}}%
2531 }

```

`xtr@do@@wrindex`

```

2532 \newcommand*{\@glxtr@do@@wrindex}{%
2533   \glxtrdoautoindexname{\@gls@label}{dualindex}}%
2534 }

```

`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.)

```

2535 \newcommand*{\glxtrdowrglossaryhook}[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.

```

2536 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2537   \let\glslinkvar\@firstofthree
2538   \let\@gls@hyp@opt@cs#1\relax
2539   \@ifstar{\s@gls@hyp@opt}%
2540   {\@ifnextchar+%
2541     {\@firstoftwo{\p@gls@hyp@opt}}}%
2542   {%
2543     \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2544     {\@firstoftwo{\@alt@gls@hyp@opt}}}%
2545     {\#1}%
2546   }%
2547 }%
2548 }

```

`alt@gls@hyp@opt` User version

```

2549 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%

```

```

2550 \let\glslinkvar\@firstofthree
2551 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}

lt@hyp@opt@char Contains the character used as the command modifier.
2552 \newcommand*{\@gls@alt@hyp@opt@char}{}

lt@hyp@opt@keys Contains the option list used as the command modifier.
2553 \newcommand*{\@gls@alt@hyp@opt@keys}{}

rSetAltModifier
2554 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2555   \let\@gls@hyp@opt\@gls@alt@hyp@opt
2556   \def\@gls@alt@hyp@opt@char{#1}%
2557   \def\@gls@alt@hyp@opt@keys{#2}%
2558 }

org@dohyperlink
2559 \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.
2560 \ifdef\glsnavhyperlink
2561 {
2562   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
2563     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
2564     {%
2565       \let\glsdohyperlink\glsxtr@org@dohyperlink
2566       \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2567     }%
2568   }%
2569 }
2570 {}

\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 tem-
porarily makes \gls behave like \glstext[hyper=false,noindex]. (This will be overrid-
den 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.
2571 \renewcommand*{\glsdohyperlink}[2]{%
2572   \glsattribute{\glslabel}{targeturl}%
2573   {%

```

```

2574 \glshasattribute{\glslabel}{targetname}%
2575 {%
2576   \glshasattribute{\glslabel}{targetcategory}%
2577   {%
2578     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2579       {\glsgetattribute{\glslabel}{targetcategory}}}%
2580     {\glsgetattribute{\glslabel}{targetname}}}%
2581     {{\glsxtrprotectlinks#2}}}%
2582   }%
2583   {%
2584     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2585       {}}%
2586       {\glsgetattribute{\glslabel}{targetname}}}%
2587       {{\glsxtrprotectlinks#2}}}%
2588   }%
2589 }%
2590 {%
2591   \href{\glsgetattribute{\glslabel}{targeturl}}{%
2592     {{\glsxtrprotectlinks#2}}}%
2593   }%
2594 }%
2595 {%

```

Check for alias.

```

2596 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2597 \ifdefvoid\gloaliaslabel
2598 {%
2599   \glsxtrhyperlink{#1}{{\glsxtrprotectlinks#2}}}%
2600 }%
2601 {%

```

Redirect link to the alias target.

```

2602 \glsxtrhyperlink
2603 {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2604 {{\glsxtrprotectlinks#2}}}%
2605 }%
2606 }%
2607 }

```

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

```

2608 \ifdef\@glsshowtarget
2609 {
2610   \newcommand{\glsxtrhyperlink}[2]{%
2611     \@glsshowtarget{#1}%
2612     \hyperlink{#1}{#2}%
2613   }%
2614 }
2615 {
2616   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2617 }

```

`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.

```
2618 \renewrobustcmd*{\glsdohyperlink}[2][\glstext{\@glo@label}]{%
2619   \glsdoifexists{#2}%
2620   {%
2621     \def\@glo@label{#2}%
2622     {\edef\glslabel{#2}%
2623       \@glslink{\glo@linkprefix\glslabel}{#1}}%
2624   }%
2625 }
```

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

```
2626 \renewcommand{\glsdisablehyper}{%
2627   \KV@glslink@hyperfalse
2628   \def\@glslink{\glsdonohyperlink}%
2629   \let\@glstarget\@secondoftwo
2630 }
```

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

```
2631 \renewcommand{\glsenablehyper}{%
2632   \KV@glslink@hypertrue
2633   \def\@glslink{\glsdohyperlink}%
2634   \def\@glstarget{\glsdohypertarget}%
2635 }
```

`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.

```
2636 \def\glsdonohyperlink#1#2{{\glstxtprotectlinks #2}}
```

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

```
2637 \ifcsundef{hyperlink}%
2638 {%
2639   \def\@glslink{\glsdonohyperlink}
2640 }%
2641 {%
2642   \def\@glslink{\glsdohyperlink}
2643 }
```

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

```
2644 \newcommand*{\glstxtprotectlinks}{%
2645   \KV@glslink@hyperfalse
```



```

2646 \KV@glslink@noindextrue
2647 \let\@gls@\@glsxtr@p@text@
2648 \let\@Gls@\@Glsxtr@p@text@
2649 \let\@GLS@\@GLSxtr@p@text@
2650 \let\@glspl@\@glsxtr@p@plural@
2651 \let\@Glspl@\@Glsxtr@p@plural@
2652 \let\@GLSpl@\@GLSxtr@p@plural@
2653 \let\@glsxtrshort\@glsxtr@p@short@
2654 \let\@Glsxtrshort\@Glsxtr@p@short@
2655 \let\@GLSxtrshort\@GLSxtr@p@short@
2656 \let\@glsxtrlong\@glsxtr@p@long@
2657 \let\@Glsxtrlong\@Glsxtr@p@long@
2658 \let\@GLSxtrlong\@GLSxtr@p@long@
2659 \let\@glsxtrshortpl\@glsxtr@p@shortpl@
2660 \let\@Glsxtrshortpl\@Glsxtr@p@shortpl@
2661 \let\@GLSxtrshortpl\@GLSxtr@p@shortpl@
2662 \let\@glsxtrlongpl\@glsxtr@p@longpl@
2663 \let\@Glsxtrlongpl\@Glsxtr@p@longpl@
2664 \let\@GLSxtrlongpl\@GLSxtr@p@longpl@
2665 \let\@acrshort\@glsxtr@p@acrshort@
2666 \let\@Acrshort\@Glsxtr@p@acrshort@
2667 \let\@ACRshort\@GLSxtr@p@acrshort@
2668 \let\@acrshortpl\@glsxtr@p@acrshortpl@
2669 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
2670 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
2671 \let\@acrlong\@glsxtr@p@acrlong@
2672 \let\@Acrlong\@Glsxtr@p@acrlong@
2673 \let\@ACRlong\@GLSxtr@p@acrlong@
2674 \let\@acrlongpl\@glsxtr@p@acrlongpl@
2675 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
2676 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
2677 }

```

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

@glsxtr@p@text@

```
2678 \def\@glsxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtr@p@text@

```
2679 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtr@p@text@

```
2680 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2681 \def\@glsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtr@p@plural@

```
2682 \def\@Glsxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtr@p@plural@

```
2683 \def\@GLSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glxtr@p@short@

```
2684 \def\@glxtr@p@short@#1#2[#3]{%
2685   {%
2686     \glsetabbrvfmt{\glscategory{#2}}%
2687     \glabbrvfont{\glentryshort{#2}}#3%
2688   }%
2689 }
```

Glsxtr@p@short@

```
2690 \def\@Glsxtr@p@short@#1#2[#3]{%
2691   {%
2692     \glsetabbrvfmt{\glscategory{#2}}%
2693     \glabbrvfont{\Glsentryshort{#2}}#3%
2694   }%
2695 }
```

GLSxtr@p@short@

```
2696 \def\@GLSxtr@p@short@#1#2[#3]{%
2697   {%
2698     \glsetabbrvfmt{\glscategory{#2}}%
2699     \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
2700   }%
2701 }
```

sxtr@p@shortpl@

```
2702 \def\@glxtr@p@shortpl@#1#2[#3]{%
2703   {%
2704     \glsetabbrvfmt{\glscategory{#2}}%
2705     \glabbrvfont{\glentryshortpl{#2}}#3%
2706   }%
2707 }
```

sxtr@p@shortpl@

```
2708 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2709   {%
2710     \glsetabbrvfmt{\glscategory{#2}}%
2711     \glabbrvfont{\Glsentryshortpl{#2}}#3%
2712   }%
2713 }
```

Sxtr@p@shortpl@

```
2714 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2715   {%
2716     \glsetabbrvfmt{\glscategory{#2}}%
2717     \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
2718   }%
2719 }
```

```

2718 }%
2719 }

@glxtr@p@long@
2720 \def\@glxtr@p@long@#1#2[#3]{\glentrylong{#2}#3}}

@Glsxtr@p@long@
2721 \def\@Glsxtr@p@long@#1#2[#3]{\Glsentrylong{#2}#3}}

@GLSxtr@p@long@
2722 \def\@GLSxtr@p@long@#1#2[#3]{%
2723   {\mfirstucMakeUppercase{\glslongfont{\glentrylong{#2}}#3}}}}

lsxtr@p@longpl@
2724 \def\@lsxtr@p@longpl@#1#2[#3]{\glentrylongpl{#2}#3}}

lsxtr@p@longpl@
2725 \def\@Glsxtr@p@longpl@#1#2[#3]{\glslongfont{\Glsentrylongpl{#2}}#3}}

LSxtr@p@longpl@
2726 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2727   {\mfirstucMakeUppercase{\glslongfont{\glentrylongpl{#2}}#3}}}}

xtr@p@acrshort@
2728 \def\@gxtr@p@acrshort@#1#2[#3]{\acronymfont{\glentryshort{#2}}#3}}

xtr@p@acrshort@
2729 \def\@Glsxtr@p@acrshort@#1#2[#3]{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2730 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2731   {\mfirstucMakeUppercase{\acronymfont{\glentryshort{#2}}#3}}}}

r@p@acrshortpl@
2732 \def\@gxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\glentryshortpl{#2}}#3}}

r@p@acrshortpl@
2733 \def\@Glsxtr@p@acrshortpl@#1#2[#3]{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2734 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2735   {\mfirstucMakeUppercase{\acronymfont{\glentryshortpl{#2}}#3}}}}

sxtr@p@acrlong@
2736 \def\@gsxtr@p@acrlong@#1#2[#3]{\glentrylong{#2}#3}}

sxtr@p@acrlong@
2737 \def\@Glsxtr@p@acrlong@#1#2[#3]{\Glsentrylong{#2}#3}}

```

Sxtr@p@acrlong@

```
2738 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2739 {\mfirstucMakeUppercase{\glstentrylong{#2}#3}}}
```

tr@p@acrlongpl@

```
2740 \def\@glstxtr@p@acrlongpl@#1#2[#3]{\glstentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
2741 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\Glsentrylongpl{#2}#3}}
```

tr@p@acrlongpl@

```
2742 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2743 {\mfirstucMakeUppercase{\glstentrylongpl{#2}#3}}}
```

Commands to minimise conflict.

\@glstxtrp@opt

```
2744 \newcommand*{\@glstxtrp@opt}{hyper=false,noindex}
```

\glstxtrsetpopts Used in glossary to switch hyperlinks on for the \glstxtrp type of commands.

```
2745 \newcommand*{\glstxtrsetpopts}[1]{%
2746 \renewcommand*{\@glstxtrp@opt}{#1}%
2747 }
```

lossxtrsetpopts Used in glossary to switch hyperlinks on for the \glstxtrp type of commands.

```
2748 \newcommand*{\glossxtrsetpopts}{%
2749 \glstxtrsetpopts{noindex}%
2750 }
```

\@@glstxtrp

```
2751 \newrobustcmd*{\@@glstxtrp}[2]{%
```

Add scope.

```
2752 {%
2753 \let\glspostlinkhook\relax
2754 \csname#1\expandafter\endcsname\expandafter[\@glstxtrp@opt]{#2}[]%
2755 }%
2756 }
```

\@glstxtrp

```
2757 \newrobustcmd*{\@glstxtrp}[2]{%
2758 \ifcsdef{gls#1}%
2759 {%
2760 \@glstxtrp{gls#1}{#2}%
2761 }%
2762 {%
2763 \ifcsdef{glsxtr#1}%
2764 {%
```

```

2765     \@@glsxtrp{glsxtr#1}{#2}%
2766   }%
2767   {%
2768     \PackageError{glossaries-extra}{‘#1’ not recognised by
2769       \string\glsxtrp}{}%
2770   }%
2771 }%
2772 }

```

\@Glsxtrp

```

2773 \newrobustcmd*{\@Glsxtrp}[2]{%
2774   \ifcsdef{Gls#1}%
2775   {%
2776     \@@glsxtrp{Gls#1}{#2}%
2777   }%
2778   {%
2779     \ifcsdef{Glsxtr#1}%
2780     {%
2781       \@@glsxtrp{Glsxtr#1}{#2}%
2782     }%
2783     {%
2784       \PackageError{glossaries-extra}{‘#1’ not recognised by
2785         \string\Glsxtrp}{}%
2786     }%
2787   }%
2788 }

```

\@GLSxtrp

```

2789 \newrobustcmd*{\@GLSxtrp}[2]{%
2790   \ifcsdef{GLS#1}%
2791   {%
2792     \@@glsxtrp{GLS#1}{#2}%
2793   }%
2794   {%
2795     \ifcsdef{GLSxtr#1}%
2796     {%
2797       \@@glsxtrp{GLSxtr#1}{#2}%
2798     }%
2799     {%
2800       \PackageError{glossaries-extra}{‘#1’ not recognised by
2801         \string\GLSxtrp}{}%
2802     }%
2803   }%
2804 }

```

\glsxtr@entry@p

```

2805 \newrobustcmd*{\glsxtr@headentry@p}[2]{%
2806   \glsifattribute{#1}{headuc}{true}%
2807   {%

```

```

2808 \mfirstucMakeUppercase{\@gls@entry@field{#1}{#2}}%
2809 }%
2810 {%
2811 \@gls@entry@field{#1}{#2}%
2812 }%
2813 }

```

`\glsxtrp` Not robust as it needs to expand somewhat.

```

2814 \ifdef\teorpdfstring
2815 {
2816 \newcommand{\glsxtrp}[2]{%
2817 \protect\NoCaseChange
2818 {%
2819 \protect\teorpdfstring
2820 {%
2821 \protect\glsxtrifinmark
2822 {%
2823 \ifcsdef{glsxtrhead#1}%
2824 {%
2825 {\protect\csuse{glsxtrhead#1}{#2}}%
2826 }%
2827 {%
2828 \glsxtr@headentry@p{#2}{#1}%
2829 }%
2830 }%
2831 {%
2832 \@glsxtrp{#1}{#2}%
2833 }%
2834 }%
2835 {%
2836 \protect\@gls@entry@field{#2}{#1}%
2837 }%
2838 }%
2839 }
2840 }
2841 {
2842 \newcommand{\glsxtrp}[2]{%
2843 \protect\NoCaseChange
2844 {%
2845 \protect\glsxtrifinmark
2846 {%
2847 \ifcsdef{glsxtrhead#1}%
2848 {%
2849 {\protect\csuse{glsxtrhead#1}}%
2850 }%
2851 {%
2852 \glsxtr@headentry@p{#2}{#1}%
2853 }%
2854 }%

```

```

2855      {%
2856      \@glsxtrp{#1}{#2}%
2857      }%
2858      }%
2859  }
2860 }

```

Provide short synonyms for the most common option.

`\glsps`

```
2861 \newcommand*{\glsps}{\glsxtrp{short}}
```

`\glspt`

```
2862 \newcommand*{\glspt}{\glsxtrp{text}}
```

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

```

2863 \ifdef\teorpdfstring
2864 {
2865   \newcommand{\Glsxtrp}[2]{%
2866     \protect\NoCaseChange
2867     {%
2868       \protect\teorpdfstring
2869       {%
2870         \protect\glsxtrifinmark
2871         {%
2872           \ifcsdef{Glsxtrhead#1}%
2873           {%
2874             {\protect\csuse{Glsxtrhead#1}{#2}}%
2875           }%
2876           {%
2877             \protect\@Gls@entry@field{#2}{#1}%
2878           }%
2879         }%
2880         {%
2881           \@Glsxtrp{#1}{#2}%
2882         }%
2883       }%
2884       {%
2885         \protect\@gls@entry@field{#2}{#1}%
2886       }%
2887     }%
2888   }
2889 }
2890 {
2891   \newcommand{\Glsxtrp}[2]{%
2892     \protect\NoCaseChange
2893     {%
2894       \protect\glsxtrifinmark

```

```

2895     {%
2896       \ifcsdef{Glsxtrhead#1}%
2897       {%
2898         {\protect\csuse{Glsxtrhead#1}}%
2899       }%
2900     {%
2901       \protect\@Gls@entry@field{#2}{#1}%
2902     }%
2903   }%
2904   {%
2905     \@Glsxtrp{#1}{#2}%
2906   }%
2907 }%
2908 }
2909 }

```

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

```

2910 \ifdef\texorpdfstring
2911 {
2912   \newcommand{\GLSxtrp}[2]{%
2913     \protect\NoCaseChange
2914     {%
2915       \protect\texorpdfstring
2916       {%
2917         \protect\glsxtrifinmark
2918         {%
2919           \ifcsdef{GLSxtr#1}%
2920           {%
2921             {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2922           }%
2923           {%
2924             \protect\mfirstucMakeUppercase
2925             {%
2926               \protect\@gls@entry@field{#2}{#1}%
2927             }%
2928           }%
2929         }%
2930       }%
2931       \@GLSxtrp{#1}{#2}%
2932     }%
2933   }%
2934   {%
2935     \protect\@gls@entry@field{#2}{#1}%
2936   }%
2937 }%
2938 }
2939 }
2940 {
2941   \newcommand{\GLSxtrp}[2]{%

```



```

2942 \protect\NoCaseChange
2943 {%
2944 \protect\glxstrifinmark
2945 {%
2946 \ifcsdef{GLSxtr#1}%
2947 {%
2948 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2949 }%
2950 {%
2951 \protect\mfirstucMakeUppercase
2952 {%
2953 \protect\@gls@entry@field{#2}{#1}%
2954 }%
2955 }%
2956 }%
2957 {%
2958 \@GLSxtrp{#1}{#2}%
2959 }%
2960 }%
2961 }
2962 }

```

1.3.5 Entry Counting

The (use) entry counting mechanism from glossaries is adjusted here to work with category attributes. Provide a convenient command to enable entry counting, set the entrycount attribute for given categories and redefine \gls etc to use \cgl instead. This form of entry counting is provided to adjust the formatting if the number of times an entry has been used (through commands that unset the first use flag) doesn't exceeding the specified threshold. For link counting, see Section 1.4.

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

```

\@glsunset Global unset.
2963 \renewcommand*{\@glsunset}[1]{%
2964 \@@glsunset{#1}%
2965 \glxstrpostunset{#1}%
2966 }%

glxstrpostunset
2967 \newcommand*{\glxstrpostunset}[1]{%

\@glslocalunset Local unset.
2968 \renewcommand*{\@glslocalunset}[1]{%
2969 \@@glslocalunset{#1}%
2970 \glxstrpostlocalunset{#1}%
2971 }%

rpostlocalunset
2972 \newcommand*{\glxstrpostlocalunset}[1]{%

```

```

\@glsreset   Global reset.
2973 \renewcommand*{\@glsreset}[1]{%
2974   \@glsreset{#1}%
2975   \glsxtrpostreset{#1}%
2976 }%

glsxtrpostreset
2977 \newcommand*{\glsxtrpostreset}[1]{%

\@glslocalreset   Local reset.
2978 \renewcommand*{\@glslocalreset}[1]{%
2979   \@glslocalreset{#1}%
2980   \glsxtrpostlocalreset{#1}%
2981 }%

rpostlocalreset
2982 \newcommand*{\glsxtrpostlocalreset}[1]{%

leEntryCounting   The first argument is the list of categories and the second argument is the value of the en-
                    trycount attribute.
2983 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
    Enable entry counting:
2984   \glsenableentrycount
    Redefine \gls etc:
2985   \renewcommand*{\gls}{\cglsl}%
2986   \renewcommand*{\Gls}{\cGls}%
2987   \renewcommand*{\glspl}{\cglspl}%
2988   \renewcommand*{\Glspl}{\cGlspl}%
2989   \renewcommand*{\GLS}{\cGLS}%
2990   \renewcommand*{\GLSpl}{\cGLSpl}%
    Set the entrycount attribute:
2991   \@glsxtr@setentrycountunsetattr{#1}{#2}%
    In case this command is used again:
2992   \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2993   \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2994     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2995       can't be used with \string\GlsXtrEnableEntryCounting}%
2996     {Use one or other but not both commands}}%
2997 }

ycountunsetattr
2998 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2999   \@for\@glsxtr@cat:=#1\do
3000   {%
3001     \ifdefempty{\@glsxtr@cat}{}%
3002     {%

```

```

3003     \glssetcategoryattribute{\@glstr@cat}{entrycount}{#2}%
3004 }%
3005 }%
3006 }

```

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

nableentrycount

```

3007 \renewcommand*{\glsenableentrycount}{%
    Enable new fields:
3008   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
    Just in case the user has switched on the docdef option.
3009   \renewcommand*{\gls@defdocnewglossaryentry}{%
3010     \renewcommand*{\newglossaryentry[2]}{%
3011       \PackageError{glossaries}{\string\newglossaryentry\space
3012         may only be used in the preamble when entry counting has
3013         been activated}{If you use \string\glsenableentrycount\space
3014         you must place all entry definitions in the preamble not in
3015         the document environment}%
3016     }%
3017 }%

```

New commands to access new fields:

```

3018 \newcommand*{\glstentrycurrrent}[1]{%
3019   \ifcsundef{glo@\glsdetoklabel{##1}@currrent}%
3020   {0}{\@gls@entry@field{##1}{currrent}}}%
3021 }%
3022 \newcommand*{\glstentryprevrent}[1]{%
3023   \ifcsundef{glo@\glsdetoklabel{##1}@prevrent}%
3024   {0}{\@gls@entry@field{##1}{prevrent}}}%
3025 }%

```

Adjust post unset and reset:

```

3026 \let\@glstr@entrycount@org@unset\glstrpostunset
3027 \renewcommand*{\glstrpostunset}[1]{%
3028   \@glstr@entrycount@org@unset{##1}%
3029   \@gls@increment@currrent{##1}%
3030 }%
3031 \let\@glstr@entrycount@org@localunset\glstrpostlocalunset
3032 \renewcommand*{\glstrpostlocalunset}[1]{%
3033   \@glstr@entrycount@org@localunset{##1}%
3034   \@gls@local@increment@currrent{##1}%
3035 }%
3036 \let\@glstr@entrycount@org@reset\glstrpostreset
3037 \renewcommand*{\glstrpostreset}[1]{%
3038   \@glstr@entrycount@org@reset{##1}%
3039   \csgdef{glo@\glsdetoklabel{##1}@currrent}{0}%
3040 }%
3041 \let\@glstr@entrycount@org@localreset\glstrpostlocalreset

```

```

3042 \renewcommand*{\glxtrpostlocalreset}[1]{%
3043   \@glxtr@entrycount@org@localreset{##1}%
3044   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
3045 }%

```

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

```

3046 \let\@cgl@\@cgl@
3047 \let\@cgl@pl@\@cgl@pl@

3048 \let\@cGls@\@cGls@
3049 \let\@cGlspl@\@cGlspl@
3050 \let\@cGLS@\@cGLS@
3051 \let\@cGLSpl@\@cGLSpl@

```

The rest is as the original definition.

```

3052 \AtEndDocument{\@gls@write@entrycounts}%
3053 \renewcommand*{\@gls@entry@count}[2]{%
3054   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
3055 }%
3056 \let\glsenableentrycount\relax
3057 \renewcommand*{\glsenableentryunitcount}{%
3058   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
3059     can't be used with \string\glsenableentrycount}%
3060   {Use one or other but not both commands}%
3061 }%
3062 }

```

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

```

3063 \renewcommand*{\@gls@write@entrycounts}{%
3064   \immediate\write\@auxout
3065     {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
3066   \count@=0\relax
3067   \forallglsentries{\@glsentry}{%
3068     \gls@hasattribute{\@glsentry}{entrycount}%
3069     {%
3070       \ifglsused{\@glsentry}%
3071       {%
3072         \immediate\write\@auxout
3073           {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3074       }%
3075     }%
3076     \advance\count@ by \@ne
3077   }%
3078 }%
3079 }%
3080 \ifnum\count@=0
3081   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3082     \MessageBreak with \string\glsenableentrycount\space but the

```

```

3083     \MessageBreak attribute 'entrycount' hasn't
3084     \MessageBreak been assigned to any of the defined
3085     \MessageBreak entries}%
3086 \fi
3087 }

```

```

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

```

```

3088 \newcommand*\glxtrifcounttrigger}[3]{%
3089   \glshasattribute{#1}{entrycount}%
3090   {%
3091     \ifnum\gl Sentryprevcount{#1}>\gl sgetattribute{#1}{entrycount}\relax
3092     #3%
3093   \else
3094     #2%
3095   \fi
3096 }%
3097 {#3}%
3098 }

```

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

\@@cgl@

```

3099 \def\@@cgl@#1#2[#3]{%
3100   \glxtrifcounttrigger{#2}%
3101   {%
3102     \cgl sformat{#2}{#3}%
3103     \gl sunset{#2}%
3104   }%
3105   {%
3106     \@gls@{#1}{#2}[#3]%
3107   }%
3108 }%

```

\@@cgl spl@

```

3109 \def\@@cgl spl@#1#2[#3]{%
3110   \glxtrifcounttrigger{#2}%
3111   {%
3112     \cgl splformat{#2}{#3}%
3113     \gl sunset{#2}%
3114   }%
3115   {%
3116     \@glspl@{#1}{#2}[#3]%
3117   }%
3118 }%

```

\@@cGls@

```
3119 \def\@@cGls@#1#2[#3]{%
3120   \glstrifcounttrigger{#2}%
3121   {%
3122     \cGlsformat{#2}{#3}%
3123     \glset{#2}%
3124   }%
3125   {%
3126     \@Gls@{#1}{#2}[#3]%
3127   }%
3128 }%
```

\@@cGlspl@

```
3129 \def\@@cGlspl@#1#2[#3]{%
3130   \glstrifcounttrigger{#2}%
3131   {%
3132     \cGlsplformat{#2}{#3}%
3133     \glset{#2}%
3134   }%
3135   {%
3136     \@Glspl@{#1}{#2}[#3]%
3137   }%
3138 }%
```

\@@cGLS@

```
3139 \def\@@cGLS@#1#2[#3]{%
3140   \glstrifcounttrigger{#2}%
3141   {%
3142     \cGLSformat{#2}{#3}%
3143     \glset{#2}%
3144   }%
3145   {%
3146     \@GLS@{#1}{#2}[#3]%
3147   }%
3148 }%
```

\@@cGLSpl@

```
3149 \def\@@cGLSpl@#1#2[#3]{%
3150   \glstrifcounttrigger{#2}%
3151   {%
3152     \cGLSplformat{#2}{#3}%
3153     \glset{#2}%
3154   }%
3155   {%
3156     \@GLSpl@{#1}{#2}[#3]%
3157   }%
3158 }%
```

Remove default warnings from \cglS etc so that it can be used interchangeable with \glS etc.

```

\@cgl@s@
3159 \def\@cgl@s@#1#2[#3]{\@gls@{#1}-{#2} [#3]}

\@cGls@
3160 \def\@cGls@#1#2[#3]{\@Gls@{#1}-{#2} [#3]}

\@cgl spl@
3161 \def\@cgl spl@#1#2[#3]{\@glspl@{#1}-{#2} [#3]}

\@cGlspl@
3162 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}-{#2} [#3]}

Add all upper case versions not provided by glossaries.

\cGLS
3163 \newrobustcmd*{\cGLS}{\@gls@hyp@opt\@cGLS}

\@cGLS Defined the un-starred form. Need to determine if there is a final optional argument
3164 \newcommand*{\@cGLS}[2] [] {%
3165   \new@ifnextchar [\@cGLS@{#1}-{#2}]{\@cGLS@{#1}-{#2} []}%
3166 }

\@cGLS@
3167 \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.
3168 \newcommand*{\cGLSformat}[2] {%
3169   \expandafter\mfirstucMakeUppercase\expandafter{\cgl sformat{#1}-{#2}}%
3170 }

\cGLSpl
3171 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl Defined the un-starred form. Need to determine if there is a final optional argument
3172 \newcommand*{\@cGLSpl}[2] [] {%
3173   \new@ifnextchar [\@cGLSpl@{#1}-{#2}]{\@cGLSpl@{#1}-{#2} []}%
3174 }

\@cGLSpl@
3175 \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.
3176 \newcommand*{\cGLSplformat}[2] {%
3177   \expandafter\mfirstucMakeUppercase\expandafter{\cgl splformat{#1}-{#2}}%
3178 }

```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
3179 \renewcommand*{\cglformat}[2]{%
3180   \glsifregular{#1}
3181   {\glsentryfirst{#1}}%
3182   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%2%
3183 }
```

`\cGlsformat`

```
3184 \renewcommand*{\cGlsformat}[2]{%
3185   \glsifregular{#1}
3186   {\Glsentryfirst{#1}}%
3187   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%2%
3188 }
```

`\cglsplformat`

```
3189 \renewcommand*{\cglsplformat}[2]{%
3190   \glsifregular{#1}
3191   {\glsentryfirstplural{#1}}%
3192   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%2%
3193 }
```

`\cGlsplformat`

```
3194 \renewcommand*{\cGlsplformat}[2]{%
3195   \glsifregular{#1}
3196   {\Glsentryfirstplural{#1}}%
3197   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}%2%
3198 }
```

New code similar to above for unit counting.

`defunitcounters`

```
3199 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3200   \edef\@glo@countunit{\csuse{@glsxtr@categoryattr@@\@glo@category @unitcount}}%
3201   \ifdefvoid\@glo@countunit
3202   {}%
3203   {%
3204     \@glsxtr@ifunitcounter{\@glo@countunit}%
3205     {}%
3206     {\expandafter\@glsxtr@addunitcounter\expandafter{\@glo@countunit}}%
3207   }%
3208 }
```

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

```
3209 \newcommand*{\@glsxtr@unitcountlist}{}
```


@addunitcounter

```
3210 \newcommand*{\@glxstr@addunitcounter}[1]{%
3211 \listadd{\@glxstr@unitcountlist}{#1}%
3212 \ifcsundef{glxstr@theunit@#1}
3213 {%
3214 \ifcsdef{theH#1}%
3215 {\csdef{glxstr@theunit@#1}{\csuse{theH#1}}}%
3216 {\csdef{glxstr@theunit@#1}{\csuse{the#1}}}%
3217 }%
3218 {}}%
3219 }
```

r@ifunitcounter

```
3220 \newcommand*{\@glxstr@ifunitcounter}[3]{%
3221 \xifinlist{#1}{\@glxstr@unitcountlist}{#2}{#3}%
3222 }
```

urrentunitcount

```
3223 \newcommand*\@glxstr@currentunitcount[1]{%
3224 glo@\glsdetoklabel{#1}@currunit@\glsggetattribute{#1}{unitcount}.%
3225 \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}%
3226 }
```

eviousunitcount

```
3227 \newcommand*\@glxstr@previousunitcount[1]{%
3228 glo@\glsdetoklabel{#1}@prevunit@\glsggetattribute{#1}{unitcount}.%
3229 \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}%
3230 }
```

t@currunitcount

```
3231 \newcommand*{\@glx@increment@currunitcount}[1]{%
3232 \glshasattribute{#1}{unitcount}%
3233 {%
3234 \edef\@glxstr@csname{\@glxstr@currentunitcount{#1}}%
3235 \ifcsundef{\@glxstr@csname}%
3236 {%
3237 \csgdef{\@glxstr@csname}{1}%
3238 \listcsxadd
3239 {glo@\glsdetoklabel{#1}@unitlist}%
3240 {\glsggetattribute{#1}{unitcount}.%
3241 \csuse{glxstr@theunit@\glsggetattribute{#1}{unitcount}}}%
3242 }%
3243 }%
3244 {%
3245 \csxdef{\@glxstr@csname}%
3246 {\number\numexpr\csname\@glxstr@csname\endcsname+1}%
3247 }%
3248 }%
3249 {}}%
```

3250 }

t@currunitcount

```
3251 \newcommand*{\@gls@local@increment@currunitcount}[1]{%
3252   \gls@attribute{#1}{unitcount}%
3253   {%
3254     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3255     \ifcsundef{\@glsxtr@csname}%
3256     {%
3257       \csdef{\@glsxtr@csname}{1}%
3258       \listcseadd
3259         {glo\glsdetoklabel{#1}@unitlist}%
3260         {\glsgetattribute{#1}{unitcount}.%
3261          \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3262       }%
3263     }%
3264     {%
3265       \csedef{\@glsxtr@csname}%
3266         {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3267       }%
3268     }%
3269   }%
3270 }
```

r@currunitcount

```
3271 \newcommand*{\@glsxtr@currunitcount}[2]{%
3272   \ifcsundef
3273     {glo\glsdetoklabel{#1}@currunit@#2}%
3274     {0}%
3275     {\csuse{glo\glsdetoklabel{#1}@currunit@#2}}%
3276   }%
```

r@prevunitcount

```
3277 \newcommand*{\@glsxtr@prevunitcount}[2]{%
3278   \ifcsundef
3279     {glo\glsdetoklabel{#1}@prevunit@#2}%
3280     {0}%
3281     {\csuse{glo\glsdetoklabel{#1}@prevunit@#2}}%
3282   }%
```

entryunitcount

```
3283 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3284   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3285   \renewcommand*{\gls@defdocnewglossaryentry}{%
3286     \renewcommand*\newglossaryentry[2]{%
3287       \PackageError{glossaries}{\string\newglossaryentry\space
```

```

3288     may only be used in the preamble when entry counting has
3289     been activated}{If you use \string\glsenableentryunitcount\space
3290     you must place all entry definitions in the preamble not in
3291     the document environment}%
3292 }%
3293 }%

```

New commands to access new fields:

```

3294 \newcommand*{\glentrycurrcount}[1]{%
3295   \@glxtr@currunitcount{##1}{\glsetattribute{##1}{unitcount}.%
3296   \csuse{glxtr@theunit@\glsetattribute{##1}{unitcount}}}%
3297 }%
3298 \newcommand*{\glentryprevcount}[1]{%
3299   \@glxtr@prevunitcount{##1}{\glsetattribute{##1}{unitcount}.%
3300   \csuse{glxtr@theunit@\glsetattribute{##1}{unitcount}}}%
3301 }%

```

Access total count:

```

3302 \newcommand*{\glentryprevtotalcount}[1]{%
3303   \ifcsundef{glo@\glsetoklabel{##1}@prevunittotal}%
3304   {0}%
3305   {%
3306     \number\csuse{glo@\glsetoklabel{##1}@prevunittotal}
3307   }%
3308 }%

```

Access max value:

```

3309 \newcommand*{\glentryprevmaxcount}[1]{%
3310   \ifcsundef{glo@\glsetoklabel{##1}@prevunitmax}%
3311   {0}%
3312   {%
3313     \number\csuse{glo@\glsetoklabel{##1}@prevunitmax}
3314   }%
3315 }%

```

Adjust post unset and reset:

```

3316 \let\@glxtr@entryunitcount@org@unset\glxtrpostunset
3317 \renewcommand*{\glxtrpostunset}[1]{%
3318   \@glxtr@entryunitcount@org@unset{##1}%
3319   \@gl@increment@currunitcount{##1}%
3320 }%
3321 \let\@glxtr@entryunitcount@org@localunset\glxtrpostlocalunset
3322 \renewcommand*{\glxtrpostlocalunset}[1]{%
3323   \@glxtr@entryunitcount@org@localunset{##1}%
3324   \@gl@local@increment@currunitcount{##1}%
3325 }%
3326 \let\@glxtr@entryunitcount@org@reset\glxtrpostreset
3327 \renewcommand*{\glxtrpostreset}[1]{%
3328   \glshasattribute{##1}{unitcount}%
3329   {%
3330     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%

```

```

3331 \ifcsundef{\@glxtr@csname}%
3332 {}%
3333 {\csgdef{\@glxtr@csname}{0}}%
3334 }%
3335 {}%
3336 }%
3337 \let\@glxtr@entryunitcount@org@localreset\glxtrpostlocalreset
3338 \renewcommand*{\glxtrpostlocalreset}[1]{%
3339 \@glxtr@entryunitcount@org@localreset{##1}%
3340 \glshasattribute{##1}{unitcount}%
3341 {%
3342 \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
3343 \ifcsundef{\@glxtr@csname}%
3344 {}%
3345 {\csdef{\@glxtr@csname}{0}}%
3346 }%
3347 {}%
3348 }%

```

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

```

3349 \let\@cgl@\@cgl@
3350 \let\@cgl@pl@\@cgl@pl@

3351 \let\@cGls@\@cGls@
3352 \let\@cGls@pl@\@cGls@pl@
3353 \let\@cGLS@\@cGLS@
3354 \let\@cGLS@pl@\@cGLS@pl@

```

Write information to the aux file.

```

3355 \AtEndDocument{\@gls@write@entryunitcounts}%
3356 \renewcommand*{\@gls@entry@unitcount}[3]{%
3357 \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
3358 \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
3359 {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
3360 {%
3361 \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
3362 \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
3363 }%
3364 \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
3365 {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
3366 {%
3367 \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
3368 \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
3369 \fi
3370 }%
3371 }%
3372 \let\glsenableentryunitcount\relax
3373 \renewcommand*{\glsenableentrycount}{%
3374 \PackageError{glossaries-extra}{\string\glsenableentrycount\space

```

```

3375     can't be used with \string\glsenableentryunitcount}%
3376     {Use one or other but not both commands}%
3377 }%
3378 }
3379 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3380 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3381 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3382   \immediate\write\@auxout
3383   {\string\@gls@entry@unitcount
3384    {\@gls@entry}%
3385    {\@gls@xtr@currunitcount{\@gls@entry}{#1}%
3386    }%
3387    {#1}}%
3388 }

```

entryunitcounts

```

3389 \newcommand*{\@gls@write@entryunitcounts}{%
3390   \immediate\write\@auxout
3391   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3392   \count@=0\relax
3393   \forallglsentries{\@gls@entry}{%
3394     \gls@hasattribute{\@gls@entry}{unitcount}%
3395     {%
3396       \ifglsused{\@gls@entry}%
3397       {%
3398         \forlistcsloop
3399           {\@gls@write@entryunitcounts@do}%
3400           {glo@\gls@detoklabel{\@gls@entry}@unitlist}%
3401       }%
3402     }%
3403     \advance\count@ by \@ne
3404   }%
3405 }%
3406 }%
3407 \ifnum\count@=0
3408   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3409     \MessageBreak with \string\glsenableentryunitcount\space but the
3410     \MessageBreak attribute 'unitcount' hasn't
3411     \MessageBreak been assigned to any of the defined
3412     \MessageBreak entries}%
3413 \fi
3414 }

```

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.

```

3415 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
    Enable entry counting:
3416   \glsenableentryunitcount
    Redefine \gls etc:
3417   \renewcommand*{\gls}{\cglss}%
3418   \renewcommand*{\Gls}{\cGls}%
3419   \renewcommand*{\glspl}{\cglsspl}%
3420   \renewcommand*{\Glspl}{\cGlspl}%
3421   \renewcommand*{\GLS}{\cGLS}%
3422   \renewcommand*{\GLSpl}{\cGLSpl}%
    Set the entrycount attribute:
3423   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%
    In case this command is used again:
3424   \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
3425   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3426     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3427       can't be used with \string\GlsXtrEnableEntryUnitCounting}%
3428     {Use one or other but not both commands}}%
3429 }

```

countunsetattr

```

3430 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3431   \@for\@glsxtr@cat:=#1\do
3432   {%
3433     \ifdefempty{\@glsxtr@cat}{}%
3434     {%
3435       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3436       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3437     }%
3438   }%
3439 }

```

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.

genericNewAcronym

```

3440 \renewcommand*{\SetGenericNewAcronym}{%
3441   \let\@Gls@entryname\@Gls@acrenryname
3442   \renewcommand{\newacronym}[4][ ]{%
3443     \ifdefempty{\@glsacronymlists}%

```

```

3444 {%
3445   \def\@glo@type{\acronymtype}%
3446   \setkeys{glossentry}{##1}%
3447   \DeclareAcronymList{\@glo@type}%
3448 }%
3449 {}%
3450 \glskeylisttok{##1}%
3451 \glslabeltok{##2}%
3452 \glsshorttok{##3}%
3453 \glslongtok{##4}%
3454 \newacronymhook
3455 \protected@edef\@do@newglossaryentry{%
3456   \noexpand\newglossaryentry{\the\glslabeltok}%
3457   {%
3458     type=\acronymtype,%
3459     name={\expandonce{\acronymentry{##2}}},%
3460     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
3461     text={\the\glsshorttok},%
3462     short={\the\glsshorttok},%
3463     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3464     long={\the\glslongtok},%
3465     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3466     category=acronym,
3467     \GenericAcronymFields,%
3468     \the\glskeylisttok
3469   }%
3470 }%
3471 \@do@newglossaryentry
3472 }%
3473 \renewcommand*{\acrfullfmt}[3]{%
3474   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3475 \renewcommand*{\Acrfullfmt}[3]{%
3476   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3477 \renewcommand*{\ACRfullfmt}[3]{%
3478   \glslink[##1]{##2}{%
3479     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3480 \renewcommand*{\acrfullplfmt}[3]{%
3481   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3482 \renewcommand*{\Acrfullplfmt}[3]{%
3483   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3484 \renewcommand*{\ACRfullplfmt}[3]{%
3485   \glslink[##1]{##2}{%
3486     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3487 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}%
3488 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}%
3489 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}%
3490 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}%
3491 }

```

This will cause a problem for glossaries that contain a mixture of acronyms and abbrevia-

tions, so redefine `\newacronym` to use the new abbreviation interface.

First save the original definitions:

```
3492 \let\@glxtr@org@setacronymstyle\setacronymstyle
3493 \let\@glxtr@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`.

```
3494 \newcommand*{\MakeAcronymsAbbreviations}{%
3495   \renewcommand*{\newacronym}[4][]{%
3496     \glxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3497   }%
3498   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
3499   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%
3500   \renewcommand*{\setacronymstyle}[1]{%
3501     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
3502       unavailable.
3503       Use \string\setabbreviationstyle\space instead.
3504       The original acronym interface can be restored with
3505       \string\RestoreAcronyms}{}}%
3506   }%
3507   \renewcommand*{\newacronymstyle}[1]{%
3508     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3509       available unless you restore the original acronym interface with
3510       \string\RestoreAcronyms}%
3511     \@glxtr@org@newacronymstyle{##1}%
3512   }%
3513 }
```

Switch acronyms to abbreviations:

```
3514 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```
3515 \newcommand*{\RestoreAcronyms}{%
3516   \SetGenericNewAcronym
3517   \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3518   \renewcommand{\acronymfont}[1]{##1}%
3519   \let\setacronymstyle\@glxtr@org@setacronymstyle
3520   \let\newacronymstyle\@glxtr@org@newacronymstyle
```

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

```
3521 \renewcommand*\@gls@link@checkfirsthyper{%
3522   \ifglused{\glslabel}%
3523   {\let\glxtrifwasfirstuse\@secondoftwo}
3524   {\let\glxtrifwasfirstuse\@firstoftwo}%
3525   \@glxtr@org@checkfirsthyper
3526 }
3527 \glsssetcategoryattribute{acronym}{regular}{false}%
```



```

3528 \setacronymstyle{long-short}%
3529 }

```

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

```

3530 \renewcommand*{\glsacspace}[1]{%
3531   \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3532   \ifdim\dimen@<\glsacspacemax~\else\space\fi
3533 }

```

`\glsacspacemax` Value used in the above.

```

3534 \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`

```

3535 \newcommand*{\@glxtr@reg@glosslist}{}

```

Save the original definition of `\makeglossaries`:

```

3536 \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`

```

3537 \renewcommand*{\makeglossaries}[1] [] {%
3538   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3539     \PackageError{glossaries-extra}{\string\makeglossaries\space
3540       not permitted\MessageBreak with record=only package option}%
3541     {You may only use \string\makeglossaries\space with
3542       record=off or record=alsoindex options}%
3543   \else
3544     \ifblank{#1}%
3545     {\@glxtr@org@makeglossaries}%
3546     {%
3547       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3548         \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3549           not permitted\MessageBreak with record=alsoindex package option}%
3550         {You may only use the hybrid \string\makeglossaries[...]\space with
3551           record=off option}%
3552       \else
3553         \edef\@glxtr@reg@glosslist{#1}%
3554         \ifundef{\glswrite}{\newwrite\glswrite}{}%

```

```

3555 \protected@write\@auxout{}\string\providecommand
3556 \string\@glsorder[1]{}
3557 \protected@write\@auxout{}\string\providecommand
3558 \string\@istfilename[1]{}
3559 \protected@write\@auxout{}\string\@istfilename{\istfilename}}%
3560 \protected@write\@auxout{}\string\@glsorder{\glsorder}}
3561 \protected@write\@auxout{}\string\glsxtr@makeglossaries{#1}}
3562 \write\@auxout{\string\providecommand\string\@gls@reference[3]{}}%

```

Iterate through each supplied glossary type and activate it.

```

3563 \@for\@glo@type:=#1\do{%
3564 \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3565 }%

```

New glossaries must be created before \makeglossaries:

```

3566 \renewcommand*\newglossary[4][]{%
3567 \PackageError{glossaries}{New glossaries
3568 must be created before \string\makeglossaries}{You need
3569 to move \string\makeglossaries\space after all your
3570 \string\newglossary\space commands}}%

```

Any subsequent instances of this command should have no effect

```

3571 \let\@makeglossary\relax
3572 \let\makeglossary\relax
3573 \renewcommand\makeglossaries[1][]{}%

```

Disable all commands that have no effect after \makeglossaries

```

3574 \@disable@onlypremakeg

```

Allow see key:

```

3575 \let\gls@checkseeallowed\relax

```

Adjust \@do@seeglossary. This needs to check for the entries existence.

```

3576 \renewcommand*\@do@seeglossary[2]{%
3577 \glsdoifexists{##1}%
3578 {%
3579 \edef\@gls@label{\glsdetoklabel{##1}}%
3580 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3581 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3582 {\@glsxtr@org@doseeglossary{##1}{##2}}%
3583 {%
3584 \@glsxtrwrglossmark
3585 \protected@write\@auxout{}\string\@gls@reference
3586 \string\@gls@reference
3587 {\@gls@type}{\@gls@label}{\string\glsseeformat##2}}%
3588 }%
3589 }%
3590 }%
3591 }%

```

Adjust \@do@wrglossary

```

3592 \let\glsxtr@do@wrglossary\@do@wrglossary

```

```

3593 \def\@do@wrglossary{%
3594 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3595 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3596 {\@glsxtr@do@wrglossary}%
3597 {\@gls@noidxglossary}%
3598 }%

```

Suppress warning about no \makeglossaries

```

3599 \let\warn@nomakeglossaries\relax
3600 \def\warn@noprintglossary{%
3601 \GlossariesWarningNoLine{No \string\printglossary\space
3602 or \string\printglossaries\space
3603 found.^^J(Remove \string\makeglossaries\space if you don't want
3604 any glossaries.)^^JThis document will not have a glossary}%
3605 }%

```

Only warn for glossaries not listed.

```

3606 \renewcommand{\@gls@noref@warn}[1]{%
3607 \edef\@gls@type{##1}%
3608 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3609 {%
3610 \GlossariesExtraWarning{Can't use
3611 \string\printnoidxglossary[type={\@gls@type}]
3612 when '\@gls@type' is listed in the optional argument of
3613 \string\makeglossaries}%
3614 }%
3615 {%
3616 \GlossariesWarning{Empty glossary for
3617 \string\printnoidxglossary[type={##1}].
3618 Rerun may be required (or you may have forgotten to use
3619 commands like \string\gls)}%
3620 }%
3621 }%

```

Adjust display number list to check for type:

```

3622 \renewcommand*\@glsdisplaynumberlist}[1]{%
3623 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3624 {\@glsxtr@idx@displaynumberlist{##1}}%
3625 {\@glsxtr@noidx@displaynumberlist{##1}}%
3626 }%

```

Adjust entry list:

```

3627 \renewcommand*\@glsentrynumberlist}[1]{%
3628 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3629 {\@glsxtr@idx@entrynumberlist{##1}}%
3630 {\@glsxtr@noidx@entrynumberlist{##1}}%
3631 }%

```

Adjust number list loop

```

3632 \renewcommand*\@glsnumberlistloop}[2]{%
3633 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%

```

```

3634      {%
3635          \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3636              not available for glossary ‘##1’}{}%
3637      }%
3638      {\@glxtr@noidx@numberlistloop{##1}{##2}}%
3639  }%

```

Only sanitize sort for normal indexing glossaries.

```

3640      \renewcommand*{\glsprestandardsort}[3]{%
3641          \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
3642      {%
3643          \glsdosanitizesort
3644      }%
3645      {%
3646          \ifglssanitizesort
3647              \@gls@noidx@sanitizesort
3648          \else
3649              \@gls@noidx@nosanitizesort
3650          \fi
3651      }%
3652  }%

```

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

```

3653      \renewcommand*\new@glossaryentry[2]{%
3654          \PackageError{glossaries-extra}{Glossary entries must be defined
3655              in the preamble\MessageBreak when you use the optional argument
3656              of \string\makeglossaries}{Either move your definitions to the
3657              preamble or don’t use the optional argument of
3658              \string\makeglossaries}%
3659  }%

```

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).

```

3660      \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
3661      \renewcommand*{\@printgloss@setsort}{%

```

Need to extract just the type value.

```

3662          \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
3663          type=\glsdefaulttype,\@end@glxtr@gettype
3664          \def\@glo@sorttype{\@glo@default@sorttype}%
3665      }%

```

Check automake setting:

```

3666      \ifglsautomake
3667          \renewcommand*{\@gls@doautomake}{%
3668              \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
3669                  \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3670              }%
3671          }%
3672      \fi

```

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

```

3673     \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3674     \fi
3675   }%
3676 \fi
3677 }

```

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

`rgprintglossary` 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.)

```

3678 \newcommand{\@glxtr@orgprintglossary}[2]{%
3679   \def\@glo@type{\glstexttype}%

```

Add check here.

```

3680   \def\glossarytitle{%
3681     \ifcsdef{\@glo@type@\@glo@type @title}%
3682     {\csuse{\@glo@type@\@glo@type @title}}%
3683     {\glossaryname}}%
3684   \def\glossarytoctitle{\glossarytitle}%
3685   \let\org@glossarytitle\glossarytitle
3686   \def\@glossarystyle{%
3687     \ifx\@glossary@default@style\relax
3688       \GlossariesWarning{No default glossary style provided \MessageBreak
3689         for the glossary '\@glo@type'. \MessageBreak
3690         Using deprecated fallback. \MessageBreak
3691         To fix this set the style with \MessageBreak
3692         \string\setglossarystyle\space or use the \MessageBreak
3693         style key=value option}%
3694     \fi
3695   }%
3696   \def\gls@dotoc@title{\gls@set@toc@title{\@glo@type}}%
3697   \let\org@glossaryentrynumbers\glossaryentrynumbers
3698   \bgroup
3699     \@printgloss@set@sort
3700     \setkeys{printgloss}{#1}%
3701     \ifx\glossarytitle\org@glossarytitle
3702     \else
3703       \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3704     \fi
3705     \let\currentglossary\@glo@type
3706     \let\org@glossaryentrynumbers\glossaryentrynumbers
3707     \let\gls@nonextpages\gls@nonextpages
3708     \let\gls@nextpages\gls@nextpages
3709
3710     \glx@extract@iven@post
3711     \gls@dotoc@title

```

```

3711 \glossarystyle
3712 \let\gls@org@glossaryentryfield\glossentry
3713 \let\gls@org@glossarysubentryfield\subglossentry
3714 \renewcommand{\glossentry}[1]{%
3715 \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3716 \gls@org@glossaryentryfield{##1}%
3717 }%
3718 \renewcommand{\subglossentry}[2]{%
3719 \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3720 \gls@org@glossarysubentryfield{##1}{##2}%
3721 }%
3722 \@gls@preglossaryhook
3723 #2%
3724 \egroup
3725 \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3726 \global\let\warn@noprintglossary\relax
3727 }

```

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

```

3728 \newcommand*{\glxtractivatenopost}{%
3729 \let\nopostdesc\@nopostdesc
3730 \let\glxtrnopostpunc\@glxtr@nopostpunc
3731 }

```

lsxtrnopostpunc

```

3732 \newrobustcmd*{\glxtrnopostpunc}{%

```

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

```

3733 \newcommand{\@glxtr@nopostpunc}{%
3734 \let\@glxtr@org@postdescription\glspostdescription
3735 \ifglsnopostdot
3736 \renewcommand{\glspostdescription}{%
3737 \glsnopostdottrue
3738 \let\glspostdescription\@glxtr@org@postdescription
3739 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3740 \glxtrpostdescription
3741 \@glxtr@nopostpunc@postdesc}%
3742 \else
3743 \renewcommand{\glspostdescription}{%
3744 \let\glspostdescription\@glxtr@org@postdescription
3745 \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3746 \glxtrpostdescription
3747 \@glxtr@nopostpunc@postdesc}%
3748 \fi
3749 \glsnopostdotfalse
3750 }

```

stpunc@postdesc

```

3751 \newcommand*{\@glxtr@nopostpunc@postdesc}{%
restore@postpunc
3752 \newcommand*{\@glxtr@restore@postpunc}{%
3753   \def\@glxtr@nopostpunc@postdesc{%
3754     \@glxtr@org@postdescription
3755     \let\@glxtr@nopostpunc@postdesc\@empty
3756     \let\glxtrrestorepostpunc\@empty
3757   }%
3758 }

restorepostpunc Does nothing outside of glossary.
3759 \newcommand*{\glxtrrestorepostpunc}{%

\@printglossary Redefine.
3760 \renewcommand{\@printglossary}[2]{%
3761   \def\@glxtr@printglossopts{#1}%
3762   \@glxtr@orgprintglossary{#1}{#2}%
3763 }

Add a key that switches off the entry targets:
3764 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3765   \ifcase\nr
3766     \let\glstarget\glsdohypertarget
3767   \else
3768     \let\glstarget\@secondoftwo
3769   \fi
3770 }

hypernameprefix
3771 \newcommand*{\@glxtrhypernameprefix}{%

New to v1.20:
3772 \define@key{printgloss}{targetnameprefix}{%
3773   \renewcommand*{\@glxtrhypernameprefix}{#1}%
3774 }

glsdohypertarget Redefine to insert \@glxtrhypernameprefix before the target name.
3775 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3776 \renewcommand*{\glsdohypertarget}[2]{%
3777   \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3778 }

@makeglossaries For the benefit of makeglossaries
3779 \newcommand*{\glxtr@makeglossaries}[1]{%

```

@glxtr@gettype Get just the type.

```
3780 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3781   \def\@glo@type{#2}%
3782 }
```

@assign@sortkey Assign the sort key.

```
3783 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3784   \edef\@glo@type{\@glo@type}%
3785   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3786   {%
3787     \@glo@no@assign@sortkey{#1}%
3788   }%
3789   {%
3790     \@glo@assign@sortkey{#1}%
3791   }%
3792 }
```

Display number list for the regular version:

splaynumberlist

```
3793 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
3794 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
3795   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3796   \ifdef\@gls@loclist
3797   {%
3798     \def\@gls@noidxloclist@sep{%
3799       \def\@gls@noidxloclist@sep{%
3800         \def\@gls@noidxloclist@sep{%
3801           \glsnumlistsep
3802         }%
3803         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3804       }%
3805     }%
3806     \def\@gls@noidxloclist@finalsep{}%
3807     \def\@gls@noidxloclist@prev{}%
3808     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3809     \@gls@noidxloclist@finalsep
3810     \@gls@noidxloclist@prev
3811   }%
3812   {%
3813     \glxtrundeftag
3814     \glsdoifexists{#1}%
3815     {%
3816       \GlossariesWarning{Missing location list for ‘#1’. Either
3817         a rerun is required or you haven’t referenced the entry.}%

```



```

3818 }%
3819 }%
3820 }%
3821

```

And for the number list loop:

@numberlistloop

```

3822 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3823   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3824   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3825   \let\@gls@org@glssseeformat\glssseeformat
3826   \let\glsnoidxdisplayloc#2\relax
3827   \let\glssseeformat#3\relax
3828   \ifdef\@gls@loclist
3829   {%
3830     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3831   }%
3832   {%
3833     \glsxtrundeftag
3834     \glsoifexists{#1}%
3835     {%
3836       \GlossariesWarning{Missing location list for ‘##1’. Either
3837         a rerun is required or you haven’t referenced the entry.}%
3838     }%
3839   }%
3840   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3841   \let\glssseeformat\@gls@org@glssseeformat
3842 }%

```

Same for entry number list.

entrynumberlist

```

3843 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3844   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3845   \ifdef\@gls@loclist
3846   {%
3847     \glsnoidxloclist{\@gls@loclist}%
3848   }%
3849   {%
3850     \glsxtrundeftag
3851     \glsoifexists{#1}%
3852     {%
3853       \GlossariesWarning{Missing location list for ‘#1’. Either
3854         a rerun is required or you haven’t referenced the entry.}%
3855     }%
3856   }%
3857 }%

```

entrynumberlist

```
3858 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}
```

x@getgrouptitle Patch.

```
3859 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3860   \protected@edef\@glsxtr@titlelabel{#1}%
3861   \ifdefvoid\@glsxtr@titlelabel
3862   {%
3863     {%
3864       \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
3865     }%
3866     \ifdefvoid{\@glsxtr@titlelabel}%
3867     {%
3868       \DTLifint{#1}%
3869       {%
3870         \ifnum#1<256\relax
3871           \edef#2{\char#1\relax}%
3872         \else
3873           \edef#2{#1}%
3874         \fi
3875       }%
3876       {%
3877         \ifcsundef{#1groupname}%
3878         {\def#2{#1}}%
3879         {\letcs#2{#1groupname}}%
3880       }%
3881     }%
3882     {%
3883       \let#2\@glsxtr@titlelabel
3884     }%
3885 }
```

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

```
3886 \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.

```
3887 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3888   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
3889   \@onelevel@sanitize\@glsxtr@titlelabel
3890   \ifcsdef{\@glsxtr@titlelabel}
3891   {\letcs{#2}{\@glsxtr@titlelabel}}%
3892   {\glsxtr@org@getgrouptitle{#1}{#2}}%
3893 }
3894 \let\@gls@getgrouptitle\glsxtrgetgrouptitle
```

trsetgrouptitle Sets the title for the given group label.

```
3895 \newcommand{\glsxtrsetgrouptitle}[2]{%
```

```

3896 \protected@edef\@glstr@titlelabel{glstr@grouptitle@#1}%
3897 \@onelevel@sanitize\@glstr@titlelabel
3898 \csxdef{\@glstr@titlelabel}{#2}%
3899 }

```

`\alsetgrouptitle` As above put only locally defines the title.

```

3900 \newcommand{\glstrlocalsetgrouptitle}[2]{%
3901 \protected@edef\@glstr@titlelabel{glstr@grouptitle@#1}%
3902 \@onelevel@sanitize\@glstr@titlelabel
3903 \csedef{\@glstr@titlelabel}{#2}%
3904 }

```

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

```

3905 \renewcommand*{\glsnavigation}{%
3906 \def\@gls@between{}%
3907 \ifcsundef{\@gls@hypergroup@list@\@glo@type}%
3908 {%
3909 \def\@gls@list{}%
3910 }%
3911 {%
3912 \expandafter\let\expandafter\@gls@list
3913 \csname \@gls@hypergroup@list@\@glo@type\endcsname
3914 }%
3915 \@for\@gls@tmp:=\@gls@list\do{%
3916 \@gls@between
3917 \glstrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3918 \glsnavigationhyperlink{\@gls@tmp}{\@gls@grptitle}%
3919 \let\@gls@between\glshypernavsep
3920 }%
3921 }

```

`@noidx@glossary`

```

3922 \renewcommand*{\@print@noidx@glossary}{%
3923 \ifcsdef{\@glsref@\@glo@type}%
3924 {%
3925 \ifcsdef{\@glo@sortmacro@\@glo@sorttype}%
3926 {%
3927 \csuse{\@glo@sortmacro@\@glo@sorttype}{\@glo@type}%
3928 }%
3929 {%
3930 \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3931 }%
3932 \glossarysection[\glossarytoctitle]{\glossarytitle}%
3933 \glossary preamble

```

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

```

3934 \def\@gls@currentlettergroup{}%
3935 \begin{theglossary}%

```

```

3936 \glossaryheader
3937 \glsresetentrylist
3938 \forlistcsloop{\@gls@noidx@do}{\@glsref@{\@glo@type}}%
3939 \end{theglossary}%
3940 \glossarypostamble
3941 }%
3942 {%

```

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

```

3943 \glstrifemptyglossary{\@glo@type}%
3944 }%
3945 {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3946 \@gls@noref@warn{\@glo@type}%
3947 }%
3948 }

```

`noidxdisplayloc` Patch to check for range formations.

```

3949 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3950 \setentrycounter[#1]{#2}%
3951 \@glstr@display@loc#3\empty\end@glstr@display@loc{#4}%
3952 }

```

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

```

3953 \def\@glstr@display@loc#1#2\end@glstr@display@loc#3{%
3954 \ifx#1\relax
3955 \glstrdisplaystartloc{#2}{#3}%
3956 \else
3957 \ifx#1)\relax
3958 \glstrdisplayendloc{#2}{#3}%
3959 \else
3960 \glstrdisplaysingleloc{#1#2}{#3}%
3961 \fi
3962 \fi
3963 }

```

`isplaysingleloc` Single location.

```

3964 \newcommand*{\glstrdisplaysingleloc}[2]{%
3965 \csuse{#1}{#2}%
3966 }

```

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

`displaystartloc` Start of a location range.

```

3967 \newcommand*{\glstrdisplaystartloc}[2]{%
3968 \edef\glstrlocrangefmt{#1}%
3969 \ifx\glstrlocrangefmt\empty
3970 \def\glstrlocrangefmt{\glsnumberformat}%

```

```

3971 \fi
3972 \expandafter\glxtrdisplaysingleloc
3973 \expandafter{\glxtrlocrangefmt}{#2}%
3974 }

```

trdisplayendloc End of a location range.

```

3975 \newcommand*\glxtrdisplayendloc}[2]{%
3976 \edef\@glxtr@tmp{#1}%
3977 \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glsnumberformat}}{}%
3978 \ifx\glxtrlocrangefmt\@glxtr@tmp
3979 \else
3980 \GlossariesExtraWarning{Mismatched end location range
3981 (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3982 \fi
3983 \expandafter\glxtrdisplayendlochook\expandafter{\@glxtr@tmp}{#2}%
3984 \expandafter\glxtrdisplaysingleloc
3985 \expandafter{\glxtrlocrangefmt}{#2}%
3986 \def\glxtrlocrangefmt{}%
3987 }

```

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

```

3988 \newcommand*\glxtrdisplayendlochook}[2]{

```

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

```

3989 \newcommand*\glxtrlocrangefmt{}

```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```

3990 \def\@gls@removespaces#1 #2\@nil{%
3991 \toks@=\expandafter{\the\toks@#1}%
3992 \ifx\#2\%
3993 \edef\x{\the\toks@}%
3994 \ifx\x\empty
3995 \else
3996 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3997 \fi
3998 \else
3999 \@gls@ReturnAfterFi{%
4000 \@gls@removespaces#2\@nil
4001 }%
4002 \fi
4003 }

```

cationhyperlink

```

4004 \newcommand*\glxtrlocationhyperlink}[3]{%
4005 \ifdefvoid\glxtrsupplocationurl
4006 {%
4007 \glxtrhyperlink{#1#2#3}{#3}%
4008 }%
4009 {%

```

```

4010 \hyperref{\glxtrsupplocationurl}{\#1\#2\#3}{\#3}%
4011 }%
4012 }

```

supphypernumber

```

4013 \newcommand*\glxtrsupphypernumber[1]{%
4014 {%
4015 \glshasattribute{\glscurrententrylabel}{externallocation}%
4016 {%
4017 \def\glxtrsupplocationurl{%
4018 \glsggetattribute{\glscurrententrylabel}{externallocation}}%
4019 }%
4020 {%
4021 \def\glxtrsupplocationurl{%
4022 }%
4023 \glshypernumber{#1}%
4024 }%
4025 }

```

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

@print@glossary

```

4026 \renewcommand*\@print@glossary{%
4027 \makeatletter
4028 \@input{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4029 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4030 {}%
4031 {\glxtrNoGlossaryWarning{\@glo@type}}%
4032 \ifglxindy
4033 \ifcsundef{\xdy@\@glo@type @language}%
4034 {%
4035 \edef\@do@auxoutstuff{%
4036 \noexpand\AtEndDocument{%
4037 \noexpand\immediate\noexpand\write\@auxout{%
4038 \string\providecommand\string\@xdylanguage[2]{}}%
4039 \noexpand\immediate\noexpand\write\@auxout{%
4040 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4041 }%
4042 }%
4043 }%
4044 {%
4045 \edef\@do@auxoutstuff{%
4046 \noexpand\AtEndDocument{%
4047 \noexpand\immediate\noexpand\write\@auxout{%
4048 \string\providecommand\string\@xdylanguage[2]{}}%
4049 \noexpand\immediate\noexpand\write\@auxout{%
4050 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4051 @language\endcsname}}%

```

```

4052     }%
4053 }%
4054 }%
4055 \do@auxoutstuff
4056 \edef\do@auxoutstuff{%
4057     \noexpand\AtEndDocument{%
4058         \noexpand\immediate\noexpand\write\@auxout{%
4059             \string\providecommand\string\@gls@codepage[2]{}}%
4060         \noexpand\immediate\noexpand\write\@auxout{%
4061             \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4062     }%
4063 }%
4064 \do@auxoutstuff
4065 \fi
4066 \renewcommand*{\@warn@nomakeglossaries}{%
4067     \GlossariesWarningNoLine{\string\makeglossaries\space
4068     hasn't been used,^^Jthe glossaries will not be updated}%
4069 }%
4070 }

```

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

`oGlsWarningHead` Header message.

```

4071 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4072     This document is incomplete. The external file associated with
4073     the glossary '#1' (which should be called \texttt{#2})
4074     hasn't been created.%
4075 }

```

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

```

4076 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4077     This has probably happened because there are no entries defined
4078     in this glossary.%
4079 }

```

`arningEmptyMain` The default “main” glossary is empty.

```

4080 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4081     If you don't want this glossary,
4082     add \texttt{nomain} to your package option list when you load
4083     \texttt{glossaries-extra.sty}. For example:%
4084 }

```

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

```

4085 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4086     Did you forget to use \texttt{type=#1} when you defined your
4087     entries? If you tried to load entries into this glossary with
4088     \texttt{\string\loadglsentries} did you remember to use
4089     \texttt{[#1]} as the optional argument? If you did, check that
4090     the definitions in the file you loaded all had the type set

```

```

4091 to \texttt{\string\glsdefaulttype}.%
4092 }

```

WarningCheckFile Advisory message to check the file contents.

```

4093 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4094   Check the contents of the file \texttt{#1}. If
4095   it's empty, that means you haven't indexed any of your entries in this
4096   glossary (using commands like \texttt{\string\gls} or
4097   \texttt{\string\glsadd}) so this list can't be generated.
4098   If the file isn't empty, the document build process hasn't been
4099   completed.%
4100 }

```

WarningAutoMake Message when automake option has been used.

```

4101 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4102   You may need to rerun \LaTeX. If you already have, it may be that
4103   \TeX's shell escape doesn't allow you to run
4104   \ifglxindy xindy\else makeindex\fi. Check the
4105   transcript file \texttt{\jobname.log}. If the shell escape is
4106   disabled, try one of the following:
4107
4108   \begin{itemize}
4109     \item Run the external (Lua) application:
4110
4111       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4112
4113     \item Run the external (Perl) application:
4114
4115       \texttt{makeglossaries \string"\jobname\string"}
4116   \end{itemize}
4117
4118   Then rerun \LaTeX\ on this document.
4119   \GlossariesExtraWarning{Rerun required to build the
4120   glossary '#1' or check TeX's shell escape allows
4121   you to run \ifglxindy xindy\else makeindex\fi}%
4122 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4123 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
4124   You need to either replace \texttt{\string\makenoidxglossaries}
4125   with \texttt{\string\makeglossaries} or replace
4126   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4127   \texttt{\string\printnoidxglossary}
4128   (or \texttt{\string\printnoidxglossaries}) and then rebuild
4129   this document.%
4130 }

```

WarningBuildInfo Build advice.


```

4131 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4132   Try one of the following:
4133   \begin{itemize}
4134     \item Add \texttt{automake} to your package option list when you load
4135       \texttt{glossaries-extra.sty}. For example:
4136
4137       \texttt{\string\usepackage[automake]%
4138         \glspopenbrace glossaries-extra\glsclosebrace}
4139
4140     \item Run the external (Lua) application:
4141
4142     \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4143
4144     \item Run the external (Perl) application:
4145
4146     \texttt{makeglossaries \string"\jobname\string"}
4147   \end{itemize}
4148
4149   Then rerun \LaTeX\ on this document.%
4150 }

```

`oGlsWarningTail` Final paragraph.

```

4151 \newcommand{\GlsXtrNoGlsWarningTail}{%
4152   This message will be removed once the problem has been fixed.%
4153 }

```

`GlsWarningNoOut` No out file created. Build advice.

```

4154 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4155   The file \texttt{#1} doesn't exist. This most likely means you haven't used
4156   \texttt{\string\makeglossaries} or you have used
4157   \texttt{\string\nofiles}. If this is just a draft version of the
4158   document, you can suppress this message using the
4159   \texttt{\string\nomissingglstext} package option.%
4160 }

```

`glossarywarning`

```

4161 \newcommand*{@@glxtr@defaultnoglossarywarning}[1]{%
4162   \glossarysection[\glossarytoctitle]{\glossarytitle}
4163   \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
4164   \par
4165   \glxtrifemptyglossary{#1}%
4166   {%
4167     \GlsXtrNoGlsWarningEmptyStart\space
4168     \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4169     \medskip
4170     \noindent\texttt{\string\usepackage[nomain\ifglacronym ,acronym\fi]%
4171       \glspopenbrace glossaries-extra\glsclosebrace}
4172     \medskip
4173   }%

```

```

4174     {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4175 }%
4176 {%
4177   \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
4178   {%
4179     \GlsXtrNoGlsWarningCheckFile
4180     {\jobname.\csname @glotype@\@glo@type @out\endcsname}
4181
4182     \ifglsautomake
4183
4184     \GlsXtrNoGlsWarningAutoMake{#1}
4185
4186     \else
4187
4188     \ifthenelse{\equal{#1}{main}}%
4189     {%
4190       \GlsXtrNoGlsWarningEmptyMain\par
4191       \medskip
4192       \noindent\texttt{\string\usepackage[nomain]%
4193         \glsopenbrace glossaries-extra\glsclosebrace}
4194       \medskip
4195     }%
4196     {}%
4197
4198     \ifdefequal\makeglossaries\@no@makeglossaries
4199     {%
4200       \GlsXtrNoGlsWarningMismatch
4201     }%
4202     {%
4203       \GlsXtrNoGlsWarningBuildInfo
4204     }%
4205     \fi
4206   }%
4207   {%
4208     \GlsXtrNoGlsWarningNoOut
4209     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4210   }%
4211 }%
4212 \par
4213 \GlsXtrNoGlsWarningTail
4214 }

```

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

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

```
4215 \newcommand*{\glxtrresourcefile}[2] [] {%
```

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

```
4216   \disable@keys{glossaries-extra.sty}{record}%
```

```

4217 \glsxtr@writefields
4218 \protected@write\@auxout{\glsxtrresourceinit}{\string\glsxtr@resource{#1}{#2}}%
4219 \let\@glsxtr@org@see@noindex\@gls@see@noindex
4220 \let\@gls@see@noindex\relax
4221 \IfFileExists{#2.glstex}%
4222 {%

```

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

```

4223 \edef\@bibgls@restoreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4224 \makeatletter
4225 \@input{#2.glstex}%
4226 \@bibgls@restoreat
4227 }%
4228 {%
4229 \GlossariesExtraWarning{No file '#2.glstex'}%
4230 }%
4231 \let\@gls@see@noindex\@glsxtr@org@see@noindex
4232 }
4233 \@onlypreamble\glsxtrresourcefile

```

trresourcecount

```

4234 \newcount\glsxtrresourcecount

```

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

```

4235 \newcommand*{\GlsXtrLoadResources}[1][]{%
4236 \ifnum\glsxtrresourcecount=0\relax
4237 \glsxtrresourcefile[#1]{\jobname}%
4238 \else
4239 \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
4240 \fi
4241 \advance\glsxtrresourcecount by 1\relax
4242 }

```

glsxtr@resource

```

4243 \newcommand*{\glsxtr@resource}[2]{%

```

\glsxtr@fields

```

4244 \newcommand*{\glsxtr@fields}[1]{%

```

xtr@texencoding

```

4245 \newcommand*{\glsxtr@texencoding}[1]{%

```

\glsxtr@langtag

```

4246 \newcommand*{\glsxtr@langtag}[1]{%

```

@pluralsuffixes

```

4247 \newcommand*{\glsxtr@pluralsuffixes}[4]{%

```

tr@shortcutsval

```
4248 \newcommand*{\glxtr@shortcutsval}[1]{}
```

sxtr@linkprefix

```
4249 \newcommand*{\glxtr@linkprefix}[1]{}
```

xtr@writefields This information only needs to be written once, so disable it after it's been used.

```
4250 \newcommand*{\glxtr@writefields}{%
```

```
4251 \protected@write\@auxout{}%
```

```
4252 {\string\providecommand*{\string\glxtr@fields}[1]{}}%
```

```
4253 \protected@write\@auxout{}%
```

```
4254 {\string\providecommand*{\string\glxtr@resource}[2]{}}%
```

```
4255 \protected@write\@auxout{}%
```

```
4256 {\string\providecommand*{\string\glxtr@pluralsuffixes}[4]{}}%
```

```
4257 \protected@write\@auxout{}%
```

```
4258 {\string\providecommand*{\string\glxtr@shortcutsval}[1]{}}%
```

```
4259 \protected@write\@auxout{}%
```

```
4260 {\string\providecommand*{\string\glxtr@linkprefix}[1]{}}%
```

```
4261 \protected@write\@auxout{}{\string\glxtr@fields{\@gls@keymap}}%
```

```
4262 \protected@write\@auxout{}%
```

```
4263 {\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`.

```
4264 \ifdef\CurrentTrackedLanguageTag
```

```
4265 {%
```

```
4266 \protected@write\@auxout{}{%
```

```
4267 \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
```

```
4268 }%
```

```
4269 {}%
```

```
4270 \protected@write\@auxout{}{\string\glxtr@pluralsuffixes
```

```
4271 {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
```

```
4272 {\glxtrabbrvpluralsuffix}}%
```

```
4273 \ifdef\inputencodingname
```

```
4274 {%
```

```
4275 \protected@write\@auxout{}{\string\glxtr@texencoding{\inputencodingname}}%
```

```
4276 }%
```

```
4277 {%
```

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.)

```
4278 \@ifpackageloaded{fontspec}%
```

```
4279 {\protected@write\@auxout{}{\string\glxtr@texencoding{utf8}}}%
```

```
4280 {}%
```

```
4281 }%
```

```
4282 \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.

```
4283 \AtBeginDocument
4284   {\protected@write\@auxout{}\string\glstr@linkprefix{\glolinkprefix}}}%
4285 \let\glstr@writefields\relax
```

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

```
4286 \ifglstrautomake
4287   \IfFileExists{\jobname.aux}%
4288   {\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.

```
4289 \ifx\@glstr@doautomake\@glstr@doautomake@err
4290   \let\@glstr@doautomake\relax
4291 \fi
4292 \fi
4293 }
```

do@automake@err

```
4294 \newcommand*{\@glstr@doautomake@err}{%
4295   \PackageError{glossaries}{You must use
4296     \string\makeglossaries\space with automake=true}
4297   {%
4298     Either remove the automake=true setting or
4299     add \string\makeglossaries\space to your document preamble.%
4300   }}%
4301 }
```

Allow locations specific to a particular counter to be recorded.

\glstr@record

```
4302 \newcommand*{\glstr@record}[5]{}
```

r@counterrecord Aux file command.

```
4303 \newcommand*{\glstr@counterrecord}[3]{%
4304   \glstrfieldlistgadd{#1}{record.#2}{#3}%
4305 }
```

unterrecordhook Hook used by \@glstr@dorecord.

```
4306 \newcommand*{\@glstr@counterrecordhook}{}
```

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

```
4307 \newcommand*{\GlsXtrRecordCounter}[1]{%
4308   \@glstr@recordcounter{#1}%
4309 }
4310 \@onlypreamble\GlsXtrRecordCounter
```

docounterrecord

```
4311 \newcommand*{\@glxstr@docounterrecord}[1]{%
4312   \protected@write\@auxout{}\string\glxstr@counterrecord
4313     {\@glsl@label}{#1}{\csuse{the#1}}}%
4314 }
```

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.

```
4315 \newcommand*{\glxtrglossentry}[1]{%
4316   \glxstrtitleorpdforheading
4317   {\@glxtrglossentry{#1}}%
4318   {\glsentryname{#1}}%
4319   {\glxtrheadname{#1}}%
4320 }
```

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

```
4321 \newrobustcmd*{\@glxtrglossentry}[1]{%
4322   \glxstrtitleorpdforheading
4323   {%
4324     \glsdofexists{#1}%
4325     {%
4326       \begingroup
4327         \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4328         \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4329         \ifglshasparent{#1}%
4330           {\glssubentryitem{#1}}%
4331           {\glsentryitem{#1}}%
4332           \glstarget{#1}{\glossentryname{#1}}%
4333       \endgroup
4334     }%
4335   }%
4336   {\glsentryname{#1}}%
4337   {\glxtrheadname{#1}}%
4338 }
```

glossentryother As `\glxtrglossentry` 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.

```
4339 \newcommand*{\glxtrglossentryother}[3]{%
4340   \ifstrepty{#1}%
4341   {%
4342     \ifcsdef{glxtrhead#3}%
```

```

4343   {%
4344       \glstrtitleorpdforheading
4345       {\@glstrglossentryother{#2}{#3}{#1}}%
4346       {\@gls@entry@field{#2}{#3}}%
4347       {\csuse{glstrhead#3}{#2}}%
4348   }%
4349   {%
4350       \glstrtitleorpdforheading
4351       {\@glstrglossentryother{#2}{#3}{#1}}%
4352       {\@gls@entry@field{#2}{#3}}%
4353       {\@gls@entry@field{\NoCaseChange{#2}}{#3}}%
4354   }%
4355 }%
4356 {%
4357     \glstrtitleorpdforheading
4358     {\@glstrglossentryother{#2}{#3}{#1}}%
4359     {\@gls@entry@field{#2}{#3}}%
4360     {#1}%
4361 }%
4362 }

```

`glossentryother` As `\@glstrglossentry` but uses a different field.

```

4363 \newrobustcmd*{\@glstrglossentryother}[3]{%
4364     \glstrtitleorpdforheading
4365     {%
4366         \glsdoifexists{#1}%
4367         {%
4368             \begingroup
4369                 \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4370                 \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4371                 \ifglshasparent{#1}%
4372                 {\glssubentryitem{#1}}%
4373                 {\glsentryitem{#1}}%
4374                 \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4375             \endgroup
4376         }%
4377     }%
4378     {\@gls@entry@field{#1}{#2}}%
4379     {#3}%
4380 }

```

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

```

4381 \newcommand*{\printunsrtglossary}{%
4382     \@ifstar\s@printunsrtglossary\@printunsrtglossary
4383 }

```

`ntunsrtglossary` Unstarred version.

```

4384 \newcommand*{\@printunsrtglossary}[1][\]{%

```

```

4385 \printglossary{type=\glsdefaulttype,#1}{\print@unsrt@glossary}%
4386 }

```

`\printunsrtglossary` Starred version.

```

4387 \newcommand*{\s@printunsrtglossary}[2][\]{%
4388   \begin{group}
4389     #2%
4390     \printglossary{type=\glsdefaulttype,#1}{\print@unsrt@glossary}%
4391   \end{group}
4392 }

```

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

```

4393 \newcommand*{\printunsrtglossaries}{%
4394   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4395 }

```

`\print@unsrt@glossary`

```

4396 \newcommand*{\print@unsrt@glossary}{%
4397   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4398   \glossarypreamble
4399   check for empty list
4400   {\%
4401     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4402   }%
4403   {\%
4404     \key@ifundefined{glossentry}{group}%
4405     {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4406     {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4407     \def\@gls@currentlettergroup{}}%

```

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

```

4408   \def\@glsxtr@doglossary{%
4409     \begin{theglossary}%
4410     \glossaryheader
4411     \glsresetentrylist
4412   }%
4413   \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4414     :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
4415     \ifdefempty{\glscurrententrylabel}
4416     }%
4417   {%

```

Provide a hook (for example to measure width).

```

4418     \let\glsxtr@process\@firstofone
4419     \let\printunsrtglossaryskipentry
4420     \@glsxtr@printunsrtglossaryskipentry
4421     \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```


Don't check group for child entries.

```

4422 \glstr@process
4423 {%
4424 \ifglshasparent{\glscurrententrylabel}{}%
4425 {%
4426 \@glstr@checkgroup\glscurrententrylabel
4427 \expandafter\appto\expandafter\@glstr@doglossary\expandafter
4428 {\@glstr@groupheading}%
4429 }%
4430 \eappto\@glstr@doglossary{%
4431 \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4432 }%
4433 }%
4434 }%
4435 \appto\@glstr@doglossary{\end{theglossary}}%
4436 \printunsrtglossarypredoglossary
4437 \@glstr@doglossary
4438 }%
4439 \glossarypostamble
4440 }

```

entryprocesshook

```

4441 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}

```

entryprocesshook

```

4442 \newcommand*{\printunsrtglossaryskipentry}{%
4443 \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4444 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4445 }

```

entryprocesshook

```

4446 \newcommand*{\@glstr@printunsrtglossaryskipentry}{%
4447 \let\glstr@process\@gobble
4448 }

```

rypredoglossary

```

4449 \newcommand*{\printunsrtglossarypredoglossary}{}

```

lossary@handler

```

4450 \newcommand{\@printunsrt@glossary@handler}[1]{%
4451 \xdef\glscurrententrylabel{#1}%
4452 \printunsrtglossaryhandler\glscurrententrylabel
4453 }

```

glossaryhandler

```

4454 \newcommand{\printunsrtglossaryhandler}[1]{%
4455 \glstrunsrtdo{#1}%
4456 }

```

triflabelinlist Might be useful for the handler to check if an entry label or category label is contained in a list, so provide a user-level version of `\@gls@ifinlist` which ensures the label and list are fully expanded.

```
4457 \newrobustcmd*{\glstriflabelinlist}[4]{%
4458   \protected@edef\@glstr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
4459   \@glstr@doiflabelinlist{#3}{#4}}%
4460 }
```

srtglossaryunit

```
4461 \newcommand{\print@op@unsrtglossaryunit}[2][{}]{%
4462   \s@printunsrtglossary[type=\glsdefaulttype,#1]{%
4463     \printunsrtglossaryunitsetup{#2}}%
4464   }%
4465 }
```

ossaryunitsetup

```
4466 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4467   \renewcommand{\printunsrtglossaryhandler}[1]{%
4468     \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4469     {\glstrunsrtdo{##1}}}%
4470   }%
4471   }%
```

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

```
4472 \ifcsundef{theH#1}%
4473 {%
4474   \renewcommand*{\@glstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4475   }%
4476   {%
4477     \renewcommand*{\@glstrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4478     }%
4479   \renewcommand*{\glossarysection}[2][{}]{%
4480     \appto\glossarypostamble{\glspare\medskip\glspare}%
4481   }
```

srtglossaryunit

```
4482 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
4483   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4484     requires the record=only or record=alsoindex package option}{}%
4485 }
```

t@getgrouptitle

```
4486 \newrobustcmd*{\@glstr@unsrt@getgrouptitle}[2]{%
4487   \protected@edef\@glstr@titlelabel{\glstr@grouptitle@#1}%
4488   \@onelevel@sanitize\@glstr@titlelabel
4489   \ifcsdef{\@glstr@titlelabel}
4490   {\letcs{#2}{\@glstr@titlelabel}}%
```

```

4491  {\def#2{#1}}%
4492 }

```

`\glxstrunsrtdo` Provide a user-level call to `\@glxstr@noidx@do` to make it easier to define a new handler.

```

4493 \newcommand{\glxstrunsrtdo}{\@glxstr@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).

```

4494 \newcommand*{\glxtrgroupfield}{group}

```

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

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

```

4495 \newcommand*{\@glxstr@checkgroup}[1]{%
4496   \def\@glxstr@groupheading{}%
4497   \key@ifundefined{glossentry}{group}%
4498   {%
4499     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4500     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4501   }%
4502   {%
4503     \protected@edef\@glo@thislettergrp{%
4504       \csuse{glo@\glsdetoklabel{#1}@\glxtrgroupfield}}%
4505   }%
4506   \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4507   {}%
4508   {%
4509     \ifdefempty{\@gls@currentlettergroup}{}%
4510     {\def\@glxstr@groupheading{\glsgroupskip}}%
4511     \eappto\@glxstr@groupheading{%
4512       \noexpand\glsgroupheading{\expandonce\@glo@thislettergrp}%
4513     }%
4514   }%
4515   \let\@gls@currentlettergroup\@glo@thislettergrp
4516 }

```

`glxstr@noidx@do` Minor modification of `\@gls@noidx@do` to check for location field if present, but also need to check for the group field.

```

4517 \newcommand{\@glxstr@noidx@do}[1]{%
4518   \ifgl Sentryexists{#1}%
4519   {%

```

```

4520 \global\letcs{\@gls@loclist}{glo@glstetoklabel{#1}@loclist}%
4521 \global\letcs{\@gls@location}{glo@glstetoklabel{#1}@location}%
4522 \ifglshasparent{#1}%
4523 {%
4524   \gls@level=\csuse{glo@glstetoklabel{#1}@level}\relax
4525   \ifdefvoid{\@gls@location}%
4526   {%
4527     \ifdefvoid{\@gls@loclist}%
4528     {%
4529       \subglossentry{\gls@level}{#1}{}%
4530     }%
4531     {%
4532       \subglossentry{\gls@level}{#1}%
4533       {%
4534         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4535       }%
4536     }%
4537   }%
4538   {%
4539     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4540   }%
4541 }%
4542 {%

4543   \ifdefvoid{\@gls@location}%
4544   {%
4545     \ifdefvoid{\@gls@loclist}
4546     {%
4547       \glossentry{#1}{}%
4548     }%
4549     {%
4550       \glossentry{#1}%
4551       {%
4552         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4553       }%
4554     }%
4555   }%
4556   {%
4557     \glossentry{#1}%
4558     {%
4559       \glossaryentrynumbers{\@gls@location}%
4560     }%
4561   }%
4562 }%
4563 }%
4564 {}%
4565 }

```

1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4566 \newcommand*{\glshex}{\string\u}
```

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

```
4567 \newcommand*{\glstrresourceinit}{}
```

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.

`\glstrnewgls`

```
4568 \newcount\@glstrnewgls@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>}`

```
4569 \newcommand*{\@glstrnewgls}[4]{%
```

```
4570   \ifdef{#3}%
```

```
4571   {%
```

```
4572     \PackageError{glossaries-extra}{Command \string#3\space already
```

```
4573 defined}{}%
```

```
4574   }%
```

```
4575   {%
```

```
4576     \ifcsdef{@#4like@#2}%
```

```
4577     {%
```

```
4578       \advance\@glstrnewgls@inner by \@ne
```

```
4579       \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
```

```
4580     }%
```

```
4581     {\def\@glstrnewgls@innercsname{@#4like@#2}}%
```

```
4582     \expandafter\newrobustcmd\expandafter*\expandafter
```

```
4583       #3\expandafter{\expandafter\@gls@hyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
```

```
4584     \ifstrempy{#1}%
```

```
4585     {%
```

```
4586       \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][ ]{%
```

```
4587         \new@ifnextchar[%
```

```
4588         {\csname @#4@\endcsname{##1}{#2##2}}%
```

```
4589         {\csname @#4@\endcsname{##1}{#2##2}[]}%
```

```
4590       }%
```

```
4591     }%
```

```

4592      {%
4593      \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2] [] {%
4594      \new@ifnextchar [%
4595      {\csname @#4@\endcsname{#1,##1}{#2##2}}}%
4596      {\csname @#4@\endcsname{#1,##1}{#2##2} [] }%
4597      }%
4598      }%
4599  }%
4600 }

```

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

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

```

4601 \newrobustcmd*{\glstrnewgls}[3] [] {%
4602   \@glstrnewgls{#1}{#2}{#3}{gls}%
4603 }

```

`\glstrnewglslike` 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.

```

4604 \newrobustcmd*{\glstrnewglslike}[6] [] {%
4605   \@glstrnewgls{#1}{#2}{#3}{gls}%
4606   \@glstrnewgls{#1}{#2}{#4}{glspl}%
4607   \@glstrnewgls{#1}{#2}{#5}{Gls}%
4608   \@glstrnewgls{#1}{#2}{#6}{Glspl}%
4609 }

```

`\glstrnewGLSlike` 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.

```

4610 \newrobustcmd*{\glstrnewGLSlike}[4] [] {%
4611   \@glstrnewgls{#1}{#2}{#3}{GLS}%
4612   \@glstrnewgls{#1}{#2}{#4}{GLSpl}%
4613 }

```

`\glstrnewrgls` As `\glstrnewgls` but for `\rgls`.

```

4614 \newrobustcmd*{\glstrnewrgls}[3] [] {%
4615   \@glstrnewgls{#1}{#2}{#3}{rgls}%
4616 }

```

`\glstrnewrglslike` As `\glstrnewglslike` but for `\rgls` etc.

```

4617 \newrobustcmd*{\glstrnewrglslike}[6] [] {%
4618   \@glstrnewgls{#1}{#2}{#3}{rgls}%
4619   \@glstrnewgls{#1}{#2}{#4}{rglspl}%
4620   \@glstrnewgls{#1}{#2}{#5}{rGls}%
4621   \@glstrnewgls{#1}{#2}{#6}{rGlspl}%
4622 }

```

`\sxtrenwrGLSlike` As `\glxtrnewGLSlike` but for `\rGLS` etc.

```
4623 \newrobustcmd*{\glxtrnewrGLSlike}[4][\]{%
4624   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4625   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4626 }
```

Provide easy access to record count fields.

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

```
4627 \newcommand*{\GlsXtrTotalRecordCount}[1]{%
4628   \ifcsdef{glo@\glsetoklabel{#1}@recordcount}%
4629   {\csname glo@\glsetoklabel{#1}@recordcount\endcsname}%
4630   {0}%
4631 }
```

`\sXtrRecordCount` Access record count for a particular counter. The first argument is the label. The second argument is the counter name.

```
4632 \newcommand*{\GlsXtrRecordCount}[2]{%
4633   \ifcsdef{glo@\glsetoklabel{#1}@recordcount.#2}%
4634   {\csname glo@\glsetoklabel{#1}@recordcount.#2\endcsname}%
4635   {0}%
4636 }
```

`\locationRecordCount` Access record count for a particular counter and location. The first argument is the label. The second argument is the counter name. The third argument is the location. This command shouldn't be used if the location doesn't fully expand unless `\glxtrdetoklocation` can be set to something sensible.

```
4637 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
4638   \ifcsdef{glo@\glsetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
4639   {\csname glo@\glsetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
4640   {0}%
4641 }
```

`\trdetoklocation`

```
4642 \newcommand*{\glxtrdetoklocation}[1]{#1}
```

`\ablerecordcount`

```
4643 \newcommand*{\glxtrenablerecordcount}{%
4644   \renewcommand*{\gls}{\rgls}%
4645   \renewcommand*{\Gls}{\rGls}%
4646   \renewcommand*{\glspl}{\rglspl}%
4647   \renewcommand*{\Glspl}{\rGlspl}%
4648   \renewcommand*{\GLS}{\rGLS}%
4649   \renewcommand*{\GLSpl}{\rGLSpl}%
4650 }
```

`\ordtriggervalue` The value used by the record trigger test. The argument is the entry's label.

```
4651 \newcommand*{\glxtrrecordtriggervalue}[1]{%

```

```

4652 \GlsXtrTotalRecordCount{#1}%
4653 }

```

dCountAttribute

```

4654 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4655 \@for\@glxstr@cat:=#1\do
4656 {%
4657 \ifdefempty{\@glxstr@cat}{}%
4658 {%
4659 \glsssetcategoryattribute{\@glxstr@cat}{recordcount}{#2}%
4660 }%
4661 }%
4662 }

```

trifrecordtrigger

```
\glxstrifrecordtrigger{<label>}{<trigger format>}{<normal>}
```

```

4663 \newcommand*{\glxstrifrecordtrigger}[3]{%
4664 \glshasattribute{#1}{recordcount}%
4665 {%
4666 \ifnum\glxstrrecordtriggervalue{#1}>\glsggetattribute{#1}{recordcount}\relax
4667 #3%
4668 \else
4669 #2%
4670 \fi
4671 }%
4672 {#3}%
4673 }

```

strigger@record Still need a record to ensure that bib2gls selects the entry.

```

4674 \newcommand*{\@glxstr@rglstrigger@record}[3]{%
4675 \edef\glslabel{\glsdetoklabel{#2}}%
4676 \let\@gls@link@label\glslabel
4677 \def\@glxstr@thevalue{%
4678 \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4679 \def\@glsnumberformat{glstriggerrecordformat}%
4680 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4681 \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4682 \def\@glxstr@thevalue{%
4683 \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4684 \glxstrinitwrgloss
4685 \glslinkpresetkeys
4686 \setkeys{glslink}{#1}%
4687 \glslinkpostsetkeys
4688 \ifdefempty{\@glxstr@thevalue}%
4689 {%
4690 \@gls@saveentrycounter

```



```

4691 }%
4692 {%
4693   \let\theglsentrycounter\@glstr@thevalue
4694   \def\theHglentrycounter{\@glstr@theHvalue}%
4695 }%
4696 \ifglstrinitwrglossbefore
4697   \@do@wrglossary{#2}%
4698 \fi
4699 #3%
4700 \ifglstrinitwrglossbefore
4701 \else
4702   \@do@wrglossary{#2}%
4703 \fi
4704 \ifKV@glslink@local
4705   \glslocalunset{#2}%
4706 \else
4707   \glsunset{#2}%
4708 \fi
4709 }

```

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by bib2gls.

```

4710 \newcommand*{\glstriggerrecordformat}[1]{%

```

\rgls

```

4711 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}

```

\@rgls

```

4712 \newcommand*{\@rgls}[2][{}]{%
4713   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4714 }

```

\@rgls@

```

4715 \def\@rgls@#1#2[#3]{%
4716   \glstrifrecordtrigger{#2}%
4717   {%
4718     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4719   }%
4720   {%
4721     \@gls@{#1}{#2}[#3]%
4722   }%
4723 }%

```

\rglsp1

```

4724 \newrobustcmd*{\rglsp1}{\@gls@hyp@opt\@rglsp1}

```

\@rglsp1

```

4725 \newcommand*{\@rglsp1}[2][{}]{%

```

```

4726 \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
4727 }

```

\@rglspl@

```

4728 \def\@rglspl@#1#2[#3]{%
4729 \glxtrifrecordtrigger{#2}%
4730 {%
4731 \@glxtr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
4732 }%
4733 {%
4734 \@rglspl@{#1}{#2}[#3]%
4735 }%
4736 }%

```

\rGls

```

4737 \newrobustcmd*{\rGls}{\@gls@hyp@opt\rGls}

```

\@rGls

```

4738 \newcommand*{\@rGls}[2][]{%
4739 \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4740 }

```

\@rGls@

```

4741 \def\@rGls@#1#2[#3]{%
4742 \glxtrifrecordtrigger{#2}%
4743 {%
4744 \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4745 }%
4746 {%
4747 \@Gls@{#1}{#2}[#3]%
4748 }%
4749 }%

```

\rGlspl

```

4750 \newrobustcmd*{\rGlspl}{\@gls@hyp@opt\rGlspl}

```

\@rGlspl

```

4751 \newcommand*{\@rGlspl}[2][]{%
4752 \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4753 }

```

\@rGlspl@

```

4754 \def\@rGlspl@#1#2[#3]{%
4755 \glxtrifrecordtrigger{#2}%
4756 {%
4757 \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4758 }%
4759 {%

```

```

4760 \@GLspl@{#1}-{#2}[#3]%
4761 }%
4762 }%

```

\rGLS

```

4763 \newrobustcmd*{\rGLS}{\@gls@hyp@opt\rGLS}

```

\@rGLS

```

4764 \newcommand*{\@rGLS}[2][{}]{%
4765 \new@ifnextchar[{\@rGLS@{#1}-{#2}}{\@rGLS@{#1}-{#2}[]}%
4766 }

```

\@rGLS@

```

4767 \def\rGLS@#1#2[#3]{%
4768 \glsxtrifrecordtrigger{#2}%
4769 {%
4770 \@glsxtr@rglstrigger@record{#1}-{#2}{\rGLSformat{#2}{#3}}%
4771 }%
4772 {%
4773 \@GLS@{#1}-{#2}[#3]%
4774 }%
4775 }%

```

\rGLSpl

```

4776 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}

```

\@rGLSpl

```

4777 \newcommand*{\@rGLSpl}[2][{}]{%
4778 \new@ifnextchar[{\@rGLSpl@{#1}-{#2}}{\@rGLSpl@{#1}-{#2}[]}%
4779 }

```

\@rGLSpl@

```

4780 \def\rGLSpl@#1#2[#3]{%
4781 \glsxtrifrecordtrigger{#2}%
4782 {%
4783 \@glsxtr@rglstrigger@record{#1}-{#2}{\rGLSplformat{#2}{#3}}%
4784 }%
4785 {%
4786 \@GLSpl@{#1}-{#2}[#3]%
4787 }%
4788 }%

```

\rglsformat

```

4789 \newcommand*{\rglsformat}[2]{%
4790 \glsifregular{#1}
4791 {\glsentryfirst{#1}}%
4792 {\ifglsashaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4793 }

```

```

\rglsplformat
4794 \newcommand*{\rglsplformat}[2]{%
4795   \glsifregular{#1}
4796   {\glsentryfirstplural{#1}}%
4797   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4798 }

\rglsformat
4799 \newcommand*{\rglsformat}[2]{%
4800   \glsifregular{#1}
4801   {\Glsentryfirst{#1}}%
4802   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4803 }

\rglsplformat
4804 \newcommand*{\rglsplformat}[2]{%
4805   \glsifregular{#1}
4806   {\Glsentryfirstplural{#1}}%
4807   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4808 }

\rglsformat
4809 \newcommand*{\rglsformat}[2]{%
4810   \expandafter\mfirstucMakeUppercase\expandafter{\rglsformat{#1}{#2}}%
4811 }

\rglsplformat
4812 \newcommand*{\rglsplformat}[2]{%
4813   \expandafter\mfirstucMakeUppercase\expandafter{\rglsplformat{#1}{#2}}%
4814 }

```

1.4 Link Counting

This is different to the entry counting provided by the base package (which counts the number of times the first use flag is unset). Instead, this method hooks into `\@gls@link` (through `\glsxtr@inc@linkcount`) to increment an associated counter. To preserve resources, the counter is only defined if it needs to be incremented. This method is independent of the presence of hyperlinks. (The “link” part of the name refers to `\@gls@link` not `\hyperlink`.)

`\@inc@linkcount` This performs the actual incrementing and counter definition. The counter is given by `\c@glxtr@linkcount@<label>` where `<label>` is the entry’s label. Since this is performed within `\@gls@link` the label can be accessed with `\glslabel`.

```

4815 \newcommand{\@glsxtr@do@inc@linkcount}{%
  Does this entry have the linkcount attribute set?
4816   \glsifattribute{\glslabel}{linkcount}{true}%
4817   {%

```

Does the counter exist?

```
4818 \ifcsdef{c@glxtr@linkcount@\glslabel}{}%  
4819 {%
```

Counter doesn't exist, so define it.

```
4820 \newcounter{glxtr@linkcount@\glslabel}%
```

If linkcountmaster is set, add to counter reset.

```
4821 \glshasattribute{\glslabel}{linkcountmaster}%  
4822 {%
```

Need to ensure values are fully expanded.

```
4823 \begingroup  
4824 \edef\x{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%  
4825 {\glsggetattribute{\glslabel}{linkcountmaster}}}%  
4826 \x  
4827 }%  
4828 {}%  
4829 }%
```

Increment counter:

```
4830 \glxtrinclinkcounter{glxtr@linkcount@\glslabel}%  
4831 }%  
4832 {}%  
4833 }
```

`\reflinkcounter` May be redefined to use `\refstepcounter` if required.

```
4834 \newcommand*{\glxtrinclinkcounter}[1]{\stepcounter{#1}}
```

`\linkCounterValue` Expands to the associated link counter register or 0 if not defined.

```
4835 \newcommand*{\GlsXtrLinkCounterValue}[1]{%  
4836 \ifcsundef{c@glxtr@linkcount@#1}{0}{\csname c@glxtr@linkcount@#1\endcsname}%  
4837 }
```

`\rTheLinkCounter` Expands to the display value of the associated link counter or 0 if not defined.

```
4838 \newcommand*{\GlsXtrTheLinkCounter}[1]{%  
4839 \ifcsundef{theglxtr@linkcount@#1}{0}%  
4840 {\csname theglxtr@linkcount@#1\endcsname}%  
4841 }
```

`\fLinkCounterDef` Tests if the counter has been defined

```
4842 \newcommand*{\GlsXtrIfLinkCounterDef}[3]{%  
4843 \ifcsundef{theglxtr@linkcount@#1}{#3}{#2}%  
4844 }
```

`\LinkCounterName` Expands to the associated link counter name. (No check for existence.)

```
4845 \newcommand*{\GlsXtrLinkCounterName}[1]{glxtr@linkcount@#1}
```

ableLinkCounting

```
\GlsXtrEnableLinkCounting[(master counter)]{(categories)}
```

Enable link counting for the given categories.

```
4846 \newcommand*{\GlsXtrEnableLinkCounting}[2][]{%
4847   \let\glstr@inc@linkcount\@glstr@do@inc@linkcount
4848   \@for\@glstr@label:=#2\do
4849   {%
4850     \glsssetcategoryattribute{\@glstr@label}{linkcount}{true}%
4851     \ifstrepty{#1}{}%
4852     {%
4853       \ifcsundef{c@#1}%
4854       {\@nocounterr{#1}}%
4855       {\glsssetcategoryattribute{\@glstr@label}{linkcountmaster}{#1}}%
4856     }%
4857   }%
4858 }
4859 \onlypreamble\GlsXtrEnableLinkCounting
```

1.5 Integration with glossaries-accsupp

Provide better integration with the glossaries-accsupp package. (Must be loaded before the main code of glossaries-extra either explicitly or through the accsupp package option.)

These commands have their definitions set according to whether or not glossaries-extra has been loaded.

```
4860 \ifpackageloaded{glossaries-accsupp}
4861 {
```

Define (or redefine) commands to use the accessibility information.

`\glsaccessname` Display the name value (no link and no check for existence).

```
4862   \newcommand*{\glsaccessname}[1]{%
4863     \glsnameaccessdisplay
4864     {%
4865       \glstryname{#1}%
4866     }%
4867     {#1}%
4868   }
```

`\Glsaccessname` Display the name value (no link and no check for existence) with the first letter converted to upper case.

```
4869   \newcommand*{\Glsaccessname}[1]{%
4870     \glsnameaccessdisplay
4871     {%
4872       \Glsstryname{#1}%
4873     }%
4874     {#1}%
4875   }
```

`\GLSaccessname` Display the name value (no link and no check for existence) converted to upper case.

```
4876 \newcommand*{\GLSaccessname}[1]{%
4877   \glnameaccessdisplay
4878   {%
4879     \mfirstucMakeUppercase{\glentryname{#1}}%
4880   }%
4881   {#1}%
4882 }
```

`\glsaccesstext` Display the text value (no link and no check for existence).

```
4883 \newcommand*{\glsaccesstext}[1]{%
4884   \glstextaccessdisplay
4885   {%
4886     \glentrytext{#1}%
4887   }%
4888   {#1}%
4889 }
```

`\Glsaccesstext` Display the text value (no link and no check for existence) with the first letter converted to upper case.

```
4890 \newcommand*{\Glsaccesstext}[1]{%
4891   \glstextaccessdisplay
4892   {%
4893     \Glsentrytext{#1}%
4894   }%
4895   {#1}%
4896 }
```

`\GLSAccessstext` Display the text value (no link and no check for existence) converted to upper case.

```
4897 \newcommand*{\GLSAccessstext}[1]{%
4898   \glstextaccessdisplay
4899   {%
4900     \mfirstucMakeUppercase{\glentrytext{#1}}%
4901   }%
4902   {#1}%
4903 }
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```
4904 \newcommand*{\glsaccessplural}[1]{%
4905   \glspluralaccessdisplay
4906   {%
4907     \glentryplural{#1}%
4908   }%
4909   {#1}%
4910 }
```

`Glsaccessplural` Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

4911 \newcommand*{\Glsaccessplural}[1]{%
4912   \glspluralaccessdisplay
4913   {%
4914     \Glsentryplural{#1}%
4915   }%
4916   {#1}%
4917 }

```

\Glsaccessplural Display the plural value (no link and no check for existence) converted to upper case.

```

4918 \newcommand*{\GLSaccessplural}[1]{%
4919   \glspluralaccessdisplay
4920   {%
4921     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4922   }%
4923   {#1}%
4924 }

```

\glsaccessfirst Display the first value (no link and no check for existence).

```

4925 \newcommand*{\glsaccessfirst}[1]{%
4926   \glsfirstaccessdisplay
4927   {%
4928     \glsentryfirst{#1}%
4929   }%
4930   {#1}%
4931 }

```

\Glsaccessfirst Display the first value (no link and no check for existence) with the first letter converted to upper case.

```

4932 \newcommand*{\Glsaccessfirst}[1]{%
4933   \glsfirstaccessdisplay
4934   {%
4935     \Glsentryfirst{#1}%
4936   }%
4937   {#1}%
4938 }

```

\GLSaccessfirst Display the first value (no link and no check for existence) converted to upper case.

```

4939 \newcommand*{\GLSaccessfirst}[1]{%
4940   \glsfirstaccessdisplay
4941   {%
4942     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4943   }%
4944   {#1}%
4945 }

```

\glsaccessfirstplural Display the firstplural value (no link and no check for existence).

```

4946 \newcommand*{\glsaccessfirstplural}[1]{%
4947   \glsfirstpluralaccessdisplay

```



```

4948   {%
4949     \glstentryfirstplural{#1}%
4950   }%
4951   {#1}%
4952 }

```

glsaccessfirstplural Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

```

4953 \newcommand*{\Glsaccessfirstplural}[1]{%
4954   \glstfirstpluralaccessdisplay
4955   {%
4956     \glstentryfirstplural{#1}%
4957   }%
4958   {#1}%
4959 }

```

GLSaccessfirstplural Display the firstplural value (no link and no check for existence) converted to upper case.

```

4960 \newcommand*{\GLSaccessfirstplural}[1]{%
4961   \glstfirstpluralaccessdisplay
4962   {%
4963     \mfirstucMakeUppercase{\glstentryfirstplural{#1}}%
4964   }%
4965   {#1}%
4966 }

```

glsaccesssymbol Display the symbol value (no link and no check for existence).

```

4967 \newcommand*{\glsaccesssymbol}[1]{%
4968   \glssymbolaccessdisplay
4969   {%
4970     \glstentrysymbol{#1}%
4971   }%
4972   {#1}%
4973 }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4974 \newcommand*{\GLSaccesssymbol}[1]{%
4975   \glssymbolaccessdisplay
4976   {%
4977     \glstentrysymbol{#1}%
4978   }%
4979   {#1}%
4980 }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```

4981 \newcommand*{\GLSaccesssymbol}[1]{%
4982   \glssymbolaccessdisplay
4983   {%

```

```

4984     \mfirstucMakeUppercase{\glentrysymbol{#1}}%
4985   }%
4986   {#1}%
4987 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence).

```

4988 \newcommand*{\glaccesssymbolplural}[1]{%
4989   \glssymbolpluralaccessdisplay
4990   {%
4991     \glentrysymbolplural{#1}%
4992   }%
4993   {#1}%
4994 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4995 \newcommand*{\Glsaccesssymbolplural}[1]{%
4996   \glssymbolpluralaccessdisplay
4997   {%
4998     \glentrysymbolplural{#1}%
4999   }%
5000   {#1}%
5001 }

```

`\esssymbolplural` Display the symbolplural value (no link and no check for existence) converted to upper case.

```

5002 \newcommand*{\GLSaccesssymbolplural}[1]{%
5003   \glssymbolpluralaccessdisplay
5004   {%
5005     \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
5006   }%
5007   {#1}%
5008 }

```

`\glaccessdesc` Display the desc value (no link and no check for existence).

```

5009 \newcommand*{\glaccessdesc}[1]{%
5010   \glsdescriptionaccessdisplay
5011   {%
5012     \glentrydesc{#1}%
5013   }%
5014   {#1}%
5015 }

```

`\Glsaccessdesc` Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

5016 \newcommand*{\Glsaccessdesc}[1]{%
5017   \glsdescriptionaccessdisplay
5018   {%
5019     \glentrydesc{#1}%

```

```

5020     }%
5021     {#1}%
5022 }

```

`\GLSaccessdesc` Display the desc value (no link and no check for existence) converted to upper case.

```

5023 \newcommand*{\GLSaccessdesc}[1]{%
5024   \glsdescriptionaccessdisplay
5025   {%
5026     \mfirstucMakeUppercase{\glentrydesc{#1}}%
5027   }%
5028   {#1}%
5029 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence).

```

5030 \newcommand*{\glsaccessdescplural}[1]{%
5031   \glsdescriptionpluralaccessdisplay
5032   {%
5033     \glentrydescplural{#1}%
5034   }%
5035   {#1}%
5036 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.

```

5037 \newcommand*{\Glsaccessdescplural}[1]{%
5038   \glsdescriptionpluralaccessdisplay
5039   {%
5040     \Glsentrydescplural{#1}%
5041   }%
5042   {#1}%
5043 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

5044 \newcommand*{\GLSaccessdescplural}[1]{%
5045   \glsdescriptionpluralaccessdisplay
5046   {%
5047     \mfirstucMakeUppercase{\glentrydescplural{#1}}%
5048   }%
5049   {#1}%
5050 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5051 \newcommand*{\glsaccessshort}[1]{%
5052   \glsshortaccessdisplay
5053   {%
5054     \glentryshort{#1}%
5055   }%
5056   {#1}%
5057 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5058 \newcommand*\Glsaccessshort}[1]{%
5059 \glsshortaccessdisplay
5060 {%
5061 \Glsentryshort{#1}%
5062 }%
5063 {#1}%
5064 }

```

`\GLSaccessshort` Display the short value (no link and no check for existence) converted to upper case.

```

5065 \newcommand*\GLSaccessshort}[1]{%
5066 \glsshortaccessdisplay
5067 {%
5068 \mfirstucMakeUppercase{\Glsentryshort{#1}}%
5069 }%
5070 {#1}%
5071 }

```

`\lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

5072 \newcommand*\lsaccessshortpl}[1]{%
5073 \glsshortpluralaccessdisplay
5074 {%
5075 \Glsentryshortpl{#1}%
5076 }%
5077 {#1}%
5078 }

```

`\lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5079 \newcommand*\lsaccessshortpl}[1]{%
5080 \glsshortpluralaccessdisplay
5081 {%
5082 \Glsentryshortpl{#1}%
5083 }%
5084 {#1}%
5085 }

```

`\LSaccessshortpl` Display the shortplural value (no link and no check for existence) converted to upper case.

```

5086 \newcommand*\LSaccessshortpl}[1]{%
5087 \glsshortpluralaccessdisplay
5088 {%
5089 \mfirstucMakeUppercase{\Glsentryshortpl{#1}}%
5090 }%
5091 {#1}%
5092 }

```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

5093 \newcommand*\glsaccesslong}[1]{%
5094   \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
5095 }

```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```

5096
5097 \newcommand*\Glsaccesslong}[1]{%
5098   \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
5099 }

```

`\GLSaccesslong` Display the long value (no link and no check for existence) converted to upper case.

```

5100 \newcommand*\GLSaccesslong}[1]{%
5101   \glslongaccessdisplay
5102   {%
5103     \mfirstucMakeUppercase{\glsentrylong{#1}}%
5104   }%
5105   {#1}%
5106 }

```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5107 \newcommand*\glsaccesslongpl}[1]{%
5108   \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5109 }

```

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5110
5111 \newcommand*\Glsaccesslongpl}[1]{%
5112   \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5113 }

```

`GLSaccesslongpl` Display the longplural value (no link and no check for existence) converted to upper case.

```

5114 \newcommand*\GLSaccesslongpl}[1]{%
5115   \glslongpluralaccessdisplay
5116   {%
5117     \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5118   }%
5119   {#1}%
5120 }

```

End of if part

```

5121 }
5122 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

`\glsaccessname` Display the name value (no link and no check for existence).

```

5123 \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.

5124 `\newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}`

`\GLSaccessname` Display the name value (no link and no check for existence). converted to upper case.

5125 `\newcommand*{\GLSaccessname}[1]{%`
5126 `\protect\mfirstucMakeUppercase{\glsentryname{#1}}}`

`\glsaccesstext` Display the text value (no link and no check for existence).

5127 `\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.

5128 `\newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}`

`\GLSaccesstext` Display the text value (no link and no check for existence). converted to upper case.

5129 `\newcommand*{\GLSaccesstext}[1]{%`
5130 `\protect\mfirstucMakeUppercase{\glsentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).

5131 `\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.

5132 `\newcommand*{\GLsaccessplural}[1]{\GLsentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.

5133 `\newcommand*{\GLSaccessplural}[1]{%`
5134 `\protect\mfirstucMakeUppercase{\glsentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).

5135 `\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.

5136 `\newcommand*{\GLsaccessfirst}[1]{\GLsentryfirst{#1}}`

`\GLSaccessfirst` Display the first value (no link and no check for existence). converted to upper case.

5137 `\newcommand*{\GLSaccessfirst}[1]{%`
5138 `\protect\mfirstucMakeUppercase{\glsentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).

5139 `\newcommand*{\glsaccessfirstplural}[1]{\glsentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence) with the first letter converted to upper case.

5140 `\newcommand*{\GLsaccessfirstplural}[1]{\GLsentryfirstplural{#1}}`

`GLSaccessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.
5141 `\newcommand*{\GLSaccessfirstplural}[1]{%`
5142 `\protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).
5143 `\newcommand*{\glsaccesssymbol}[1]{\glentrysymbol{#1}}`

`GLsaccesssymbol` Display the symbol value (no link and no check for existence) with the first letter converted to upper case.
5144 `\newcommand*{\GLsaccesssymbol}[1]{\GLentrysymbol{#1}}`

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.
5145 `\newcommand*{\GLSaccesssymbol}[1]{%`
5146 `\protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}`

`glsaccesssymbolplural` Display the symbolplural value (no link and no check for existence).
5147 `\newcommand*{\glsaccesssymbolplural}[1]{\glentrysymbolplural{#1}}`

`GLsaccesssymbolplural` Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
5148 `\newcommand*{\GLsaccesssymbolplural}[1]{\GLentrysymbolplural{#1}}`

`GLSaccesssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.
5149 `\newcommand*{\GLSaccesssymbolplural}[1]{%`
5150 `\protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}`

`\glsaccessdesc` Display the desc value (no link and no check for existence).
5151 `\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.
5152 `\newcommand*{\GLsaccessdesc}[1]{\GLentrydesc{#1}}`

`\GLSaccessdesc` Display the desc value (no link and no check for existence). converted to upper case.
5153 `\newcommand*{\GLSaccessdesc}[1]{%`
5154 `\protect\mfirstucMakeUppercase{\glentrydesc{#1}}}`

`GLSaccessdescplural` Display the descplural value (no link and no check for existence).
5155 `\newcommand*{\GLSaccessdescplural}[1]{\GLentrydescplural{#1}}`

`GLsaccessdescplural` Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
5156 `\newcommand*{\GLsaccessdescplural}[1]{\GLentrydescplural{#1}}`

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.

```

5157 \newcommand*{\GLSaccessdescplural}[1]{%
5158 \protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}
```

\glsaccessshort Display the short form (no link and no check for existence).

```

5159 \newcommand*{\glsaccessshort}[1]{\glentryshort{#1}}
```

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5160 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}
```

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.

```

5161 \newcommand*{\GLSaccessshort}[1]{%
5162 \protect\mfirstucMakeUppercase{\glentryshort{#1}}}
```

lsaccessshortpl Display the short plural form (no link and no check for existence).

```

5163 \newcommand*{\glsaccessshortpl}[1]{\glentryshortpl{#1}}
```

lsaccessshortpl Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5164 \newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

LSaccessshortpl Display the shortplural value (no link and no check for existence). converted to upper case.

```

5165 \newcommand*{\GLSaccessshortpl}[1]{%
5166 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}
```

\glsaccesslong Display the long form (no link and no check for existence).

```

5167 \newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}
```

\Glsaccesslong Display the long form (no link and no check for existence).

```

5168 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

\GLSaccesslong Display the long value (no link and no check for existence). converted to upper case.

```

5169 \newcommand*{\GLSaccesslong}[1]{%
5170 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}
```

glsaccesslongpl Display the long plural form (no link and no check for existence).

```

5171 \newcommand*{\glsaccesslongpl}[1]{\glentrylongpl{#1}}
```

Glsaccesslongpl Display the long plural form (no link and no check for existence).

```

5172 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

GLSaccesslongpl Display the longplural value (no link and no check for existence). converted to upper case.

```

5173 \newcommand*{\GLSaccesslongpl}[1]{%
5174 \protect\mfirstucMakeUppercase{\glentrylongpl{#1}}}
```

End of else part

```

5175 }
```


1.6 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5176 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5177 \newcommand{\glsifcategory}[4]{%
5178   \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5179 }
```

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
5180 \newcommand*{\glssetcategoryattribute}[3]{%
5181   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
5182 }
```

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5183 \newcommand*{\glsgetcategoryattribute}[2]{%
5184   \csuse{@glxtr@categoryattr@@#1@#2}%
5185 }
```

`categoryattribute` `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5186 \newcommand*{\glshascategoryattribute}[4]{%
5187   \ifcvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
5188 }
```

`\glssetattribute` `\glssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5189 \newcommand*{\glssetattribute}[3]{%
```

```

5190 \glsssetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5191 }

```

`\glsggetattribute` `\glsggetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```

5192 \newcommand*{\glsggetattribute}[2]{%
5193   \glsssetcategoryattribute{\glscategory{#1}}{#2}%
5194 }

```

`\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.

```

5195 \newcommand*{\glshasattribute}[4]{%
5196   \ifglssentryexists{#1}%
5197   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5198   {#4}%
5199 }

```

`categoryattribute` `\glssifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```

5200 \newcommand{\glssifcategoryattribute}[5]{%
5201   \ifcsundef{@glssxtr@categoryattr@#1@#2}%
5202   {#5}%
5203   {\ifcsstring{@glssxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5204 }

```

`\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.

```

5205 \newcommand{\glssifattribute}[5]{%
5206   \ifglssentryexists{#1}%
5207   {\glssifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5208   {#5}%
5209 }

```

Set attributes for the default general category:

```
5210 \glsssetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5211 \glsssetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```
5212 \newcommand*{\glsssetregularcategory}[1]{%  
5213 \glsssetcategoryattribute{#1}{regular}{true}}%  
5214 }
```

`ifregularcategory` `\glssifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5215 \newcommand{\glssifregularcategory}[3]{%  
5216 \glssifcategoryattribute{#1}{regular}{true}{#2}{#3}}%  
5217 }
```

`ifnotregularcategory` `\glssifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5218 \newcommand{\glssifnotregularcategory}[3]{%  
5219 \glssifcategoryattribute{#1}{regular}{false}{#2}{#3}}%  
5220 }
```

`\glssifregular` `\glssifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5221 \newcommand{\glssifregular}[3]{%  
5222 \glssifregularcategory{\glsscategor{#1}}{#2}{#3}}%  
5223 }
```

`\glssifnotregular` `\glssifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5224 \newcommand{\glssifnotregular}[3]{%  
5225 \glssifnotregularcategory{\glsscategor{#1}}{#2}{#3}}%  
5226 }
```

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.

```
5227 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
5228   \forallglossaries[#1]{#3}%
5229   {%
5230     \forlgsentries[#3]{#4}%
5231     {%
5232       \glsifcategory{#4}{#2}{#5}{}%
5233     }%
5234   }%
5235 }
```

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.

```
5236 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5237   \forallglossaries[#1]{#4}%
5238   {%
5239     \forlgsentries[#4]{#5}%
5240     {%
5241       \glsifattribute{#5}{#2}{#3}{#6}{}%
5242     }%
5243   }%
5244 }
```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```
5245 \ifdef\newterm
5246 {%
```

`\newterm`

```
5247   \renewcommand*{\newterm}[2][ ]{%
5248     \newglossaryentry{#2}%
5249     {type={index},category=index,name={#2},%
```

```

5250     description={\glxtrpostdescription\nopostdesc},#1}%
5251 }

```

Indexed terms are regular by default.

```

5252 \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5253 \newcommand*{\glxtrpostdescindex}{}

5254 }
5255 {}

```

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.

```

5256 \ifdef\printsymbols
5257 {%

```

glxtrnewsymbol Unlike \newterm, this has a separate argument for the label (since the symbol will likely contain commands).

```

5258 \newcommand*{\glxtrnewsymbol}[3] [] {%
5259     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5260 }

```

Symbols are regular by default.

```

5261 \glsssetcategoryattribute{symbol}{regular}{true}

```

rpostdescsymbol

```

5262 \newcommand*{\glxtrpostdescsymbol}{}

5263 }
5264 {}

```

Similar for the numbers option.

```

5265 \ifdef\printnumbers
5266 {%

```

glxtrnewnumber

```

5267 \ifdef\printnumbers
5268 \newcommand*{\glxtrnewnumber}[3] [] {%
5269     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5270 }

```

Numbers are regular by default.

```

5271 \glsssetcategoryattribute{number}{regular}{true}

```

rpostdescnumber

```

5272 \newcommand*{\glxtrpostdescnumber}{}

```

```
5273 }
5274 {}
```

`\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.

```
5275 \newcommand*{\glstrsetcategory}[2]{%
5276   \@for\@glstr@label:=#1\do
5277   {%
5278     \glsfieldxdef{\@glstr@label}{category}{#2}%
5279   }%
5280 }
```

`\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.

```
5281 \newcommand*{\glstrcategoryforall}[2]{%
5282   \forallglossaries[#1]{\@glstr@type}{%
5283     \forallsentries[\@glstr@type]{\@glstr@label}%
5284     {%
5285       \glsfieldxdef{\@glstr@label}{category}{#2}%
5286     }%
5287   }%
5288 }
```

`\glstrfieldtitlecase`

```
\glstrfieldtitlecase{<label>}{<field>}
```

Apply title casing to the contents of the given field.

```
5289 \newcommand*{\glstrfieldtitlecase}[2]{%
5290   \expandafter\glstrfieldtitlecasesecs\expandafter
5291   {\csname glo@glsetoklabel{#1}@#2\endcsname}%
5292 }
```

`\glstrfieldtitlecasesecs` The command used by `\glstrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```
5293 \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.

```
5294 \@ifpackageloaded{glossaries-accsupp}
5295 {
5296   \renewcommand*{\glossentrydesc}[1]{%
5297     \glsdoifexistsorwarn{#1}%
5298     {%
5299       \glssetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the glossdescfont attribute to determine the font applied.

```

5300 \glshasattribute{#1}{glossdescfont}%
5301 {%
5302 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5303 \ifcsdef{\@glxtr@attrval}%
5304 {%
5305 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5306 }%
5307 {%
5308 \GlossariesExtraWarning{Unknown control sequence name
5309 '\@glxtr@attrval' supplied in glossdescfont attribute
5310 for entry '#1'. Ignoring}%
5311 \let\@glxtr@glossdescfont\@firstofone
5312 }%
5313 }%
5314 {\let\@glxtr@glossdescfont\@firstofone}%
5315 \glusifattribute{#1}{glossdesc}{firstuc}%
5316 {%
5317 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5318 }%
5319 {%
5320 \glusifattribute{#1}{glossdesc}{title}%
5321 {%
5322 \@glxtr@do@titlecaps@warn
5323 \glsdescriptionaccessdisplay
5324 {%
5325 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5326 }%
5327 {#1}%
5328 }%
5329 {%
5330 \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5331 }%
5332 }%
5333 }%
5334 }
5335 }
5336 {
5337 \renewcommand*{\glossentrydesc}[1]{%
5338 \glsdoifexistsorwarn{#1}%
5339 {%
5340 \glissetabbrvfmt{\glscategory{#1}}%
5341 \glshasattribute{#1}{glossdescfont}%
5342 {%
5343 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossdescfont}}%
5344 \ifcsdef{\@glxtr@attrval}%
5345 {%
5346 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5347 }%

```

```

5348      {%
5349      \GlossariesExtraWarning{Unknown control sequence name
5350      '\@glsxtr@attrval' supplied in glossdescfont attribute
5351      for entry '#1'. Ignoring}%
5352      \let\@glsxtr@glossdescfont\@firstofone
5353      }%
5354    }%
5355    {\let\@glsxtr@glossdescfont\@firstofone}%
5356    \glsifattribute{#1}{glossdesc}{firstuc}%
5357    {%
5358      \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5359    }%
5360    {%
5361      \glsifattribute{#1}{glossdesc}{title}%
5362      {%
5363        \@glsxtr@do@titlecaps@warn
5364        \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5365      }%
5366      {%
5367        \@glsxtr@glossdescfont{\Glsentrydesc{#1}}%
5368      }%
5369    }%
5370  }%
5371 }
5372 }

```

`\glossentryname` If the glossname attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5373 \ifpackageloaded{glossaries-accsupp}
5374 {
5375   \renewcommand*{\glossentryname}[1]{%
5376     \@glsdoifexistsorwarn{#1}%
5377     {%
5378       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5379   \glshasattribute{#1}{glossnamefont}%
5380   {%
5381     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5382     \ifcsdef{\@glsxtr@attrval}%
5383     {%
5384       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5385     }%
5386     {%
5387       \GlossariesExtraWarning{Unknown control sequence name
5388       '\@glsxtr@attrval' supplied in glossnamefont attribute
5389       for entry '#1'. Reverting to default \string\glsnamefont}%
5390       \let\@glsxtr@glossnamefont\glsnamefont
5391     }%
5392   }%

```



```

5393     {\let\@glsxtr@glossnamefont\glsnamefont}%
5394     \glsifattribute{#1}{glossname}{firstuc}%
5395     {%
5396         \glsnameaccessdisplay
5397         {%
5398             \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5399             }%
5400         {#1}%
5401     }%
5402     {%
5403         \glsifattribute{#1}{glossname}{title}%
5404         {%
5405             \@glsxtr@do@titlecaps@warn
5406             \glsnameaccessdisplay
5407             {%
5408                 \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
5409                 }%
5410             {#1}%
5411         }%
5412         {%
5413             \glsifattribute{#1}{glossname}{uc}%
5414             {%
5415                 \glsnameaccessdisplay
5416                 {%

```

Hide the label from the upper-casing command.

```

5417             \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5418             \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5419             }%
5420         {#1}%
5421     }%
5422     {%
5423         \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5424         \glsnameaccessdisplay
5425         {%
5426             \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
5427             }%
5428         {#1}%
5429     }%
5430 }%
5431 }%

```

Do post-name hook:

```

5432     \glsxtrpostnamehook{#1}%
5433     }%
5434 }
5435 }
5436 {
5437     \renewcommand*{\glossentryname}[1]{%
5438         \@glsdoifexistsorwarn{#1}%

```

```

5439  {%
5440      \glsetabbrvfmt{\glscategory{#1}}%
5441      \glshasattribute{#1}{glossnamefont}%
5442      {%
5443          \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5444          \ifcsdef{\@glxtr@attrval}%
5445              {%
5446                  \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5447              }%
5448              {%
5449                  \GlossariesExtraWarning{Unknown control sequence name
5450                      '\@glxtr@attrval' supplied in glossnamefont attribute
5451                      for entry '#1'. Reverting to default \string\glsgnamefont}%
5452                  \let\@glxtr@glossnamefont\glsgnamefont
5453              }%
5454          }%
5455          {\let\@glxtr@glossnamefont\glsgnamefont}%
5456          \glsgifattribute{#1}{glossname}{firstuc}%
5457          {%
5458              \@glxtr@glossnamefont{\Glsentryname{#1}}%
5459          }%
5460          {%
5461              \glsgifattribute{#1}{glossname}{title}%
5462              {%
5463                  \@glxtr@do@titlecaps@warn
5464                  \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
5465              }%
5466              {%
5467                  \glsgifattribute{#1}{glossname}{uc}%
5468                  {%

```

Hide the label from the upper-casing command.

```

5469          \letcs{\glo@name}{\glo@glsgdetoklabel{#1}@name}%
5470          \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5471          }%
5472          {%

```

This little trick is used by glossaries to allow the user to redefine \glsgnamefont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

5473          \letcs{\glo@name}{\glo@glsgdetoklabel{#1}@name}%
5474          \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5475          }%
5476          }%
5477      }%

```

Do post-name hook.

```

5478      \glxtrpostnamehook{#1}%
5479      }%
5480  }
5481 }

```

`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
5482 \@ifpackageloaded{glossaries-accsupp}
5483 {
5484   \renewcommand*{\Glossentryname}[1]{%
5485     \@glsdoifexistsorwarn{#1}%
5486     {%
5487       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
5488     \glshasattribute{#1}{glossnamefont}%
5489     {%
5490       \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5491       \ifcsdef{\@glstr@attrval}%
5492       {%
5493         \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5494       }%
5495       {%
5496         \GlossariesExtraWarning{Unknown control sequence name
5497           '\@glstr@attrval' supplied in glossnamefont attribute
5498           for entry '#1'. Reverting to default \string\glnamefont}%
5499         \let\@glstr@glossnamefont\glnamefont
5500       }%
5501     }%
5502     {\let\@glstr@glossnamefont\glnamefont}%
5503     \glnameaccessdisplay
5504     {%
5505       \@glstr@glossnamefont{\Glsentryname{#1}}%
5506     }%
5507     {#1}%
```

Do post-name hook:

```
5508     \glstrpostnamehook{#1}%
5509   }%
5510 }
5511 }
5512 {
5513   \renewcommand*{\Glossentryname}[1]{%
5514     \@glsdoifexistsorwarn{#1}%
5515     {%
5516       \glsetabbrvfmt{\glscategory{#1}}%
5517       \glshasattribute{#1}{glossnamefont}%
5518       {%
5519         \edef\@glstr@attrval{\glsetattribute{#1}{glossnamefont}}%
5520         \ifcsdef{\@glstr@attrval}%
5521         {%
5522           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5523         }%
5524         {%
5525           \GlossariesExtraWarning{Unknown control sequence name
5526             '\@glstr@attrval' supplied in glossnamefont attribute
```

```

5527         for entry '#1'. Reverting to default \string\glsnamefont}%
5528         \let\@glsxtr@glossnamefont\glsnamefont
5529     }%
5530 }%
5531 {\let\@glsxtr@glossnamefont\glsnamefont}%
5532 \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5533     \glsxtrpostnamehook{#1}%
5534 }%
5535 }
5536 }

```

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.

```

5537 \newcommand*{\glsxtrpostnamehook}[1]{%
5538   \let\@glsnumberformat\@glsxtr@defaultnumberformat
5539   \glsxtrdoautoindexname{#1}{indexname}%

```

Allow additional code regardless of category:

```

5540   \glsextrapostnamehook{#1}%

```

Allow categories to hook in here.

```

5541   \csuse{glsxtrpostname\glscategory{#1}}%
5542 }

```

`trapostnamehook`

```

5543 \newcommand*{\glsextrapostnamehook}[1]{}%

```

`etaccessdisplay`

```

5544 \@ifpackageloaded{glossaries-accsupp}
5545 {
5546   \newcommand*{\glsxtr@setaccessdisplay}[1]{%
5547     \ifcsdef{gls#1accessdisplay}%
5548     {\letcs\@glsxtr@accessdisplay{gls#1accessdisplay}}%
5549     {%

```

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.

```

5550     \edef\@gls@thisval{#1}%
5551     \@for\@gls@map:=\@gls@keymap\do{%
5552       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5553       \ifdefequal{\@this@key}{\@gls@thisval}%
5554       {%
5555         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5556         \@endfortrue

```

```

5557     }%
5558     {}%
5559     }%
5560     \ifcsdef{gls\@gls@thisval accessdisplay}%
5561     {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5562     {\let\@glxtr@accessdisplay\@firstoftwo}%
5563     }%
5564 }
5565 }
5566 {%
5567 \newcommand*{\glxtr@setaccessdisplay}[1]{%
5568 \let\@glxtr@accessdisplay\@firstoftwo}
5569 }

```

sentrynameother Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

5570 \newrobustcmd*{\glossentrynameother}[2]{%
5571 \@glstdoifexistsorwarn{#1}%
5572 {%

```

Accessibility support:

```

5573 \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5574 \glsssetabbrvfmt{\glscategory{#1}}%
5575 \glshasattribute{#1}{glossnamefont}%
5576 {%
5577 \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5578 \ifcsdef{\@glxtr@attrval}%
5579 {%
5580 \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5581 }%
5582 {%
5583 \GlossariesExtraWarning{Unknown control sequence name
5584 '@@glxtr@attrval' supplied in glossnamefont attribute
5585 for entry '#1'. Reverting to default \string\glsgnamefont}%
5586 \let\@glxtr@glossnamefont\glsgnamefont
5587 }%
5588 }%
5589 {\let\@glxtr@glossnamefont\glsgnamefont}%
5590 \glsgifattribute{#1}{glossname}{firstuc}%
5591 {%
5592 \@glxtr@accessdisplay
5593 {\@glxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5594 {#1}%
5595 }%
5596 {%
5597 \glsgifattribute{#1}{glossname}{title}%
5598 {%
5599 \@glxtr@do@titlecaps@warn

```

```

5600 \@glsxtr@accessdisplay
5601 {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
5602 {#1}%
5603 }%
5604 {%
5605 \glsifattribute{#1}{glossname}{uc}%
5606 {%
5607 \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5608 \@glsxtr@accessdisplay
5609 {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5610 {#1}%
5611 }%
5612 {%
5613 \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5614 \@glsxtr@accessdisplay
5615 {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
5616 {#1}%
5617 }%
5618 }%
5619 }%
    Do post-name hook.
5620 \glsxtrpostnamehook{#1}%
5621 }%
5622 }

```

`format@override` Determines if the `format` key should override the indexing attribute value.

```

5623 \newif\if@glsxtr@format@override
5624 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5625 \@ifpackageloaded{hyperref}
5626 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
5627 \ifHy@hyperindex
5628 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5629 \@glsxtr@format@override true
5630 \appto\theindex{\let\glshypernumber\@firstofone}%
5631 }
5632 \else
5633 \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5634 \@glsxtr@format@override true
5635 \appto\theindex{\let\glshypernumber\hyperpage}%
5636 }
5637 \fi

```

```

5638 }
5639 {
5640   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5641     \@glsxtr@format@override true
5642   }
5643 }
5644 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

doautoindexname

```

5645 \newcommand*{\glxtrdoautoindexname}[2]{%
5646   \glshasattribute{#1}{#2}%
5647   {%

```

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.

```

5648   \@glsxtr@autoindex@setname{#1}%

```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```

5649   \protected@edef\@glsxtr@attrval{\glsggetattribute{#1}{#2}}%
5650   \if@glsxtr@format@override

5651     \ifx\@glsnumberformat\@glsxtr@defaultnumberformat
5652     \else
5653       \let\@glsxtr@attrval\@glsnumberformat
5654     \fi
5655   \fi
5656   \ifdefstring{\@glsxtr@attrval}{true}%
5657   {}%
5658   {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
5659   \expandafter\glxtrautoindex\expandafter{\@glo@name}%
5660   }%
5661   {}%
5662 }

```

glxtrautoindex

```

5663 \newcommand*{\glxtrautoindex}{\index}

```

toindex@setname Assign \@glo@name for use with indexname attribute.

```

5664 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
5665   \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
5666   \glxtrautoindexassignsort{\@glo@sort}{#1}%
5667   \@gls@checkmkidxchars\@glo@sort
5668   \@glsxtr@autoindex@doextra@esc\@glo@sort
5669   \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
5670 }

```

rautoindexentry Command used for the actual part when auto-indexing.

```

5671 \newcommand*{\glxtrautoindexentry}[1]{\string\glstryname{#1}}

```

trautoindexsort Used to assign the sort value when auto-indexing.

```
5672 \newcommand*{\glxtrautoindexassignsort}[2]{%
5673   \glsletentryfield{#1}{#2}{sort}%
5674 }
```

dex@doextra@esc

```
5675 \newcommand*{\@glxstr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5676   \ifx\@glxstr@autoindex@esc\@gls@quotechar
5677   \else
5678     \def\@gls@checkedmkidx{}%
5679     \edef\@glxstr@checkspch{%
5680       \noexpand\@glxstr@autoindex@escquote\expandonce{#1}%
5681       \noexpand\@empty\@glxstr@autoindex@esc\noexpand\@nnil
5682       \@glxstr@autoindex@esc\noexpand\@empty\noexpand\@glxstr@endescspch}%
5683     \@glxstr@checkspch
5684     \let#1\@gls@checkedmkidx\relax
5685   \fi
```

Escape actual character unless it has already been escaped.

```
5686   \ifx\@glxstr@autoindex@at\@gls@actualchar
5687   \else
5688     \def\@gls@checkedmkidx{}%
5689     \edef\@glxstr@checkspch{%
5690       \noexpand\@glxstr@autoindex@escat\expandonce{#1}%
5691       \noexpand\@empty\@glxstr@autoindex@at\noexpand\@nnil
5692       \@glxstr@autoindex@at\noexpand\@empty\noexpand\@glxstr@endescspch}%
5693     \@glxstr@checkspch
5694     \let#1\@gls@checkedmkidx\relax
5695   \fi
```

Escape level character unless it has already been escaped.

```
5696   \ifx\@glxstr@autoindex@level\@gls@levelchar
5697   \else
5698     \def\@gls@checkedmkidx{}%
5699     \edef\@glxstr@checkspch{%
5700       \noexpand\@glxstr@autoindex@esclevel\expandonce{#1}%
5701       \noexpand\@empty\@glxstr@autoindex@level\noexpand\@nnil
5702       \@glxstr@autoindex@level\noexpand\@empty\noexpand\@glxstr@endescspch}%
5703     \@glxstr@checkspch
5704     \let#1\@gls@checkedmkidx\relax
5705   \fi
```

Escape encap character unless it has already been escaped.

```
5706   \ifx\@glxstr@autoindex@encap\@gls@encapchar
5707   \else
5708     \def\@gls@checkedmkidx{}%
5709     \edef\@glxstr@checkspch{%
5710       \noexpand\@glxstr@autoindex@escencap\expandonce{#1}%
5711       \noexpand\@empty\@glxstr@autoindex@encap\noexpand\@nnil
```



```

5712      \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5713      \@glsxtr@checkspch
5714      \let#1\@gls@checkedmkidx\relax
5715      \fi
5716 }

```

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`.

```
5717 \newcommand*{\@glsxtr@autoindex@at}{}

```

`\trSetActualChar` Set the actual character.

```

5718 \newcommand*{\GlsXtrSetActualChar}[1]{%
5719   \gdef\@glsxtr@autoindex@at{#1}%
5720   \def\@glsxtr@autoindex@escat##1#1##2#1##3\@glsxtr@endescspch{%
5721     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
5722   }%
5723 }
5724 \@onlypreamble\GlsXtrSetActualChar
5725 \makeatother
5726 \GlsXtrSetActualChar{}
5727 \makeatletter

```

`\autoindex@encap` Encap character for use with `\index`.

```
5728 \newcommand*{\@glsxtr@autoindex@encap}{}

```

`\XtrSetEncapChar` Set the encap character.

```

5729 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5730   \gdef\@glsxtr@autoindex@encap{#1}%
5731   \def\@glsxtr@autoindex@escencap##1#1##2#1##3\@glsxtr@endescspch{%
5732     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
5733   }%
5734 }
5735 \GlsXtrSetEncapChar{}
5736 \@onlypreamble\GlsXtrSetEncapChar

```

`\autoindex@level` Level character for use with `\index`.

```
5737 \newcommand*{\@glsxtr@autoindex@level}{}

```

`\XtrSetLevelChar` Set the encap character.

```

5738 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5739   \gdef\@glsxtr@autoindex@level{#1}%
5740   \def\@glsxtr@autoindex@esclevel##1#1##2#1##3\@glsxtr@endescspch{%
5741     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@esclevel}{##1}{##2}{##3}%
5742   }%
5743 }
5744 \GlsXtrSetLevelChar{!}
5745 \@onlypreamble\GlsXtrSetLevelChar

```

r@autoindex@esc Escape character for use with \index.

```
5746 \newcommand*{\@glsxtr@autoindex@esc}{"
```

lsXtrSetEscChar Set the escape character.

```
5747 \newcommand*{\GlsXtrSetEscChar}[1]{%
5748   \gdef\@glsxtr@autoindex@esc{#1}%
5749   \def\@glsxtr@autoindex@escquote##1##2##3\@glsxtr@endescspch{%
5750     \@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escquote}{##1}{##2}{##3}%
5751   }%
5752 }
5753 \GlsXtrSetEscChar{"}
5754 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
5755 \ifdef\actualchar
5756 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5757 {}
```

Quote character \quotechar:

```
5758 \ifdef\quotechar
5759 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5760 {}
```

Level character \levelchar:

```
5761 \ifdef\levelchar
5762 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5763 {}
```

Encap character \encapchar:

```
5764 \ifdef\encapchar
5765 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5766 {}
```

leto@endescspch

```
5767 \def\@glsxtr@gobbleto@endescspch#1\@glsxtr@endescspch{}
```

toindex@esc@spch

<code>\@glsxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}</code>

```
5768 \newcommand*{\@glsxtr@autoindex@escspch}[5]{%
5769   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
5770   \toks@={#3}%
5771   \ifx\@nnil#3\relax
5772     \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch#5\@glsxtr@endescspch}%
5773   \else
5774     \ifx\@nnil#4\relax
5775       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
5776     \def\@glsxtr@checkspch{\@glsxtr@gobbleto@endescspch
```

```

5777      #4#5\@glxtr@endescspch}%
5778 \else
5779 \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
5780 \@glxtr@autoindex@esc#1}%
5781 \def\@@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
5782 \fi
5783 \fi
5784 \@glxtr@checkspch
5785 }

```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```

5786 \renewcommand*{\Glossentrydesc}[1]{%
5787 \glsdoifexistsorwarn{#1}%
5788 {%
5789 \glssetabbrvfmt{\glscategory{#1}}%
5790 \Glsaccessdesc{#1}%
5791 }%
5792 }

```

\Glossentrysymbol Redefine to set the abbreviation format and accessibility support.

```

5793 \renewcommand*{\Glossentrysymbol}[1]{%
5794 \glsdoifexistsorwarn{#1}%
5795 {%
5796 \glssetabbrvfmt{\glscategory{#1}}%
5797 \Glsaccesssymbol{#1}%
5798 }%
5799 }

```

\Glossentrysymbol Redefine to set the abbreviation format and accessibility support.

```

5800 \renewcommand*{\Glossentrysymbol}[1]{%
5801 \glsdoifexistsorwarn{#1}%
5802 {%
5803 \glssetabbrvfmt{\glscategory{#1}}%
5804 \Glsaccesssymbol{#1}%
5805 }%
5806 }

```

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.

```

5807 \newcommand*{\GlsXtrEnableInitialTagging}{%
5808 \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
5809 }
5810 \@onlypreamble\GlsXtrEnableInitialTagging

```

\GlsXtrEnableInitialTagging Starred version undefines command.

```

5811 \newcommand*{\s@glstr@enabletagging}[2]{%
5812   \undef#2%
5813   \@glstr@enabletagging{#1}{#2}%
5814 }

```

`r@enabletagging` Internal command.

```

5815 \newcommand*{\@glstr@enabletagging}[2]{%
    Set attributes for categories given in the first argument.
5816   \@for\@glstr@cat:=#1\do
5817   {%
5818     \ifdefempty\@glstr@cat
5819     {}%
5820     {\glsssetcategoryattribute{\@glstr@cat}{tagging}{true}}%
5821   }%
5822   \newrobustcmd*#2[1]{##1}%
5823   \def\@glstr@taggingcs{#2}%
5824   \renewcommand*{\@glstr@activate@initialtagging{%
5825     \let#2\@glstr@tag
5826   }%
5827   \ifundef\@glstr@preglossaryhook
5828   {\GlossariesExtraWarning{Initial tagging requires at least
5829     glossaries.sty v4.19 to work correctly}}%
5830   }%
5831 }

```

Are we using an old version of `mfirstuc` that has a bug in `\capitalisewords`? If so, patch it so we don't have a problem with a combination of tagging and title case.

`mfu@checkword@do` If this command hasn't been defined, then we have pre v2.02 of `mfirstuc`

```

5832 \ifundef\mfu@checkword@do
5833 {
5834   \newcommand*{\mfu@checkword@do}[1]{%
5835     \ifdefstring{\mfu@checkword@arg}{#1}%
5836     {%
5837       \let\@mfu@do@makefirstuc\@firstofone
5838       \listbreak
5839     }%
5840   }%
5841 }

```

`\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.

```

5842 \ifundef\mfu@checkword
5843 {
5844   \newcommand{\@glstr@do@titlecaps@warn}{%
5845     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5846       support not available}%

```

One warning should suffice.

```
5847 \let\@glxtr@do@titlecaps@warn\relax
5848 }
5849 }
5850 {
5851 \renewcommand*{\mfu@checkword}[1]{%
5852 \def\mfu@checkword@arg{#1}%
5853 \let\@mfu@domakefirstuc\makefirstuc
5854 \forlistloop\mfu@checkword@do\@mfu@nocaplist
5855 }
5856 }
5857 }
5858 {}% no patch required
```

@titlecaps@warn Do warning if title case not supported.

```
5859 \newcommand*{\@glxtr@do@titlecaps@warn}{}%
```

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.

```
5860 \newcommand*{\@glxtr@activate@initialtagging}{}%
```

\@glxtr@tag Definition of tagging command when used in glossary.

```
5861 \newrobustcmd*{\@glxtr@tag}[1]{%
5862 \glsifattribute{\glscurrententrylabel}{tagging}{true}%
5863 {\glxtrtagfont{#1}}{#1}%
5864 }
```

\glxtrtagfont Used in the glossary.

```
5865 \newcommand*{\glxtrtagfont}[1]{\underline{#1}}
```

preglossaryhook This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't been defined this feature is unavailable. A check is added for the entry's existence to prevent errors from occurring if the user removes an entry or changes the label, which can interrupt the build process.

```
5866 \ifdef\@gls@preglossaryhook
5867 {
5868 \renewcommand*{\@gls@preglossaryhook}{%
5869 \@glxtr@activate@initialtagging
```

Since the glossaries are automatically scoped, \@glxtr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

```
5870 \ifundef\@glxtr@org@postdescription
5871 {%
5872 \let\@glxtr@org@postdescription\glspostdescription
5873 \renewcommand*{\glspostdescription}{%
5874 \ifglentryexists{\glscurrententrylabel}%
5875 {%
5876 \glxtrpostdescription
5877 \@glxtr@org@postdescription
```

```

5878     }%
5879     {}%
5880     }%
5881     }%
5882     {}%

```

Enable the options used by \@glsxtrp:

```

5883     \glossxtrsetpopts
5884     }%
5885 }
5886 {}

```

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.

```

5887 \newcommand*{\glsxtrpostdescription}{%
5888   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5889 }

```

postdescgeneral

```

5890 \newcommand*{\glsxtrpostdescgeneral}{}

```

xtrpostdescterm

```

5891 \newcommand*{\glsxtrpostdescterm}{}

```

postdescacronym

```

5892 \newcommand*{\glsxtrpostdescacronym}{}

```

escabbreviation

```

5893 \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.

```

5894 \renewcommand*{\glspostlinkhook}{%
5895   \ifglsentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
5896 }

```

xtrpostlinkhook The entry label should already be stored in \glslabel by \@gls@link.

```

5897 \newcommand*{\glsxtrpostlinkhook}{%
5898   \glsxtrdiscardperiod{\glslabel}%
5899   {\glsxtrpostlinkendsentence}%
5900   {\glsxtrifcustomdiscardperiod
5901     {\glsxtrifperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}}%
5902   {\glsxtrpostlink}%
5903 }%
5904 }

```

`omdiscardperiod` Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
5905 \newcommand*{\glxtrifcustomdiscardperiod}[2]{#2}
```

`\glxtrpostlink`

```
5906 \newcommand*{\glxtrpostlink}{%
5907   \csuse{\glxtrpostlink\glscategory{\glslabel}}%
5908 }
```

`linkendsentence` Done by `\glxtrpostlinkhook` if a full stop is discarded.

```
5909 \newcommand*{\glxtrpostlinkendsentence}{%
5910   \ifcsdef{\glxtrpostlink\glscategory{\glslabel}}
5911     {%
5912       \csuse{\glxtrpostlink\glscategory{\glslabel}}%
```

Put the full stop back.

```
5913   .\spacefactor\sfcode'\. \relax
5914   }%
5915   {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the space-factor.

```
5916   \spacefactor\sfcode'\. \relax
5917   }%
5918 }
```

`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.

```
5919 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
5920   \glxtrifwasfirstuse{\space\glxtrparen{\glssaccessdesc{\glslabel}}}{}%
5921 }
```

`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.

```
5922 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5923   \glxtrifwasfirstuse
5924   {%
5925     \ifglshassymbol{\glslabel}%
5926     {\space\glxtrparen{\glssaccesssymbol{\glslabel}}}%
5927     }%
5928   }%
5929   {}%
5930 }
```

`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).

```

5931 \newcommand*{\glxtrdiscardperiod}[3]{%
5932   \glxtrifwasfirstuse
5933   {%
5934     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5935     {#3}%
5936     {%
5937       \glusifattribute{#1}{discardperiod}{true}%
5938       {%
5939         \glusifplural
5940         {%
5941           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5942           {\glxtrifperiod{#2}{#3}}%
5943           {#3}%
5944         }%
5945         {%
5946           \glxtrifperiod{#2}{#3}%
5947         }%
5948       }%
5949     }%
5950   }%
5951 }%
5952 {%
5953   \glusifattribute{#1}{discardperiod}{true}%
5954   {%
5955     \glusifplural
5956     {%
5957       \glusifattribute{#1}{pluraldiscardperiod}{true}%
5958       {\glxtrifperiod{#2}{#3}}%
5959       {#3}%
5960     }%
5961     {%
5962       \glxtrifperiod{#2}{#3}%
5963     }%
5964   }%
5965   {#3}%
5966 }%
5967 }

```

`\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`

```

5968 \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 ' ').

```

5969 \newcommand*{\glxtr@punclist}{.,;?!}

```

`\punctuationmark` Add character to punctuation list.

```

5970 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}

```


unctuationmarks Reset the punctuation list.

```
5971 \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.)

```
5972 \newcommand*{\glxtrifnextpunc}[2]{%
5973   \def\reserved@a{#1}%
5974   \def\reserved@b{#2}%
5975   \futurelet\@glspunc@token\glxtr@ifnextpunc
5976 }
```

glxtr@ifnextpunc

```
5977 \newcommand*{\glxtr@ifnextpunc}{%
5978   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5979   \reserved@b
5980 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
5981 \newcommand*{\glxtr@ifpunctoken}[1]{%
5982   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5983 }
```

glxtr@ifpunctoken

```
5984 \def\@glxtr@ifpunctoken#1#2{%
5985   \let\reserved@d=#2%
5986   \ifx\reserved@d\@nnil
5987     \let\glxtr@next\@glxtr@notfoundinlist
5988   \else
5989     \ifx#1\reserved@d
5990       \let\glxtr@next\@glxtr@foundinlist
5991     \else
5992       \let\glxtr@next\@glxtr@ifpunctoken
5993     \fi
5994   \fi
5995   \glxtr@next#1%
5996 }
```

glxtr@foundinlist

```
5997 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5998 \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.

```
5999 \newcommand{\glsxtrdopostpunc}[1]{%
6000   \glsxtrifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
6001 }
```

@glsxtr@swaptwo

```
6002 \newcommand{\@glsxtr@swaptwo}[2]{#2#1}
```

1.7 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```
6003 \define@key{glsxtrabbrv}{category}{%
6004   \edef\glscategorylabel{#1}%
6005   \ifcsdef{@glsabbrv@current@#1}%
6006   {%
```

Warning should already have been issued.

```
6007   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
6008   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
6009   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
6010   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
6011 }%
6012 {}%
6013 }
```

Save the short plural form. This may be needed before the entry is defined.

```
6014 \define@key{glsxtrabbrv}{shortplural}{%
6015   \def\@gls@shortpl{#1}%
6016 }
```

Similarly for the long plural form.

```
6017 \define@key{glsxtrabbrv}{longplural}{%
6018   \def\@gls@longpl{#1}%
6019 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
6020 \newtoks\glsshortpltok
```

```

\glslongpltok
6021 \newtoks\glslongpltok

\glsxtr@insertdots
  Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to \newabbreviation. Note that explicitly using the short or shortplural keys will override this.
6022 \newcommand*{\@glsxtr@insertdots}[2]{%
6023   \def#1{}%
6024   \@glsxtr@insert@dots#1#2\@nnil
6025 }

\glsxtr@insert@dots
6026 \newcommand*{\@glsxtr@insert@dots}[2]{%
6027   \ifx\@nnil#2\relax
6028   \let\@glsxtr@insert@dots@next\@gobble
6029   \else
6030   \ifx\relax#2\relax
6031   \else
6032     \appto#1{#2.}%
6033   \fi
6034   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
6035   \fi
6036   \@glsxtr@insert@dots@next#1%
6037 }

  Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep
6038 \newcommand*{\glsxtrwordsep}{\space}

  Each word is marked with

\glsxtrword
6039 \newcommand*{\glsxtrword}[1]{#1}

\glsxtr@markwordseps
6040 \newcommand*{\@glsxtr@markwordseps}[2]{%
6041   \def#1{}%
6042   \@glsxtr@mark@wordseps#1#2 \@nnil
6043 }

\glsxtr@mark@wordseps
6044 \def\@glsxtr@mark@wordseps#1#2 #3{%
6045   \ifdefempty{#1}%

```

```

6046 {\def#1{\protect\glstrword{#2}}}%
6047 {\appto#1{\protect\glstrwordsep\protect\glstrword{#2}}}%
6048 \ifx\@nnil#3\relax
6049 \let\@glstr@mark@wordseps@next\relax
6050 \else
6051 \def\@glstr@mark@wordseps@next{%
6052 \glstr@mark@wordseps#1#3}%
6053 \fi
6054 \@glstr@mark@wordseps@next
6055 }

```

newabbreviation Define a new generic abbreviation.

```

6056 \newcommand*{\newabbreviation}[4][]{%
6057 \glstr@newabbreviation{#1}{#2}{#3}{#4}%
6058 }

```

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.)

```

6059 \newcommand*{\glstr@newabbreviation}[4]{%
6060 \glscopylisttok{#1}%
6061 \glscopylabeltok{#2}%
6062 \glscopyshorttok{#3}%
6063 \glscopylongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

6064 \def\glstr@orgshort{#3}%
6065 \def\glstr@orglong{#4}%

```

Get the category.

```

6066 \def\glscategorylabel{abbreviation}%
6067 \glstr@applyabbrvstyle{\@glsabrv@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```

6068 \setkeys*{\glstr@abbrv}{shortplural,longplural}{#1}%

```

Set the default long plural

```

6069 \def\@glslongpl{#4\glsppluralsuffix}%
6070 \let\@glslongpl\@glslongpl

```

Has the markwords attribute been set?

```

6071 \glscopycategoryattribute{\glscategorylabel}{markwords}{true}%
6072 {%
6073 \@glstr@markwordseps\@glslong{#4}%
6074 \expandafter\def\expandafter\@glslongpl\expandafter
6075 {\@glslong\glsppluralsuffix}%
6076 \let\@glslongpl\@glslongpl

```

Update \glslongtok.

```

6077 \expandafter\glslongtok\expandafter{\@glslong}%
6078 }%
6079 {}%

```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
6080 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
6081 {%
6082   \@glstr@markwordseps\@gls@short{#3}%
6083 }%
6084 {%
```

Has the insertdots attribute been set?

```
6085 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
6086 {%
6087   \@glstr@insertdots\@gls@short{#3}%
6088   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6089 }%
6090 {\def\@gls@short{#3}}%
6091 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
6092 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
6093 {%
6094   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6095     '\abbrvpluralsuffix}%
6096 }%
6097 {%
```

Has the noshortplural attribute been set?

```
6098 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
6099 {%
6100   \let\@gls@shortpl\@gls@short
6101 }%
6102 {%
6103   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6104     \abbrvpluralsuffix}%
6105 }%
6106 }%
```

Update \glsshorttok:

```
6107 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
6108 \glstrnewabbrevpresetkeyhook{#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).

```
6109 \setkeys*{glstrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
6110 \ifx\@gls@default@longpl\@gls@longpl
6111 \else
```

Has the markwords attribute been set?

```
6112 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6113 {%
```

```

6114 \expandafter\@glxtr@markwordseps\expandafter\@gls@longpl\expandafter
6115 {\@gls@longpl}%
6116 }%
6117 {}%
6118 \fi

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

6119 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6120 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

6121 \newabbreviationhook

```

Define this entry:

```

6122 \protected@edef\@do@newglossaryentry{%
6123 \noexpand\newglossaryentry{\the\glslabeltok}%
6124 {%
6125 type=\glxtrabbrvtype,%
6126 category=abbreviation,%
6127 short={\the\glsshorttok},%
6128 shortplural={\the\glsshortpltok},%
6129 long={\the\glslongtok},%
6130 longplural={\the\glslongpltok},%
6131 name={\the\glsshorttok},%
6132 \CustomAbbreviationFields,%
6133 \the\glskeylisttok
6134 }%
6135 }%
6136 \@do@newglossaryentry
6137 \GlsXtrPostNewAbbreviation
6138 }

```

`evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```

6139 \newcommand*{\glxtrnewabbrevpresetkeyhook}[3]{}

```

`NewAbbreviation` Hook used by abbreviation styles.

```

6140 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`bbreviationhook` Hook for use with `\newabbreviation`.

```

6141 \newcommand*{\newabbreviationhook}{}

```

`reviationFields`

```

6142 \newcommand*{\CustomAbbreviationFields}{}

```

`\glxtrparen` For the parenthetical styles.

```

6143 \newcommand*{\glxtrparen}[1]{(#1)}

```

`lsxtrfullformat` Full format without case change.

```

6144 \newcommand*{\glxtrfullformat}[2]{%

```

```

6145 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6146 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6147 }

```

`\glsxtrfullformat` Full format with case change.

```

6148 \newcommand*{\Glsxtrfullformat}[2]{%
6149 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6150 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6151 }

```

`\glsxtrfullplformat` Plural full format without case change.

```

6152 \newcommand*{\glsxtrfullplformat}[2]{%
6153 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6154 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6155 }

```

`\glsxtrfullplformat` Plural full format with case change.

```

6156 \newcommand*{\Glsxtrfullplformat}[2]{%
6157 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6158 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6159 }

```

`\glsxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```

6160 \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.

```

6161 \newcommand*{\glsxtrinfullformat}{\glsxtrfullformat}

```

`\glsxtrinfullformat` Full format with case change.

```

6162 \newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}

```

`\glsxtrfullplformat` Plural full format without case change.

```

6163 \newcommand*{\glsxtrinfullplformat}{\glsxtrfullplformat}

```

`\glsxtrinfullplformat` Plural full format with case change.

```

6164 \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`

```

6165 \renewcommand*{\glsentryfull}[1]{\glsxtrinfullformat{#1}{}}

```

`\Glsentryfull`

```

6166 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}

```

`\glsentryfullpl` 6167 `\renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}`

`\Glsentryfullpl` 6168 `\renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}`

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.
6169 `\newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.
6170 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.
6171 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`\glsabbrvdefaultfont` 6172 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.
6173 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`\glslongdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.
6174 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`\glsfirstlongfont` Font changing command used for the long form on first use or in the full format.
6175 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`\glsfirstlongdefaultfont` 6176 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`\glsbrvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.
6177 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

`\glsbrvpluralsuffix` Default plural suffix.
6178 `\newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).
6179 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}`
6180 `\newcommand*{\ns@glsxtrfull}[2][{}]{%`
6181 `\new@ifnextchar[{\@glsxtr@full{#1}{#2}}{%`
6182 `{\@glsxtr@full{#1}{#2}}[{}]{%`
6183 `}`

\@glxtr@full Low-level macro:

```
6184 \def\@glxtr@full#1#2[#3]{%
6185   \glstoifexists{#2}%
6186   {%
6187     \glsetabbrvfmt{\glscategory{#2}}%
6188     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6189     \let\gl@ifplural\@secondoftwo
6190     \let\glscapscase\@firstofthree
6191     \let\glinsert\@empty
6192     \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.

```
6193   \glxtrsetupfulldefs
6194   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6195   }%
6196   \glspostlinkhook
6197 }
```

trsetupfulldefs

```
6198 \newcommand*{\glxtrsetupfulldefs}{%
6199   \let\glxtrifwasfirstuse\@firstoftwo
6200 }
```

\Glsxtrfull Full form (first letter uppercase).

```
6201 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\@ns@Glsxtrfull}
6202 \newcommand*\ns@Glsxtrfull[2][ ]{%
6203   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}%
6204   {\@Glsxtr@full{#1}{#2} [ ]}%
6205 }
```

\@Glsxtr@full Low-level macro:

```
6206 \def\@Glsxtr@full#1#2[#3]{%
6207   \glstoifexists{#2}%
6208   {%
6209     \glsetabbrvfmt{\glscategory{#2}}%
6210     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6211     \let\gl@ifplural\@secondoftwo
6212     \let\glscapscase\@secondofthree
6213     \let\glinsert\@empty
6214     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6215     \glxtrsetupfulldefs
6216     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6217   }%
6218   \glspostlinkhook
6219 }
```

`\GLSxtrfull` Full form (all uppercase).

```
6220 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\ns@GLSxtrfull}
6221 \newcommand*\ns@GLSxtrfull[2][\%
6222 \new@ifnextchar[\@GLSxtr@full{#1}{#2}]{%
6223     {\@GLSxtr@full{#1}{#2}[]}%
6224 }
```

`\@GLSxtr@full` Low-level macro:

```
6225 \def\@GLSxtr@full#1#2[#3]{%
6226   \glsdoifexists{#2}%
6227   {%
6228     \glssetabbrvfmt{\glscategory{#2}}%
6229     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6230     \let\glsifplural\@secondoftwo
6231     \let\glsapscase\@thirdofthree
6232     \let\glsinsert\@empty
6233     \def\glscustomtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
6234     \glsxtrsetupfulldefs
6235     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6236   }%
6237   \glspostlinkhook
6238 }
```

`\glsxtrfullpl` Plural full form (no case-change).

```
6239 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\ns@glsxtrfullpl}
6240 \newcommand*\ns@glsxtrfullpl[2][\%
6241 \new@ifnextchar[\@glsxtr@fullpl{#1}{#2}]{%
6242     {\@glsxtr@fullpl{#1}{#2}[]}%
6243 }
```

`\@glsxtr@fullpl` Low-level macro:

```
6244 \def\@glsxtr@fullpl#1#2[#3]{%
6245   \glsdoifexists{#2}%
6246   {%
6247     \glssetabbrvfmt{\glscategory{#2}}%
6248     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6249     \let\glsifplural\@firstoftwo
6250     \let\glsapscase\@firstofthree
6251     \let\glsinsert\@empty
6252     \def\glscustomtext{\glsxtrinlinefullplformat{#2}{#3}}%
6253     \glsxtrsetupfulldefs
6254     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6255   }%
6256   \glspostlinkhook
6257 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
6258 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\ns@Glsxtrfullpl}
6259 \newcommand*\ns@Glsxtrfullpl[2][\%
```

```

6260 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6261         {\@Glsxtr@fullpl{#1}{#2}[]}%
6262 }

```

\@Glsxtr@fullpl Low-level macro:

```

6263 \def\@Glsxtr@fullpl#1#2[#3]{%
6264     \glsdoifexists{#2}%
6265     {%
6266         \glssetabbrvfmt{\glscategory{#2}}%
6267         \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
6268         \let\glsifplural\@firstoftwo
6269         \let\glscapscase\@secondofthree
6270         \let\glsinsert\@empty
6271         \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6272         \glsxtrsetupfulldefs
6273         \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6274     }%
6275     \glspostlinkhook
6276 }

```

\Glsxtrfullpl Plural full form (all upper case).

```

6277 \newrobustcmd*{\Glsxtrfullpl}{\@glshyp@opt\@ns@Glsxtrfullpl}
6278 \newcommand*\ns@Glsxtrfullpl[2][]{%
6279     \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6280         {\@Glsxtr@fullpl{#1}{#2}[]}%
6281 }

```

\@Glsxtr@fullpl Low-level macro:

```

6282 \def\@Glsxtr@fullpl#1#2[#3]{%
6283     \glsdoifexists{#2}%
6284     {%
6285         \let\do@glsl@link@checkfirsthyper\@glsl@link@nocheckfirsthyper
6286         \let\glsifplural\@firstoftwo
6287         \let\glscapscase\@thirdofthree
6288         \let\glsinsert\@empty
6289         \def\glscustomtext{%
6290             \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6291         \glsxtrsetupfulldefs
6292         \@glsl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6293     }%
6294     \glspostlinkhook
6295 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

6296 \newrobustcmd*{\glsxtrshort}{\@glshyp@opt\@ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6297 \newcommand*{\ns@glstrshort}[2] [] {%
6298   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} []}%
6299 }

```

Read in the final optional argument:

```

6300 \def\@glstrshort#1#2[#3] {%
6301   \glsdofexists{#2}%
6302   {%

```

Need to make sure \glabbrvfont is set correctly.

```

6303     \glsetabbrvfmt{\glscategory{#2}}%
6304     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6305     \let\glstrifwasfirstuse\@secondoftwo
6306     \let\gl@ifplural\@secondoftwo
6307     \let\glscapscase\@firstofthree
6308     \let\glinsert\@empty
6309     \def\glscustomtext{%
6310       \glabbrvfont{\glaccessshort{#2}\ifglstrinsertinside#3\fi}%
6311       \ifglstrinsertinside\else#3\fi
6312     }%
6313     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6314   }%
6315   \glspostlinkhook
6316 }

```

\Glsxtrshort

```

6317 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6318 \newcommand*{\ns@Glsxtrshort}[2] [] {%
6319   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
6320 }

```

Read in the final optional argument:

```

6321 \def\@Glsxtrshort#1#2[#3] {%
6322   \glsdofexists{#2}%
6323   {%
6324     \glsetabbrvfmt{\glscategory{#2}}%
6325     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6326     \let\glstrifwasfirstuse\@secondoftwo
6327     \let\gl@ifplural\@secondoftwo
6328     \let\glscapscase\@secondofthree
6329     \let\glinsert\@empty
6330     \def\glscustomtext{%
6331       \glabbrvfont{\Glsaccessshort{#2}\ifglstrinsertinside#3\fi}%
6332       \ifglstrinsertinside\else#3\fi
6333     }%
6334     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6335   }%
6336   \glspostlinkhook
6337 }

```

\GLSxtrshort

```
6338 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\@ns@GLSxtrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6339 \newcommand*{\ns@GLSxtrshort}[2][\%  
6340   \new@ifnextchar[\@GLSxtrshort{#1}{#2}]{\@GLSxtrshort{#1}{#2}[]}%  
6341 }
```

Read in the final optional argument:

```
6342 \def\@GLSxtrshort#1#2[#3]{%  
6343   \glsdoifexists{#2}%  
6344   {%  
6345     \glssetabbrvfmt{\glscategory{#2}}%  
6346     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper  
6347     \let\glsxtrifwasfirstuse\@secondoftwo  
6348     \let\glsifplural\@secondoftwo  
6349     \let\glsapscase\@thirdofthree  
6350     \let\glsinsert\@empty  
6351     \def\glscustomtext{%  
6352       \mfirstucMakeUppercase  
6353       {\glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%  
6354       \ifglsxtrininsertinside\else#3\fi  
6355     }%  
6356   }%  
6357   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%  
6358   }%  
6359   \glspostlinkhook  
6360 }
```

\glsxtrlong

```
6361 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6362 \newcommand*{\ns@glsxtrlong}[2][\%  
6363   \new@ifnextchar[\@glsxtrlong{#1}{#2}]{\@glsxtrlong{#1}{#2}[]}%  
6364 }
```

Read in the final optional argument:

```
6365 \def\@glsxtrlong#1#2[#3]{%  
6366   \glsdoifexists{#2}%  
6367   {%  
6368     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper  
6369     \let\glsxtrifwasfirstuse\@secondoftwo  
6370     \let\glsifplural\@secondoftwo  
6371     \let\glsapscase\@firstofthree  
6372     \let\glsinsert\@empty  
6373     \def\glscustomtext{%  
6374       \glsfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%  
6375       \ifglsxtrininsertinside\else#3\fi  
6376     }%  
6377     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%  
6378   }%
```

```

6378 }%
6379 \glspostlinkhook
6380 }

```

\Glsxtrlong

```

6381 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6382 \newcommand*{\ns@Glsxtrlong}[2] [] {%
6383   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
6384 }

    Read in the final optional argument:
6385 \def\@Glsxtrlong#1#2[#3]{%
6386   \glsdoifexists{#2}%
6387   {%
6388     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6389     \let\glstrifwasfirstuse\@secondoftwo
6390     \let\glsifplural\@secondoftwo
6391     \let\glscapscase\@thirdofthree
6392     \let\glsinsert\@empty
6393     \def\glscustomtext{%
6394       \glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6395       \ifglstrinsertinside\else#3\fi
6396     }%
6397     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6398   }%
6399   \glspostlinkhook
6400 }

```

\GLSxtrlong

```

6401 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\@ns@GLSxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6402 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6403   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6404 }

    Read in the final optional argument:
6405 \def\@GLSxtrlong#1#2[#3]{%
6406   \glsdoifexists{#2}%
6407   {%
6408     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6409     \let\glstrifwasfirstuse\@secondoftwo
6410     \let\glsifplural\@secondoftwo
6411     \let\glscapscase\@thirdofthree
6412     \let\glsinsert\@empty
6413     \def\glscustomtext{%
6414       \mfirstucMakeUppercase
6415       {\glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6416       \ifglstrinsertinside\else#3\fi

```

```

6417     }%
6418     }%
6419     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6420     }%
6421     \glspostlinkhook
6422 }

```

Plural short forms:

\glsxtrshortpl

```

6423 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\ns@glsxtrshortpl}

Define the un-starred form. Need to determine if there is a final optional argument
6424 \newcommand*{\ns@glsxtrshortpl}[2][ ]{%
6425   \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2}[]}%
6426 }

Read in the final optional argument:
6427 \def\@glsxtrshortpl#1#2[#3]{%
6428   \glsdoifexists{#2}%
6429   {%
6430     \glssetabbrvfmt{\glscategory{#2}}%
6431     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6432     \let\glsxtrifwasfirstuse\@secondoftwo
6433     \let\glsifplural\@firstoftwo
6434     \let\glscapscase\@firstofthree
6435     \let\glsinsert\@empty
6436     \def\glscustomtext{%
6437       \glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6438       \ifglsxtrininsertinside\else#3\fi
6439     }%
6440     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6441   }%
6442   \glspostlinkhook
6443 }

```

\Glsxtrshortpl

```

6444 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\ns@Glsxtrshortpl}

Define the un-starred form. Need to determine if there is a final optional argument
6445 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
6446   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[]}%
6447 }

Read in the final optional argument:
6448 \def\@Glsxtrshortpl#1#2[#3]{%
6449   \glsdoifexists{#2}%
6450   {%
6451     \glssetabbrvfmt{\glscategory{#2}}%
6452     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6453     \let\glsxtrifwasfirstuse\@secondoftwo

```

```

6454 \let\glsifplural\@firstoftwo
6455 \let\glscapscase\@secondofthree
6456 \let\glsinsert\@empty
6457 \def\glscustomtext{%
6458   \glsabbrvfont{\Glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
6459   \ifglstrinsertinside\else#3\fi
6460 }%
6461 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6462 }%
6463 \glspostlinkhook
6464 }

```

\GLSxtrshortpl

```

6465 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6466 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6467   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6468 }

    Read in the final optional argument:
6469 \def\@GLSxtrshortpl#1#2[#3] {%
6470   \glsdoifexists{#2}%
6471   {%
6472     \glssetabbrvfmt{\glscategory{#2}}%
6473     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6474     \let\glsxtrifwasfirstuse\@secondoftwo
6475     \let\glsifplural\@firstoftwo
6476     \let\glsapscase\@thirdofthree
6477     \let\glsinsert\@empty
6478     \def\glscustomtext{%
6479       \mfirstucMakeUppercase
6480       {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
6481       \ifglstrinsertinside\else#3\fi
6482     }%
6483   }%
6484   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6485 }%
6486 \glspostlinkhook
6487 }

```

Plural long forms:

\glsxtrlongpl

```

6488 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\ns@glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6489 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
6490   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
6491 }

```


Read in the final optional argument:

```
6492 \def\@glxstrlongpl#1#2[#3]{%
6493   \glstoifexists{#2}%
6494   {%
6495     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6496     \let\glxstrifwasfirstuse\@secondoftwo
6497     \let\gl@ifplural\@firstoftwo
6498     \let\glscapscase\@firstofthree
6499     \let\glinsert\@empty
6500     \def\glscustomtext{%
6501       \glslongfont{\glaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
6502       \ifglxstrinsertinside\else#3\fi
6503     }%
6504     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6505   }%
6506   \glspostlinkhook
6507 }
```

\Glsxtrlongpl

```
6508 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\ns@Glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6509 \newcommand*{\ns@Glsxtrlongpl}[2][]{%
6510   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
6511 }
```

Read in the final optional argument:

```
6512 \def\@Glsxtrlongpl#1#2[#3]{%
6513   \glstoifexists{#2}%
6514   {%
6515     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6516     \let\glxstrifwasfirstuse\@secondoftwo
6517     \let\gl@ifplural\@firstoftwo
6518     \let\glscapscase\@secondofthree
6519     \let\glinsert\@empty
6520     \def\glscustomtext{%
6521       \glslongfont{\Glsaccesslongpl{#2}\ifglxstrinsertinside#3\fi}%
6522       \ifglxstrinsertinside\else#3\fi
6523     }%
6524     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6525   }%
6526   \glspostlinkhook
6527 }
```

\GLSxtrlongpl

```
6528 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6529 \newcommand*{\ns@GLSxtrlongpl}[2][]{%
6530   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2}[]}%
6531 }
```

Read in the final optional argument:

```

6532 \def\@GLSxtrlongpl#1#2[#3]{%
6533   \glsdoifexists{#2}%
6534   {%
6535     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6536     \let\glxtrifwasfirstuse\@secondoftwo
6537     \let\glsifplural\@firstoftwo
6538     \let\glscapscase\@thirdofthree
6539     \let\glsinsert\@empty
6540     \def\glscustomtext{%
6541       \mfirstucMakeUppercase
6542       {\glslongfont{\glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6543       \ifglxtrininsertinside\else#3\fi
6544       }%
6545     }%
6546     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6547   }%
6548   \glspostlinkhook
6549 }

```

`\glsetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6550 \newcommand*{\glsetabbrvfmt}[1]{%
6551   \ifcsdef{@glsabbrv@current@#1}%
6552   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6553   {\glxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6554 }

```

`glsuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6555 \newrobustcmd*{\glsuseabbrvfont}[2]{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\glsuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6556 \newrobustcmd*{\glsuselongfont}[2]{\glsetabbrvfmt{#2}\glslongfont{#1}}

```

`sxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6557 \newcommand*{\glsxtrgenabbrvfmt}{%
6558   \ifdefempty\glscustomtext
6559   {%
6560     \ifglused\glslabel
6561     {%

```

Subsequent use:

```

6562     \glsifplural
6563     {%

```

Subsequent plural form:

```

6564     \glscapscase
6565     {%

```

Subsequent plural form, don't adjust case:

```
6566      \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6567      }%  
6568      {%
```

Subsequent plural form, make first letter upper case:

```
6569      \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6570      }%  
6571      {%
```

Subsequent plural form, all caps:

```
6572      \mfirstucMakeUppercase  
6573      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%  
6574      }%  
6575      }%  
6576      {%
```

Subsequent singular form

```
6577      \glscapscase  
6578      {%
```

Subsequent singular form, don't adjust case:

```
6579      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6580      }%  
6581      {%
```

Subsequent singular form, make first letter upper case:

```
6582      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6583      }%  
6584      {%
```

Subsequent singular form, all caps:

```
6585      \mfirstucMakeUppercase  
6586      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6587      }%  
6588      }%  
6589      }%  
6590      {%
```

First use:

```
6591      \glsifplural  
6592      {%
```

First use plural form:

```
6593      \glscapscase  
6594      {%
```

First use plural form, don't adjust case:

```
6595      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6596      }%  
6597      {%
```

First use plural form, make first letter upper case:

```
6598      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
6599      }%
6600      {%
```

First use plural form, all caps:

```
6601      \mfirstucMakeUppercase
6602      {\glsxtrfullplformat{\glslabel}{\glsinsert}}%
6603      }%
6604      }%
6605      {%
```

First use singular form

```
6606      \glscapscase
6607      {%
```

First use singular form, don't adjust case:

```
6608      \glsxtrfullformat{\glslabel}{\glsinsert}%
6609      }%
6610      {%
```

First use singular form, make first letter upper case:

```
6611      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6612      }%
6613      {%
```

First use singular form, all caps:

```
6614      \mfirstucMakeUppercase
6615      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6616      }%
6617      }%
6618      }%
6619      }%
6620      {%
```

User supplied text.

```
6621      \glscustomtext
6622      }%
6623 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6624 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6625   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6626   \ifglsxtrininsertinside \else#2\fi
6627 }
6628 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6629 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6630   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6631   \ifglsxtrininsertinside \else#2\fi
```

```

6632 }
6633 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

trsubsequentfmt Subsequent use format (singular, first letter uppercase).
6634 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6635   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
6636   \ifglxtrininsertinside \else#2\fi
6637 }
6638 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

subsequentplfmt Subsequent use format (plural, first letter uppercase).
6639 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6640   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
6641   \ifglxtrininsertinside \else#2\fi
6642 }
6643 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.7.1 Abbreviation Styles Setup

```

breviationstyle
6644 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6645   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6646   {%
6647     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6648   }%
6649   {%

Have abbreviations already been defined for this category?
6650   \ifcsstring{@glsabbrv@current@#1}{#2}%
6651   {%

Style already set.
6652   }%
6653   {%
6654     \def\@glxtr@dostylewarn{%
6655       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6656       {%
6657         \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
6658           style has been switched \MessageBreak
6659           for category ‘#1’, \MessageBreak
6660           but there have already been entries \MessageBreak
6661           defined for this category. Unwanted \MessageBreak
6662           side-effects may result}}%
6663         \@endfortrue
6664       }%
6665       \@glxtr@dostylewarn

Set up the style for the given category.
6666       \csdef{@glsabbrv@current@#1}{#2}%
6667       \glxtr@applyabbrvstyle{#2}%

```

```

6668     }%
6669 }%
6670 }

```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```

6671 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
6672   \csuse{@glsabbrv@dispstyle@setup@#1}%
6673   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6674 }

```

`\r@applyabbrvfmt` Only apply the style formats.

```

6675 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
6676   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6677 }

```

`\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.

```

6678 \newcommand*{\newabbreviationstyle}[3]{%
6679   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6680     {%
6681       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6682         defined}{}%
6683     }%
6684     {%
6685       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```

6686       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6687       #2}%
6688       \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

6689       \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6690       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6691       \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6692       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glxtrsubsequentfmt` etc in case a style changes this.

```

6693       \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
6694       \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
6695       \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6696       \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6697       #3}%
6698     }%
6699 }

```

`\abbreviationstyle`

```

6700 \newcommand*{\renewabbreviationstyle}[3]{%
6701   \ifcsundef{@glsabbrv@dispstyle@setup@#1}

```

```

6702  {%
6703    \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6704  }%
6705  {%
6706    \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
6707      \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6708      #2}%
6709    \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
6710      \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6711      \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6712      \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6713      \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6714      #3}%
6715    }%
6716  }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

6717 \newcommand*{\letabbreviationstyle}[2]{%
6718   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6719   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6720 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

6721 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6722   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6723     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6724     \csuse{@glsabbrv@dispstyle@setup@#2}%
6725   }%
6726   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6727 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

6728 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6729   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6730     use ‘#2’ instead}%
6731 }

```

eAbbrStyleSetup

```

6732 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6733   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%

```

```

6734 {%
6735     \PackageError{glossaries-extra}%
6736     {Unknown abbreviation style definitions ‘#1’}{}%
6737 }%
6738 {%
6739     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6740 }%
6741 }

```

seAbbrStyleFmts

```

6742 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
6743     \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6744     {%
6745         \PackageError{glossaries-extra}%
6746         {Unknown abbreviation style formats ‘#1’}{}%
6747     }%
6748     {%
6749         \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6750     }%
6751 }

```

1.7.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn’t set to “true”. If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```

6752 \newif\ifglsxtrinsertinside
6753 \glsxtrinsertinsidefalse

```

trlongshortname

```

6754 \newcommand*{\glsxtrlongshortname}{%
6755     \protect\glsabbrvfont{\the\glsshorttok}%
6756 }

```

long-short

```

6757 \newabbreviationstyle{long-short}%
6758 {%
6759     \renewcommand*{\CustomAbbreviationFields}{%
6760         name={\glsxtrlongshortname},
6761         sort={\the\glsshorttok},

```



```

6762 first={\protect\glsfirstlongfont{\the\glslongtok}%
6763 \protect\glsxtrfullsep{\the\glslabeltok}%
6764 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6765 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6766 \protect\glsxtrfullsep{\the\glslabeltok}%
6767 \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

6768 plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6769 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

6770 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6771 \glsasattribute{\the\glslabeltok}{regular}%
6772 {%
6773 \glssetattribute{\the\glslabeltok}{regular}{false}%
6774 }%
6775 {}%
6776 }%
6777 }%
6778 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6779 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6780 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6781 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6782 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6783 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6784 \renewcommand*{\glsxtrfullformat}[2]{%
6785 \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6786 \ifglsxtrininsertinside\else##2\fi
6787 \glsxtrfullsep{##1}%
6788 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6789 }%
6790 \renewcommand*{\glsxtrfullplformat}[2]{%
6791 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6792 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6793 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6794 }%
6795 \renewcommand*{\Glsxtrfullformat}[2]{%
6796 \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6797 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6798 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6799 }%
6800 \renewcommand*{\Glsxtrfullplformat}[2]{%
6801 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6802 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6803 \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6804 }%
6805 }

```

Set this as the default style for general abbreviations:

```
6806 \setabbreviationstyle{long-short}
```

ngshortdescsort

```
6807 \newcommand*{\glxtrlongshortdescsort}{%
6808 \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6809 }
```

ngshortdescname

```
6810 \newcommand*{\glxtrlongshortdescname}{%
6811 \protect\glslongfont{\the\glslongtok}
6812 \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}}%
6813 }
```

long-short-desc User supplies description. The long form is included in the name.

```
6814 \newabbreviationstyle{long-short-desc}%
6815 {%
6816 \renewcommand*{\CustomAbbreviationFields}{%
6817 name={\glxtrlongshortdescname},
6818 sort={\glxtrlongshortdescsort},%
6819 first={\protect\glsfirstlongfont{\the\glslongtok}%
6820 \protect\glxtrfullsep{\the\glslabeltok}%
6821 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6822 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6823 \protect\glxtrfullsep{\the\glslabeltok}%
6824 \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
6825 text={\protect\glsabbrvfont{\the\glsshorttok}},%
6826 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6827 }%
```

Unset the regular attribute if it has been set.

```
6828 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6829 \glshasattribute{\the\glslabeltok}{regular}%
6830 {%
6831 \glssetattribute{\the\glslabeltok}{regular}{false}%
6832 }%
6833 {}%
6834 }%
6835 }%
6836 {%
6837 \GlsXtrUseAbbrStyleFmts{long-short}%
6838 }
```

trshortlongname

```
6839 \newcommand*{\glxtrshortlongname}{%
6840 \protect\glsabbrvfont{\the\glsshorttok}%
6841 }
```

short-long Short form followed by long form in parenthesis on first use.

```
6842 \newabbreviationstyle{short-long}%
6843 {%
6844   \renewcommand*{\CustomAbbreviationFields}{%
6845     name={\glxtrshortlongname},
6846     sort={\the\glsshorttok},
6847     description={\the\gslongtok},%
6848     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6849       \protect\glxtrfullsep{\the\glslabeltok}%
6850       \glxtrparen{\protect\glsfirstlongfont{\the\gslongtok}}},%
6851     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6852       \protect\glxtrfullsep{\the\glslabeltok}%
6853       \glxtrparen{\protect\glsfirstlongfont{\the\gslongpltok}}},%
6854     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6855 }
```

Unset the regular attribute if it has been set.

```
6855 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6856   \glshasattribute{\the\glslabeltok}{regular}%
6857   {%
6858     \glissetattribute{\the\glslabeltok}{regular}{false}%
6859   }%
6860   {}%
6861 }%
6862 }%
6863 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6864 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6865 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6866 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6867 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6868 \renewcommand*{\gslongfont}[1]{\gslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6869 \renewcommand*{\glxtrfullformat}[2]{%
6870   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6871   \ifglxtrininsertinside\else##2\fi
6872   \glxtrfullsep{##1}%
6873   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6874 }%
6875 \renewcommand*{\glxtrfullplformat}[2]{%
6876   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6877   \ifglxtrininsertinside\else##2\fi
6878   \glxtrfullsep{##1}%
6879   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6880 }%
6881 \renewcommand*{\Glsxtrfullformat}[2]{%
6882   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6883   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6884 }
```

```

6884 \glstrparen{\glfirstlongfont{\glaccesslong{##1}}}%
6885 }%
6886 \renewcommand*{\Glsxtrfullplformat}[2]{%
6887 \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
6888 \ifglstrinsertinside\else##2\fi\glxtrfullsep{##1}%
6889 \glstrparen{\glfirstlongfont{\glaccesslongpl{##1}}}%
6890 }%
6891 }

```

ortlongdescsort

```

6892 \newcommand*{\glstrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

6893 \newcommand*{\glstrshortlongdescname}{%
6894 \protect\glsabbrvfont{\the\glsshorttok}
6895 \glstrparen{\protect\glslongfont{\the\glslongtok}}}%
6896 }

```

short-long-desc User supplies description. The long form is included in the name.

```

6897 \newabbreviationstyle{short-long-desc}%
6898 {%
6899 \renewcommand*{\CustomAbbreviationFields}{%
6900 name={\glstrshortlongdescname},
6901 sort={\glstrshortlongdescsort},
6902 first={\protect\glfirstabbrvfont{\the\glsshorttok}}%
6903 \protect\glxtrfullsep{\the\glslabeltok}}%
6904 \glstrparen{\protect\glfirstlongfont{\the\glslongtok}}},%
6905 firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}}%
6906 \protect\glxtrfullsep{\the\glslabeltok}}%
6907 \glstrparen{\protect\glfirstlongfont{\the\glslongpltok}}},%
6908 text={\protect\glsabbrvfont{\the\glsshorttok}},%
6909 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6910 }%

```

Unset the regular attribute if it has been set.

```

6911 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6912 \glshasattribute{\the\glslabeltok}{regular}}%
6913 {%
6914 \glissetattribute{\the\glslabeltok}{regular}{false}}%
6915 }%
6916 {}%
6917 }%
6918 }%
6919 {%
6920 \GlsXtrUseAbbrStyleFmts{short-long}}%
6921 }

```

ongfootnotefont Only used by the “footnote” styles.

```
6922 \newcommand*{\glfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6923 \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`).

```
6924 \newcommand*{\glxtrabbrvfootnote}[2]{\footnote{#2}}
```

xtrfootnotename

```
6925 \newcommand*{\glxtrfootnotename}{%
6926   \protect\glsabbrvfont{\the\glsshorttok}%
6927 }
```

footnote Short form followed by long form in footnote on first use.

```
6928 \newabbreviationstyle{footnote}%
6929 {%
6930   \renewcommand*{\CustomAbbreviationFields}{%
6931     name={\glxtrfootnotename},
6932     sort={\the\glsshorttok},
6933     description={\the\glslongtok},%
6934     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6935       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6936       {\protect\glfirstlongfootnotefont{\the\glslongtok}}},%
6937     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6938       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
6939       {\protect\glfirstlongfootnotefont{\the\glslongpltok}}},%
6940     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
6941   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6942     \glsssetAttribute{\the\glslabeltok}{nohyperfirst}{true}%
6943     \glshasattribute{\the\glslabeltok}{regular}%
6944     {%
6945       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
6946     }%
6947   {}%
6948 }
```

6949 }%
 6950 {%

In case the user wants to mix and match font styles, these are redefined here.

```
6951 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
6952 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
6953 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
6954 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
6955 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
6956 \renewcommand*\glxtrfullformat[2]{%
6957   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6958   \ifglxtrininsertinside\else##2\fi
6959   \protect\glxtrabbrvfootnote{##1}%
6960   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6961 }%
6962 \renewcommand*\glxtrfullplformat[2]{%
6963   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6964   \ifglxtrininsertinside\else##2\fi
6965   \protect\glxtrabbrvfootnote{##1}%
6966   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6967 }%
6968 \renewcommand*\Glsxtrfullformat[2]{%
6969   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6970   \ifglxtrininsertinside\else##2\fi
6971   \protect\glxtrabbrvfootnote{##1}%
6972   {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6973 }%
6974 \renewcommand*\Glsxtrfullplformat[2]{%
6975   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6976   \ifglxtrininsertinside\else##2\fi
6977   \protect\glxtrabbrvfootnote{##1}%
6978   {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6979 }%
```

The first use full form and the inline full form use the short (long) style.

```
6980 \renewcommand*\glxtrininlinefullformat[2]{%
6981   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6982   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6983   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6984 }%
6985 \renewcommand*\glxtrininlinefullplformat[2]{%
6986   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6987   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6988   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6989 }%
6990 \renewcommand*\Glsxtrininlinefullformat[2]{%
6991   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6992   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6993   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6994 }
```

```

6994 }%
6995 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6996   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6997   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6998   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6999 }%
7000 }

```

short-footnote

```

7001 \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.

```

7002 \newabbreviationstyle{postfootnote}%
7003 {%
7004   \renewcommand*{\CustomAbbreviationFields}{%
7005     name={\glxtrfootnotename},
7006     sort={\the\glsshorttok},
7007     description={\the\glslongtok},%
7008     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
7009     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
7010     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.

```

7011 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7012   \csdef{glxtrpostlink\glscategorylabel}{%
7013     \glxtrifwasfirstuse
7014     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7015       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7016       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7017     }%
7018   }%
7019 }%
7020 \glshasattribute{\the\glslabeltok}{regular}%
7021 {%
7022   \glissetattribute{\the\glslabeltok}{regular}{false}%
7023 }%
7024 {}%
7025 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7026 \renewcommand*{\glxtrsetupfulldefs}{%
7027   \let\glxtrifwasfirstuse\@secondoftwo

```

```

7028 }%
7029 }%
7030 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7031 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
7032 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
7033 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
7034 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
7035 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7036 \renewcommand*\glxtrfullformat[2]{%
7037   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7038   \ifglxtrininsertinside\else##2\fi
7039 }%
7040 \renewcommand*\glxtrfullplformat[2]{%
7041   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7042   \ifglxtrininsertinside\else##2\fi
7043 }%
7044 \renewcommand*\Glsxtrfullformat[2]{%
7045   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7046   \ifglxtrininsertinside\else##2\fi
7047 }%
7048 \renewcommand*\Glsxtrfullplformat[2]{%
7049   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7050   \ifglxtrininsertinside\else##2\fi
7051 }%

```

The first use full form and the inline full form use the short (long) style.

```

7052 \renewcommand*\glxtrininlinefullformat[2]{%
7053   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7054   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7055   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7056 }%
7057 \renewcommand*\glxtrininlinefullplformat[2]{%
7058   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7059   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7060   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7061 }%
7062 \renewcommand*\Glsxtrininlinefullformat[2]{%
7063   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7064   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7065   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7066 }%
7067 \renewcommand*\Glsxtrininlinefullplformat[2]{%
7068   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7069   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7070   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7071 }%
7072 }

```


rt-postfootnote

```
7073 \letabbreviationstyle{short-postfootnote}{postfootnote}
```

shortnolongname

```
7074 \newcommand*{\glxtrshortnolongname}{%
7075   \protect\glsabbrvfont{\the\glsshorttok}%
7076 }
```

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.

```
7077 \newabbreviationstyle{short}%
7078 {%
7079   \renewcommand*{\CustomAbbreviationFields}{%
7080     name={\glxtrshortnolongname},
7081     sort={\the\glsshorttok},
7082     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7083     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7084     text={\protect\glsabbrvfont{\the\glsshorttok}},
7085     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7086     description={\the\glslongtok}}%
7087   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7088     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7089 }%
7090 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7091 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7092 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7093 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7094 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7095 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7096 \renewcommand*{\glxtrinlinefullformat}[2]{%
7097   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7098   \ifglxtrininsertinside##2\fi}%
7099   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7100   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7101 }%
7102 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7103   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
7104   \ifglxtrininsertinside##2\fi}%
7105   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7106   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7107 }%
7108 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7109   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
```

```

7110     \ifglxtrinsertinside##2\fi}%
7111     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7112     \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7113 }%
7114 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7115     \protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}%
7116         \ifglxtrinsertinside##2\fi}%
7117     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7118     \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7119 }%

```

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.

```

7120 \renewcommand*{\glxtrfullformat}[2]{%
7121     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7122     \ifglxtrinsertinside\else##2\fi
7123 }%
7124 \renewcommand*{\glxtrfullplformat}[2]{%
7125     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7126     \ifglxtrinsertinside\else##2\fi
7127 }%
7128 \renewcommand*{\Glsxtrfullformat}[2]{%
7129     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7130     \ifglxtrinsertinside\else##2\fi
7131 }%
7132 \renewcommand*{\Glsxtrfullplformat}[2]{%
7133     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7134     \ifglxtrinsertinside\else##2\fi
7135 }%
7136 }

```

Set this as the default style for acronyms:

```

7137 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7138 \letabbreviationstyle{short-nolong}{short}

```

short-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7139 \newabbreviationstyle{short-nolong-noreg}%
7140 {%
7141     \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7142 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7143     \glshasattribute{\the\glslabeltok}{regular}%
7144     {%
7145         \glissetattribute{\the\glslabeltok}{regular}{false}%
7146     }%
7147     {}%
7148 }%

```

```

7149 }%
7150 {%
7151   \GlsXtrUseAbbrStyleFmts{short-nolong}%
7152 }

```

trshortdescname

```

7153 \newcommand*{\glxtrshortdescname}{%
7154   \protect\glsabbrvfont{\the\glsshorttok}%
7155 }

```

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.

```

7156 \newabbreviationstyle{short-desc}%
7157 {%
7158   \renewcommand*{\CustomAbbreviationFields}{%
7159     name={\glxtrshortdescname},
7160     sort={\the\glsshorttok},
7161     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7162     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7163     text={\protect\glsabbrvfont{\the\glsshorttok}},
7164     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7165     description={\the\glslongtok}}%
7166   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7167     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7168 }%
7169 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7170 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7171 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7172 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7173 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7174 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7175 \renewcommand*{\glxtrinlinefullformat}[2]{%
7176   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7177   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7178   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7179 }%
7180 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7181   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7182   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7183   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7184 }%
7185 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7186   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7187   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7188   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7189 }%

```

```

7190 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7191   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7192   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7193   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7194 }%

```

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.

```

7195 \renewcommand*{\glsxtrfullformat}[2]{%
7196   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7197   \ifglxtrinsertinside\else##2\fi
7198 }%
7199 \renewcommand*{\glsxtrfullplformat}[2]{%
7200   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7201   \ifglxtrinsertinside\else##2\fi
7202 }%
7203 \renewcommand*{\Glsxtrfullformat}[2]{%
7204   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7205   \ifglxtrinsertinside\else##2\fi
7206 }%
7207 \renewcommand*{\Glsxtrfullplformat}[2]{%
7208   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7209   \ifglxtrinsertinside\else##2\fi
7210 }%
7211 }

```

ort-nolong-desc

```

7212 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7213 \newabbreviationstyle{short-nolong-desc-noreg}%
7214 {%
7215   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

7216 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7217   \glshasattribute{\the\glslabeltok}{regular}%
7218   {%
7219     \glssetattribute{\the\glslabeltok}{regular}{false}%
7220   }%
7221   {}}%
7222 }%
7223 }%
7224 {%
7225   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7226 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7227 \newabbreviationstyle{nolong-short}%
7228 {%
7229   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7230 }%
7231 {%
7232   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7233 \renewcommand*{\glxtrinlinefullformat}[2]{%
7234   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7235   \ifglxtrininsertinside##2\fi}%
7236   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7237   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7238 }%
7239 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7240   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7241   \ifglxtrininsertinside##2\fi}%
7242   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7243   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7244 }%
7245 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7246   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7247   \ifglxtrininsertinside##2\fi}%
7248   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7249   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7250 }%
7251 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7252   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7253   \ifglxtrininsertinside##2\fi}%
7254   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7255   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7256 }%
7257 }

```

ong-short-noreg Like `nolong-short` but doesn't set the regular attribute.

```

7258 \newabbreviationstyle{nolong-short-noreg}%
7259 {%
7260   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7261 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7262   \glshasattribute{\the\glslabeltok}{regular}%
7263   {%
7264     \glissetattribute{\the\glslabeltok}{regular}{false}%
7265   }%
7266   {}%
7267 }%
7268 }%
7269 {%
7270   \GlsXtrUseAbbrStyleFmts{nolong-short}%

```

7271 }

noshortdescname

```
7272 \newcommand*{\glxtrlongnoshortdescname}{%
7273   \protect\glslongfont{\the\glslongtok}%
7274 }
```

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.

```
7275 \newabbreviationstyle{long-desc}%
7276 {%
7277   \renewcommand*{\CustomAbbreviationFields}{%
7278     name={\glxtrlongnoshortdescname},
7279     sort={\the\glslongtok},
7280     first={\protect\glsfirstlongfont{\the\glslongtok}},
7281     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7282     text={\glslongfont{\the\glslongtok}},
7283     plural={\glslongfont{\the\glslongpltok}}}%
7284   }%
7285   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7286     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7287 }%
7288 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7289 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7290 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7291 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7292 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7293 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7294 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7295   \glslongfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7296   \ifglxtrinsertinside \else##2\fi
7297 }%
7298 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7299   \glslongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7300   \ifglxtrinsertinside \else##2\fi
7301 }%
7302 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7303   \glslongfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7304   \ifglxtrinsertinside \else##2\fi
7305 }%
7306 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7307   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7308   \ifglxtrinsertinside \else##2\fi
7309 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```

7310 \renewcommand*{\glxtrinlinefullformat}[2]{%
7311   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7312   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7313   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7314 }%
7315 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7316   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7317   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7318   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7319 }%
7320 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7321   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7322   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7323   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7324 }%
7325 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7326   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7327   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7328   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7329 }%

```

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.

```

7330 \renewcommand*{\glxtrfullformat}[2]{%
7331   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7332   \ifglxtrinsertinside\else##2\fi
7333 }%
7334 \renewcommand*{\glxtrfullplformat}[2]{%
7335   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7336   \ifglxtrinsertinside\else##2\fi
7337 }%
7338 \renewcommand*{\Glsxtrfullformat}[2]{%
7339   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7340   \ifglxtrinsertinside\else##2\fi
7341 }%
7342 \renewcommand*{\Glsxtrfullplformat}[2]{%
7343   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7344   \ifglxtrinsertinside\else##2\fi
7345 }%
7346 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
7347 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

hort-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```

7348 \newabbreviationstyle{long-noshort-desc-noreg}%
7349 {%
7350   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

```

Unset the regular attribute if it has been set.

```
7351 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7352   \glshasattribute{\the\glslabeltok}{regular}%
7353   {%
7354     \glsselattribute{\the\glslabeltok}{regular}{false}%
7355   }%
7356   {}%
7357 }%
7358}%
7359{%
7360 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7361 }
```

longnoshortname

```
7362 \newcommand*{\glsxtrlongnoshortname}{%
7363   \protect\glsabbrvfont{\the\glsshorttok}%
7364 }
```

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.

```
7365 \newabbreviationstyle{long}%
7366 {%
7367   \renewcommand*{\CustomAbbreviationFields}{%
7368     name={\glsxtrlongnoshortname},
7369     sort={\the\glsshorttok},
7370     first={\protect\glsfirstlongfont{\the\glslongtok}},
7371     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7372     text={\glslongfont{\the\glslongtok}},
7373     plural={\glslongfont{\the\glslongpltok}},%
7374     description={\the\glslongtok}%
7375   }%
7376   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7377     \glsselattribute{\the\glslabeltok}{regular}{true}}%
7378 }%
7379 {%
7380   \GlsXtrUseAbbrStyleFmts{long-desc}%
7381 }
```

long-noshort Provide a synonym that matches similar styles.

```
7382 \letabbreviationstyle{long-noshort}{long}
```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```
7383 \newabbreviationstyle{long-noshort-noreg}%
7384 {%
7385   \GlsXtrUseAbbrStyleSetup{long-noshort}%
7386 }
```

Unset the regular attribute if it has been set.

```
7386 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```



```

7387 \glshasattribute{\the\glslabeltok}{regular}%
7388 {%
7389 \glissetattribute{\the\glslabeltok}{regular}{false}%
7390 }%
7391 {}%
7392 }%
7393 }%
7394 {%
7395 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7396 }

```

1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

`\glsxtrscfont` Maintained for backward-compatibility.

```
7397 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

`\glsabbrvscfont` Added for consistent naming.

```
7398 \newcommand*{\glsabbrvscfont}{\glsxtrscfont}
```

`\glsxtrfirstscfont` Maintained for backward-compatibility.

```
7399 \newcommand*{\glsxtrfirstscfont}[1]{\glsabbrvscfont{#1}}
```

`\glsfirstabbrvscfont` Added for consistent naming.

```
7400 \newcommand*{\glsfirstabbrvscfont}{\glsxtrfirstscfont}
```

and for the default short form suffix:

`\glsxtrscsuffix`

```
7401 \newcommand*{\glsxtrscsuffix}{\glstextup{\glsxtrabbrvpluralsuffix}}
```

`long-short-sc`

```

7402 \newabbreviationstyle{long-short-sc}%
7403 {%
7404 \renewcommand*{\CustomAbbreviationFields}{%
7405 name={\glsxtrlongshortname},
7406 sort={\the\glsshorttok},
7407 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7408 \protect\glsxtrfullsep{\the\glslabeltok}%
7409 \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7410 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7411 \protect\glsxtrfullsep{\the\glslabeltok}%
7412 \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7413 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}},%
7414 description={\the\glslongtok}}%
7415 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7416 \glshasattribute{\the\glslabeltok}{regular}%
7417 {%

```

```

7418     \glsetattribute{\the\glslabeltok}{regular}{false}%
7419   }%
7420   {}%
7421 }%
7422 }%
7423 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7424 \renewcommand*\abbrvpluralsuffix{\protect\glxtrscsuffix}%
7425 \renewcommand*\glsabbrvfont[1]{\glsabbrvscfont{##1}}%
7426 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%

```

Use the default long fonts.

```

7427 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7428 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7429 \renewcommand*\glsxtrfullformat[2]{%
7430   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7431   \ifglsxtrininsertinside\else##2\fi
7432   \glsxtrfullsep{##1}%
7433   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7434 }%
7435 \renewcommand*\glsxtrfullplformat[2]{%
7436   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7437   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7438   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7439 }%
7440 \renewcommand*\Glsxtrfullformat[2]{%
7441   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7442   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7443   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7444 }%
7445 \renewcommand*\Glsxtrfullplformat[2]{%
7446   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7447   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7448   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7449 }%
7450 }

```

g-short-sc-desc

```

7451 \newabbreviationstyle{long-short-sc-desc}%
7452 {%
7453   \renewcommand*\CustomAbbreviationFields{%
7454     name={\glsxtrlongshortdescname},
7455     sort={\glsxtrlongshortdescsort},%
7456     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7457       \protect\glsxtrfullsep{\the\glslabeltok}%
7458       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7459     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%

```

```

7460 \protect\glxtrfullsep{\the\glslabeltok}%
7461 \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7462 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7463 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7464 }%

```

Unset the regular attribute if it has been set.

```

7465 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7466 \glshasattribute{\the\glslabeltok}{regular}%
7467 {%
7468 \glissetattribute{\the\glslabeltok}{regular}{false}%
7469 }%
7470 {}%
7471 }%
7472 }%
7473 {%

```

As long-short-sc style:

```

7474 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7475 }

```

Now the short (long) version

```

7476 \newabbreviationstyle{short-sc-long}%
7477 {%
7478 \renewcommand*{\CustomAbbreviationFields}{%
7479 name={\glxtrshortlongname},
7480 sort={\the\glsshorttok},
7481 description={\the\glslongtok},%
7482 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7483 \protect\glxtrfullsep{\the\glslabeltok}%
7484 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7485 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7486 \protect\glxtrfullsep{\the\glslabeltok}%
7487 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7488 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7489 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7490 \glshasattribute{\the\glslabeltok}{regular}%
7491 {%
7492 \glissetattribute{\the\glslabeltok}{regular}{false}%
7493 }%
7494 {}%
7495 }%
7496 }%
7497 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7498 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7499 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvscfont{##1}}%
7500 \renewcommand*{\glsfirstabbrvfont[1]}{\glsfirstabbrvscfont{##1}}%

```

```

7501 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7502 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7503 \renewcommand*{\glsxtrfullformat}[2]{%
7504   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7505   \ifglsxtrininsertinside\else##2\fi
7506   \glsxtrfullsep{##1}%
7507   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7508 }%
7509 \renewcommand*{\glsxtrfullplformat}[2]{%
7510   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7511   \ifglsxtrininsertinside\else##2\fi
7512   \glsxtrfullsep{##1}%
7513   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7514 }%
7515 \renewcommand*{\Glsxtrfullformat}[2]{%
7516   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7517   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7518   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7519 }%
7520 \renewcommand*{\Glsxtrfullplformat}[2]{%
7521   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7522   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7523   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7524 }%
7525 }

```

As before but user provides description

```

7526 \newabbreviationstyle{short-sc-long-desc}%
7527 {%
7528   \renewcommand*{\CustomAbbreviationFields}{%
7529     name={\glsxtrshortlongdescname},
7530     sort={\glsxtrshortlongdescsort},
7531     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7532       \protect\glsxtrfullsep{\the\glslabeltok}%
7533       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7534     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7535       \protect\glsxtrfullsep{\the\glslabeltok}%
7536       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7537     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7538     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7539 }%

```

Unset the regular attribute if it has been set.

```

7540 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7541   \glshasattribute{\the\glslabeltok}{regular}%
7542   {%
7543     \glssetattribute{\the\glslabeltok}{regular}{false}%
7544   }%

```

```

7545 {}%
7546 }%
7547 }%
7548 {%

```

As short-sc-long style:

```

7549 \GlsXtrUseAbbrStyleFmts{short-sc-long}%
7550 }

```

short-sc

```

7551 \newabbreviationstyle{short-sc}%
7552 {%
7553 \renewcommand*{\CustomAbbreviationFields}{%
7554 name={\glxtrshortnolongname},
7555 sort={\the\glsshorttok},
7556 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7557 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7558 text={\protect\glsabbrvscfont{\the\glsshorttok}},
7559 plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7560 description={\the\gslongtok}}%
7561 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7562 \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7563 }%
7564 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7565 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7566 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7567 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7568 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7569 \renewcommand*{\gslongfont}[1]{\gslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7570 \renewcommand*{\glxtrinlinefullformat}[2]{%
7571 \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
7572 \ifglxtrininsertinside##2\fi}%
7573 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7574 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7575 }%
7576 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7577 \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
7578 \ifglxtrininsertinside##2\fi}%
7579 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7580 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7581 }%

7582 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7583 \protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}%
7584 \ifglxtrininsertinside##2\fi}%
7585 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7586 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%

```

```

7587 }%
7588 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7589   \protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}%
7590     \ifglxtrinsertinside##2\fi}%
7591   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7592   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7593 }%

```

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.

```

7594 \renewcommand*{\glxtrfullformat}[2]{%
7595   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7596   \ifglxtrinsertinside\else##2\fi
7597 }%
7598 \renewcommand*{\glxtrfullplformat}[2]{%
7599   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7600   \ifglxtrinsertinside\else##2\fi
7601 }%
7602 \renewcommand*{\Glsxtrfullformat}[2]{%
7603   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7604   \ifglxtrinsertinside\else##2\fi
7605 }%
7606 \renewcommand*{\Glsxtrfullplformat}[2]{%
7607   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7608   \ifglxtrinsertinside\else##2\fi
7609 }%
7610 }

```

short-sc-nolong

```

7611 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7612 \newabbreviationstyle{short-sc-desc}%
7613 {%
7614   \renewcommand*{\CustomAbbreviationFields}{%
7615     name={\glxtrshortdescname},
7616     sort={\the\glsshorttok},
7617     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7618     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7619     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7620     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7621     description={\the\glslongtok}}%
7622   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7623     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7624 }%
7625 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7626 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7627 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%

```

```

7628 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvscfont{##1}}%
7629 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7630 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7631 \renewcommand*\glsxtrinlinefullformat}[2]{%
7632   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7633   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7634   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7635 }%
7636 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7637   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7638   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7639   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7640 }%
7641 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7642   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7643   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7644   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7645 }%
7646 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7647   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7648   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7649   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7650 }%

```

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.

```

7651 \renewcommand*\glsxtrfullformat}[2]{%
7652   \glsfirstabbrvscfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7653   \ifglsxtrininsertinside\else##2\fi
7654 }%
7655 \renewcommand*\glsxtrfullplformat}[2]{%
7656   \glsfirstabbrvscfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7657   \ifglsxtrininsertinside\else##2\fi
7658 }%
7659 \renewcommand*\Glsxtrfullformat}[2]{%
7660   \glsfirstabbrvscfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7661   \ifglsxtrininsertinside\else##2\fi
7662 }%
7663 \renewcommand*\Glsxtrfullplformat}[2]{%
7664   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7665   \ifglsxtrininsertinside\else##2\fi
7666 }%
7667 }

```

-sc-nolong-desc

```

7668 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7669 \newabbreviationstyle{nolong-short-sc}%
7670 {%
7671   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
7672 }%
7673 {%
7674   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7675 \renewcommand*{\glxtrinlinefullformat}[2]{%
7676   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%
7677   \ifglxtrininsertinside##2\fi}%
7678 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7679 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7680 }%
7681 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7682   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%
7683   \ifglxtrininsertinside##2\fi}%
7684 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7685 \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7686 }%
7687 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7688   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%
7689   \ifglxtrininsertinside##2\fi}%
7690 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7691 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7692 }%
7693 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7694   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%
7695   \ifglxtrininsertinside##2\fi}%
7696 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7697 \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7698 }%
7699 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7700 \newabbreviationstyle{long-noshort-sc}%
7701 {%
7702   \renewcommand*{\CustomAbbreviationFields}{%
7703     name={\glxtrlongnoshortname},
7704     sort={\the\glsshorttok},
7705     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7706     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7707     text={\protect\glslongdefaultfont{\the\glslongtok}},
7708     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7709     description={\the\glslongtok}%
7710   }%
7711   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7712     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7713 }%

```


7714 {%

Use smallcaps and adjust the plural suffix to revert to upright.

```
7715 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7716 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7717 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7718 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7719 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
7720 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7721   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7722   \ifglxtrininsertinside \else##2\fi
7723 }%
7724 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7725   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7726   \ifglxtrininsertinside \else##2\fi
7727 }%
7728 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7729   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7730   \ifglxtrininsertinside \else##2\fi
7731 }%
7732 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7733   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7734   \ifglxtrininsertinside \else##2\fi
7735 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
7736 \renewcommand*{\glxtrinlinefullformat}[2]{%
7737   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7738   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7739   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7740 }%
7741 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7742   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7743   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7744   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7745 }%
7746 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7747   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7748   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7749   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7750 }%
7751 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7752   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7753   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7754   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7755 }%
```

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.

```

7756 \renewcommand*{\glxtrfullformat}[2]{%
7757   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7758   \ifglxtrinsertinside\else##2\fi
7759 }%
7760 \renewcommand*{\glxtrfullplformat}[2]{%
7761   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7762   \ifglxtrinsertinside\else##2\fi
7763 }%
7764 \renewcommand*{\Glsxtrfullformat}[2]{%
7765   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7766   \ifglxtrinsertinside\else##2\fi
7767 }%
7768 \renewcommand*{\Glsxtrfullplformat}[2]{%
7769   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7770   \ifglxtrinsertinside\else##2\fi
7771 }%
7772 }

```

long-sc Backward compatibility:

```

7773 \@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.

```

7774 \newabbreviationstyle{long-noshort-sc-desc}%
7775 {%
7776   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7777 }%
7778 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7779 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7780 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7781 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7782 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7783 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7784 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7785   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7786   \ifglxtrinsertinside \else##2\fi
7787 }%
7788 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7789   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7790   \ifglxtrinsertinside \else##2\fi
7791 }%
7792 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7793   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7794   \ifglxtrinsertinside \else##2\fi
7795 }%
7796 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```

```

7797 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7798 \ifglxtrinsertinside \else##2\fi
7799 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7800 \renewcommand*{\glxtrinlinefullformat}[2]{%
7801 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7802 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7803 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7804 }%
7805 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7806 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7807 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7808 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7809 }%
7810 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7811 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7812 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7813 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7814 }%
7815 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7816 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7817 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7818 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7819 }%

```

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.

```

7820 \renewcommand*{\glxtrfullformat}[2]{%
7821 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7822 \ifglxtrinsertinside\else##2\fi
7823 }%
7824 \renewcommand*{\glxtrfullplformat}[2]{%
7825 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7826 \ifglxtrinsertinside\else##2\fi
7827 }%
7828 \renewcommand*{\Glsxtrfullformat}[2]{%
7829 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7830 \ifglxtrinsertinside\else##2\fi
7831 }%
7832 \renewcommand*{\Glsxtrfullplformat}[2]{%
7833 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7834 \ifglxtrinsertinside\else##2\fi
7835 }%
7836 }

```

long-desc-sc Backward compatibility:

```

7837 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7838 \newabbreviationstyle{short-sc-footnote}%
7839 {%
7840   \renewcommand*{\CustomAbbreviationFields}{%
7841     name={\glxtrfootnotename},
7842     sort={\the\glsshorttok},
7843     description={\the\glslongtok},%
7844     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7845       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7846       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7847     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7848       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7849       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7850     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.

```

7851   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7852     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7853     \glsattribute{\the\glslabeltok}{regular}%
7854     {%
7855       \glssetattribute{\the\glslabeltok}{regular}{false}%
7856     }%
7857   }%
7858 }%
7859 }%
7860 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7861   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7862   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7863   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7864   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7865   \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7866   \renewcommand*{\glxtrfullformat}[2]{%
7867     \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7868     \ifglxtrinsertinside\else##2\fi
7869     \protect\glxtrabbrvfootnote{##1}%
7870     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7871   }%
7872   \renewcommand*{\glxtrfullplformat}[2]{%
7873     \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7874     \ifglxtrinsertinside\else##2\fi
7875     \protect\glxtrabbrvfootnote{##1}%
7876     {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7877   }%
7878   \renewcommand*{\Glsxtrfullformat}[2]{%
7879     \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7880     \ifglxtrinsertinside\else##2\fi
7881     \protect\glxtrabbrvfootnote{##1}%

```

```

7882     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7883 }%
7884 \renewcommand*{\Glsxtrfullplformat}[2]{%
7885   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7886   \ifglxtrinsertinside\else##2\fi
7887   \protect\glxtrabbrvfootnote{##1}%
7888   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7889 }%

```

The first use full form and the inline full form use the short (long) style.

```

7890 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7891   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7892   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7893   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7894 }%
7895 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7896   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7897   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7898   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7899 }%
7900 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7901   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7902   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7903   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7904 }%
7905 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7906   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7907   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7908   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7909 }%
7910 }

```

footnote-sc Backward compatibility:

```

7911 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

7912 \newabbreviationstyle{short-sc-postfootnote}%
7913 {%
7914   \renewcommand*{\CustomAbbreviationFields}{%
7915     name={\glxtrfootnotename},
7916     sort={\the\glsshorttok},
7917     description={\the\glslongtok},%
7918     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7919     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7920     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.

```

7921 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7922   \csdef{glxtrpostlink\glscategorylabel}{%

```

```

7923 \glxtrifwasfirstuse
7924 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7925 \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7926 {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7927 }%
7928 {%
7929 }%
7930 \glshasattribute{\the\glslabeltok}{regular}%
7931 {%
7932 \glissetattribute{\the\glslabeltok}{regular}{false}%
7933 }%
7934 {}%
7935 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7936 \renewcommand*{\glxtrsetupfulldefs}{%
7937 \let\glxtrifwasfirstuse\@secondoftwo
7938 }%
7939 }%
7940 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7941 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7942 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
7943 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
7944 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
7945 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7946 \renewcommand*{\glxtrfullformat}[2]{%
7947 \glfirstabbrvscfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7948 \ifglxtrininsertinside\else##2\fi
7949 }%
7950 \renewcommand*{\glxtrfullplformat}[2]{%
7951 \glfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7952 \ifglxtrininsertinside\else##2\fi
7953 }%
7954 \renewcommand*{\Glsxtrfullformat}[2]{%
7955 \glfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7956 \ifglxtrininsertinside\else##2\fi
7957 }%
7958 \renewcommand*{\Glsxtrfullplformat}[2]{%
7959 \glfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7960 \ifglxtrininsertinside\else##2\fi
7961 }%

```

The first use full form and the inline full form use the short (long) style.

```

7962 \renewcommand*{\glxtrinlinefullformat}[2]{%
7963   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7964   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7965   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7966 }%
7967 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7968   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7969   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7970   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7971 }%
7972 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7973   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7974   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7975   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7976 }%
7977 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7978   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7979   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7980   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7981 }%
7982 }

```

postfootnote-sc Backward compatibility:

```

7983 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

1.7.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

```

\glxtrsmfont Maintained for backward compatibility.
7984 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

\glsabbrvsmfont Added for consistent naming.
7985 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
7986 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
7987 \newcommand*{\glsfirstabbrvsmfont}{\glxtrfirstsmfont}

and for the default short form suffix:

\glxtrsmsuffix
7988 \newcommand*{\glxtrsmsuffix}{\glxtrabbrvpluralsuffix}

```

long-short-sm

```
7989 \newabbreviationstyle{long-short-sm}%
7990 {%
7991   \renewcommand*{\CustomAbbreviationFields}{%
7992     name={\glxtrlongshortname},
7993     sort={\the\glsshorttok},
7994     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7995       \protect\glxtrfullsep{\the\glslabeltok}%
7996       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7997     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7998       \protect\glxtrfullsep{\the\glslabeltok}%
7999       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
8000     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
8001     description={\the\glslongtok}}%
8002   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8003     \glshasattribute{\the\glslabeltok}{regular}%
8004     {%
8005       \glissetattribute{\the\glslabeltok}{regular}{false}%
8006     }%
8007   }%
8008 }%
8009 }%
8010 {%
8011   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8012   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8013   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%

```

Use the default long fonts.

```
8014 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8015 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
8016 \renewcommand*{\glxtrfullformat}[2]{%
8017   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8018   \ifglxtrininsertinside\else##2\fi
8019   \glxtrfullsep{##1}%
8020   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8021 }%
8022 \renewcommand*{\glxtrfullplformat}[2]{%
8023   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8024   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8025   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8026 }%
8027 \renewcommand*{\Glsxtrfullformat}[2]{%
8028   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8029   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8030   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8031 }%
8032 \renewcommand*{\Glsxtrfullplformat}[2]{%
8033   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%

```



```

8034 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8035 \glxtrparen{\glsfirstabbrvsmfont{\glssaccessshortpl{##1}}}%
8036 }%
8037 }

```

g-short-sm-desc

```

8038 \newabbreviationstyle{long-short-sm-desc}%
8039 {%
8040 \renewcommand*{\CustomAbbreviationFields}{%
8041   name={\glxtrlongshortdescname},
8042   sort={\glxtrlongshortdescsort},%
8043   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8044     \protect\glxtrfullsep{\the\glslabeltok}%
8045     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
8046   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8047     \protect\glxtrfullsep{\the\glslabeltok}%
8048     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
8049   text={\protect\glssabbrvsmfont{\the\glsshorttok}},%
8050   plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%
8051 }%

```

Unset the regular attribute if it has been set.

```

8052 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8053   \glshasattribute{\the\glslabeltok}{regular}%
8054   {%
8055     \glissetattribute{\the\glslabeltok}{regular}{false}%
8056   }%
8057   {}%
8058 }%
8059 }%
8060 {%

```

As long-short-sm style:

```

8061 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
8062 }

```

short-sm-long Now the short (long) version

```

8063 \newabbreviationstyle{short-sm-long}%
8064 {%
8065 \renewcommand*{\CustomAbbreviationFields}{%
8066   name={\glxtrshortlongname},
8067   sort={\the\glsshorttok},
8068   description={\the\glslongtok},%
8069   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8070     \protect\glxtrfullsep{\the\glslabeltok}%
8071     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8072   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8073     \protect\glxtrfullsep{\the\glslabeltok}%
8074     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8075   plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8076 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8077   \glshasattribute{\the\glslabeltok}{regular}%
8078   {%
8079     \glsselattribute{\the\glslabeltok}{regular}{false}%
8080   }%
8081   {}%
8082 }%
8083 }%
8084 {%
8085 \renewcommand*{\glssabrvfont[1]{\glssabrvsmfont{##1}}}%
8086 \renewcommand*{\glssfirstabrvfont[1]{\glssfirstabrvsmfont{##1}}}%
8087 \renewcommand*{\abbrvpluralsuffix}{\protect\glssxrsmssuffix}%
8088 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
8089 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8090 \renewcommand*{\glssxtrfullformat}[2]{%
8091   \glssfirstabrvsmfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8092   \ifglssxtrininsertinside\else##2\fi
8093   \glssxtrfullsep{##1}%
8094   \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslong{##1}}}%
8095 }%
8096 \renewcommand*{\glssxtrfullplformat}[2]{%
8097   \glssfirstabrvsmfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8098   \ifglssxtrininsertinside\else##2\fi
8099   \glssxtrfullsep{##1}%
8100   \glssxtrparen{\glssfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8101 }%
8102 \renewcommand*{\Glsxtrfullformat}[2]{%
8103   \glssfirstabrvsmfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8104   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8105   \glssxtrparen{\glssfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8106 }%
8107 \renewcommand*{\Glsxtrfullplformat}[2]{%
8108   \glssfirstabrvsmfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8109   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8110   \glssxtrparen{\glssfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8111 }%
8112 }

```

rt-sm-long-desc As before but user provides description

```

8113 \newabbreviationstyle{short-sm-long-desc}%
8114 {%
8115 \renewcommand*{\CustomAbbreviationFields}{%
8116   name={\glssxtrshortlongdescname},
8117   sort={\glssxtrshortlongdescsort},
8118   first={\protect\glssfirstabrvsmfont{\the\glssshorttok}}%
8119   \protect\glssxtrfullsep{\the\glslabeltok}%

```

```

8120     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8121     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8122     \protect\glxtrfullsep{\the\glslabeltok}%
8123     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8124     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
8125     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
8126 }%

```

Unset the regular attribute if it has been set.

```

8127 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8128     \glshasattribute{\the\glslabeltok}{regular}%
8129     {%
8130         \glssetattribute{\the\glslabeltok}{regular}{false}%
8131     }%
8132 }%
8133 }%
8134 }%
8135 {%

```

As short-sm-long style:

```

8136 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8137 }

```

short-sm

```

8138 \newabbreviationstyle{short-sm}%
8139 {%
8140     \renewcommand*{\CustomAbbreviationFields}{%
8141         name={\glxtrshortnolongname},
8142         sort={\the\glsshorttok},
8143         first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8144         firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8145         text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8146         plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8147         description={\the\glslongtok}}%
8148     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8149         \glssetattribute{\the\glslabeltok}{regular}{true}}%
8150 }%
8151 {%
8152     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8153     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8154     \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8155     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8156     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8157 \renewcommand*{\glxtrinlinefullformat}[2]{%
8158     \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8159     \ifglxtrininsertinside##2\fi}%
8160     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8161     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%

```

```

8162 }%
8163 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8164   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}%
8165     \ifglxtrininsertinside##2\fi}%
8166   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8167   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8168 }%

8169 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8170   \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}%
8171     \ifglxtrininsertinside##2\fi}%
8172   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8173   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8174 }%

8175 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8176   \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}%
8177     \ifglxtrininsertinside##2\fi}%
8178   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8179   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8180 }%

```

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.

```

8181 \renewcommand*{\glxtrfullformat}[2]{%
8182   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8183   \ifglxtrininsertinside\else##2\fi
8184 }%
8185 \renewcommand*{\glxtrfullplformat}[2]{%
8186   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8187   \ifglxtrininsertinside\else##2\fi
8188 }%
8189 \renewcommand*{\Glsxtrfullformat}[2]{%
8190   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8191   \ifglxtrininsertinside\else##2\fi
8192 }%
8193 \renewcommand*{\Glsxtrfullplformat}[2]{%
8194   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8195   \ifglxtrininsertinside\else##2\fi
8196 }%
8197 }

```

short-sm-nolong

```

8198 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

8199 \newabbreviationstyle{short-sm-desc}{%
8200 {%
8201   \renewcommand*{\CustomAbbreviationFields}{%
8202     name={\glxtrshortdescname},

```

```

8203     sort={\the\glsshorttok},
8204     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8205     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8206     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8207     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8208     description={\the\glslongtok}}%
8209 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8210   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8211 }%
8212 {%
8213   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8214   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8215   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrmsuffix}%
8216   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8217   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8218   \renewcommand*{\glsxtrinlinefullformat}[2]{%
8219     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8220     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8221     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8222   }%
8223   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8224     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8225     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8226     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8227   }%
8228   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8229     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8230     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8231     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8232   }%
8233   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8234     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8235     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8236     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8237   }%

```

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.

```

8238   \renewcommand*{\glsxtrfullformat}[2]{%
8239     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8240     \ifglsxtrininsertinside\else##2\fi
8241   }%
8242   \renewcommand*{\glsxtrfullplformat}[2]{%
8243     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8244     \ifglsxtrininsertinside\else##2\fi
8245   }%
8246   \renewcommand*{\Glsxtrfullformat}[2]{%
8247     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

8248     \ifglxtrinsertinside\else##2\fi
8249 }%
8250 \renewcommand*{\Glsxtrfullplformat}[2]{%
8251     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8252     \ifglxtrinsertinside\else##2\fi
8253 }%
8254 }

```

-sm-nolong-desc

```

8255 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8256 \newabbreviationstyle{nolong-short-sm}%
8257 {%
8258     \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8259 }%
8260 {%
8261     \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8262 \renewcommand*{\glxtrinlinefullformat}[2]{%
8263     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8264         \ifglxtrinsertinside##2\fi}%
8265     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8266     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8267 }%
8268 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8269     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8270         \ifglxtrinsertinside##2\fi}%
8271     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8272     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8273 }%
8274 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8275     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8276         \ifglxtrinsertinside##2\fi}%
8277     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8278     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8279 }%
8280 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8281     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8282         \ifglxtrinsertinside##2\fi}%
8283     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8284     \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8285 }%
8286 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8287 \newabbreviationstyle{long-noshort-sm}%

```

```

8288 {%
8289   \renewcommand*{\CustomAbbreviationFields}{%
8290     name={\glstrlongnosshortname},
8291     sort={\the\glssshorttok},
8292     first={\protect\glstrfirstlongdefaultfont{\the\glstrlongtok}},
8293     firstplural={\protect\glstrfirstlongdefaultfont{\the\glstrlongpltok}},
8294     text={\protect\glstrlongdefaultfont{\the\glstrlongtok}},
8295     plural={\protect\glstrlongdefaultfont{\the\glstrlongpltok}},%
8296     description={\the\glstrlongtok}%
8297   }%
8298   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8299     \glsssetAttribute{\the\glstrlabeltok}{regular}{true}}%
8300 }%
8301 {%
8302   \renewcommand*{\glstrabbrvfont}[1]{\glstrabbrvsmfont{##1}}%
8303   \renewcommand*{\glstrfirstabbrvfont}[1]{\glstrfirstabbrvsmfont{##1}}%
8304   \renewcommand*{\abbrvpluralsuffix}{\protect\glstrsrmsuffix}%
8305   \renewcommand*{\glstrfirstlongfont}[1]{\glstrfirstlongdefaultfont{##1}}%
8306   \renewcommand*{\glstrlongfont}[1]{\glstrlongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8307   \renewcommand*{\glstrsubsequentfmt}[2]{%
8308     \glstrlongdefaultfont{\glstraccesslong{##1}\ifglstrinsertinside ##2\fi}%
8309     \ifglstrinsertinside \else##2\fi
8310   }%
8311   \renewcommand*{\glstrsubsequentplfmt}[2]{%
8312     \glstrlongdefaultfont{\glstraccesslongpl{##1}\ifglstrinsertinside ##2\fi}%
8313     \ifglstrinsertinside \else##2\fi
8314   }%
8315   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8316     \glstrlongdefaultfont{\Glsstraccesslong{##1}\ifglstrinsertinside ##2\fi}%
8317     \ifglstrinsertinside \else##2\fi
8318   }%
8319   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8320     \glstrlongdefaultfont{\Glsstraccesslongpl{##1}\ifglstrinsertinside ##2\fi}%
8321     \ifglstrinsertinside \else##2\fi
8322   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8323   \renewcommand*{\glstrinlinefullformat}[2]{%
8324     \glstrfirstlongdefaultfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
8325     \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
8326     \glstrparen{\protect\glstrfirstabbrvsmfont{\glstraccessshort{##1}}}%
8327   }%
8328   \renewcommand*{\glstrinlinefullplformat}[2]{%
8329     \glstrfirstlongdefaultfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
8330     \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
8331     \glstrparen{\protect\glstrfirstabbrvsmfont{\glstraccessshortpl{##1}}}%
8332   }%
8333   \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

8334 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8335 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8336 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8337 }%
8338 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8339 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8340 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8341 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8342 }%

```

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.

```

8343 \renewcommand*{\glxtrfullformat}[2]{%
8344 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8345 \ifglxtrinsertinside\else##2\fi
8346 }%
8347 \renewcommand*{\glxtrfullplformat}[2]{%
8348 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8349 \ifglxtrinsertinside\else##2\fi
8350 }%
8351 \renewcommand*{\Glsxtrfullformat}[2]{%
8352 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8353 \ifglxtrinsertinside\else##2\fi
8354 }%
8355 \renewcommand*{\Glsxtrfullplformat}[2]{%
8356 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8357 \ifglxtrinsertinside\else##2\fi
8358 }%
8359 }

```

long-sm Backward compatibility:

```

8360 \@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`.

```

8361 \newabbreviationstyle{long-noshort-sm-desc}%
8362 {%
8363 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8364 }%
8365 {%
8366 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8367 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8368 \renewcommand*\{abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8369 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8370 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8371 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8372 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8373 \ifglxtrinsertinside \else##2\fi

```



```

8374 }%
8375 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8376   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8377   \ifglxtrinsertinside \else##2\fi
8378 }%
8379 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8380   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8381   \ifglxtrinsertinside \else##2\fi
8382 }%
8383 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8384   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8385   \ifglxtrinsertinside \else##2\fi
8386 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8387 \renewcommand*{\glxtrinlinefullformat}[2]{%
8388   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8389   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8390   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8391 }%
8392 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8393   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8394   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8395   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8396 }%
8397 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8398   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8399   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8400   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8401 }%
8402 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8403   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8404   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8405   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8406 }%

```

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.

```

8407 \renewcommand*{\glxtrfullformat}[2]{%
8408   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8409   \ifglxtrinsertinside\else##2\fi
8410 }%
8411 \renewcommand*{\glxtrfullplformat}[2]{%
8412   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8413   \ifglxtrinsertinside\else##2\fi
8414 }%
8415 \renewcommand*{\Glsxtrfullformat}[2]{%
8416   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8417   \ifglxtrinsertinside\else##2\fi
8418 }%

```

```

8419 \renewcommand*{\Glsxtrfullplformat}[2]{%
8420   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8421   \ifglxtrinsertinside\else##2\fi
8422 }%
8423 }

```

long-desc-sm Backward compatibility:

```

8424 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

8425 \newabbreviationstyle{short-sm-footnote}%
8426 {%
8427   \renewcommand*{\CustomAbbreviationFields}{%
8428     name={\glsxtrfootnotename},
8429     sort={\the\glsshorttok},
8430     description={\the\glslongtok},%
8431     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8432       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8433       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8434     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8435       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8436       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8437     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.

```

8438 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8439   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8440   \glshasattribute{\the\glslabeltok}{regular}%
8441   {%
8442     \glssetattribute{\the\glslabeltok}{regular}{false}%
8443   }%
8444   {}%
8445 }%
8446 }%
8447 {%
8448   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8449   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8450   \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
8451   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8452   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8453 \renewcommand*{\glsxtrfullformat}[2]{%
8454   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8455   \ifglxtrinsertinside\else##2\fi
8456   \protect\glxtrabbrvfootnote{##1}%
8457   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8458 }%
8459 \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

8460 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8461 \ifglxtrinsertinside\else##2\fi
8462 \protect\glxtrabbrvfootnote{##1}%
8463 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8464 }%
8465 \renewcommand*{\Glsxtrfullformat}[2]{%
8466 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8467 \ifglxtrinsertinside\else##2\fi
8468 \protect\glxtrabbrvfootnote{##1}%
8469 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8470 }%
8471 \renewcommand*{\Glsxtrfullplformat}[2]{%
8472 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8473 \ifglxtrinsertinside\else##2\fi
8474 \protect\glxtrabbrvfootnote{##1}%
8475 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8476 }%

```

The first use full form and the inline full form use the short (long) style.

```

8477 \renewcommand*{\glxtrininlinefullformat}[2]{%
8478 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8479 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8480 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8481 }%
8482 \renewcommand*{\glxtrininlinefullplformat}[2]{%
8483 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8484 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8485 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8486 }%
8487 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
8488 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8489 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8490 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8491 }%
8492 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
8493 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8494 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8495 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8496 }%
8497 }

```

footnote-sm Backward compatibility:

```

8498 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8499 \newabbreviationstyle{short-sm-postfootnote}%
8500 {%
8501 \renewcommand*{\CustomAbbreviationFields}{%
8502 name={\glxtrfootnotename},
8503 sort={\the\glsshorttok},

```

```

8504   description={\the\glslongtok},%
8505   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8506   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8507   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.

```

8508   \renewcommand*\GlsXtrPostNewAbbreviation{%
8509     \csdef{glsxtrpostlink\glscategorylabel}{%
8510       \glsxtrifwasfirstuse
8511       {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8512         \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}}%
8513         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
8514       }%
8515     }%
8516   }%
8517   \glshasattribute{\the\glslabeltok}{regular}%
8518   {%
8519     \glssetattribute{\the\glslabeltok}{regular}{false}%
8520   }%
8521   {}%
8522 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

8523   \renewcommand*\glsxtrsetupfulldefs{%
8524     \let\glsxtrifwasfirstuse\@secondoftwo
8525   }%
8526 }%
8527 {%
8528   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8529   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8530   \renewcommand*\{abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
8531   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8532   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8533   \renewcommand*\glsxtrfullformat[2]{%
8534     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8535     \ifglsxtrininsertinside\else##2\fi
8536   }%
8537   \renewcommand*\glsxtrfullplformat[2]{%
8538     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8539     \ifglsxtrininsertinside\else##2\fi
8540   }%
8541   \renewcommand*\Glsxtrfullformat[2]{%
8542     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8543     \ifglsxtrininsertinside\else##2\fi

```

```

8544 }%
8545 \renewcommand*{\Glsxtrfullplformat}[2]{%
8546   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8547   \ifglxtrinsertinside\else##2\fi
8548 }%

```

The first use full form and the inline full form use the short (long) style.

```

8549 \renewcommand*{\glxtrinlinefullformat}[2]{%
8550   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8551   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8552   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8553 }%
8554 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8555   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8556   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8557   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8558 }%
8559 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8560   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8561   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8562   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
8563 }%
8564 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8565   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8566   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8567   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8568 }%
8569 }

```

postfootnote-sm Backward compatibility:

```

8570 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

8571 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%

```

`\glsfirstabbrvemfont`

```

8572 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%

```

The default short form suffix:

`\glxtremsuffix`

```

8573 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```

8574 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```
8575 \newcommand*{\glslongemfont}[1]{\emph{#1}}%
```

`long-short-em` The long form is just set in the default long font.

```
8576 \newabbreviationstyle{long-short-em}%
8577 {%
8578   \renewcommand*{\CustomAbbreviationFields}{%
8579     name={\glsxtrlongshortname},
8580     sort={\the\glsshorttok},
8581     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8582       \protect\glsxtrfullsep{\the\glslabeltok}%
8583       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8584     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8585       \protect\glsxtrfullsep{\the\glslabeltok}%
8586       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8587     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8588     description={\the\glslongtok}}%
8589   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8590     \glsattribute{\the\glslabeltok}{regular}%
8591     {%
8592       \glssetattribute{\the\glslabeltok}{regular}{false}%
8593     }%
8594   }%
8595 }%
8596 }%
8597 {%
8598   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8599   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8600   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```
8601 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8602 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8603 \renewcommand*{\glsxtrfullformat}[2]{%
8604   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8605   \ifglsxtrininsertinside\else##2\fi
8606   \glsxtrfullsep{##1}%
8607   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8608 }%
8609 \renewcommand*{\glsxtrfullplformat}[2]{%
8610   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8611   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8612   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8613 }%
8614 \renewcommand*{\Glsxtrfullformat}[2]{%
8615   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8616   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8617   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

8618 }%
8619 \renewcommand*{\Glsxtrfullplformat}[2]{%
8620   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8621   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8622   \glsxtrparen{\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8623 }%
8624 }

```

g-short-em-desc

```

8625 \newabbreviationstyle{long-short-em-desc}%
8626 {%
8627   \renewcommand*{\CustomAbbreviationFields}{%
8628     name={\glsxtrlongshortdescname},
8629     sort={\glsxtrlongshortdescsort},%
8630     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8631       \protect\glsxtrfullsep{\the\glslabeltok}%
8632       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8633     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8634       \protect\glsxtrfullsep{\the\glslabeltok}%
8635       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8636     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8637     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8638 }%

```

Unset the regular attribute if it has been set.

```

8639 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8640   \glshasattribute{\the\glslabeltok}{regular}%
8641   {%
8642     \glissetattribute{\the\glslabeltok}{regular}{false}%
8643   }%
8644   {}}%
8645 }%
8646 }%
8647 {%

```

As long-short-em style:

```

8648 \GlsXtrUseAbbrStyleFmts{long-short-em}%
8649 }

```

ong-em-short-em

```

8650 \newabbreviationstyle{long-em-short-em}%
8651 {%
8652   \glslongemfont is used in the description since \glsdesc doesn't set the style.
8652   \renewcommand*{\CustomAbbreviationFields}{%
8653     name={\glsxtrlongshortname},
8654     sort={\the\glsshorttok},
8655     first={\protect\glsfirstlongemfont{\the\glslongtok}%
8656       \protect\glsxtrfullsep{\the\glslabeltok}%
8657       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%

```

```

8658 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}}%
8659 \protect\glsxtrfullsep{\the\glslabeltok}}%
8660 \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8661 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8662 description={\protect\glsfirstlongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8663 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8664 \glshasattribute{\the\glslabeltok}{regular}}%
8665 {%
8666 \glssetattribute{\the\glslabeltok}{regular}{false}}%
8667 }%
8668 {}%
8669 }%
8670 }%
8671 {%
8672 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8673 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8674 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8675 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8676 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8677 \renewcommand*{\glsxtrfullformat}[2]{%
8678 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8679 \ifglsxtrininsertinside\else##2\fi
8680 \glsxtrfullsep{##1}}%
8681 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8682 }%
8683 \renewcommand*{\glsxtrfullplformat}[2]{%
8684 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8685 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8686 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8687 }%
8688 \renewcommand*{\Glsxtrfullformat}[2]{%
8689 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8690 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8691 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8692 }%
8693 \renewcommand*{\Glsxtrfullplformat}[2]{%
8694 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8695 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8696 \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8697 }%
8698 }

```

m-short-em-desc

```

8699 \newabbreviationstyle{long-em-short-em-desc}%
8700 {%

```



```

8701 \renewcommand*{\CustomAbbreviationFields}{%
8702   name={\glxtrlongshortdescname},
8703   sort={\glxtrlongshortdescsort},%
8704   first={\protect\glsfirstlongemfont{\the\glslongtok}%
8705     \protect\glxtrfullsep{\the\glslabeltok}%
8706     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8707   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8708     \protect\glxtrfullsep{\the\glslabeltok}%
8709     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8710   text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8711   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8712 }%

```

Unset the regular attribute if it has been set.

```

8713 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8714   \glshasattribute{\the\glslabeltok}{regular}%
8715   {%
8716     \glissetattribute{\the\glslabeltok}{regular}{false}%
8717   }%
8718   {}%
8719 }%
8720 }%
8721 {%
8722   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8723 }

```

short-em-long Now the short (long) version

```

8724 \newabbreviationstyle{short-em-long}%
8725 {%
8726   \renewcommand*{\CustomAbbreviationFields}{%
8727     name={\glxtrshortlongname},
8728     sort={\the\glsshorttok},
8729     description={\the\glslongtok},%
8730     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8731       \protect\glxtrfullsep{\the\glslabeltok}%
8732       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8733     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8734       \protect\glxtrfullsep{\the\glslabeltok}%
8735       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8736     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8737 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8738   \glshasattribute{\the\glslabeltok}{regular}%
8739   {%
8740     \glissetattribute{\the\glslabeltok}{regular}{false}%
8741   }%
8742   {}%
8743 }%
8744 }%

```

8745 {%

Mostly as short-long style:

```
8746 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8747 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8748 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8749 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8750 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8751 \renewcommand*{\glsxtrfullformat}[2]{%
8752   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8753   \ifglsxtrininsertinside\else##2\fi
8754   \glsxtrfullsep{##1}%
8755   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8756 }%
8757 \renewcommand*{\glsxtrfullplformat}[2]{%
8758   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8759   \ifglsxtrininsertinside\else##2\fi
8760   \glsxtrfullsep{##1}%
8761   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8762 }%
8763 \renewcommand*{\Glsxtrfullformat}[2]{%
8764   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8765   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8766   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8767 }%
8768 \renewcommand*{\Glsxtrfullplformat}[2]{%
8769   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8770   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8771   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8772 }%
8773 }
```

rt-em-long-desc As before but user provides description

```
8774 \newabbreviationstyle{short-em-long-desc}%
8775 {%
8776   \renewcommand*{\CustomAbbreviationFields}{%
8777     name={\glsxtrshortlongdescname},
8778     sort={\glsxtrshortlongdescsort},
8779     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8780     \protect\glsxtrfullsep{\the\glslabeltok}%
8781     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8782     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8783     \protect\glsxtrfullsep{\the\glslabeltok}%
8784     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8785     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8786     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8787   }%
```

Unset the regular attribute if it has been set.

```

8788 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8789   \glshasattribute{\the\glslabeltok}{regular}%
8790   {%
8791     \glissetattribute{\the\glslabeltok}{regular}{false}%
8792   }%
8793   {}%
8794 }%
8795 }%
8796 {%
8797   \GlsXtrUseAbbrStyleFmts{short-em-long}%
8798 }

```

hort-em-long-em

```

8799 \newabbreviationstyle{short-em-long-em}%
8800 {%

```

\glslongemfont is used in the description since \glsgdesc doesn't set the style.

```

8801 \renewcommand*{\CustomAbbreviationFields}{%
8802   name={\glsxtrshortlongname},
8803   sort={\the\glsshorttok},
8804   description={\protect\glslongemfont{\the\glslongtok}},%
8805   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8806     \protect\glsxtrfullsep{\the\glslabeltok}%
8807     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8808   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8809     \protect\glsxtrfullsep{\the\glslabeltok}%
8810     \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8811   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8812 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8813   \glshasattribute{\the\glslabeltok}{regular}%
8814   {%
8815     \glissetattribute{\the\glslabeltok}{regular}{false}%
8816   }%
8817   {}%
8818 }%
8819 }%
8820 {%
8821 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8822 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8823 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8824 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8825 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8826 \renewcommand*{\glsxtrfullformat}[2]{%
8827   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8828   \ifglsxtrininsertinside\else##2\fi
8829   \glsxtrfullsep{##1}%

```

```

8830 \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8831 }%
8832 \renewcommand*{\glsxtrfullplformat}[2]{%
8833 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8834 \ifglsxtrininsertinside\else##2\fi
8835 \glsxtrfullsep{##1}%
8836 \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8837 }%
8838 \renewcommand*{\Glsxtrfullformat}[2]{%
8839 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8840 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8841 \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8842 }%
8843 \renewcommand*{\Glsxtrfullplformat}[2]{%
8844 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8845 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8846 \glsxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8847 }%
8848 }

```

em-long-em-desc

```

8849 \newabbreviationstyle{short-em-long-em-desc}%
8850 {%
8851 \renewcommand*{\CustomAbbreviationFields}{%
8852 name={\glsxtrshortlongdescname},%
8853 sort={\glsxtrshortlongdescsort},%
8854 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8855 \protect\glsxtrfullsep{\the\glslabeltok}%
8856 \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8857 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8858 \protect\glsxtrfullsep{\the\glslabeltok}%
8859 \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8860 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8861 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8862 }%

```

Unset the regular attribute if it has been set.

```

8863 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8864 \glshasattribute{\the\glslabeltok}{regular}%
8865 {%
8866 \glissetattribute{\the\glslabeltok}{regular}{false}%
8867 }%
8868 {}%
8869 }%
8870 }%
8871 {%
8872 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8873 }

```

short-em

```

8874 \newabbreviationstyle{short-em}%
8875 {%
8876   \renewcommand*{\CustomAbbreviationFields}{%
8877     name={\glxtrshortnolongname},
8878     sort={\the\glsshorttok},
8879     first={\protect\glfirstabbrvemfont{\the\glsshorttok}},
8880     firstplural={\protect\glfirstabbrvemfont{\the\glsshortpltok}},
8881     text={\protect\glabbrvemfont{\the\glsshorttok}},
8882     plural={\protect\glabbrvemfont{\the\glsshortpltok}},
8883     description={\the\glslongtok}}%
8884   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8885     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8886 }%
8887 {%
8888   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8889   \renewcommand*{\glabbrvfnt}[1]{\glabbrvemfont{##1}}%
8890   \renewcommand*{\glfirstabbrvfnt}[1]{\glfirstabbrvemfont{##1}}%
8891   \renewcommand*{\glfirstlongfnt}[1]{\glfirstlongdefaultfont{##1}}%
8892   \renewcommand*{\glslongfnt}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8893   \renewcommand*{\glxtrinlinefullformat}[2]{%
8894     \protect\glfirstabbrvemfont{\glaccessshort{##1}}%
8895     \ifglxtrininsertinside##2\fi}%
8896   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8897   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
8898 }%
8899   \renewcommand*{\glxtrinlinefullplformat}[2]{%
8900     \protect\glfirstabbrvemfont{\glaccessshortpl{##1}}%
8901     \ifglxtrininsertinside##2\fi}%
8902   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8903   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8904 }%
8905   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8906     \protect\glfirstabbrvemfont{\Glsaccessshort{##1}}%
8907     \ifglxtrininsertinside##2\fi}%
8908   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8909   \glxtrparen{\glfirstlongdefaultfont{\glaccesslong{##1}}}%
8910 }%
8911   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8912     \protect\glfirstabbrvemfont{\Glsaccessshortpl{##1}}%
8913     \ifglxtrininsertinside##2\fi}%
8914   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8915   \glxtrparen{\glfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8916 }%

```

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.

```

8917   \renewcommand*{\glxtrfullformat}[2]{%

```

```

8918 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8919 \ifglxtrinsertinside\else##2\fi
8920 }%
8921 \renewcommand*{\glxtrfullplformat}[2]{%
8922 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8923 \ifglxtrinsertinside\else##2\fi
8924 }%
8925 \renewcommand*{\Glsxtrfullformat}[2]{%
8926 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8927 \ifglxtrinsertinside\else##2\fi
8928 }%
8929 \renewcommand*{\Glsxtrfullplformat}[2]{%
8930 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8931 \ifglxtrinsertinside\else##2\fi
8932 }%
8933 }

```

short-em-nolong

```

8934 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8935 \newabbreviationstyle{short-em-desc}%
8936 {%
8937 \renewcommand*{\CustomAbbreviationFields}{%
8938 name={\glxtrshortdescname},
8939 sort={\the\glsshorttok},
8940 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8941 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8942 text={\protect\glsabbrvemfont{\the\glsshorttok}},
8943 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8944 description={\the\glslongtok}}%
8945 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8946 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8947 }%
8948 {%
8949 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8950 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8951 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8952 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8953 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8954 \renewcommand*{\glxtrinlinefullformat}[2]{%
8955 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8956 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8957 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8958 }%
8959 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8960 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8961 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

8962 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8963 }%
8964 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8965 \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8966 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8967 \glstrparen{\glfirstlongdefaultfont{\glsaccesslong{##1}}}%
8968 }%
8969 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8970 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8971 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8972 \glstrparen{\glfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8973 }%

```

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.

```

8974 \renewcommand*{\glxtrfullformat}[2]{%
8975 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8976 \ifglxtrinsertinside\else##2\fi
8977 }%
8978 \renewcommand*{\glxtrfullplformat}[2]{%
8979 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8980 \ifglxtrinsertinside\else##2\fi
8981 }%
8982 \renewcommand*{\Glsxtrfullformat}[2]{%
8983 \glfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8984 \ifglxtrinsertinside\else##2\fi
8985 }%
8986 \renewcommand*{\Glsxtrfullplformat}[2]{%
8987 \glfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8988 \ifglxtrinsertinside\else##2\fi
8989 }%
8990 }

```

-em-nolong-desc

```

8991 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8992 \newabbreviationstyle{nolong-short-em}%
8993 {%
8994 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8995 }%
8996 {%
8997 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8998 \renewcommand*{\glxtrinlinefullformat}[2]{%
8999 \protect\glfirstlongdefaultfont{\glsaccesslong{##1}%
9000 \ifglxtrinsertinside##2\fi}%
9001 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9002 \glstrparen{\glfirstabbrvemfont{\glsaccessshort{##1}}}%

```

```

9003 }%
9004 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9005   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
9006     \ifglxtrinsertinside##2\fi}%
9007   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9008   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9009 }%
9010 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9011   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
9012     \ifglxtrinsertinside##2\fi}%
9013   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9014   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9015 }%
9016 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9017   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
9018     \ifglxtrinsertinside##2\fi}%
9019   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9020   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9021 }%
9022 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

9023 \newabbreviationstyle{long-noshort-em}%
9024 {%
9025   \renewcommand*{\CustomAbbreviationFields}{%
9026     name={\glxtrlongnoshortname},
9027     sort={\the\glsshorttok},
9028     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9029     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9030     text={\protect\glslongdefaultfont{\the\glslongtok}},
9031     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9032     description={\the\glslongtok}%
9033   }%
9034   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9035     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9036 }%
9037 {%
9038   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9039   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9040   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9041   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9042   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9043 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9044   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9045   \ifglxtrinsertinside \else##2\fi
9046 }%
9047 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9048   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%

```



```

9049 \ifglxtrinsertinside \else##2\fi
9050 }%
9051 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9052 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9053 \ifglxtrinsertinside \else##2\fi
9054 }%
9055 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9056 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9057 \ifglxtrinsertinside \else##2\fi
9058 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9059 \renewcommand*{\glxtrinlinefullformat}[2]{%
9060 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9061 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9062 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9063 }%
9064 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9065 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9066 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9067 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9068 }%
9069 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9070 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9071 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9072 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9073 }%
9074 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9075 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9076 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9077 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9078 }%

```

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.

```

9079 \renewcommand*{\glxtrfullformat}[2]{%
9080 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9081 \ifglxtrinsertinside\else##2\fi
9082 }%
9083 \renewcommand*{\glxtrfullplformat}[2]{%
9084 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9085 \ifglxtrinsertinside\else##2\fi
9086 }%
9087 \renewcommand*{\Glsxtrfullformat}[2]{%
9088 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9089 \ifglxtrinsertinside\else##2\fi
9090 }%
9091 \renewcommand*{\Glsxtrfullplformat}[2]{%
9092 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9093 \ifglxtrinsertinside\else##2\fi

```

```

9094 }%
9095 }

```

long-em Backward compatibility:

```

9096 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

9097 \newabbreviationstyle{long-em-noshort-em}%
9098 {%
9099   \renewcommand*{\CustomAbbreviationFields}{%
9100     name={\glxtrlongnoshortname},
9101     sort={\the\glsshorttok},
9102     first={\protect\glslongemfont{\the\glslongtok}},
9103     firstplural={\protect\glslongemfont{\the\glslongpltok}},
9104     text={\protect\glslongemfont{\the\glslongtok}},
9105     plural={\protect\glslongemfont{\the\glslongpltok}},%
9106     description={\protect\glslongemfont{\the\glslongtok}}%
9107   }%
9108   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9109     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9110 }%
9111 {%
9112   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9113   \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
9114   \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
9115   \renewcommand*{\glfirstlongfont}[1]{\glfirstlongemfont{##1}}%
9116   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9117 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9118   \glslongemfont{\glaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9119   \ifglxtrininsertinside \else##2\fi
9120 }%
9121 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9122   \glslongemfont{\glaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9123   \ifglxtrininsertinside \else##2\fi
9124 }%
9125 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9126   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9127   \ifglxtrininsertinside \else##2\fi
9128 }%
9129 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9130   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9131   \ifglxtrininsertinside \else##2\fi
9132 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9133 \renewcommand*{\glxtrinlinefullformat}[2]{%
9134   \glslongemfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9135   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

9136 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9137 }%
9138 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9139 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9140 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9141 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9142 }%
9143 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9144 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9145 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9146 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9147 }%
9148 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9149 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9150 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
9151 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9152 }%

```

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.

```

9153 \renewcommand*{\glsxtrfullformat}[2]{%
9154 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9155 \ifglsxtrinsertinside\else##2\fi
9156 }%
9157 \renewcommand*{\glsxtrfullplformat}[2]{%
9158 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9159 \ifglsxtrinsertinside\else##2\fi
9160 }%
9161 \renewcommand*{\Glsxtrfullformat}[2]{%
9162 \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
9163 \ifglsxtrinsertinside\else##2\fi
9164 }%
9165 \renewcommand*{\Glsxtrfullplformat}[2]{%
9166 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
9167 \ifglsxtrinsertinside\else##2\fi
9168 }%
9169 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9170 \newabbreviationstyle{long-em-noshort-em-noreg}%
9171 {%
9172 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9173 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9174 \glshasattribute{\the\glslabeltok}{regular}%
9175 {%
9176 \glssetattribute{\the\glslabeltok}{regular}{false}%
9177 }%
9178 }%

```

```

9179 }%
9180 }%
9181 {%
9182   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9183 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9184 \newabbreviationstyle{long-noshort-em-desc}%
9185 {%
9186   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9187 }%
9188 {%
9189   \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9190   \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
9191   \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
9192   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9193   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9194   \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9195     \glslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9196     \ifglsxtrininsertinside \else##2\fi
9197   }%
9198   \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9199     \glslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9200     \ifglsxtrininsertinside \else##2\fi
9201   }%
9202   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9203     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9204     \ifglsxtrininsertinside \else##2\fi
9205   }%
9206   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9207     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9208     \ifglsxtrininsertinside \else##2\fi
9209   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9210   \renewcommand*{\glsxtrinlinefullformat}[2]{%
9211     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9212     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9213     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshort{##1}}}%
9214   }%
9215   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9216     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9217     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9218     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshortpl{##1}}}%
9219   }%
9220   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9221     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%

```

```

9222     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9223     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9224 }%
9225 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9226     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9227     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9228     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9229 }%

```

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.

```

9230 \renewcommand*{\glxtrfullformat}[2]{%
9231     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9232     \ifglxtrinsertinside\else##2\fi
9233 }%
9234 \renewcommand*{\glxtrfullplformat}[2]{%
9235     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9236     \ifglxtrinsertinside\else##2\fi
9237 }%
9238 \renewcommand*{\Glsxtrfullformat}[2]{%
9239     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9240     \ifglxtrinsertinside\else##2\fi
9241 }%
9242 \renewcommand*{\Glsxtrfullplformat}[2]{%
9243     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9244     \ifglxtrinsertinside\else##2\fi
9245 }%
9246 }

```

long-desc-em Backward compatibility:

```

9247 \@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.

```

9248 \newabbreviationstyle{long-em-noshort-em-desc}%
9249 {%
9250 \renewcommand*{\CustomAbbreviationFields}{%
9251     name={\glxtrlongnoshortdescname},
9252     sort={\the\glslongtok},
9253     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9254     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9255     text={\glslongemfont{\the\glslongtok}},
9256     plural={\glslongemfont{\the\glslongpltok}}}%
9257 }%
9258 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9259     \glsssetattribute{\the\glslabeltok}{regular}{true}}%
9260 }%
9261 {%
9262 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%

```

```

9263 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
9264 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
9265 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongemfont{##1}}%
9266 \renewcommand*\glslongfont[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9267 \renewcommand*\glsxtrsubsequentfmt[2]{%
9268   \glslongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9269   \ifglsxtrininsertinside \else##2\fi
9270 }%
9271 \renewcommand*\glsxtrsubsequentplfmt[2]{%
9272   \glslongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9273   \ifglsxtrininsertinside \else##2\fi
9274 }%
9275 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9276   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9277   \ifglsxtrininsertinside \else##2\fi
9278 }%
9279 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9280   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9281   \ifglsxtrininsertinside \else##2\fi
9282 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9283 \renewcommand*\glsxtrinlinefullformat[2]{%
9284   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9285   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9286   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9287 }%
9288 \renewcommand*\glsxtrinlinefullplformat[2]{%
9289   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9290   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9291   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9292 }%
9293 \renewcommand*\Glsxtrinlinefullformat[2]{%
9294   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9295   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9296   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9297 }%
9298 \renewcommand*\Glsxtrinlinefullplformat[2]{%
9299   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9300   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9301   \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9302 }%

```

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.

```

9303 \renewcommand*\glsxtrfullformat[2]{%
9304   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9305   \ifglsxtrininsertinside\else##2\fi
9306 }%

```

```

9307 \renewcommand*{\glstrfullplformat}[2]{%
9308   \glstrfirstlongemfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
9309   \ifglstrinsertinside\else##2\fi
9310 }%
9311 \renewcommand*{\GlsXtrfullformat}[2]{%
9312   \glstrfirstlongemfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
9313   \ifglstrinsertinside\else##2\fi
9314 }%
9315 \renewcommand*{\GlsXtrfullplformat}[2]{%
9316   \glstrfirstlongemfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
9317   \ifglstrinsertinside\else##2\fi
9318 }%
9319 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9320 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9321 {%
9322   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
   Unset the regular attribute if it has been set.
9323   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9324     \glshasattribute{\the\glslabeltok}{regular}%
9325     {%
9326       \glissetattribute{\the\glslabeltok}{regular}{false}%
9327     }%
9328   }%
9329 }%
9330 }%
9331 {%
9332   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9333 }

```

ort-em-footnote

```

9334 \newabbreviationstyle{short-em-footnote}%
9335 {%
9336   \renewcommand*{\CustomAbbreviationFields}{%
9337     name={\glstrfootnotename},
9338     sort={\the\glsshorttok},
9339     description={\the\glslongtok},%
9340     first={\protect\glstrfirstabbrvemfont{\the\glsshorttok}%
9341       \protect\glstrabbrvfootnote{\the\glslabeltok}%
9342       {\protect\glstrfirstlongfootnotefont{\the\glslongtok}}},%
9343     firstplural={\protect\glstrfirstabbrvemfont{\the\glsshortpltok}%
9344       \protect\glstrabbrvfootnote{\the\glslabeltok}%
9345       {\protect\glstrfirstlongfootnotefont{\the\glslongpltok}}},%
9346     plural={\protect\glstrabbrvemfont{\the\glsshortpltok}}}%
   Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute
   if it has been set.
9347   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

9348 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9349 \glsattribute{\the\glslabeltok}{regular}%
9350 {%
9351 \glssetattribute{\the\glslabeltok}{regular}{false}%
9352 }%
9353 {}%
9354 }%
9355 }%
9356 {%
9357 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
9358 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9359 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9360 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9361 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9362 \renewcommand*{\glsxtrfullformat}[2]{%
9363 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9364 \ifglsxtrininsertinside\else##2\fi
9365 \protect\glsxtrabbrvfootnote{##1}%
9366 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9367 }%
9368 \renewcommand*{\glsxtrfullplformat}[2]{%
9369 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9370 \ifglsxtrininsertinside\else##2\fi
9371 \protect\glsxtrabbrvfootnote{##1}%
9372 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9373 }%
9374 \renewcommand*{\Glsxtrfullformat}[2]{%
9375 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9376 \ifglsxtrininsertinside\else##2\fi
9377 \protect\glsxtrabbrvfootnote{##1}%
9378 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9379 }%
9380 \renewcommand*{\Glsxtrfullplformat}[2]{%
9381 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9382 \ifglsxtrininsertinside\else##2\fi
9383 \protect\glsxtrabbrvfootnote{##1}%
9384 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9385 }%

```

The first use full form and the inline full form use the short (long) style.

```

9386 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9387 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9388 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9389 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9390 }%
9391 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9392 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9393 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```



```

9394 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
9395 }%
9396 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9397 \glfirstabbrvemfont{\Glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
9398 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
9399 \glstrparen{\glfirstlongfootnotefont{\glaccesslong{##1}}}%
9400 }%
9401 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9402 \glfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglstrinsertinside##2\fi}%
9403 \ifglstrinsertinside\else##2\fi\glstrfullsep{##1}%
9404 \glstrparen{\glfirstlongfootnotefont{\glaccesslongpl{##1}}}%
9405 }%
9406 }

```

footnote-em Backward compatibility:

```

9407 \@glstr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9408 \newabbreviationstyle{short-em-postfootnote}%
9409 {%
9410 \renewcommand*{\CustomAbbreviationFields}{%
9411 name={\glstrfootnotename},
9412 sort={\the\glsshorttok},
9413 description={\the\glslongtok},%
9414 first={\protect\glfirstabbrvemfont{\the\glsshorttok}},%
9415 firstplural={\protect\glfirstabbrvemfont{\the\glsshortpltok}},%
9416 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

9417 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9418 \csdef{glstrpostlink\glscategorylabel}{%
9419 \glstrifwasfirstuse
9420 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9421 \glstrdopostpunc{\protect\glstrabbrvfootnote{\glslabel}%
9422 {\glfirstlongfootnotefont{\glentrylong{\glslabel}}}}%
9423 }%
9424 {}%
9425 }%
9426 \glshasattribute{\the\glslabeltok}{regular}%
9427 {%
9428 \glissetattribute{\the\glslabeltok}{regular}{false}%
9429 }%
9430 {}%
9431 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9432 \renewcommand*\glxtrsetupfulldefs}{%
9433 \let\glxtrifwasfirstuse\@secondoftwo
9434 }%
9435 }%
9436 {%
9437 \renewcommand*\abbrvpluralsuffix{\protect\glxxtremsuffix}%
9438 \renewcommand*\glxabbrvfont[1]{\glxabbrvemfont{##1}}%
9439 \renewcommand*\glxfirstabbrvfont[1]{\glxfirstabbrvemfont{##1}}%
9440 \renewcommand*\glxfirstlongfont[1]{\glxfirstlongfootnotefont{##1}}%
9441 \renewcommand*\glxlongfont[1]{\glxlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9442 \renewcommand*\glxtrfullformat}[2]{%
9443 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9444 \ifglxtrininsertinside\else##2\fi
9445 }%
9446 \renewcommand*\glxtrfullplformat}[2]{%
9447 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9448 \ifglxtrininsertinside\else##2\fi
9449 }%
9450 \renewcommand*\Glxtrfullformat}[2]{%
9451 \glxfirstabbrvemfont{\Glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9452 \ifglxtrininsertinside\else##2\fi
9453 }%
9454 \renewcommand*\Glxtrfullplformat}[2]{%
9455 \glxfirstabbrvemfont{\Glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9456 \ifglxtrininsertinside\else##2\fi
9457 }%

```

The first use full form and the inline full form use the short (long) style.

```

9458 \renewcommand*\glxtrininlinefullformat}[2]{%
9459 \glxfirstabbrvemfont{\glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9460 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9461 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslong{##1}}}%
9462 }%
9463 \renewcommand*\glxtrininlinefullplformat}[2]{%
9464 \glxfirstabbrvemfont{\glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9465 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9466 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslongpl{##1}}}%
9467 }%
9468 \renewcommand*\Glxtrininlinefullformat}[2]{%
9469 \glxfirstabbrvemfont{\Glxaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9470 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9471 \glxtrparen{\glxfirstlongfootnotefont{\glxaccesslong{##1}}}%
9472 }%
9473 \renewcommand*\Glxtrininlinefullplformat}[2]{%
9474 \glxfirstabbrvemfont{\Glxaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9475 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

9476 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9477 }%
9478 }

```

postfootnote-em Backward compatibility:

```

9479 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

1.7.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glsxtruserfield Default is the useri field.

```

9480 \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.

```

9481 \ifdef\glscurrentfieldvalue
9482 {
9483   \newcommand*{\glsxtruserparen}[2]{%
9484     \glsxtrfullsep{#2}%
9485     \glsxtrparen
9486       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9487   }
9488 }
9489 {
9490   \newcommand*{\glsxtruserparen}[2]{%
9491     \glsxtrfullsep{#2}%
9492     \glsxtrparen
9493       {#1\ifglshasfield{\glsxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9494   }
9495 }

```

Font used for short form:

lsabbrvuserfont

```

9496 \newcommand*{\glsabbrvuserfont}[1]{\glsabbrvdefaultfont{#1}}

```

Font used for short form on first use:

stabbrvuserfont

```

9497 \newcommand*{\glsfirstabbrvuserfont}[1]{\glsabbrvuserfont{#1}}

```

Font used for long form:

glslonguserfont

```

9498 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

```

Font used for long form on first use:

rstlonguserfont

```
9499 \newcommand*{\glsfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
9500 \newcommand*{\glsxtrusersuffix}{\glsxtrabbrvpluralsuffix}
```

long-short-user

```
9501 \newabbreviationstyle{long-short-user}%
9502 {%
9503   \renewcommand*{\CustomAbbreviationFields}{%
9504     name={\glsxtrlongshortname},
9505     sort={\the\glsshorttok},
9506     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9507       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9508     {\the\glslabeltok}},%
9509     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9510       \protect\glsxtruserparen
9511         {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9512     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9513     description={\protect\glslonguserfont{\the\glslongtok}}}%
9514   }
```

Unset the regular attribute if it has been set.

```
9514   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9515     \glshasattribute{\the\glslabeltok}{regular}%
9516     {%
9517       \glssetattribute{\the\glslabeltok}{regular}{false}%
9518     }%
9519   }%
9520 }%
9521 }%
9522 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9523   \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9524   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9525   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9526   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9527   \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9528   \renewcommand*{\glsxtrfullformat}[2]{%
9529     \glsfirstlonguserfont{\glsaccesslong{##1}}\ifglxtrinsertinside##2\fi}%
9530   \ifglxtrinsertinside\else##2\fi
9531   \glsxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9532 }%
9533   \renewcommand*{\glsxtrfullplformat}[2]{%
9534     \glsfirstlonguserfont{\glsaccesslongpl{##1}}\ifglxtrinsertinside##2\fi}%
9535   \ifglxtrinsertinside\else##2\fi
```

```

9536 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9537 }%
9538 \renewcommand*{\Glsxtrfullformat}[2]{%
9539 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9540 \ifglxtrinsertinside\else##2\fi
9541 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9542 }%
9543 \renewcommand*{\Glsxtrfullplformat}[2]{%
9544 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9545 \ifglxtrinsertinside\else##2\fi
9546 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9547 }%
9548 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9549 \newabbreviationstyle{long-postshort-user}%
9550 {%
9551 \renewcommand*{\CustomAbbreviationFields}{%
9552 name={\glxtrlongshortname},
9553 sort={\the\glsshorttok},
9554 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9555 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9556 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9557 description={\protect\glslonguserfont{\the\glslongtok}}}%
9558 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9559 \csdef{glxtrpostlink\glscategorylabel}{%
9560 \glxtrifwasfirstuse
9561 {%
9562 \glxtruserparen
9563 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9564 {\glslabel}%
9565 }%
9566 }%
9567 }%
9568 \glshasattribute{\the\glslabeltok}{regular}%
9569 {%
9570 \glssetattribute{\the\glslabeltok}{regular}{false}%
9571 }%
9572 }%
9573 }%
9574 }%
9575 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9576 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9577 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9578 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9579 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9580 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9581 \renewcommand*{\glxtrfullformat}[2]{%
9582   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9583   \ifglxtrinsertinside\else##2\fi
9584 }%
9585 \renewcommand*{\glxtrfullplformat}[2]{%
9586   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9587   \ifglxtrinsertinside\else##2\fi
9588 }%
9589 \renewcommand*{\Glsxtrfullformat}[2]{%
9590   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9591   \ifglxtrinsertinside\else##2\fi
9592 }%
9593 \renewcommand*{\Glsxtrfullplformat}[2]{%
9594   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9595   \ifglxtrinsertinside\else##2\fi
9596 }%

```

In-line format:

```

9597 \renewcommand*{\glxtrinlinefullformat}[2]{%
9598   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9599   \ifglxtrinsertinside\else##2\fi
9600   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9601 }%
9602 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9603   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9604   \ifglxtrinsertinside\else##2\fi
9605   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9606 }%
9607 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9608   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9609   \ifglxtrinsertinside\else##2\fi
9610   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9611 }%
9612 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9613   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9614   \ifglxtrinsertinside\else##2\fi
9615   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9616 }%
9617 }

```

ortuserdesname

```

9618 \newcommand*{\glxtrlongshortuserdesname}{%
9619   \protect\glslonguserfont{\the\glslongtok}%
9620   \protect\glxtruserparen
9621   {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
9622 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9623 \newabbreviationstyle{long-postshort-user-desc}%
9624 {%
9625   \renewcommand*{\CustomAbbreviationFields}{%
9626     name={\glxtrlongshortuserdescname},
9627     sort={\the\glslongtok},
9628     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9629     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9630     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9631     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9632   }%
9633   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9634     \csdef{glxtrpostlink\glscategorylabel}{%
9635       \glxtrifwasfirstuse
9636       {%
9637         \glxtruserparen
9638         {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
9639         {\glslabel}%
9640       }%
9641     }%
9642   }%
9643   \glshasattribute{\the\glslabeltok}{regular}%
9644   {%
9645     \glssetattribute{\the\glslabeltok}{regular}{false}%
9646   }%
9647   {}%
9648 }%
9649 }%
9650 {%
9651   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9652 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9653 \newabbreviationstyle{short-postlong-user}%
9654 {%
9655   \renewcommand*{\CustomAbbreviationFields}{%
9656     name={\glxtrshortlongname},
9657     sort={\the\glsshorttok},
9658     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9659     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9660     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9661     description={\protect\glslonguserfont{\the\glslongtok}}%
9662   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9663     \csdef{glxtrpostlink\glscategorylabel}{%
9664       \glxtrifwasfirstuse
9665       {%
9666         \glxtruserparen
9667         {\glsfirstlonguserfont{\glstrylong{\glslabel}}}%
9668         {\glslabel}%

```

```

9669     }%
9670     {}%
9671     }%
9672     \glshasattribute{\the\glslabeltok}{regular}%
9673     {%
9674         \glissetattribute{\the\glslabeltok}{regular}{false}%
9675     }%
9676     {}%
9677 }%
9678 }%
9679 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9680 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9681 \renewcommand*{\glsabrvfont}[1]{\glsabrvuserfont{##1}}%
9682 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9683 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9684 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9685 \renewcommand*{\glsxtrfullformat}[2]{%
9686     \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9687     \ifglsxtrininsertinside\else##2\fi
9688 }%
9689 \renewcommand*{\glsxtrfullplformat}[2]{%
9690     \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9691     \ifglsxtrininsertinside\else##2\fi
9692 }%
9693 \renewcommand*{\Glsxtrfullformat}[2]{%
9694     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9695     \ifglsxtrininsertinside\else##2\fi
9696 }%
9697 \renewcommand*{\Glsxtrfullplformat}[2]{%
9698     \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9699     \ifglsxtrininsertinside\else##2\fi
9700 }%

```

In-line format:

```

9701 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9702     \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9703     \ifglsxtrininsertinside\else##2\fi
9704     \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9705 }%
9706 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9707     \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9708     \ifglsxtrininsertinside\else##2\fi
9709     \glsxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9710 }%
9711 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9712     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```



```

9713 \ifglxtrinsertinside\else##2\fi
9714 \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9715 }%
9716 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9717 \glsfirstabbruserfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9718 \ifglxtrinsertinside\else##2\fi
9719 \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9720 }%
9721 }

```

onguserdescname

```

9722 \newcommand*{\glxtrshortlonguserdescname}{%
9723 \protect\glssabbruserfont{\the\glssshorttok}%
9724 \protect\glxtruserparen
9725 {\protect\glslonguserfont{\the\glslongpltok}}}%
9726 {\the\glslabeltok}%
9727 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

9728 \newabbreviationstyle{short-postlong-user-desc}%
9729 {%
9730 \renewcommand*{\CustomAbbreviationFields}{%
9731 name={\glxtrshortlonguserdescname},
9732 sort={\the\glssshorttok},
9733 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9734 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9735 text={\protect\glssabbruserfont{\the\glssshorttok}},%
9736 plural={\protect\glssabbruserfont{\the\glssshortpltok}}}%
9737 }%
9738 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9739 \csdef{glxtrpostlink\glscategorylabel}{%
9740 \glxtrifwasfirstuse
9741 {%
9742 \glxtruserparen
9743 {\glsfirstlonguserfont{\glssentrylong{\glslabel}}}%
9744 {\glslabel}%
9745 }%
9746 {}%
9747 }%
9748 \glshasattribute{\the\glslabeltok}{regular}%
9749 {%
9750 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9751 }%
9752 {}%
9753 }%
9754 }%
9755 {%
9756 \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9757 }

```

short-user-desc

```
9758 \newabbreviationstyle{long-short-user-desc}%
9759 {%
9760   \renewcommand*{\CustomAbbreviationFields}{%
9761     name={\glxtrlongshortuserdescname},
9762     sort={\glxtrlongshortdescsort},%

9763     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9764       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9765     {\the\glslabeltok}},%
9766     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9767       \protect\glxtruserparen
9768       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9769     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9770     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9771   }%

   Unset the regular attribute if it has been set.
9772   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9773     \glshasattribute{\the\glslabeltok}{regular}%
9774     {%
9775       \glissetattribute{\the\glslabeltok}{regular}{false}%
9776       }%
9777     {}}%
9778   }%
9779 }%
9780 {%
9781   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9782 }
```

short-long-user

```
9783 \newabbreviationstyle{short-long-user}%
9784 {%

   \glslonguserfont is used in the description since \glsdesc doesn't set the style.
9785   \renewcommand*{\CustomAbbreviationFields}{%
9786     name={\glxtrshortlongname},
9787     sort={\the\glsshorttok},
9788     description={\protect\glslonguserfont{\the\glslongtok}},%
9789     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9790       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9791     {\the\glslabeltok}},%
9792     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9793       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9794     {\the\glslabeltok}},%

9795     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

   Unset the regular attribute if it has been set.
9796   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```

9797 \glshasattribute{\the\glslabeltok}{regular}%
9798 {%
9799 \glissetattribute{\the\glslabeltok}{regular}{false}%
9800 }%
9801 {}%
9802 }%
9803 }%
9804 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9805 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9806 \renewcommand*{\glssabrvfont}[1]{\glssabrvuserfont{##1}}%
9807 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvuserfont{##1}}%
9808 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonguserfont{##1}}%
9809 \renewcommand*{\glsslongfont}[1]{\glsslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9810 \renewcommand*{\glsxtrfullformat}[2]{%
9811 \glssfirstabrvuserfont{\glssaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9812 \ifglsxtrinsertinside\else##2\fi
9813 \glsxtruserparen{\glssfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9814 }%
9815 \renewcommand*{\glsxtrfullplformat}[2]{%
9816 \glssfirstabrvuserfont{\glssaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9817 \ifglsxtrinsertinside\else##2\fi
9818 \glsxtruserparen{\glssfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9819 }%
9820 \renewcommand*{\Glsxtrfullformat}[2]{%
9821 \glssfirstabrvuserfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9822 \ifglsxtrinsertinside\else##2\fi
9823 \glsxtruserparen{\glssfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9824 }%
9825 \renewcommand*{\Glsxtrfullplformat}[2]{%
9826 \glssfirstabrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9827 \ifglsxtrinsertinside\else##2\fi
9828 \glsxtruserparen{\glssfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9829 }%
9830 }

```

-long-user-desc

```

9831 \newabbreviationstyle{short-long-user-desc}%
9832 {%
9833 \renewcommand*{\CustomAbbreviationFields}{%
9834 name={\glsxtrshortlonguserdescname},
9835 sort={\glsxtrshortlongdescsort},%
9836 first={\protect\glssfirstabrvuserfont{\the\glssshorttok}}%
9837 \protect\glsxtruserparen{\protect\glssfirstlonguserfont{\the\glsslongtok}}%
9838 {\the\glslabeltok}},%
9839 firstplural={\protect\glssfirstabrvuserfont{\the\glssshortpltok}}%

```

```

9840 \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9841 {\the\glslabeltok}},%
9842 text={\protect\glsabbrvfont{\the\glsshorttok}},%
9843 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9844 }%

```

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 \GlsXtrUseAbbrStyleFmts{short-long-user}%
9855 }

```

1.7.7 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.

```

9856 \newrobustcmd*{\glxtrifhyphenstart}[3]{%
9857 \ifx\glsinsert#1\relax
9858 \expandafter@\glxtrifhyphenstart#1\relax\relax
9859 \@end@glxtrifhyphenstart{#2}{#3}%
9860 \else
9861 \@glxtrifhyphenstart#1\relax\relax\@end@glxtrifhyphenstart{#2}{#3}%
9862 \fi
9863 }

```

`trifhyphenstart`

```

9864 \def\@glxtrifhyphenstart#1#2\@end@glxtrifhyphenstart#3#4{%
9865 \ifx-#1\relax#3\else #4\fi
9866 }

```

`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.

9867 \newcommand*{\glxtrlonghyphenshort}[4]{%

Grouping is needed to localise the redefinitions.

9868 {%

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.

9869 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}}%

9870 \glsfirslonghyphenfont{#2\ifglxtrininsertinside{#4}\fi}%

9871 \ifglxtrininsertinside\else{#4}\fi

9872 \glxtrfullsep{#1}%

9873 \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrininsertinside{#4}\fi}%

9874 \ifglxtrininsertinside\else{#4}\fi}%

9875 }%

9876 }

abbrvhyphenfont

9877 \newcommand*{\glsabrvhyphenfont}{\glsabrvdefaultfont}%

abbrvhyphenfont

9878 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabrvhyphenfont}%

slonghyphenfont

9879 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%

tlonghyphenfont

9880 \newcommand*{\glsfirslonghyphenfont}{\glslonghyphenfont}%

The default short form suffix:

xtrhyphensuffix

9881 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}

en-short-hyphen Designed for use with the markwords attribute.

9882 \newabbreviationstyle{long-hyphen-short-hyphen}%

9883 {%

9884 \renewcommand*{\CustomAbbreviationFields}{%

9885 name={\glxtrlongshortname},

9886 sort={\the\glsshorttok},

9887 first={\protect\glsfirslonghyphenfont{\the\glslongtok}}%

9888 \protect\glxtrfullsep{\the\glslabeltok}}%

9889 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%

9890 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}}%

9891 \protect\glxtrfullsep{\the\glslabeltok}}%

9892 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%

9893 plural={\protect\glsabrvhyphenfont{\the\glsshortpltok}}},%

9894 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

Unset the regular attribute if it has been set.

```

9895 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9896   \glshasattribute{\the\glslabeltok}{regular}%
9897   {%
9898     \glissetattribute{\the\glslabeltok}{regular}{false}%
9899   }%
9900   {}%
9901 }%
9902}%
9903{%
9904 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9905 \renewcommand*{\glssabrvfont}[1]{\glssabrvhyphenfont{##1}}%
9906 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvhyphenfont{##1}}%
9907 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonghyphenfont{##1}}%
9908 \renewcommand*{\glsslongfont}[1]{\glsslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9909 \renewcommand*{\glsxtrfullformat}[2]{%
9910   \glsxtrlonghyphenshort{##1}{\glssaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9911 }%
9912 \renewcommand*{\glsxtrfullplformat}[2]{%
9913   \glsxtrlonghyphenshort{##1}{\glssaccesslongpl{##1}}%
9914   {\glssaccessshortpl{##1}}{##2}%
9915 }%
9916 \renewcommand*{\Glsxtrfullformat}[2]{%
9917   \glsxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9918 }%
9919 \renewcommand*{\Glsxtrfullplformat}[2]{%
9920   \glsxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9921   {\glssaccessshortpl{##1}}{##2}%
9922 }%
9923}

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9924 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9925 {%
9926   \renewcommand*{\CustomAbbreviationFields}{%
9927     name={\glsxtrlongshortdescname},
9928     sort={\glsxtrlongshortdescsort},
9929     first={\protect\glssfirstlonghyphenfont{\the\glsslongtok}}%
9930     \protect\glsxtrfullsep{\the\glslabeltok}%
9931     \glsxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshorttok}}},%
9932     firstplural={\protect\glssfirstlonghyphenfont{\the\glsslongpltok}}%
9933     \protect\glsxtrfullsep{\the\glslabeltok}%
9934     \glsxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshortpltok}}},%
9935     text={\protect\glssabrvhyphenfont{\the\glssshorttok}},%
9936     plural={\protect\glssabrvhyphenfont{\the\glssshortpltok}}}%
9937 }%

```

Unset the regular attribute if it has been set.

```

9938 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9939   \glshasattribute{\the\glslabeltok}{regular}%
9940   {%
9941     \glissetattribute{\the\glslabeltok}{regular}{false}%
9942   }%
9943   {}%
9944 }%
9945 }%
9946 {%
9947   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9948 }

```

onghyphennoshort

```
\glsxtrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```

9949 \newcommand*{\glsxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```
9950 {%
```

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.

```

9951 \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9952 \glsfirstlonghyphenfont{#2\ifglsxtrininsertinside{#3}\fi}%
9953 \ifglsxtrininsertinside\else{#3}\fi
9954 }%
9955 }

```

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.

```

9956 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
9957 {%
9958   \renewcommand*{\CustomAbbreviationFields}{%
9959     name={\glsxtrlongnoshortdescname},
9960     sort={\expandonce\glxtrorglong},
9961     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9962     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9963     plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
9964   }%

```

Unset the regular attribute if it has been set.

```

9965 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9966   \glshasattribute{\the\glslabeltok}{regular}%
9967   {%
9968     \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9969 }%
9970 {}%
9971 }%
9972 }%
9973 {%
9974 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9975 \renewcommand*\abbrvpluralsuffix{\glsxtrabbrvpluralsuffix}%
9976 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
9977 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
9978 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
9979 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9980 \renewcommand*\glxtrsubsequentfmt[2]{%
9981   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9982 }%
9983 \renewcommand*\glxtrsubsequentplfmt[2]{%
9984   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9985 }%
9986 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9987   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9988 }%
9989 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9990   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9991 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9992 \renewcommand*\glxtrinlinefullformat[2]{%
9993   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9994   \glxtrfullsep{##1}%
9995   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9996 }%
9997 \renewcommand*\glxtrinlinefullplformat[2]{%
9998   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9999   \glxtrfullsep{##1}%
10000   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
10001 }%
10002 \renewcommand*\Glsxtrinlinefullformat[2]{%
10003   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10004   \glxtrfullsep{##1}%
10005   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
10006 }%
10007 \renewcommand*\Glsxtrinlinefullplformat[2]{%
10008   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10009   \glxtrfullsep{##1}%
10010   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
10011 }%

```

The first use full form only displays the long form.


```

10012 \renewcommand*{\glxtrfullformat}[2]{%
10013   \glxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
10014 }%
10015 \renewcommand*{\glxtrfullplformat}[2]{%
10016   \glxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
10017 }%
10018 \renewcommand*{\Glsxtrfullformat}[2]{%
10019   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10020 }%
10021 \renewcommand*{\Glsxtrfullplformat}[2]{%
10022   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10023 }%
10024 }

```

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.

```

10025 \newabbreviationstyle{long-hyphen-noshort-noreg}%
10026 {%
10027   \renewcommand*{\CustomAbbreviationFields}{%
10028     name={\glxtrlongnoshortname},
10029     sort={\the\glsshorttok},
10030     first={\protect\glxtrfirstlonghyphenfont{\the\glslongtok}},%
10031     firstplural={\protect\glxtrfirstlonghyphenfont{\the\glslongpltok}},%
10032     text={\protect\glxtrlonghyphenfont{\the\glslongtok}},%
10033     plural={\protect\glxtrlonghyphenfont{\the\glslongpltok}},%
10034     description={\the\glslongtok}%
10035   }%

```

Unset the regular attribute if it has been set.

```

10036 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10037   \glshasattribute{\the\glslabeltok}{regular}%
10038   {%
10039     \glissetattribute{\the\glslabeltok}{regular}{false}%
10040   }%
10041   {}}%
10042 }%
10043 }%
10044 {%
10045   \GlsXtrUseAbbrStyleFmts{long-desc}%
10046 }

```

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.

```

10047 \newcommand*{\glxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10048 {%
10049   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}}%
10050   \glsfirslonghyphenfont{#1}%
10051 }%
10052 }

```

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.

```

10053 \newcommand*{\glxtrposthyphenshort}[2]{%
10054   {%
10055     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}}%
10056     \ifglxtrininsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
10057     \glxtrfullsep{#1}%
10058     \glxtrparen
10059     {\glsfirstabbrvhyphenfont{\glstryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
10060     \ifglxtrininsertinside\else{#2}\fi
10061   }%
10062 }%
10063 }

```

hyphensubsequent `\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

10064 \newcommand*{\glxtrposthyphensubsequent}[2]{%
10065   \glssabrvfont{\ifglxtrininsertinside {#2}\fi}%
10066   \ifglxtrininsertinside \else{#2}\fi
10067 }

```

postshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10068 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
10069 {%
10070   \renewcommand*{\CustomAbbreviationFields}{%
10071     name={\glxtrlongshortname},
10072     sort={\the\glsshorttok},
10073     first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
10074     firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
10075     plural={\protect\glssabrvhyphenfont{\the\glsshortpltok}},%
10076     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10077   \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

10078 \csdef{glxstrpostlink\glscategorylabel}{%
10079 \glxstrifwasfirstuse
10080 {%
10081 \glxstrposthyphenshort{\glslabel}{\glsininsert}%
10082 }%
10083 {%

```

Put the insertion into the post-link:

```

10084 \glxstrposthyphensubsequent{\glslabel}{\glsininsert}%
10085 }%
10086 }%
10087 \glshasattribute{\the\glslabeltok}{regular}%
10088 {%
10089 \glissetattribute{\the\glslabeltok}{regular}{false}%
10090 }%
10091 {}%
10092 }%
10093 }%
10094 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10095 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10096 \renewcommand*{\glssabrvfont}[1]{\glssabrvhyphenfont{##1}}%
10097 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvhyphenfont{##1}}%
10098 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonghyphenfont{##1}}%
10099 \renewcommand*{\glsslongfont}[1]{\glsslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10100 \renewcommand*{\glxtrs subsequentfmt}[2]{%
10101 \glssabrvfont{\glssaccessshort{##1}}%
10102 }%
10103 \renewcommand*{\glxtrs subsequentplfmt}[2]{%
10104 \glssabrvfont{\glssaccessshortpl{##1}}%
10105 }%
10106 \renewcommand*{\Glsxtrs subsequentfmt}[2]{%
10107 \glssabrvfont{\Glsaccessshort{##1}}%
10108 }%
10109 \renewcommand*{\Glsxtrs subsequentplfmt}[2]{%
10110 \glssabrvfont{\Glsaccessshortpl{##1}}%
10111 }%

```

First use full form:

```

10112 \renewcommand*{\glxtrfullformat}[2]{%
10113 \glxtrlonghyphen{\glssaccesslong{##1}}{##1}{##2}%
10114 }%
10115 \renewcommand*{\glxtrfullplformat}[2]{%
10116 \glxtrlonghyphen{\glssaccesslongpl{##1}}{##1}{##2}%
10117 }%
10118 \renewcommand*{\Glsxtrfullformat}[2]{%
10119 \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
10120 }%

```

```

10121 \renewcommand*{\Glsxtrfullplformat}[2]{%
10122   \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
10123 }%

```

In-line format.

```

10124 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10125   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
10126     \ifglsxtrininsertinside{##2}\fi}%
10127   \ifglsxtrininsertinside \else{##2}\fi
10128 }%
10129 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10130   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10131     \ifglsxtrininsertinside{##2}\fi}%
10132   \ifglsxtrininsertinside \else{##2}\fi
10133 }%
10134 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10135   \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
10136     \ifglsxtrininsertinside{##2}\fi}%
10137   \ifglsxtrininsertinside \else{##2}\fi
10138 }%
10139 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10140   \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10141     \ifglsxtrininsertinside{##2}\fi}%
10142   \ifglsxtrininsertinside \else{##2}\fi
10143 }%
10144 }

```

ort-hyphen-desc Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10145 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10146 {%
10147   \renewcommand*{\CustomAbbreviationFields}{%
10148     name={\glsxtrlongshortdescname},
10149     sort={\glsxtrlongshortdescsort},%
10150     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10151     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10152     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10153     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10154 }%
10155 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10156   \csdef{glsxtrpostlink\glscategorylabel}{%
10157     \glsxtrifwasfirstuse
10158     {%
10159       \glsxtrposthyphenshort{\glslabel}{\glsinsert}%
10160     }%
10161     {%

```

Put the insertion into the post-link:

```

10162       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10163     }%
10164   }%

```

```

10165 \glshasattribute{\the\glslabeltok}{regular}%
10166 {%
10167 \glissetattribute{\the\glslabeltok}{regular}{false}%
10168 }%
10169 {}%
10170 }%
10171 }%
10172 {%
10173 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10174 }

```

rsorthyphenlong

```
\glsxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
10175 \newcommand*{\glsxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10176 {%
```

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.)

```

10177 \glsxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{}%
10178 \glsfirstabbrvhyphenfont{#2\ifglsxtrininsertinside{#4}\fi}%
10179 \ifglsxtrininsertinside\else{#4}\fi
10180 \glsxtrfullsep{#1}%
10181 \glsxtrparen{\glsfirstlonghyphenfont{#3\ifglsxtrininsertinside{#4}\fi}%
10182 \ifglsxtrininsertinside\else{#4}\fi}%
10183 }%
10184 }

```

hen-long-hyphen Designed for use with the keywords attribute.

```

10185 \newabbreviationstyle{short-hyphen-long-hyphen}%
10186 {%
10187 \renewcommand*{\CustomAbbreviationFields}{%
10188 name={\glsxtrshortlongname},
10189 sort={\the\glsshorttok},
10190 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10191 \protect\glsxtrfullsep{\the\glslabeltok}%
10192 \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10193 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10194 \protect\glsxtrfullsep{\the\glslabeltok}%
10195 \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10196 plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}}},%
10197 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10198 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10199   \glshasattribute{\the\glslabeltok}{regular}%
10200   {%
10201     \glissetattribute{\the\glslabeltok}{regular}{false}%
10202   }%
10203   {}%
10204 }%
10205 }%
10206 {%
10207 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
10208 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10209 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10210 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10211 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10212 \renewcommand*{\glsxtrfullformat}[2]{%
10213   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10214 }%
10215 \renewcommand*{\glsxtrfullplformat}[2]{%
10216   \glsxtrshorthyphenlong{##1}%
10217   {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10218 }%
10219 \renewcommand*{\Glsxtrfullformat}[2]{%
10220   \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10221 }%
10222 \renewcommand*{\Glsxtrfullplformat}[2]{%
10223   \glsxtrshorthyphenlong{##1}%
10224   {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10225 }%
10226 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10227 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10228 {%
10229   \renewcommand*{\CustomAbbreviationFields}{%
10230     name={\glsxtrshortlongdescname},
10231     sort={\glsxtrshortlongdescsort},
10232     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
10233     \protect\glsxtrfullsep{\the\glslabeltok}%
10234     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10235     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
10236     \protect\glsxtrfullsep{\the\glslabeltok}%
10237     \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10238     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10239     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10240 }%

```

Unset the regular attribute if it has been set.

```

10241 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10242   \glshasattribute{\the\glslabeltok}{regular}}%
10243   {%
10244     \glissetattribute{\the\glslabeltok}{regular}{false}}%
10245   }%
10246   {}%
10247 }%
10248 }%
10249 {%
10250   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10251 }

```

`\glstrshorthyphen` `{\langle short \rangle}{\langle label \rangle}{\langle insert \rangle}`

Used by short-hyphen-postlong-hyphen. The `\langle insert \rangle` is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

10252 \newcommand*{\glstrshorthyphen}[3]{%
    Grouping is needed to localise the redefinitions.
10253   {%
10254     \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}}%
10255     \glsfirstabbrvhyphenfont{#1}%
10256   }%
10257 }

```

`\glstrposthyphenlong` `{\langle label \rangle}{\langle insert \rangle}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glstrshorthyphenlong` but omits the `\langle short \rangle` part. This always uses the singular long form.

```

10258 \newcommand*{\glstrposthyphenlong}[2]{%
10259   {%
10260     \glstrifhyphenstart{#2}{\def\glstrwordsep{-}}}%
10261     \ifglstrinsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10262     \glstrfullsep{#1}%
10263     \glstrparen
10264     {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglstrinsertinside{#2}\fi}%
10265     \ifglstrinsertinside\else{#2}\fi
10266   }%
10267 }%
10268 }

```

`postlong-hyphen` Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10269 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10270 {%
10271   \renewcommand*{\CustomAbbreviationFields}{%
10272     name={\glxtrshortlongname},
10273     sort={\the\glsshorttok},
10274     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10275     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10276     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10277     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10278 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10279   \csdef{glxtrpostlink\glscategorylabel}{%
10280     \glxtrifwasfirstuse
10281     {%
10282       \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10283     }%
10284   }%

```

Put the insertion into the post-link:

```

10285     \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10286   }%
10287 }%
10288 \glshasattribute{\the\glslabeltok}{regular}%
10289 {%
10290   \glssetattribute{\the\glslabeltok}{regular}{false}%
10291 }%
10292 {}%
10293 }%
10294 }%
10295 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10296 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10297 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10298 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10299 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10300 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10301 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10302   \glsabbrvfont{\glsaccessshort{##1}}%
10303 }%
10304 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10305   \glsabbrvfont{\glsaccessshortpl{##1}}%
10306 }%
10307 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10308   \glsabbrvfont{\Glsaccessshort{##1}}%
10309 }%
10310 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10311   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10312 }%

```


First use full form:

```

10313 \renewcommand*{\glxstrfullformat}[2]{%
10314   \glxstrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10315 }%
10316 \renewcommand*{\glxstrfullplformat}[2]{%
10317   \glxstrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10318 }%
10319 \renewcommand*{\Glsxstrfullformat}[2]{%
10320   \glxstrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10321 }%
10322 \renewcommand*{\Glsxstrfullplformat}[2]{%
10323   \glxstrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10324 }%

```

In-line format. Commands like `\glxstrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10325 \renewcommand*{\glxstrinlinefullformat}[2]{%
10326   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
10327   \ifglxstrinsertinside{##2}\fi}%
10328   \ifglxstrinsertinside \else{##2}\fi
10329 }%
10330 \renewcommand*{\glxstrinlinefullplformat}[2]{%
10331   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
10332   \ifglxstrinsertinside{##2}\fi}%
10333   \ifglxstrinsertinside \else{##2}\fi
10334 }%
10335 \renewcommand*{\Glsxstrinlinefullformat}[2]{%
10336   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10337   \ifglxstrinsertinside{##2}\fi}%
10338   \ifglxstrinsertinside \else{##2}\fi
10339 }%
10340 \renewcommand*{\Glsxstrinlinefullplformat}[2]{%
10341   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10342   \ifglxstrinsertinside{##2}\fi}%
10343   \ifglxstrinsertinside \else{##2}\fi
10344 }%
10345 }

```

`ong-hyphen-desc` Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

10346 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10347 {%
10348   \renewcommand*{\CustomAbbreviationFields}{%
10349     name={\glxstrshortlongdescname},
10350     sort={\glxstrshortlongdescsort},%
10351     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10352     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10353     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10354     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10355 }%

```

```

10356 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10357   \csdef{glxtrpostlink\glscategorylabel}{%
10358     \glxtrifwasfirstuse
10359     {%
10360       \glxtrposthyphenlong{\glslabel}{\glinsert}%
10361     }%
10362     {%

```

Put the insertion into the post-link:

```

10363       \glxtrposthyphensubsequent{\glslabel}{\glinsert}%
10364     }%
10365   }%
10366   \glshasattribute{\the\glslabeltok}{regular}%
10367   {%
10368     \glissetattribute{\the\glslabeltok}{regular}{false}%
10369   }%
10370   {}%
10371 }%
10372 }%
10373 {%
10374   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10375 }

```

1.7.8 Predefined Styles (No Short on First Use)

These styles show only the long form on first use and only the short form on subsequent use.

`lsabbrvonlyfont`

```

10376 \newcommand*{\glabbrvonlyfont}{\glabbrvdefaultfont}%

```

`stabbrvonlyfont`

```

10377 \newcommand*{\glfirstabbrvonlyfont}{\glabbrvonlyfont}%

```

`glslongonlyfont`

```

10378 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

`rstlongonlyfont`

```

10379 \newcommand*{\glfirstlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

`lsxtronlysuffix`

```

10380 \newcommand*{\glsxtronlysuffix}{\glxtrabbrvpluralsuffix}

```

`\glsxtronlyname` The default name format for this style.

```

10381 \newcommand*{\glsxtronlyname}{%
10382   \protect\glabbrvonlyfont{\the\glsshorttok}%
10383 }

```

only-short-only

```
10384 \newabbreviationstyle{long-only-short-only}%
10385 {%
10386   \renewcommand*{\CustomAbbreviationFields}{%
10387     name={\glstronlyname},
10388     sort={\the\glsshorttok},
10389     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10390     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10391     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10392     description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
10393   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10394     \glshasattribute{\the\glslabeltok}{regular}%
10395     {%
10396       \glissetattribute{\the\glslabeltok}{regular}{false}%
10397     }%
10398   }%
10399 }%
10400 }%
10401 {%
10402   \renewcommand*{\abbrvpluralsuffix}{\protect\glstronlysuffix}%
10403   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
10404   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10405   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10406   \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
10407   \renewcommand*{\glsxtrfullformat}[2]{%
10408     \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10409     \ifglsxtrininsertinside\else##2\fi
10410   }%
10411   \renewcommand*{\glsxtrfullplformat}[2]{%
10412     \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10413     \ifglsxtrininsertinside\else##2\fi
10414   }%
10415   \renewcommand*{\Glsxtrfullformat}[2]{%
10416     \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10417     \ifglsxtrininsertinside\else##2\fi
10418   }%
10419   \renewcommand*{\Glsxtrfullplformat}[2]{%
10420     \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10421     \ifglsxtrininsertinside\else##2\fi
10422   }%
```

The inline full form does show the short form.

```
10423   \renewcommand*{\glsxtrinlinefullformat}[2]{%
10424     \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10425     \ifglsxtrininsertinside\else##2\fi
10426     \glsxtrfullsep{##1}%
```

```

10427 \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10428 }%
10429 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10430 \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10431 \ifglxtrininsertinside\else##2\fi
10432 \glxtrfullsep{##1}%
10433 \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10434 }%
10435 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10436 \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10437 \ifglxtrininsertinside\else##2\fi
10438 \glxtrfullsep{##1}%
10439 \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10440 }%
10441 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10442 \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10443 \ifglxtrininsertinside\else##2\fi
10444 \glxtrfullsep{##1}%
10445 \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10446 }%
10447 }

```

xtronlydescsort

```

10448 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

10449 \newcommand*{\glxtronlydescname}{%
10450 \protect\glslongfont{\the\glslongtok}%
10451 }

```

short-only-desc

```

10452 \newabbreviationstyle{long-only-short-only-desc}%
10453 {%
10454 \renewcommand*{\CustomAbbreviationFields}{%
10455 name={\glxtronlydescname},
10456 sort={\glxtronlydescsort},%
10457 first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10458 firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10459 text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
10460 plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
10461 }%

```

Unset the regular attribute if it has been set.

```

10462 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10463 \glshasattribute{\the\glslabeltok}{regular}%
10464 {%
10465 \glissetattribute{\the\glslabeltok}{regular}{false}%
10466 }%
10467 {}%

```

```

10468 }%
10469 }%
10470 {%
10471 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10472 }

```

1.8 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\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:

```
10473 \let\@glstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

10474 \renewcommand*{\markright}[1]{%
10475 \glstrmarkhook
10476 \@glstr@org@markright{\@glstrinmark#1\@glstrnotinmark}%
10477 \glstrrestoremarkhook
10478 }

```

`\markboth` Save original definition:

```
10479 \let\@glstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```

10480 \renewcommand*{\markboth}[2]{%
10481 \glstrmarkhook

```

```

10482 \@glsxtr@org@markboth
10483   {\@glsxtrinmark#1\@glsxtrnotinmark}%
10484   {\@glsxtrinmark#2\@glsxtrnotinmark}%
10485 \glsxtrrestoremarkhook
10486 }

```

Also do this for \@starttoc

\@starttoc Save original definition:

```

10487 \let\@glsxtr@org@@starttoc\@starttoc

```

Redefine:

```

10488 \renewcommand*{\@starttoc}[1]{%
10489   \glsxtrmarkhook
10490   \@glsxtrinmark
10491   \@glsxtr@org@@starttoc{#1}%
10492   \@glsxtrnotinmark
10493   \glsxtrrestoremarkhook
10494 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

10495 \newcommand*{\glsxtrRevertMarks}{%
10496   \let\markright\@glsxtr@org@markright
10497   \let\markboth\@glsxtr@org@markboth
10498   \let\@starttoc\@glsxtr@org@@starttoc
10499 }

```

\glsxtrifinmark

```

10500 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

10501 \newrobustcmd*{\@glsxtrinmark}{%
10502   \let\glsxtrifinmark\@firstoftwo
10503 }

```

glsxtrnotinmark

```

10504 \newrobustcmd*{\@glsxtrnotinmark}{%
10505   \let\glsxtrifinmark\@secondoftwo
10506 }

```

eorpdforheading

```

10507 \ifdef\texorpdfstring
10508 {
10509   \newcommand*{\glsxtrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10510 }
10511 {
10512   \newcommand*{\glsxtrtitleorpdforheading}[3]{#1}
10513 }

```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```
10514 \newcommand*{\glxtrmarkhook}{%
```

Save current definitions:

```
10515 \let\@glxtr@org@MakeUppercase\MakeUppercase
10516 \let\@glxtr@org@glxtrtitleorpdforheading\glxtrtitleorpdforheading
10517 \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
10518 \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
10519 \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10520 \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
10521 \let\@glxtr@org@glxtrtitlename\glxtrtitlename
10522 \let\@glxtr@org@Glsxtrtitlename\Glsxtrtitlename
10523 \let\@glxtr@org@glxtrtitletext\glxtrtitletext
10524 \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
10525 \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
10526 \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10527 \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
10528 \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10529 \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
10530 \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10531 \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
10532 \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl
10533 \let\@glxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
10534 \let\@glxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10535 \let\@glxtr@org@glxtrtitlefull\glxtrtitlefull
10536 \let\@glxtr@org@glxtrtitlefullpl\glxtrtitlefullpl
10537 \let\@glxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
10538 \let\@glxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl
```

New definitions

```
10539 \let\glxtrifinmark\@firstoftwo
10540 \let\MakeUppercase\MakeTextUppercase
10541 \let\glxtrtitleorpdforheading\@thirdofthree
10542 \let\glxtrtitleshort\glxtrheadshort
10543 \let\glxtrtitleshortpl\glxtrheadshortpl
10544 \let\Glsxtrtitleshort\Glsxtrheadshort
10545 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
10546 \let\glxtrtitlename\glxtrheadname
10547 \let\Glsxtrtitlename\Glsxtrheadname
10548 \let\glxtrtitletext\glxtrheadtext
10549 \let\Glsxtrtitletext\Glsxtrheadtext
10550 \let\glxtrtitleplural\glxtrheadplural
10551 \let\Glsxtrtitleplural\Glsxtrheadplural
10552 \let\glxtrtitlefirst\glxtrheadfirst
10553 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10554 \let\glxtrtitlefirstplural\glxtrheadfirstplural
10555 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10556 \let\glxtrtitlelong\glxtrheadlong
10557 \let\glxtrtitlelongpl\glxtrheadlongpl
```

```

10558 \let\Glsxtrtitlelong\Glsxtrheadlong
10559 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10560 \let\glxstrtitlefull\glxstrheadfull
10561 \let\glxstrtitlefullpl\glxstrheadfullpl
10562 \let\Glsxtrtitlefull\Glsxtrheadfull
10563 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10564 }

```

`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.)

```

10565 \newcommand*{\glxstrrestoremarkhook}{%
10566 \let\glxstrifinmark\@secondoftwo
10567 \let\MakeUppercase\@glxstr@org@MakeUppercase
10568 \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
10569 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
10570 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
10571 \let\Glsxtrtitleshort\@glxstr@org@Glsxtrtitleshort
10572 \let\Glsxtrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
10573 \let\glxstrtitlename\@glxstr@org@glxstrtitlename
10574 \let\Glsxtrtitlename\@glxstr@org@Glsxtrtitlename
10575 \let\glxstrtitletext\@glxstr@org@glxstrtitletext
10576 \let\Glsxstrtitletext\@glxstr@org@Glsxstrtitletext
10577 \let\glxstrtitleplural\@glxstr@org@glxstrtitleplural
10578 \let\Glsxstrtitleplural\@glxstr@org@Glsxstrtitleplural
10579 \let\glxstrtitlefirst\@glxstr@org@glxstrtitlefirst
10580 \let\Glsxstrtitlefirst\@glxstr@org@Glsxstrtitlefirst
10581 \let\glxstrtitlefirstplural\@glxstr@org@glxstrtitlefirstplural
10582 \let\Glsxstrtitlefirstplural\@glxstr@org@Glsxstrtitlefirstplural
10583 \let\glxstrtitlelong\@glxstr@org@glxstrtitlelong
10584 \let\glxstrtitlelongpl\@glxstr@org@glxstrtitlelongpl
10585 \let\Glsxstrtitlelong\@glxstr@org@Glsxstrtitlelong
10586 \let\Glsxstrtitlelongpl\@glxstr@org@Glsxstrtitlelongpl
10587 \let\glxstrtitlefull\@glxstr@org@glxstrtitlefull
10588 \let\glxstrtitlefullpl\@glxstr@org@glxstrtitlefullpl
10589 \let\Glsxstrtitlefull\@glxstr@org@Glsxstrtitlefull
10590 \let\Glsxstrtitlefullpl\@glxstr@org@Glsxstrtitlefullpl
10591 }

```

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.

```

10592 \newcommand*{\glxstrheadshort}[1]{%
10593 \protect\NoCaseChange
10594 {%
10595 \gl@ifattribute{#1}{headuc}{true}%
10596 {%

```



```

10597      \GLSxtrshort[noindex,hyper=false]{#1}[]%
10598    }%
10599    {%
10600      \glsxtrshort[noindex,hyper=false]{#1}[]%
10601    }%
10602  }%
10603 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

10604 \newrobustcmd*{\glsxtrtitleshort}[1]{%
10605   \glsxtrshort[noindex,hyper=false]{#1}[]%
10606 }

```

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.

```

10607 \newcommand*{\glsxtrheadshortpl}[1]{%
10608   \protect\NoCaseChange
10609   {%
10610     \glsifattribute{#1}{headuc}{true}%
10611     {%
10612       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10613     }%
10614     {%
10615       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10616     }%
10617   }%
10618 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

10619 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10620   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10621 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

10622 \newcommand*{\Glsxtrheadshort}[1]{%
10623   \protect\NoCaseChange
10624   {%
10625     \glsifattribute{#1}{headuc}{true}%
10626     {%
10627       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10628     }%
10629     {%
10630       \Glsxtrshort[noindex,hyper=false]{#1}[]%
10631     }%
10632   }%
10633 }

```

`\glxtrtitleshort` Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10634 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10635   \Glsxtrshort[noindex,hyper=false]{#1}[]%
10636 }
```

`\glxtrheadshortpl` Command used to display plural short form in the page header with the first letter converted to upper case.

```
10637 \newcommand*{\Glsxtrheadshortpl}[1]{%
10638   \protect\NoCaseChange
10639   {%
10640     \glsifattribute{#1}{headuc}{true}%
10641     {%
10642       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10643     }%
10644     {%
10645       \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10646     }%
10647   }%
10648 }
```

`\glxtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10649 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10650   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10651 }
```

`\glxtrheadname` As above but for the name value.

```
10652 \newcommand*{\glxtrheadname}[1]{%
10653   \protect\NoCaseChange
10654   {%
10655     \glsifattribute{#1}{headuc}{true}%
10656     {%
10657       \GLSname[noindex,hyper=false]{#1}[]%
10658     }%
10659     {%
10660       \glsname[noindex,hyper=false]{#1}[]%
10661     }%
10662   }%
10663 }
```

`\glxtrtitlename` Command to display name value in section title and table of contents.

```
10664 \newrobustcmd*{\glxtrtitlename}[1]{%
10665   \glsname[noindex,hyper=false]{#1}[]%
10666 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10667 \newcommand*{\Glsxtrheadname}[1]{%
```

```

10668 \protect\NoCaseChange
10669 {%
10670   \glsifattribute{#1}{headuc}{true}%
10671   {%
10672     \GLSname[noindex,hyper=false]{#1}[]%
10673   }%
10674   {%
10675     \Glsname[noindex,hyper=false]{#1}[]%
10676   }%
10677 }%
10678 }

```

Glsxtrtitlename Command to display name value in section title and table of contents with the first letter changed to upper case.

```

10679 %\changes{1.21}{2017-11-03}{new}
10680 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10681   \Glsname[noindex,hyper=false]{#1}[]%
10682 }

```

\glsxtrheadtext As above but for the text value.

```

10683 \newcommand*{\glsxtrheadtext}[1]{%
10684   \protect\NoCaseChange
10685   {%
10686     \glsifattribute{#1}{headuc}{true}%
10687     {%
10688       \GLStext[noindex,hyper=false]{#1}[]%
10689     }%
10690     {%
10691       \glstext[noindex,hyper=false]{#1}[]%
10692     }%
10693   }%
10694 }

```

glsxtrtitletext Command to display text value in section title and table of contents.

```

10695 \newrobustcmd*{\glsxtrtitletext}[1]{%
10696   \glstext[noindex,hyper=false]{#1}[]%
10697 }

```

\Glsxtrheadtext First letter converted to upper case

```

10698 \newcommand*{\Glsxtrheadtext}[1]{%
10699   \protect\NoCaseChange
10700   {%
10701     \glsifattribute{#1}{headuc}{true}%
10702     {%
10703       \GLStext[noindex,hyper=false]{#1}[]%
10704     }%
10705     {%
10706       \Glsstext[noindex,hyper=false]{#1}[]%
10707     }%

```

```

10708 }%
10709 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

10710 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10711   \Gls{text[noindex,hyper=false]}{#1}[]%
10712 }

```

lsxtrheadplural As above but for the plural value.

```

10713 \newcommand*{\glxtrheadplural}[1]{%
10714   \protect\NoCaseChange
10715   {%
10716     \gl@ifattribute{#1}{headuc}{true}%
10717     {%
10718       \GLSplural[noindex,hyper=false]{#1}[]%
10719     }%
10720     {%
10721       \glspplural[noindex,hyper=false]{#1}[]%
10722     }%
10723   }%
10724 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```

10725 \newrobustcmd*{\glsxtrtitleplural}[1]{%
10726   \glspplural[noindex,hyper=false]{#1}[]%
10727 }

```

lsxtrheadplural Convert first letter to upper case.

```

10728 \newcommand*{\Glsxtrheadplural}[1]{%
10729   \protect\NoCaseChange
10730   {%
10731     \gl@ifattribute{#1}{headuc}{true}%
10732     {%
10733       \GLSplural[noindex,hyper=false]{#1}[]%
10734     }%
10735     {%
10736       \Glspplural[noindex,hyper=false]{#1}[]%
10737     }%
10738   }%
10739 }

```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

10740 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10741   \Glspplural[noindex,hyper=false]{#1}[]%
10742 }

```

glsxtrheadfirst As above but for the first value.

```
10743 \newcommand*{\glsxtrheadfirst}[1]{%
10744   \protect\NoCaseChange
10745   {%
10746     \glsifattribute{#1}{headuc}{true}%
10747     {%
10748       \GLSfirst[noindex,hyper=false]{#1}[]%
10749     }%
10750   }%
10751   \glsfirst[noindex,hyper=false]{#1}[]%
10752 }%
10753 }%
10754 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
10755 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10756   \glsfirst[noindex,hyper=false]{#1}[]%
10757 }
```

Glsxtrheadfirst First letter converted to upper case

```
10758 \newcommand*{\Glsxtrheadfirst}[1]{%
10759   \protect\NoCaseChange
10760   {%
10761     \glsifattribute{#1}{headuc}{true}%
10762     {%
10763       \GLSfirst[noindex,hyper=false]{#1}[]%
10764     }%
10765   }%
10766   \Glsfirst[noindex,hyper=false]{#1}[]%
10767 }%
10768 }%
10769 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10770 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10771   \Glsfirst[noindex,hyper=false]{#1}[]%
10772 }
```

headfirstplural As above but for the firstplural value.

```
10773 \newcommand*{\glsxtrheadfirstplural}[1]{%
10774   \protect\NoCaseChange
10775   {%
10776     \glsifattribute{#1}{headuc}{true}%
10777     {%
10778       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10779     }%
10780   }%
```

```

10781 \glsfirstplural[noindex,hyper=false]{#1}[]%
10782 }%
10783 }%
10784 }

```

titlefirstplural Command to display firstplural value in section title and table of contents.

```

10785 \newrobustcmd*{\glsxtrtitlefirstplural}[1]{%
10786 \glsfirstplural[noindex,hyper=false]{#1}[]%
10787 }

```

headfirstplural First letter converted to upper case

```

10788 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10789 \protect\NoCaseChange
10790 {%
10791 \glsifattribute{#1}{headuc}{true}%
10792 {%
10793 \GLSfirstplural[noindex,hyper=false]{#1}[]%
10794 }%
10795 {%
10796 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10797 }%
10798 }%
10799 }

```

titlefirstplural Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10800 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10801 \Glsfirstplural[noindex,hyper=false]{#1}[]%
10802 }

```

\glsxtrheadlong Command used to display long form in the page header.

```

10803 \newcommand*{\glsxtrheadlong}[1]{%
10804 \protect\NoCaseChange
10805 {%
10806 \glsifattribute{#1}{headuc}{true}%
10807 {%
10808 \GLSxtrlong[noindex,hyper=false]{#1}[]%
10809 }%
10810 {%
10811 \glsxtrlong[noindex,hyper=false]{#1}[]%
10812 }%
10813 }%
10814 }

```

glsxtrtitlelong Command to display long form of abbreviation in section title and table of contents.

```

10815 \newrobustcmd*{\glsxtrtitlelong}[1]{%
10816 \glsxtrlong[noindex,hyper=false]{#1}[]%
10817 }

```

`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.

```

10818 \newcommand*{\glxtrheadlongpl}[1]{%
10819   \protect\NoCaseChange
10820   {%
10821     \gl@ifattribute{#1}{headuc}{true}%
10822     {%
10823       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10824     }%
10825     {%
10826       \glxtrlongpl[noindex,hyper=false]{#1}[]%
10827     }%
10828   }%
10829 }

```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

10830 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
10831   \glxtrlongpl[noindex,hyper=false]{#1}[]%
10832 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

10833 \newcommand*{\Glsxtrheadlong}[1]{%
10834   \protect\NoCaseChange
10835   {%
10836     \gl@ifattribute{#1}{headuc}{true}%
10837     {%
10838       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10839     }%
10840     {%
10841       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10842     }%
10843   }%
10844 }

```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10845 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10846   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10847 }

```

`lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

10848 \newcommand*{\Glsxtrheadlongpl}[1]{%
10849   \protect\NoCaseChange
10850   {%
10851     \gl@ifattribute{#1}{headuc}{true}%

```

```

10852   {%
10853     \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10854   }%
10855   {%
10856     \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10857   }%
10858 }%
10859 }

```

sxtrtitlelongpl Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10860 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10861   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10862 }

```

\glsxtrheadfull Command used to display full form in the page header.

```

10863 \newcommand*{\glsxtrheadfull}[1]{%
10864   \protect\NoCaseChange
10865   {%
10866     \glsifattribute{#1}{headuc}{true}%
10867     {%
10868       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10869     }%
10870     {%
10871       \glsxtrfull[noindex,hyper=false]{#1}[]%
10872     }%
10873   }%
10874 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

10875 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10876   \glsxtrfull[noindex,hyper=false]{#1}[]%
10877 }

```

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.

```

10878 \newcommand*{\glsxtrheadfullpl}[1]{%
10879   \protect\NoCaseChange
10880   {%
10881     \glsifattribute{#1}{headuc}{true}%
10882     {%
10883       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10884     }%
10885     {%
10886       \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10887     }%
10888   }%
10889 }

```


`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents.

```

10890 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10891   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10892 }

```

`\Glsxtrheadfull` Command used to display full form in the page header with the first letter converted to upper case.

```

10893 \newcommand*{\Glsxtrheadfull}[1]{%
10894   \protect\NoCaseChange
10895   {%
10896     \glsifattribute{#1}{headuc}{true}%
10897     {%
10898       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10899     }%
10900     {%
10901       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10902     }%
10903   }%
10904 }

```

`Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10905 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10906   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10907 }

```

`lGlsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```

10908 \newcommand*{\Glsxtrheadfullpl}[1]{%
10909   \protect\NoCaseChange
10910   {%
10911     \glsifattribute{#1}{headuc}{true}%
10912     {%
10913       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10914     }%
10915     {%
10916       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10917     }%
10918   }%
10919 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10920 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10921   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10922 }

```

`\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.

```

10923 \ifdef\texorpdfstring
10924 {
10925   \newcommand*\glsfmtshort}[1]{%
10926     \texorpdfstring
10927       {\glsxtrtitleshort{#1}}%
10928       {\glsentryshort{#1}}%
10929   }
10930 }
10931 {
10932   \newcommand*\glsfmtshort}[1]{%
10933     \glsxtrtitleshort{#1}}
10934 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

10935 \ifdef\texorpdfstring
10936 {
10937   \newcommand*\glsfmtshortpl}[1]{%
10938     \texorpdfstring
10939       {\glsxtrtitleshortpl{#1}}%
10940       {\glsentryshortpl{#1}}%
10941   }
10942 }
10943 {
10944   \newcommand*\glsfmtshortpl}[1]{%
10945     \glsxtrtitleshortpl{#1}}
10946 }

```

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).

```

10947 \ifdef\texorpdfstring
10948 {
10949   \newcommand*\Glsfmtshort}[1]{%
10950     \texorpdfstring
10951       {\Glsxtrtitleshort{#1}}%
10952       {\glsentryshort{#1}}%
10953   }
10954 }
10955 {
10956   \newcommand*\Glsfmtshort}[1]{%
10957     \Glsxtrtitleshort{#1}}
10958 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10959 \ifdef\texorpdfstring
10960 {
10961   \newcommand*\Glsfmtshortpl}[1]{%
10962     \texorpdfstring
10963     {\Glsxtrtitleshortpl{#1}}%
10964     {\glsentryshortpl{#1}}%
10965   }
10966 }
10967 {
10968   \newcommand*\Glsfmtshortpl}[1]{%
10969     \Glsxtrtitleshortpl{#1}}
10970 }

```

`\glsfmtname` As above but for the name value.

```

10971 \ifdef\texorpdfstring
10972 {
10973   \newcommand*\glsfmtname}[1]{%
10974     \texorpdfstring
10975     {\glsxtrtitlename{#1}}%
10976     {\glsentryname{#1}}%
10977   }
10978 }
10979 {
10980   \newcommand*\glsfmtname}[1]{%
10981     \glsxtrtitlename{#1}}
10982 }

```

`\Glsfmtname` First letter converted to upper case.

```

10983 \ifdef\texorpdfstring
10984 {
10985   \newcommand*\Glsfmtname}[1]{%
10986     \texorpdfstring
10987     {\Glsxtrtitlename{#1}}%
10988     {\glsentryname{#1}}%
10989   }
10990 }
10991 {
10992   \newcommand*\Glsfmtname}[1]{%
10993     \Glsxtrtitlename{#1}}
10994 }

```

`\glsfmttext` As above but for the text value.

```

10995 \ifdef\texorpdfstring
10996 {
10997   \newcommand*\glsfmttext}[1]{%
10998     \texorpdfstring
10999     {\glsxtrtitletext{#1}}%
11000     {\glsentrytext{#1}}%
11001   }

```

```

11002 }
11003 {
11004   \newcommand*{\glsfmttext}[1]{%
11005     \glsxtrtitletext{#1}}
11006 }

```

`\Glsfmttext` First letter converted to upper case.

```

11007 \ifdef\texorpdfstring
11008 {
11009   \newcommand*{\Glsfmttext}[1]{%
11010     \texorpdfstring
11011     {\Glsxtrtitletext{#1}}%
11012     {\glsentrytext{#1}}%
11013   }
11014 }
11015 {
11016   \newcommand*{\Glsfmttext}[1]{%
11017     \Glsxtrtitletext{#1}}
11018 }

```

`\glsfmtplural` As above but for the plural value.

```

11019 \ifdef\texorpdfstring
11020 {
11021   \newcommand*{\glsfmtplural}[1]{%
11022     \texorpdfstring
11023     {\glsxtrtitleplural{#1}}%
11024     {\glsentryplural{#1}}%
11025   }
11026 }
11027 {
11028   \newcommand*{\glsfmtplural}[1]{%
11029     \glsxtrtitleplural{#1}}
11030 }

```

`\Glsfmtplural` First letter converted to upper case.

```

11031 \ifdef\texorpdfstring
11032 {
11033   \newcommand*{\Glsfmtplural}[1]{%
11034     \texorpdfstring
11035     {\Glsxtrtitleplural{#1}}%
11036     {\glsentryplural{#1}}%
11037   }
11038 }
11039 {
11040   \newcommand*{\Glsfmtplural}[1]{%
11041     \Glsxtrtitleplural{#1}}
11042 }

```

`\glsfmtfirst` As above but for the first value.

```

11043 \ifdef\texorpdfstring
11044 {
11045   \newcommand*\glsfmtfirst}[1]{%
11046     \texorpdfstring
11047     {\glsxtrtitlefirst{#1}}%
11048     {\glsentryfirst{#1}}%
11049   }
11050 }
11051 {
11052   \newcommand*\glsfmtfirst}[1]{%
11053     \glsxtrtitlefirst{#1}}
11054 }

```

\Glsfmtfirst First letter converted to upper case.

```

11055 \ifdef\texorpdfstring
11056 {
11057   \newcommand*\Glsfmtfirst}[1]{%
11058     \texorpdfstring
11059     {\Glsxtrtitlefirst{#1}}%
11060     {\glsentryfirst{#1}}%
11061   }
11062 }
11063 {
11064   \newcommand*\Glsfmtfirst}[1]{%
11065     \Glsxtrtitlefirst{#1}}
11066 }

```

\glsfmtfirstpl As above but for the firstplural value.

```

11067 \ifdef\texorpdfstring
11068 {
11069   \newcommand*\glsfmtfirstpl}[1]{%
11070     \texorpdfstring
11071     {\glsxtrtitlefirstplural{#1}}%
11072     {\glsentryfirstplural{#1}}%
11073   }
11074 }
11075 {
11076   \newcommand*\glsfmtfirstpl}[1]{%
11077     \glsxtrtitlefirstplural{#1}}
11078 }

```

\Glsfmtfirstpl First letter converted to upper case.

```

11079 \ifdef\texorpdfstring
11080 {
11081   \newcommand*\Glsfmtfirstpl}[1]{%
11082     \texorpdfstring
11083     {\Glsxtrtitlefirstplural{#1}}%
11084     {\glsentryfirstplural{#1}}%
11085   }

```

```

11086 }
11087 {
11088   \newcommand*{\Glsfmtfirstpl}[1]{%
11089     \Glsxtrtitlefirstplural{#1}}
11090 }

```

`\glsfmtlong` As above but for the long value.

```

11091 \ifdef\texorpdfstring
11092 {
11093   \newcommand*{\glsfmtlong}[1]{%
11094     \texorpdfstring
11095       {\Glsxtrtitlelong{#1}}%
11096       {\glsentrylong{#1}}%
11097   }
11098 }
11099 {
11100   \newcommand*{\glsfmtlong}[1]{%
11101     \Glsxtrtitlelong{#1}}
11102 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11103 \ifdef\texorpdfstring
11104 {
11105   \newcommand*{\Glsfmtlong}[1]{%
11106     \texorpdfstring
11107       {\Glsxtrtitlelong{#1}}%
11108       {\glsentrylong{#1}}%
11109   }
11110 }
11111 {
11112   \newcommand*{\Glsfmtlong}[1]{%
11113     \Glsxtrtitlelong{#1}}
11114 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11115 \ifdef\texorpdfstring
11116 {
11117   \newcommand*{\glsfmtlongpl}[1]{%
11118     \texorpdfstring
11119       {\Glsxtrtitlelongpl{#1}}%
11120       {\glsentrylongpl{#1}}%
11121   }
11122 }
11123 {
11124   \newcommand*{\glsfmtlongpl}[1]{%
11125     \Glsxtrtitlelongpl{#1}}
11126 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11127 \ifdef\txorpdfstring
11128 {
11129   \newcommand*\Glsfmtlongpl}[1]{%
11130     \txorpdfstring
11131     {\Glsxtrtitlelongpl{#1}}%
11132     {\glsentrylongpl{#1}}%
11133   }
11134 }
11135 {
11136   \newcommand*\Glsfmtlongpl}[1]{%
11137     \Glsxtrtitlelongpl{#1}}
11138 }

```

`\glsfmtfull` In-line full format.

```

11139 \ifdef\txorpdfstring
11140 {
11141   \newcommand*\glsfmtfull}[1]{%
11142     \txorpdfstring
11143     {\glsxtrtitlefull{#1}}%
11144     {\glsxtrinlinefullformat{#1}-{}}%
11145   }
11146 }
11147 {
11148   \newcommand*\glsfmtfull}[1]{%
11149     \glsxtrtitlefull{#1}}
11150 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11151 \ifdef\txorpdfstring
11152 {
11153   \newcommand*\Glsfmtfull}[1]{%
11154     \txorpdfstring
11155     {\Glsxtrtitlefull{#1}}%
11156     {\Glsxtrinlinefullformat{#1}-{}}%
11157   }
11158 }
11159 {
11160   \newcommand*\Glsfmtfull}[1]{%
11161     \Glsxtrtitlefull{#1}}
11162 }

```

`\glsfmtfullpl` In-line full plural format.

```

11163 \ifdef\txorpdfstring
11164 {
11165   \newcommand*\glsfmtfullpl}[1]{%
11166     \txorpdfstring
11167     {\glsxtrtitlefullpl{#1}}%
11168     {\glsxtrinlinefullplformat{#1}-{}}%
11169   }

```

```

11170 }
11171 {
11172   \newcommand*{\Glsfmtfullpl}[1]{%
11173     \Glsxtrtitlefullpl{#1}}
11174 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11175 \ifdef\texorpdfstring
11176 {
11177   \newcommand*{\Glsfmtfullpl}[1]{%
11178     \texorpdfstring
11179     {\Glsxtrtitlefullpl{#1}}%
11180     {\Glsxtrinlinefullplformat{#1}{}}%
11181   }
11182 }
11183 {
11184   \newcommand*{\Glsfmtfullpl}[1]{%
11185     \Glsxtrtitlefullpl{#1}}
11186 }

```

1.9 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

11187 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11188   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
11189 }

```

`sariesExtraLang`

```

11190 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11191   \ProvidesFile{glossariesxtr-#1.ldf}%
11192 }

```

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.)

```

11193 \@ifpackageloaded{tracklang}
11194 {%
11195   \AnyTrackedLanguages
11196   {%
11197     \ForEachTrackedDialect{\this@dialect}{%
11198       \IfTrackedLanguageFileExists{\this@dialect}%
11199       {glossariesxtr-}% prefix
11200       {.ldf}%
11201     }%

```



```

11202         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11203     }%
11204     {%
11205     }%
11206     }%
11207 }%
11208 {}%
11209 }
11210 {}

```

Load glossaries-extra-stylemods if required.

```
11211 \@glsxtr@redefstyles
```

and set the style:

```
11212 \@glsxtr@do@style
```

2 Style Adjustments (glossaries-extra-stylemods.sty)

This package adjusts the predefined styles so that they include the post description hook. Also, some other minor adjustments may be made to make existing styles more flexible.

2.1 Package Initialisation

First identify package:

```
11213 \NeedsTeXFormat{LaTeX2e}
11214 \ProvidesPackage{glossaries-extra-stylemods}[2018/01/05 v1.26 (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`

```
11215 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
11216 \DeclareOption{all}{%
11217   \appto\@glxtr@loadstyles{%
11218     \RequirePackage{glossary-inline}%
11219     \RequirePackage{glossary-list}%
11220     \RequirePackage{glossary-tree}%
11221     \RequirePackage{glossary-mcols}%
11222     \RequirePackage{glossary-long}%
11223     \RequirePackage{glossary-longragged}%
11224     \RequirePackage{glossary-longbooktabs}%
11225     \RequirePackage{glossary-super}%
11226     \RequirePackage{glossary-superragged}%
11227     \RequirePackage{glossary-bookindex}%
11228   }
11229 }

11230 \DeclareOption*{%
11231   \IfFileExists{glossary-\CurrentOption.sty}
11232   {\eappto\@glxtr@loadstyles{%
11233     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
11234   }%
11235   {%
11236     \PackageError{glossaries-extra-styles}%

```

```

11237     {Unknown option ‘\CurrentOption’}{}%
11238   }%
11239 }

```

Process the package options:

```
11240 \ProcessOptions
```

Load the required packages:

```
11241 \@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`.

```
11242 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

11243 \providecommand{\renewglossarystyle}[2]{%
11244   \ifcsundef{@glsstyle@#1}%
11245   {%
11246     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
11247   }%
11248   {%
11249     \csdef{@glsstyle@#1}{#2}%
11250   }%
11251 }

```

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.

```

11252 \ifdef{@glsstyle@listdotted}
11253 {%
11254   \renewglossarystyle{listdotted}{%
11255     \setglossarystyle{list}%
11256     \renewcommand*{\glossentry}[2]{%
11257       \item[]\makebox[\glslistdottedwidth][l]{%
11258         \glsentryitem{##1}%
11259         \glstarget{##1}{\glossentryname{##1}}%
11260         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11261         \glossentrydesc{##1}\glspostdescription}%
11262     \renewcommand*{\subglossentry}[3]{%
11263       \item[]\makebox[\glslistdottedwidth][l]{%
11264         \glssubentryitem{##2}%
11265         \glstarget{##2}{\glossentryname{##2}}%
11266         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11267         \glossentrydesc{##2}\glspostdescription}%
11268   }

```

```
11269 }
11270 {%
```

Assume the style isn't required if it hasn't already been defined.

```
11271 }
```

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.

```
11272 \ifdef{\@glsstyle@list}
11273 {%
```

listprelocation Space before number list for top-level entries.

```
11274 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11275 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11276 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11277 \renewglossarystyle{list}{%
11278 \renewenvironment{theglossary}%
11279 {\begin{description}}{\end{description}}%
11280 \renewcommand*{\glossaryheader}{}%
11281 \renewcommand*{\glsgroupheading}[1]{}%
11282 \renewcommand*{\glossentry}[2]{%
11283 \item[\glsentryitem{##1}]%
11284 \glstarget{##1}{\glossentryname{##1}}]
11285 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11286 \renewcommand*{\subglossentry}[3]{%
11287 \glssubentryitem{##2}%
11288 \glstarget{##2}{\strut}\space
11289 \glossentrydesc{##2}\glspostdescription
11290 \glslistchildprelocation ##3\glslistchildpostlocation}%
11291 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11292 }
11293 }
11294 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11295 \ifdef{\@glsstyle@altlist}
11296 {%
11297 \renewglossarystyle{altlist}{%
11298 \setglossarystyle{list}%
11299 \renewcommand*{\glossentry}[2]{%
11300 \item[\glsentryitem{##1}]%
```

```

11301      \glstarget{##1}{\glossentryname{##1}}}%
11302      \mbox{}\par\nobreak\@afterheading
11303      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11304      \renewcommand{\subglossentry}[3]{%
11305        \par
11306        \glssubentryitem{##2}%
11307        \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11308        \glslistchildprelocation ##3}%
11309    }
11310 }
11311 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11312 \ifdef{\@glsstyle@listgroup}
11313 {%
11314   \renewglossarystyle{listgroup}{%
11315     \setglossarystyle{list}%
11316     \renewcommand*\glsgroupheading[1]{%
11317       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11318       \mbox{}\par\nobreak\@afterheading
11319     }%
11320   }
11321 }
11322 {}

```

Similarly for listhypergroup.

```

11323 \ifdef{\@glsstyle@listhypergroup}
11324 {%
11325   \renewglossarystyle{listhypergroup}{%
11326     \setglossarystyle{list}%
11327     \renewcommand*\glossaryheader{%
11328       \glslistnavigationitem{\glsnavigation}}%
11329     \renewcommand*\glsgroupheading[1]{%
11330       \item[\glslistgroupheaderfmt
11331         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11332       \mbox{}\par\nobreak\@afterheading
11333     }%
11334   }
11335 }
11336 {}

```

Similarly for altlistgroup.

```

11337 \ifdef{\@glsstyle@altlistgroup}
11338 {%
11339   \renewglossarystyle{altlistgroup}{%
11340     \setglossarystyle{altlist}%
11341     \renewcommand*\glsgroupheading[1]{%
11342       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11343       \mbox{}\par\nobreak\@afterheading
11344     }%
11345   }

```

```

11346 }
11347 {}

    Similarly for altlisthypergroup.
11348 \ifdef{\@glsstyle@altlisthypergroup}
11349 {%
11350   \renewglossarystyle{altlisthypergroup}{%
11351     \setglossarystyle{altlist}%
11352     \renewcommand*\{glossaryheader\}{%
11353       \glslistnavigationitem{\glsnavigation}}%
11354     \renewcommand*\{glsgroupheading\}[1]{%
11355       \item[\glslistgroupheaderfmt
11356         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11357       \mbox{}\par\nobreak\@afterheading
11358     }%
11359   }
11360 }
11361 {}

```

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`.

```

11362 \ifcsdef{@glsstyle@long}
11363 {%
11364   \renewglossarystyle{long}{%
11365     \renewenvironment{theglossary}%
11366       {\begin{longtable}{lp{\glsdescwidth}}}%
11367       {\end{longtable}}%
11368     \renewcommand*\{glossaryheader\}{}%
11369     \renewcommand*\{glsgroupheading\}[1]{}%
11370     \renewcommand{\glossentry}[2]{%
11371       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11372       \glossentrydesc{##1}\glspostdescription
11373       \glstrprelocation ##2\tabularnewline
11374     }%
11375     \renewcommand{\subglossentry}[3]{%
11376       &
11377       \glssubentryitem{##2}%
11378       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11379       \glstrprelocation ##3\tabularnewline
11380     }%
11381     \ifglsnogroupskip
11382       \renewcommand*\{glsgroupskip\}{}%
11383     \else
11384       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11385     \fi
11386   }

```

```
11387 }
11388 {}
```

Three column style:

```
11389 \ifcsdef{@glsstyle@long3col}
11390 {%
11391   \renewglossarystyle{long3col}{%
11392     \renewenvironment{theglossary}%
11393       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
11394       {\end{longtable}}}%
11395     \renewcommand*{\glossaryheader}{}%
11396     \renewcommand*{\glsgroupheading}[1]{}%
11397     \renewcommand{\glossentry}[2]{%
11398       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11399       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11400     }%
11401     \renewcommand{\subglossentry}[3]{%
11402       &
11403       \glssubentryitem{##2}%
11404       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11405       ##3\tabularnewline
11406     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11407   \ifglsnogroupskip
11408     \renewcommand*{\glsgroupskip}{}%
11409   \else
11410     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11411   \fi
11412 }
11413 }
11414 {}
```

Four column style:

```
11415 \ifcsdef{@glsstyle@long4col}
11416 {%
11417   \renewglossarystyle{long4col}{%
11418     \renewenvironment{theglossary}%
11419       {\begin{longtable}{llll}}}%
11420       {\end{longtable}}}%
11421     \renewcommand*{\glossaryheader}{}%
11422     \renewcommand*{\glsgroupheading}[1]{}%
11423     \renewcommand{\glossentry}[2]{%
11424       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11425       \glossentrydesc{##1}\glspostdescription &
11426       \glossentrysymbol{##1} &
11427       ##2\tabularnewline
11428     }%
11429     \renewcommand{\subglossentry}[3]{%
11430       &
```

```

11431     \glssubentryitem{##2}%
11432     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11433     \glossentrysymbol{##2} & ##3\tabularnewline
11434 }%

11435 \ifglsnogroupskip
11436   \renewcommand*\{glsgroupskip}{}%
11437 \else
11438   \renewcommand*\{glsgroupskip}{& & \tabularnewline}%
11439 \fi
11440 }
11441 }
11442 {}

```

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`.

```

11443 \ifcsdef{@glstyle@longragged}
11444 {%
11445   \renewglossarystyle{longragged}{%
11446     \renewenvironment{theglossary}%
11447       {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}]}%
11448       {\end{longtable}}%
11449     \renewcommand*\{glossaryheader}{}%
11450     \renewcommand*\{glsgroupheading}[1]{}%
11451     \renewcommand*\{glossentry}[2]{%
11452       \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11453       \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11454       \tabularnewline
11455     }%
11456     \renewcommand*\{subglossentry}[3]{%
11457       &
11458       \glssubentryitem{##2}%
11459       \glstarget{##2}{\strut}\glossentrydesc{##2}%
11460       \glspostdescription\glxtrprelocation ##3%
11461       \tabularnewline
11462     }%
11463     \ifglsnogroupskip
11464       \renewcommand*\{glsgroupskip}{}%
11465     \else
11466       \renewcommand*\{glsgroupskip}{& \tabularnewline}%
11467     \fi
11468   }
11469 }

```


11470 {}

Three and four column styles don't use \glstrprelocation since the number list is in its own column.

```
11471 \ifcsdef{@glsstyle@longragged3col}
11472 {%
11473   \renewglossarystyle{longragged3col}{%
11474     \renewenvironment{theglossary}%
11475       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11476         >{\raggedright}p{\glspagelistwidth}}}%
11477       {\end{longtable}}}%
11478     \renewcommand*{\glossaryheader}{}%
11479     \renewcommand*{\glsgroupheading}[1]{}%
11480     \renewcommand{\glossentry}[2]{%
11481       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11482       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11483     }%
11484     \renewcommand{\subglossentry}[3]{%
11485       &
11486       \glssubentryitem{##2}%
11487       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11488       ##3\tabularnewline
11489     }%
11490   \ifglsgroupskip
11491     \renewcommand*{\glsgroupskip}{}%
11492   \else
11493     \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11494   \fi
11495 }
11496 }
11497 {}
```

Four column style:

```
11498 \ifcsdef{@glsstyle@altlongragged4col}
11499 {%
11500   \renewglossarystyle{altlongragged4col}{%
11501     \renewenvironment{theglossary}%
11502       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11503         >{\raggedright}p{\glspagelistwidth}}}%
11504       {\end{longtable}}}%
11505     \renewcommand*{\glossaryheader}{}%
11506     \renewcommand*{\glsgroupheading}[1]{}%
11507     \renewcommand{\glossentry}[2]{%
11508       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11509       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11510       ##2\tabularnewline
11511     }%
11512     \renewcommand{\subglossentry}[3]{%
11513       &
```

```

11514      \glssubentryitem{##2}%
11515      \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11516      \glossentrysymbol{##2} & ##3\tabularnewline
11517  }%

11518  \ifglsnogroupskip
11519      \renewcommand*{\glsgroupskip}{}%
11520  \else
11521      \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11522  \fi
11523  }
11524  }
11525  {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glxtrprelocation`.

```

11526 \ifcsdef{@glstyle@super}
11527 {%
11528   \renewglossarystyle{super}{%
11529     \renewenvironment{theglossary}%
11530       {\tablehead{}\tabletail{}}%
11531       \begin{supertabular}{lp{\glsdescwidth}}%
11532       {\end{supertabular}}%
11533   \renewcommand*{\glossaryheader}{}%
11534   \renewcommand*{\glsgroupheading}[1]{}%
11535   \renewcommand{\glossentry}[2]{%
11536     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11537     \glossentrydesc{##1}\glspostdescription
11538     \glxtrprelocation ##2\tabularnewline
11539   }%
11540   \renewcommand{\subglossentry}[3]{%
11541     &
11542     \glssubentryitem{##2}%
11543     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11544     \glxtrprelocation ##3\tabularnewline
11545   }%
11546   \ifglsnogroupskip
11547       \renewcommand*{\glsgroupskip}{}%
11548   \else
11549       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11550   \fi
11551  }
11552  }
11553  {}

```

Three column style:

```

11554 \ifcsdef{@glstyle@super3col}

```

```

11555 {%
11556   \renewglossarystyle{super3col}{%
11557     \renewenvironment{theglossary}%
11558       {\tablehead{}\tabletail{}}%
11559       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
11560       {\end{supertabular}}%
11561     \renewcommand*{\glossaryheader}{}%
11562     \renewcommand*{\glsgroupheading}[1]{}%
11563     \renewcommand{\glossentry}[2]{%
11564       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11565       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11566     }%
11567     \renewcommand{\subglossentry}[3]{%
11568       &
11569       \glssubentryitem{##2}%
11570       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11571       ##3\tabularnewline
11572     }%

11573     \ifglsgnogroupskip
11574       \renewcommand*{\glsgroupskip}{}%
11575     \else
11576       \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11577     \fi
11578   }
11579 }
11580 {}

```

Four column styles:

```

11581 \ifcsdef{@glstyle@super4col}
11582 {%
11583   \renewglossarystyle{super4col}{%
11584     \renewenvironment{theglossary}%
11585       {\tablehead{}\tabletail{}}%
11586       \begin{supertabular}{llll}}{%
11587       \end{supertabular}}%
11588     \renewcommand*{\glossaryheader}{}%
11589     \renewcommand*{\glsgroupheading}[1]{}%
11590     \renewcommand{\glossentry}[2]{%
11591       \glstentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11592       \glossentrydesc{##1}\glspostdescription &
11593       \glossentrysymbol{##1} & ##2\tabularnewline
11594     }%
11595     \renewcommand{\subglossentry}[3]{%
11596       &
11597       \glssubentryitem{##2}%
11598       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11599       \glossentrysymbol{##2} & ##3\tabularnewline
11600     }%

```

```

11601 \ifglsgroupskip
11602 \renewcommand*{\glsgroupskip}{}%
11603 \else
11604 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11605 \fi
11606 }
11607 }
11608 {}

```

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`.

```

11609 \ifcsdef{@glstyle@superragged}
11610 {%
11611 \renewglossarystyle{superragged}{%
11612 \renewenvironment{theglossary}%
11613 {\tablehead{}}\tabletail{}}%
11614 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11615 {\end{supertabular}}%
11616 \renewcommand*{\glossaryheader}{}%
11617 \renewcommand*{\glsgroupheading}[1]{}%
11618 \renewcommand{\glossentry}[2]{%
11619 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11620 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11621 \tabularnewline
11622 }%
11623 \renewcommand{\subglossentry}[3]{%
11624 &
11625 \glssubentryitem{##2}%
11626 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11627 \glstrprelocation ##3%
11628 \tabularnewline
11629 }%
11630 \ifglsgroupskip
11631 \renewcommand*{\glsgroupskip}{}%
11632 \else
11633 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11634 \fi
11635 }
11636 }
11637 {}

```

Three column style:

```

11638 \ifcsdef{@glstyle@superragged3col}
11639 {%
11640 \renewglossarystyle{superragged3col}{%

```

```

11641 \renewenvironment{theglossary}%
11642   {\tablehead{}\tabletail{}}%
11643   \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
11644     >{\raggedright}p{\glspagelistwidth}}}%
11645   {\end{supertabular}}%
11646 \renewcommand*{\glossaryheader}{}%
11647 \renewcommand*{\glsgroupheading}[1]{}%
11648 \renewcommand{\glossentry}[2]{%
11649   \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11650   \glossentrydesc{##1}\glspostdescription &
11651   ##2\tabularnewline
11652 }%
11653 \renewcommand{\subglossentry}[3]{%
11654   &
11655   \glssubentryitem{##2}%
11656   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11657   ##3\tabularnewline
11658 }%
11659 \ifglsnogroupskip
11660   \renewcommand*{\glsgroupskip}{}%
11661 \else
11662   \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11663 \fi
11664 }
11665 }
11666 {}

```

Four columns:

```

11667 \ifcsdef{@glsstyle@altsuperragged4col}
11668 {%
11669   \renewglossarystyle{altsuperragged4col}{%
11670     \renewenvironment{theglossary}%
11671       {\tablehead{}\tabletail{}}%
11672       \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
11673         >{\raggedright}p{\glspagelistwidth}}}%
11674       {\end{supertabular}}%
11675     \renewcommand*{\glossaryheader}{}%
11676     \renewcommand{\glossentry}[2]{%
11677       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11678       \glossentrydesc{##1}\glspostdescription &
11679       \glossentrysymbol{##1} & ##2\tabularnewline
11680     }%
11681     \renewcommand{\subglossentry}[3]{%
11682       &
11683       \glssubentryitem{##2}%
11684       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11685       \glossentrysymbol{##2} & ##3\tabularnewline
11686     }%

```

```

11687 \ifglsnogroupskip
11688 \renewcommand*{\glsgroupskip}{}%
11689 \else
11690 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11691 \fi
11692 }
11693 }
11694 {}

```

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`.

```

11695 \ifdef{\@glsstyle@inline}
11696 {%
11697 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11698 \renewcommand*{\glsinlinedescformat}[3]{%
11699 \space#1\glxtrpostdescription}
11700 \renewcommand*{\glsinlinesubdescformat}[3]{%
11701 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11702 }
11703 {}

```

2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11704 \ifdef{\@glsstyle@index}
11705 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11706 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11707 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

```

```

11708 \renewglossarystyle{index}{%
11709 \renewenvironment{theglossary}%
11710 {\setlength{\parindent}{0pt}%
11711 \setlength{\parskip}{0pt plus 0.3pt}%
11712 \let\item\glstreeitem
11713 \let\subitem\glstreesubitem

```

```

11714     \let\subsubitem\glstreesubsubitem
11715 }%
11716 {\par}%
11717 \renewcommand*\{\glossaryheader}\{%
11718 \renewcommand*\{\glsgroupheading}[1]\{%
11719 \renewcommand*\{\glossentry}[2]\{%
11720     \item\glstentryitem{##1}%
11721     \glstreenamfmt{\glstarget{##1}\glossentryname{##1}}%
11722     \ifglshassymbol{##1}\space(\glossentrysymbol{##1})\}%
11723     \glstreepredesc \glossentrydesc{##1}\glspostdescription
11724     \glstreeprelocation ##2%
11725 }%
11726 \renewcommand{\subglossentry}[3]\{%
11727     \ifcase##1\relax
11728         \item
11729     \or
11730         \subitem
11731         \glssubentryitem{##2}%
11732     \else
11733         \subsubitem
11734     \fi
11735     \glstreenamfmt{\glstarget{##2}\glossentryname{##2}}%
11736     \ifglshassymbol{##2}\space(\glossentrysymbol{##2})\}%
11737     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11738     \glstreechildprelocation ##3%
11739 }%
11740 \renewcommand*\{\glsgroupskip}\{\ifglsnogroupskip\else\indexspace\fi}%
11741 }
11742 }
11743 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11744 \ifdef{\@glsstyle@indexgroup}
11745 {%
11746     \renewglossarystyle{indexgroup}\{%
11747         \setglossarystyle{index}%
11748         \renewcommand*\{\glsgroupheading}[1]\{%
11749             \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}%
11750             \nopagebreak\indexspace
11751             \nobreak\@afterheading
11752         }%
11753     }
11754 }
11755 {}

```

Similarly for indexhypergroup.

```

11756 \ifdef{\@glsstyle@indexhypergroup}
11757 {%
11758     \renewglossarystyle{indexhypergroup}\{%
11759         \setglossarystyle{index}%

```

```

11760 \renewcommand*{\glossaryheader}{%
11761     \item\glstreenavigationfmt{\glsnavigation}%
11762     \nobreak\@afterheading\indexspace}%
11763 \renewcommand*{\glsgroupheading}[1]{%
11764     \item\glstreegroupheaderfmt
11765     {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
11766     \nopagebreak\indexspace
11767     \nobreak\@afterheading}%
11768 }%
11769 }
11770 {}

```

Adjust tree style to remove hard coded space before number list.

```

11771 \ifdef{\@glsstyle@tree}
11772 {%
11773 \renewglossarystyle{tree}{%
11774 \renewenvironment{theglossary}%
11775     {\setlength{\parindent}{0pt}%
11776     \setlength{\parskip}{0pt plus 0.3pt}}%
11777     {}%
11778 \renewcommand*{\glossaryheader}{}%
11779 \renewcommand*{\glsgroupheading}[1]{}%
11780 \renewcommand{\glossentry}[2]{%
11781     \hangindent0pt\relax
11782     \parindent0pt\relax
11783     \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11784     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11785     \glstreepredesc\glossentrydesc{##1}\glspostdescription
11786     \glstreeprelocation##2\par
11787 }%
11788 \renewcommand{\subglossentry}[3]{%
11789     \hangindent##1\glstreeindent\relax
11790     \parindent##1\glstreeindent\relax
11791     \ifnum##1=1\relax
11792     \glssubentryitem{##2}%
11793     \fi
11794     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11795     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11796     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11797     \glstreechildprelocation ##3\par
11798 }%
11799 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11800 }%
11801 }
11802 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11803 \ifdef{\@glsstyle@treegroup}
11804 {%
11805 \renewglossarystyle{treegroup}{%

```



```

11806 \setglossarystyle{tree}%
11807 \renewcommand{\glsgroupheading}[1]{\par
11808 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11809 \nopagebreak\indexspace\nobreak\@afterheading}%
11810 }
11811 }
11812 {}

```

Similarly for treehypergroup

```

11813 \ifdef{\@glsstyle@treehypergroup}
11814 {%
11815 \renewglossarystyle{treehypergroup}{%
11816 \setglossarystyle{tree}%
11817 \renewcommand*\glossaryheader{%
11818 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11819 \nobreak\@afterheading\indexspace}%
11820 \renewcommand*\glsgroupheading}[1]{%
11821 \par\noindent
11822 \glstreegroupheaderfmt
11823 {\glssnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
11824 \nopagebreak\indexspace\nobreak\@afterheading}%
11825 }
11826 }
11827 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11828 \ifdef{\@glsstyle@treenoname}
11829 {%
11830 \renewglossarystyle{treenoname}{%
11831 \renewenvironment{theglossary}%
11832 {\setlength{\parindent}{0pt}%
11833 \setlength{\parskip}{0pt plus 0.3pt}}%
11834 {}%
11835 \renewcommand*\glossaryheader{}%
11836 \renewcommand*\glsgroupheading}[1]{}%
11837 \renewcommand{\glossentry}[2]{%
11838 \hangindent0pt\relax
11839 \parindent0pt\relax
11840 \glssentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11841 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11842 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11843 \glstreeprelocation##2\par
11844 }%
11845 \renewcommand{\subglossentry}[3]{%
11846 \hangindent##1\glstreeindent\relax
11847 \parindent##1\glstreeindent\relax
11848 \ifnum##1=1\relax
11849 \glssubentryitem{##2}%
11850 \fi
11851 \glstarget{##2}{\strut}%

```

```

11852     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11853 }%
11854 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11855 }
11856 }
11857 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11858 \ifdef{\@glsstyle@treenonamegroup}
11859 {%
11860   \renewglossarystyle{treenonamegroup}{%
11861     \setglossarystyle{treenoname}%
11862     \renewcommand{\glsgroupheading}[1]{\par
11863       \noindent\glstreegroupheaderfmt
11864       {\glsggetgrouptitle{##1}}}%
11865     \nopagebreak\indexspace\nobreak\@afterheading
11866   }%
11867 }
11868 }
11869 {}

```

Similarly for `treenonamehypergroup`

```

11870 \ifdef{\@glsstyle@treenonamehypergroup}
11871 {%
11872   \renewglossarystyle{treenonamehypergroup}{%
11873     \setglossarystyle{treenoname}%
11874     \renewcommand*{\glossaryheader}{%
11875       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11876       \nobreak\@afterheading\indexspace}%
11877     \renewcommand*{\glsgroupheading}[1]{%
11878       \par\noindent
11879       \glstreegroupheaderfmt
11880       {\glsnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
11881     \nopagebreak\indexspace\nobreak\@afterheading}%
11882   }
11883 }
11884 {}

```

The `almtree` style is redefined to make it easier to made minor adjustments.

```

11885 \ifdef{\@glsstyle@almtree}
11886 {%

```

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.

```

11887 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```

```

11888   {%
11889     \let\par\glxstrAltTreePar
11890     \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11891     \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11892   }%
11893 }

```

trAltTreeIndent Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11894 \newlength\glxstrAltTreeIndent

```

lsxtrAltTreePar Multi-paragraph descriptions need to keep the hanging indent.

```

11895 \newcommand{\glxstrAltTreePar}{%
11896   @@par
11897   \glxstrAltTreeSetHangIndent
11898   \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11899 }

```

symbolDescLocation `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11900 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11901   \glxstralttreeSymbolDescLocation{#2}{#3}%
11902 }

```

trreetopindent The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

11903 \newlength\glxstrtreetopindent

```

sxtralttreeInit User-level initialisation for the alttree style.

```

11904 \newcommand*{\glxstralttreeInit}{%
11905   \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
11906   \glxstrAltTreeIndent=\parindent
11907 }

```

\glsglissetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\gdef`.

```

11908 \newcommand*{\glsglissetwidest}[2][0]{%
11909   \csgdef{@glswidestname\romannumeral#1}{#2}%
11910 }

```

\eglssetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

11911 \newcommand*{\eglssetwidest}[2][0]{%
11912   \protected@csedef{@glswidestname\romannumeral#1}{#2}%
11913 }

```

`\xglsetwidest` Like the above but uses `\protected@csxdef`.

```

11914 \newcommand*\xglsetwidest}[2][0]{%
11915   \protected@csxdef{#1}{#2}%
11916 }

```

`glsupdatewidest` Only sets if new value is wider than old value.

```

11917 \newcommand*\glsupdatewidest}[2][0]{%
11918   \ifcsundef{#1}%
11919   {\csdef{#1}{#2}}%
11920   {%
11921     \settowidth{\dimen0}{\csuse{#1}}%
11922     \settowidth{\dimen@ii}{#2}%
11923     \ifdim\dimen@ii>\dimen0
11924       \csdef{#1}{#2}%
11925     \fi
11926   }%
11927 }

```

`glsupdatewidest` As above but global definition.

```

11928 \newcommand*\gglsupdatewidest}[2][0]{%
11929   \ifcsundef{#1}%
11930   {\csgdef{#1}{#2}}%
11931   {%
11932     \settowidth{\dimen0}{\csuse{#1}}%
11933     \settowidth{\dimen@ii}{#2}%
11934     \ifdim\dimen@ii>\dimen0
11935       \csgdef{#1}{#2}%
11936     \fi
11937   }%
11938 }

```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```

11939 \newcommand*\eglsupdatewidest}[2][0]{%
11940   \ifcsundef{#1}%
11941   {\protected@csedef{#1}{#2}}%
11942   {%
11943     \settowidth{\dimen0}{\csuse{#1}}%
11944     \settowidth{\dimen@ii}{#2}%
11945     \ifdim\dimen@ii>\dimen0
11946       \protected@csedef{#1}{#2}%
11947     \fi
11948   }%
11949 }

```

`glsupdatewidest` As above but global.

```

11950 \newcommand*\xglsupdatewidest}[2][0]{%
11951   \ifcsundef{#1}%
11952   {\protected@csxdef{#1}{#2}}%
11953   {%

```

```

11954      \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
11955      \settowidth{\dimen@ii}{#2}%
11956      \ifdim\dimen@ii>\dimen@
11957      \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
11958      \fi
11959  }%
11960  }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

11961  \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

11962  \newcommand*{\glswidestsubname}[1]{%
11963    \ifcsundef{@glswidestname\romannumeral#1}%
11964    {\@glswidestname}%
11965    {\csuse{@glswidestname\romannumeral#1}}%
11966  }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

11967  \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.

```

11968  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
11969    \dimen@=0pt\relax
11970    \glstmpplen=0pt\relax
11971    \foralllglossaries[#1]{\@gls@type}%
11972    {%
11973      \forglsentries[\@gls@type]{\@glo@label}%
11974      {%
11975        \ifglswidest{\@glo@label}%
11976        {%
11977          \ifglshasparent{\@glo@label}%
11978          {}%
11979          {%
11980            \settowidth{\dimen@}{%
11981              {\glstreenamfmt{\glstryname{\@glo@label}}}%
11982              \ifdim\dimen@>\glstmpplen
11983              \glstmpplen=\dimen@
11984              \eglssetwidest{\glstryname{\@glo@label}}%
11985            }%
11986          }%
11987        }%
11988      }%
11989    }%
11990  }%
11991  }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11992 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11993 \dimen@=0pt\relax
11994 \gls@tmplen=0pt\relax
11995 \foralllglossaries[#1]{\@gls@type}%
11996 {%
11997 \forallglsentries[\@gls@type]{\@glo@label}%
11998 {%
11999 \ifglsused{\@glo@label}%
12000 {%
12001 \settowidth{\dimen@}%
12002 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12003 \ifdim\dimen@>\gls@tmplen
12004 \gls@tmplen=\dimen@
12005 \eglssetwidest{\glsentryname{\@glo@label}}%
12006 \fi
12007 }%
12008 }%
12009 }%
12010 }%
12011 }

```

`ndWidestAnyName` Like the above but doesn't check is the entry has been used.

```

12012 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
12013 \dimen@=0pt\relax
12014 \gls@tmplen=0pt\relax
12015 \foralllglossaries[#1]{\@gls@type}%
12016 {%
12017 \forallglsentries[\@gls@type]{\@glo@label}%
12018 {%
12019 \settowidth{\dimen@}%
12020 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12021 \ifdim\dimen@>\gls@tmplen
12022 \gls@tmplen=\dimen@
12023 \eglssetwidest{\glsentryname{\@glo@label}}%
12024 \fi
12025 }%
12026 }%
12027 }

```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

12028 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
12029 \dimen@=0pt\relax
12030 \dimen@i=0pt\relax
12031 \dimen@ii=0pt\relax
12032 \foralllglossaries[#1]{\@gls@type}%
12033 {%

```

```

12034 \forglsentries[\@gls@type]{\@glo@label}%
12035 {%
12036 \ifglsused{\@glo@label}%
12037 {%
12038 \ifglsashasparent{\@glo@label}%
12039 {%
12040 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
12041 \ifglsashasparent{\@glo@parent}%
12042 {%
12043 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
12044 \ifglsashasparent{\@glo@parent}%
12045 {}%
12046 {%
12047 \settowidth{\gls@tmplen}%
12048 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12049 \ifdim\gls@tmplen>\dimen@ii
12050 \dimen@ii=\gls@tmplen
12051 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
12052 \fi
12053 }%
12054 }%
12055 {%
12056 \settowidth{\gls@tmplen}%
12057 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12058 \ifdim\gls@tmplen>\dimen@i
12059 \dimen@i=\gls@tmplen
12060 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
12061 \fi
12062 }%
12063 }%
12064 {%
12065 \settowidth{\gls@tmplen}%
12066 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12067 \ifdim\gls@tmplen>\dimen@
12068 \dimen@=\gls@tmplen
12069 \eglssetwidest{\glsentryname{\@glo@label}}%
12070 \fi
12071 }%
12072 }%
12073 {}%
12074 }%
12075 }%
12076 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

12077 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
12078 \dimen@=0pt\relax
12079 \dimen@i=0pt\relax
12080 \dimen@ii=0pt\relax

```

```

12081 \forall glossaries[#1]{\@gls@type}%
12082 {%
12083   \forall sentries[\@gls@type]{\@glo@label}%
12084   {%
12085     \ifglshasparent{\@glo@label}%
12086     {%
12087       \edef\@glo@parent{\csuse{glo@\glsetoklabel{\@glo@label}@parent}}%
12088       \ifglshasparent{\@glo@parent}%
12089       {%
12090         \edef\@glo@parent{\csuse{glo@\glsetoklabel{\@glo@parent}@parent}}%
12091         \ifglshasparent{\@glo@parent}%
12092         {}%
12093         {%
12094           \settowidth{\gls@tmplen}%
12095             {\glstreenamefmt{\glstentryname{\@glo@label}}}%
12096           \ifdim\gls@tmplen>\dimen@ii
12097             \dimen@ii=\gls@tmplen
12098             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
12099           \fi
12100         }%
12101       }%
12102     {%
12103       \settowidth{\gls@tmplen}%
12104         {\glstreenamefmt{\glstentryname{\@glo@label}}}%
12105       \ifdim\gls@tmplen>\dimen@i
12106         \dimen@i=\gls@tmplen
12107         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
12108       \fi
12109     }%
12110   }%
12111   {%
12112     \settowidth{\gls@tmplen}%
12113       {\glstreenamefmt{\glstentryname{\@glo@label}}}%
12114     \ifdim\gls@tmplen>\dimen@
12115       \dimen@=\gls@tmplen
12116       \eglssetwidest{\glstentryname{\@glo@label}}%
12117     \fi
12118   }%
12119 }%
12120 }%
12121 }

```

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.

```

12122 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
12123   \dimen@=0pt\relax
12124   \gls@tmplen=0pt\relax
12125   #2=0pt\relax
12126   \forall glossaries[#1]{\@gls@type}%

```



```

12127  {%
12128    \forglsentries[\@gls@type]{\@glo@label}%
12129  {%
12130    \ifglsused{\@glo@label}%
12131  {%
12132    \settowidth{\dimen@}%
12133      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12134    \ifdim\dimen@>\gls@tmplen
12135      \gls@tmplen=\dimen@
12136      \eglssetwidest{\glsentryname{\@glo@label}}%
12137    \fi
12138    \settowidth{\dimen@}%
12139      {\glsentrysymbol{\@glo@label}}%
12140    \ifdim\dimen@>#2\relax
12141      #2=\dimen@
12142    \fi
12143  }%
12144  }%
12145 }%
12146 }%
12147 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

12148 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
12149   \dimen@=0pt\relax
12150   \gls@tmplen=0pt\relax
12151   #2=0pt\relax
12152   \forallglossaries[#1]{\@gls@type}%
12153   {%
12154     \forglsentries[\@gls@type]{\@glo@label}%
12155   {%
12156     \settowidth{\dimen@}%
12157       {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12158     \ifdim\dimen@>\gls@tmplen
12159       \gls@tmplen=\dimen@
12160       \eglssetwidest{\glsentryname{\@glo@label}}%
12161     \fi
12162     \settowidth{\dimen@}%
12163       {\glsentrysymbol{\@glo@label}}%
12164     \ifdim\dimen@>#2\relax
12165       #2=\dimen@
12166     \fi
12167   }%
12168 }%
12169 }

```

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.

```

12170 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
12171   \dimen@=0pt\relax
12172   \gls@tmplen=0pt\relax
12173   #2=0pt\relax
12174   #3=0pt\relax
12175   \foralllglossaries[#1]{\@gls@type}%
12176   {%
12177     \forglsentries[\@gls@type]{\@glo@label}%
12178     {%
12179       \ifglsused{\@glo@label}%
12180       {%
12181         \settowidth{\dimen@}%
12182         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12183         \ifdim\dimen@>\gls@tmplen
12184           \gls@tmplen=\dimen@
12185           \eglssetwidest{\glsentryname{\@glo@label}}%
12186         \fi
12187         \settowidth{\dimen@}%
12188         {\glsentrysymbol{\@glo@label}}%
12189         \ifdim\dimen@>#2\relax
12190           #2=\dimen@
12191         \fi
12192         \settowidth{\dimen@}%
12193         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12194         \ifdim\dimen@>#3\relax
12195           #3=\dimen@
12196         \fi
12197       }%
12198     }%
12199   }%
12200 }%
12201 }

```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

12202 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
12203   \dimen@=0pt\relax
12204   \gls@tmplen=0pt\relax
12205   #2=0pt\relax
12206   #3=0pt\relax
12207   \foralllglossaries[#1]{\@gls@type}%
12208   {%
12209     \forglsentries[\@gls@type]{\@glo@label}%
12210     {%
12211       \settowidth{\dimen@}%
12212       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12213       \ifdim\dimen@>\gls@tmplen
12214         \gls@tmplen=\dimen@
12215       \eglssetwidest{\glsentryname{\@glo@label}}%

```

```

12216      \fi
12217      \settowidth{\dimen@}%
12218      {\glstrysymbol{\@glo@label}}}%
12219      \ifdim\dimen@>#2\relax
12220      #2=\dimen@
12221      \fi
12222      \settowidth{\dimen@}%
12223      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12224      \ifdim\dimen@>#3\relax
12225      #3=\dimen@
12226      \fi
12227    }%
12228  }%
12229 }

```

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.

```

12230 \newrobustcmd*{\glstryFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
12231   \dimen@=0pt\relax
12232   \glstry@tmplen=0pt\relax
12233   #2=0pt\relax
12234   \forallglossaries[#1]{\@gls@type}%
12235   {%
12236     \forallglsentries[\@gls@type]{\@glo@label}%
12237     {%
12238       \ifglstryused{\@glo@label}%
12239       {%
12240         \settowidth{\dimen@}%
12241         {\glstrynamefmt{\glstryname{\@glo@label}}}%
12242         \ifdim\dimen@>\glstry@tmplen
12243         \glstry@tmplen=\dimen@
12244         \eglstrysetwidest{\glstryname{\@glo@label}}%
12245         \fi
12246         \settowidth{\dimen@}%
12247         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12248         \ifdim\dimen@>#2\relax
12249         #2=\dimen@
12250         \fi
12251       }%
12252     }%
12253   }%
12254 }%
12255 }

```

AnyNameLocation Like the `\glstryFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

12256 \newrobustcmd*{\glstryFindWidestAnyNameLocation}[2][\@glo@types]{%
12257   \dimen@=0pt\relax
12258   \glstry@tmplen=0pt\relax

```

```

12259      #2=Opt\relax
12260      \forallglossaries[#1]{\@gls@type}%
12261      {%
12262        \forglsentries[\@gls@type]{\@glo@label}%
12263        {%
12264          \settowidth{\dimen@}%
12265            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12266          \ifdim\dimen@>\gls@tmplen
12267            \gls@tmplen=\dimen@
12268            \eglssetwidest{\glsentryname{\@glo@label}}%
12269          \fi
12270          \settowidth{\dimen@}%
12271            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12272          \ifdim\dimen@>#2\relax
12273            #2=\dimen@
12274          \fi
12275        }%
12276      }%
12277    }

```

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`.

```

12278    \newcommand*{\glsxtrComputeTreeIndent}[1]{%
12279      \glstreeindent=\glsxtrtreetopindent\relax
12280    }

```

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.

```

12281    \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
12282      \ifcsundef{@glswidestname\romannumeral#1}%
12283      {%
12284        \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
12285      }%
12286      {%
12287        \settowidth{#3}{\glstreenamefmt{%
12288          \csname @glswidestname\romannumeral#1\endcsname\space}}%
12289      }%
12290    }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

12291    \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
12292 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine altree:

```
12293 \renewglossarystyle{alttree}{%
12294   \renewenvironment{theglossary}%
12295     {%
12296       \glxtralmtreeInit
12297       \def\@gls@prevlevel{-1}%
12298       \mbox{}\par}%
12299     {\par}%
12300   \renewcommand*{\glossaryheader}{}%
12301   \renewcommand*{\glsgroupheading}[1]{}%
12302   \renewcommand{\glossentry}[2]{%
12303     \ifnum\@gls@prevlevel=0\relax
12304     \else
12305       \glxtrComputeTreeIndent{##1}%
12306     \fi
12307     \parindent\glstreeindent
12308     \glxtrAltTreeSetHangIndent
12309     \makebox[0pt][r]%
12310     {%
12311       \glstreenamebox{\glstreeindent}%
12312       {%
12313         \glssentryitem{##1}%
12314         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
12315       }%
12316     }%
12317     \glxtralmtreeSymbolDescLocation{##1}{##2}%
12318     \def\@gls@prevlevel{0}%
12319   }
12320   \renewcommand{\subglossentry}[3]{%
12321     \ifnum##1=1\relax
12322       \glssubentryitem{##2}%
12323     \fi
12324     \ifnum\@gls@prevlevel=##1\relax
12325     \else
12326       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12327       \ifnum\@gls@prevlevel<##1\relax
12328         \setlength\glstreeindent\gls@tmplen
12329         \addtolength\glstreeindent\parindent
12330         \parindent\glstreeindent
12331       \else
12332         \ifnum\@gls@prevlevel=0\relax
12333           \glxtrComputeTreeIndent{##2}%
12334         \else
12335           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12336         \fi
12337         \addtolength\parindent{-\glstreeindent}%

```

```

12338         \setlength\glstreeindent\parindent
12339     \fi
12340 \fi
12341 \glxtrAltTreeSetSubHangIndent{##1}%
12342 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
12343     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12344 \glxtralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12345 \def\@gls@prevlevel{##1}%
12346 }%
12347 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12348 }
12349 }%
12350 {%
12351 }

```

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.

```

12352 \ifdef{\@glsstyle@almtreegroup}
12353 {%
12354     \renewglossarystyle{almtreegroup}{%
12355         \setglossarystyle{almtree}%
12356         \renewcommand{\glsgroupheading}[1]{\par
12357             \def\@gls@prevlevel{-1}%
12358             \hangindentOpt\relax
12359             \parindentOpt\relax
12360             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}%
12361             \nopagebreak\indexspace\nopagebreak
12362         }%
12363     }%
12364 }%
12365 {%
12366 }

```

Similarly for `almtreehypergroup`.

```

12367 \ifdef{\@glsstyle@almtreehypergroup}
12368 {%
12369     \renewglossarystyle{almtreehypergroup}{%
12370         \setglossarystyle{almtree}%
12371         \renewcommand*\{\glossaryheader}{%
12372             \par
12373             \def\@gls@prevlevel{-1}%
12374             \hangindentOpt\relax
12375             \parindentOpt\relax
12376             \glstreenavigationfmt{\glsnavigation}\par\indexspace
12377         }%
12378         \renewcommand*\{\glsgroupheading}[1]{%
12379             \par
12380             \def\@gls@prevlevel{-1}%
12381             \hangindentOpt\relax
12382             \parindentOpt\relax

```

```

12383     \glstreegroupheaderfmt
12384     {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12385     \nopagebreak\indexspace\nopagebreak
12386   }%
12387 }
12388 }%
12389 {%
12390 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12391 \ifdef{\@glsstyle@mcolindexgroup}
12392 {%
12393   \renewglossarystyle{mcolindexgroup}{%
12394     \setglossarystyle{mcolindex}%
12395     \renewcommand*{\glsgroupheading}[1]{%
12396       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12397       \nopagebreak\indexspace\nobreak\@afterheading
12398     }%
12399   }
12400 }%
12401 {%
12402 }

```

Similarly for mcolindexhypergroup.

```

12403 \ifdef{\@glsstyle@mcolindexhypergroup}
12404 {%
12405   \renewglossarystyle{mcolindexhypergroup}{%
12406     \setglossarystyle{mcolindex}%
12407     \renewcommand*{\glossaryheader}{%
12408       \item\glstreenavigationfmt{\glsnavigation}%
12409       \indexspace
12410     }%
12411     \renewcommand*{\glsgroupheading}[1]{%
12412       \item\glstreegroupheaderfmt
12413       {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12414       \nopagebreak\indexspace\nobreak\@afterheading
12415     }%
12416   }
12417 }%
12418 {%
12419 }

```

Similarly for mcolindexspannav.

```

12420 \ifdef{\@glsstyle@mcolindexspannav}
12421 {%
12422   \renewglossarystyle{mcolindexspannav}{%
12423     \setglossarystyle{index}%

```

```

12424 \renewenvironment{theglossary}%
12425 {%
12426 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12427 \setlength{\parindent}{0pt}%
12428 \setlength{\parskip}{0pt plus 0.3pt}%
12429 \let\item\glstreeitem}%
12430 {\end{multicols}}}%
12431 \renewcommand*{\glsgroupheading}[1]{%
12432 \item\glstreegroupheaderfmt
12433 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12434 \nopagebreak\indexspace\nobreak\@afterheading
12435 }%
12436 }
12437 }%
12438 {%
12439 }

```

Similarly for mcoltreegroup.

```

12440 \ifdef{\@glstyle@mcoltreegroup}
12441 {%
12442 \renewglossarystyle{mcoltreegroup}{%
12443 \setglossarystyle{mcoltree}%
12444 \renewcommand{\glsgroupheading}[1]{\par
12445 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12446 \nopagebreak\indexspace\nobreak\@afterheading
12447 }%
12448 }
12449 }%
12450 {%
12451 }

```

Similarly for mcoltreehypergroup.

```

12452 \ifdef{\@glstyle@mcoltreehypergroup}
12453 {%
12454 \renewglossarystyle{mcoltreehypergroup}{%
12455 \setglossarystyle{mcoltree}%
12456 \renewcommand*{\glossaryheader}{%
12457 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12458 }%
12459 \renewcommand*{\glsgroupheading}[1]{%
12460 \par\noindent
12461 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12462 \nopagebreak\indexspace\nobreak\@afterheading
12463 }%
12464 }
12465 }%
12466 {%
12467 }

```

Similarly for mcoltreespannav.

```

12468 \ifdef{\@glstyle@mcoltreespannav}

```



```

12469 {%
12470   \renewglossarystyle{mcoltreespannav}{%
12471     \setglossarystyle{tree}%
12472     \renewenvironment{theglossary}%
12473     {%
12474       \begin{multicols}{\glsmcols}%
12475       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12476       \setlength{\parindent}{0pt}%
12477       \setlength{\parskip}{0pt plus 0.3pt}%
12478     }%
12479     {\end{multicols}}}%
12480   \renewcommand*{\glsgroupheading}[1]{%
12481     \par\noindent
12482     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12483     \nopagebreak\indexspace\nobreak\@afterheading
12484   }%
12485 }
12486 }%
12487 {%
12488 }

```

Similarly for mcoltreenonamegroup.

```

12489 \ifdef{\@glsstyle@mcoltreenonamegroup}
12490 {%
12491   \renewglossarystyle{mcoltreenonamegroup}{%
12492     \setglossarystyle{mcoltreenoname}%
12493     \renewcommand*{\glsgroupheading}[1]{\par
12494       \noindent\glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%
12495     \nopagebreak\indexspace\nobreak\@afterheading
12496   }%
12497 }
12498 }%
12499 {%
12500 }

```

Similarly for mcoltreenonamehypergroup.

```

12501 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12502 {%
12503   \renewglossarystyle{mcoltreenonamehypergroup}{%
12504     \setglossarystyle{mcoltreenoname}%
12505     \renewcommand*{\glossaryheader}{%
12506       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12507     \renewcommand*{\glsgroupheading}[1]{%
12508       \par\noindent
12509       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12510       \nopagebreak\indexspace\nobreak\@afterheading}%
12511   }
12512 }%
12513 {%
12514 }

```

Similarly for mcoltreenonamespannav.

```

12515 \ifdef{\@glsstyle@mcoltreenonamespannav}
12516 {%
12517   \renewglossarystyle{mcoltreenonamespannav}{%
12518     \setglossarystyle{treenoname}%
12519     \renewenvironment{theglossary}%
12520     {%
12521       \begin{multicols}{\glsmcols}%
12522       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12523       \setlength{\parindent}{0pt}%
12524       \setlength{\parskip}{0pt plus 0.3pt}%
12525     }%
12526     {\end{multicols}}}%
12527   \renewcommand*{\glsgroupheading}[1]{%
12528     \par\noindent
12529     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12530     \nopagebreak\indexspace\nobreak\@afterheading}%
12531   }
12532 }%
12533 {%
12534 }

```

mcolalmtree needs adjusting so that it uses \glxtralttreeInit This doesn't use \mbox{} \par which would unbalance the top of the columns.

```

12535 \ifdef{\@glsstyle@mcolalmtree}
12536 {%
12537   \renewglossarystyle{mcolalmtree}{%
12538     \setglossarystyle{almtree}%
12539     \renewenvironment{theglossary}%
12540     {%
12541       \glxtralttreeInit
12542       \def\@gls@prevlevel{-1}%
12543       \begin{multicols}{\glsmcols}%
12544     }%
12545     {\par\end{multicols}}}%
12546   }
12547 }%
12548 {%
12549 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12550 \ifdef{\@glsstyle@mcolalmtreegroup}
12551 {%
12552   \renewglossarystyle{mcolalmtreegroup}{%
12553     \setglossarystyle{mcolalmtree}%
12554     \renewcommand{\glsgroupheading}[1]{\par
12555       \def\@gls@prevlevel{-1}%
12556       \hangindent0pt\relax
12557       \parindent0pt\relax
12558       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

12559     \nopagebreak\indexspace\nopagebreak
12560   }%
12561 }
12562 }%
12563 {%
12564 }

```

Similarly for mcolalmtreehypergroup.

```

12565 \ifdef{\@glsstyle@mcolalmtreehypergroup}
12566 {%
12567   \renewglossarystyle{mcolalmtreehypergroup}{%
12568     \setglossarystyle{mcolalmtree}%
12569     \renewcommand*{\glossaryheader}{%
12570       \par
12571       \def\@gls@prevlevel{-1}%
12572       \hangindent0pt\relax
12573       \parindent0pt\relax
12574       \glstreenavigationfmt{\glsnavigation}%
12575       \par\indexspace
12576     }%
12577     \renewcommand*{\glsgroupheading}[1]{%
12578       \par
12579       \def\@gls@prevlevel{-1}%
12580       \hangindent0pt\relax
12581       \parindent0pt\relax
12582       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12583       \nopagebreak\indexspace\nopagebreak
12584     }%
12585   }
12586 }%
12587 {%
12588 }

```

Similarly for mcolalmtreespannav.

```

12589 \ifdef{\@glsstyle@mcolalmtreespannav}
12590 {%
12591   \renewglossarystyle{mcolalmtreespannav}{%
12592     \setglossarystyle{almtree}%
12593     \renewenvironment{theglossary}%
12594     {%
12595       \glsextralmtreeInit
12596       \def\@gls@prevlevel{-1}%
12597       \begin{multicols}{\glsncols}%
12598         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12599     }%
12600     {\par\end{multicols}}%
12601     \renewcommand*{\glsgroupheading}[1]{%
12602       \par
12603       \def\@gls@prevlevel{-1}%
12604       \hangindent0pt\relax

```

```

12605      \parindent0pt\relax
12606      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgroupptitle{##1}}}%
12607      \nopagebreak\indexspace\nopagebreak
12608    }%
12609  }
12610}%
12611{%
12612}

  Reset the default style
12613\ifx\@glossary@default@style\relax
12614\else
12615  \setglossarystyle{\@glxtr@current@style}
12616\fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
12617 \NeedsTeXFormat{LaTeX2e}
12618 \ProvidesPackage{glossary-bookindex}[2018/01/05 v1.26 (NLCT)]
```

Load required packages.

```
12619 \RequirePackage{multicol}
12620 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12621 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12622 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`bookindexsubname` Format used for sub entries.

```
12623 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`glstrprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12624 \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.

```
12625 \newcommand*{\glstrbookindexprelocation}[1]{%
12626   \glstrifhasfield{location}{#1}%
12627   {,\glstrprelocation}%
12628   {\glstrprelocation}%
12629 }
```

`glstrsubprelocation` Separator used before location list for sub-entries.

```
12630 \newcommand*{\glstrbookindexsubprelocation}[1]{%
12631   \glstrbookindexprelocation{#1}%
12632 }
```

`glstrparentchildsep` Separator used between top-level parent and child entry.

```
12633 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`glstrparentsubchildsep` Separator used between sub-level parent and child entry.

```
12634 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

bookindexbetween Between two top-level entries identified by the labels in the arguments.

```
12635 \newcommand{\glstrbookindexbetween}[2]{}
```

indexsubbetween Between two level 1 entries identified by the labels in the arguments.

```
12636 \newcommand{\glstrbookindexsubbetween}[2]{}
```

exsubsubbetween Between two level 2 entries identified by the labels in the arguments.

```
12637 \newcommand{\glstrbookindexsubsubbetween}[2]{}
```

indexatendgroup At the end of a letter group. The argument is the index of the last top-level entry.

```
12638 \newcommand{\glstrbookindexatendgroup}[1]{}
```

exsubatendgroup At the end of a letter group. The argument is the index of the last level 1 entry.

```
12639 \newcommand{\glstrbookindexsubatendgroup}[1]{}
```

subsubatendgroup At the end of a letter group. The argument is the index of the last level 2 entry.

```
12640 \newcommand{\glstrbookindexsubsubatendgroup}[1]{}
```

kindexgroupskip Group separator.

```
12641 \newcommand{\glstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

indexformatheader Group separator.

```
12642 \newcommand*{\glstrbookindexformatheader}[1]{%
12643   \par{\centering\glstreegroupheaderfmt{#1}\par}%
12644 }
```

bookindexbookmark Book mark group heading if supported.

```
12645 \ifdef\pdfbookmark
12646 {%
12647   \newcommand*{\glstrbookindexbookmark}[2]{%
12648     \ifdefstring{\@@glossarysec}{chapter}%
12649     {\pdfbookmark[1]{#1}{#2}}%
12650     {\pdfbookmark[2]{#1}{#2}}%
12651   }
12652 }
12653 {%
12654   \newcommand*{\glstrbookindexbookmark}[2]{%
12655 }
```

kindexcolspread

```
12656 \newcommand*{\glstrbookindexcolspread}{}%
```

indexmulticolenv

```
12657 \newcommand*{\glstrbookindexmulticolenv}{multicols}
```

Define the style.

```

12658 \newglossarystyle{bookindex}{%
12659   \setglossarystyle{index}%
12660   \renewenvironment{theglossary}%
12661   {%
12662     \ifdefempty{glxstrbookindexcolspread}
12663     {%
12664       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
12665       {\glxstrbookindexcols}%
12666     }%
12667     {%
12668       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
12669       {\glxstrbookindexcols}[\glxstrbookindexcolspread]%
12670     }%
12671     \setlength{\parindent}{0pt}%
12672     \setlength{\parskip}{0pt plus 0.3pt}%
12673     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12674     \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
12675     \let\@glxstr@bookindex@between\@gobble
12676     \let\@glxstr@bookindex@subbetween\@gobble
12677     \let\@glxstr@bookindex@subsubbetween\@gobble
12678     \let\@glxstr@bookindex@atendgroup\relax
12679     \let\@glxstr@bookindex@subatendgroup\relax
12680     \let\@glxstr@bookindex@subsubatendgroup\relax
12681     \let\@glxstr@bookindex@groupskip\relax
12682   }%
12683   {%

```

Do end group hooks.

```

12684     \@glxstr@bookindex@subsubatendgroup
12685     \@glxstr@bookindex@subatendgroup
12686     \@glxstr@bookindex@atendgroup

```

End multicol environment.

```

12687     \expandafter\end\expandafter{\glxstrbookindexmulticolseenv}%
12688   }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12689   \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12690   \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12691     \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12692     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12693     \let\@glxstr@bookindex@subsep\glxstrbookindexparentschildsep
12694     \let\@glxstr@bookindex@subbetween\@gobble
12695     \let\@glxstr@bookindex@subsubbetween\@gobble
12696     \edef\@glxstr@bookindex@between{%

```

```

12697      \noexpand\glxstrbookindexbetween{##1}%
12698  }%
12699  \edef\@glxstr@bookindex@atendgroup{%
12700      \noexpand\glxstrbookindexatendgroup{##1}%
12701  }%
12702  \let\@glxstr@bookindex@subatendgroup\relax
12703  \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12704  \glstreeitem
12705      \glstryitem{##1}%
12706      \glstarget{##1}{\glxstrbookindexname{##1}}%
12707  \glxstrbookindexprelocation{##1}##2%
12708  }%
12709  \renewcommand{\subglossentry}[3]{%
12710      \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12711      \glstreeitem
12712  \or

```

Level 1.

```

12713      \@glxstr@bookindex@sep
12714      \@glxstr@bookindex@subbetween{##2}%
12715      \let\@glxstr@bookindex@sep\relax

```

Update separators.

```

12716      \let\@glxstr@bookindex@subsubbetween\@gobble
12717      \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12718      \edef\@glxstr@bookindex@subbetween{%
12719          \noexpand\glxstrbookindexsubbetween{##2}%
12720      }%
12721      \edef\@glxstr@bookindex@atsubendgroup{%
12722          \noexpand\glxstrbookindexatsubendgroup{##1}%
12723      }%

```

Start sub-item.

```

12724      \glstreesubitem
12725      \glssubentryitem{##2}%
12726  \else

```

All other levels.

```

12727      \@glxstr@bookindex@subsep
12728      \@glxstr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12729      \let\@glxstr@bookindex@subsep\relax
12730      \edef\@glxstr@bookindex@subsubbetween{%
12731          \noexpand\glxstrbookindexsubsubbetween{##2}%
12732      }%
12733      \edef\@glxstr@bookindex@atsubsubendgroup{%
12734          \noexpand\glxstrbookindexatsubsubendgroup{##1}%
12735      }%

```


Start sub-sub-item.

```
12736      \glstreesubsubitem
12737      \fi
```

Format entry.

```
12738      \glstarget{##2}{\glxtrbookindexsubname{##2}}%
12739      \glxtrbookindexsubprelocation{##2}##3%
12740  }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12741  \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12742  \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12743      \@glxtr@bookindex@subsubatendgroup
12744      \@glxtr@bookindex@subatendgroup
12745      \@glxtr@bookindex@atendgroup
12746      \@glxtr@bookindex@groupskip
```

Update separators.

```
12747      \let\@glxtr@bookindex@groupskip\glxtrbookindexgroupskip
12748      \let\@glxtr@bookindex@between\@gobble
12749      \let\@glxtr@bookindex@atendgroup\relax
12750      \let\@glxtr@bookindex@subatendgroup\relax
12751      \let\@glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12752      \glxtrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12753      \glxtrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12754      \glxtrbookindexformatheader{\thisgrptitle}%
12755      \nopagebreak\indexspace\nopagebreak\@afterheading
12756  }%
12757 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glxtrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`\glxtrbookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
12758 \newcommand{\glxtrbookindexthepage}{%
12759 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
12760 }
```

bookindexmarkentry Writes entry information to the .aux file. The argument is the entry label.

```
12761 \newcommand*{\glxtrbookindexmarkentry}[1]{%
12762   \protected@write\@auxout
12763   {\let\glxtrbookindexthepage\relax}%
12764   {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
12765 }
```

etbookindexmark

```
12766 \newcommand*{\glxtr@setbookindexmark}[2]{%
12767   \ifcsundef{glxtr@idxfirstmark@#1}%
12768   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
12769   {}%
12770   \csgdef{glxtr@idxlastmark@#1}{#2}%
12771 }
```

indexfirstmarkfmt

```
12772 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
12773   \glstryname{#1}%
12774 }
```

indexfirstmark

```
12775 \newcommand*{\glxtrbookindexfirstmark}{%
12776   \letcs{glxtr@label}{glxtr@idxfirstmark@\glxtrbookindexthepage}%
12777   \ifdefglxtr@label
12778   {\glxtrbookindexfirstmarkfmt{\glxtr@label}}%
12779   {}%
12780 }
```

indexlastmarkfmt

```
12781 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
12782   \glstryname{#1}%
12783 }
```

okindexlastmark

```
12784 \newcommand*{\glxtrbookindexlastmark}{%
12785   \letcs{glxtr@label}{glxtr@idxlastmark@\glxtrbookindexthepage}%
12786   \ifdefglxtr@label
12787   {\glxtrbookindexlastmarkfmt{\glxtr@label}}%
12788   {}%
12789 }
```

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)	
General: Initial experimental release	5
0.2 (2015-11-30)	
\Glsfmtshort: new	306
\glsfmtshort: new	306
\Glsfmtshortpl: new	306
\glsfmtshortpl: new	306
short: switched inline full form to short (long)	209
0.3 (2015-12-02)	
\@ACRlong: added redefinition	71
\@ACRlongpl: added redefinition	72
\@ACRshort: added redefinition	69
\@ACRshortpl: added redefinition	70
\@Acrlong: added redefinition	70
\@Acrlongpl: added redefinition	71
\@Acrshort: added redefinition	68
\@Acrshortpl: added redefinition	69
\@GLSdesc: added redefinition	64
\@GLSdescplural@: added redefinition	65
\@GLSfirst@: added redefinition	62
\@GLSfirstplural@: added redefinition	63
\@GLSname@: added redefinition	64
\@GLSplural@: added redefinition	62
\@GLSSymbol@: added redefinition	65
\@GLSSymbolplural@: added redefinition	66
\@GLStext@: added redefinition	61
\@GLSuseri@: added redefinition	66
\@GLSuserii@: added redefinition	66
\@GLSuseriii@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@GLSuserv@: added redefinition	67
\@GLSuservi@: added redefinition	68
\@GLSdesc@: added redefinition	64
\@GLSdescplural@: added redefinition	64
\@GLSfirst@: added redefinition	62
\@GLSfirstplural@: added redefinition	63
\@GLSname@: added redefinition	64
\@GLSplural@: added redefinition	62
\@GLSSymbol@: added redefinition	65
\@GLSSymbolplural@: added redefinition	66
\@GLStext@: added redefinition	61
\@GLSuseri@: added redefinition	66
\@GLSuserii@: added redefinition	66
\@GLSuseriii@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@GLSuserv@: added redefinition	67
\@GLSuservi@: added redefinition	67
\@acrlong: added redefinition	70
\@acrlongpl: added redefinition	71
\@acrshort: added redefinition	68
\@acrshortpl: added redefinition	69
\@gls@field@link: added optional argument	54
\@glsdescplural@: added redefinition	64
\@glsfirst@: added redefinition	61
\@glsfirstplural@: added redefinition	63
\@glsplural@: added redefinition	62
\@glssymbolplural@: added redefinition	65
\@glsxtr@defaultnoglossarywarning: new	121
\@glsxtr@field@linkdefs: new	60
\@glsxtr@insertdots: new	179
\@print@glossary: added redefinition	118
\glsabbrvdefaultfont: renamed from \abbrvdefaultfont	184
\glsaccessdesc: new	146
\glsaccessdescplural: new	147
\glsaccessfirst: new	144
\glsaccessfirstplural: new	144
\glsaccesslong: new	149
\glsaccesslong: new	148
\glsaccessname: new	142
\glsaccessplural: new	143
\glsaccessshort: new	148
\glsaccessshort: new	147
\glsaccessshortpl: new	148

\glsaccesssshortpl: new	148	\cGLSpl: new	95
\glsaccessssymbol: new	145	\cGLSpl@: new	95
\glsaccessssymbolplural: new	146	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	143	new	90
\glsentryfmt: added check for short ..	54	\cGLS: new	95
\glslongpltok: new	179	\cGLSformat: new	95
\glsshortpltok: new	178	\cGLSpl: new	95
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	95
name in \setkeys	180	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	15
for plural	175	\glsenableentrycount: new	91
\GLSxtrlongpl: new	193	\glsfirstabbrvdefaultfont: new ..	184
\Glsxtrlongpl: new	193	\glsfirstlongdefaultfont: new ...	184
\glsxtrlongpl: new	192	\Glsfmtfirst: new	309
\glsxtrNoGlossaryWarning: new	20	\glsfmtfirst: new	308
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	309
new	175	\glsfmtfirstpl: new	309
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	308
new	175	\glsfmtplural: new	308
\glsxtrpostlinkendsentence: new ..	175	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	192	\Glsxtrtitleshort	306
\Glsxtrshortpl: new	191	renamed from \Glsentryfmtshort ..	306
\glsxtrshortpl: new	191	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	306
\glslabeltok	204	renamed from \glsentryfmtshort ..	306
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	202	\Glsxtrtitleshortpl	306
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glsentryfmtshortpl	306
redefinition of \acronymtype	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	306
\Glsxtrshort	306	renamed from	
\glsfmtshort: changed to use		\glsentryfmtshortpl	306
\glsxtrshort	306	\Glsfmttext: new	308
\Glsfmtshortpl: changed to use		\glsfmttext: new	307
\glsxtrshortpl	306	\glshasattribute: new	154
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	153
\glsxtrshortpl	306	\glsxtremsuffix: new	245
\glsxtrifemptyglossary: new	25	\GlsXtrEnableEntryCounting: new ..	90
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	93
argument	157	\glsxtrscfont: new	217
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	217
argument	157	\glsxtrsmfont: new	231
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	231
default type to \acronymtype	104	short-em: new	252
\newterm: fixed name argument	156	short-em-desc: new	254
0.5 (2015-12-07)		short-em-footnote: new	263
\@cGLS: new	95	short-em-long: new	249
\@cGLS@: new	95	short-em-long-desc: new	250

short-em-postfootnote: new	265	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	227	attribute	297
short-sc-postfootnote: new	229	\Glsxtrheadtext: now uses headuc	
short-sm: new	235	attribute	299
short-sm-desc: new	236	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	242	attribute	299
short-sm-long: new	233	short-em-footnote: switch off regular	
short-sm-long-desc: new	234	attribute if set	263
short-sm-postfootnote: new	243	short-long: switch off regular attribute	
long-noshort-em: new	256	if set	203
long-noshort-em-desc: new	260	short-long-desc: switch off regular	
long-noshort-sm: new	238	attribute if set	204
long-noshort-sm-desc: new	240	short-sc-footnote: switch off regular	
long-short-em: new	246	attribute if set	228
long-short-em-desc: new	247	short-sm-footnote: switch off regular	
long-short-sm: new	232	attribute if set	242
long-short-sm-desc: new	233	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	201
\Glsaccessstext: new	143	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	202
\@glxtr@doaccsupp: new	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	296	attribute if set	219
\Glsaccessdesc: new	146	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	147	set	205
\Glsaccessfirst: new	144	postfootnote: switch off regular	
\Glsaccessfirstplural: new	145	attribute if set	207
\Glsaccessname: new	142	0.5.2 (2015-12-08)	
\Glsaccessplural: new	143	\@GLSdesc@: added accessibility support	64
\Glsaccesssymbol: new	145	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	146	support	65
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	301	support	62
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	301	support	63
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	64
headuc attribute	302	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	62
headuc attribute	301	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	65
attribute	300	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	66
attribute	300	\@GLStext@: added accessibility support	61
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	64
attribute	297	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	64
attribute	296	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	62
attribute	298	\@GLSfirstplural@: added accessibility	
		support	63

\@Glsname@: add accessibility support ..	64	\GLSaccesssymbolplural: new ..	146, 151
\@Glsplural@: added accessibility support	62	\GLSaccessstext: new	143, 150
\@Glsymbol@: added accessibility support	65	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	66	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	61	\glsxtrabbrvfmt to here	54
\@glsdesc@: added accessibility support	64	\GlsXtrEnableInitialTagging: new	171
\@glsdescplural@: added accessibility support	64	\glsxtrfieldtitlecase: new	158
\@glsfirst@: added accessibility support	61	\GlsXtrFormatLocationList: new ...	52
\@glsfirstplural@: added accessibility support	63	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	63	new	182
\@Glsplural@: added accessibility support	62	\glsxtrtagfont: new	173
\@Glsymbol@: added accessibility support	65	\KV@printgloss@nonumberlist: added	53
\@Glsymbolplural@: added accessibility support	65	\mfu@checkword@do: added	172
\@Glstext@: added accessibility support	61	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	197
new	173	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	173	\@glsxtr@autoindex@at: new	169
\@glsxtr@tag: new	173	\@glsxtr@autoindex@encap: new ...	169
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	170
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	169
\@ifpackageloaded	142	\@glsxtr@autoindex@setname: new ..	167
removed \glsxtrabbrvfmt	194	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added	51	General: removed	
\Glossentrydesc: added	171	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	163	120
\Glossentrysymbol: added	171	\glsdescwidth: added	51
\Glsaccessdesc: new	147, 151	\glspagelistwidth: added	51
\Glsaccessdescplural: new ...	147, 152	\glsxtrdoautoindexname: new	167
\Glsaccessfirst: new	144, 150	\glsxtrpostnamehook: new	164
\Glsaccessfirstplural: new ..	145, 151	\if@glsxtr@format@override: new ..	166
\Glsaccesslong: new	149, 152	\ProvidesGlossariesExtraLang: new	312
\Glsaccesslongpl: new	149, 152	\RequireGlossariesExtraLang: new	312
\Glsaccesslongpl: new	149	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new	149	\@newglossaryentry@defunitcounters:	
\Glsaccessname: new	143, 150	new	96
\Glsaccessplural: new	144, 150	\@GLSxtr@p@acrlong@: new	84
\Glsaccesssshort: new	148, 152	\@GLSxtr@p@acrlongpl@: new	84
\Glsaccesssshortpl: new	148, 152	\@GLSxtr@p@acrshort@: new	83
\Glsaccesssymbol: new	145, 151	\@GLSxtr@p@acrshortpl@: new	83
		\@GLSxtr@p@long@: new	83
		\@GLSxtr@p@longpl@: new	83
		\@GLSxtr@p@plural@: new	82
		\@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	84

\@Glsxtr@p@acrshort@: new	83	\glsenableentryunitcount: new	98
\@Glsxtr@p@acrshortpl@: new	83	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	83	entry's existence	154
\@Glsxtr@p@longpl@: new	83	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	81	entry's existence	154
\@Glsxtr@p@short@: new	82	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	82	check	174
\@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	78	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	78	new	101
\@glscincrement@currunitcount:		\GlsXtrEnableOnTheFly: new	46
new	97	\Glsxtrpl: new	48
\@glsclocal@increment@currunitcount:		\glxtrpl: new	48
new	98	\glxtrpostlocalreset: new	90
\@glscsetdefault@glslink@opts:		\glxtrpostlocalunset: new	89
new	75	\glxtrpostreset: new	90
\@glsxtr: new	47	\glxtrpostunset: new	89
\@glsxtr@addunitcounter: new	97	\glxtrprotectlinks: new	80
\@glsxtr@currunitcount: new	98	\GlsXtrSetAltModifier: new	78
\@glsxtr@ifunitcounter: new	97	\GlsXtrSetDefaultGlsOpts: new	76
\@glsxtr@p@acrlong@: new	83	\glxtrstarflywarn: new	47
\@glsxtr@p@acrlongpl@: new	84	\GlsXtrWarning: new	49
\@glsxtr@p@acrshort@: new	83	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	83	disables \setacronymstyle	104
\@glsxtr@p@long@: new	83	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	83	\@glsxtr@autoindexcrossrefs: new .	15
\@glsxtr@p@plural@: new	81	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	82	new	112
\@glsxtr@p@shortpl@: new	82	\@glsxtr@idx@entrynumberlist: new	114
\@glsxtr@p@text@: new	81	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	98	new	112
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	102	new	113
\@glsxtr@unitcountlist: new	96	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	48	new	113
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	105
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	96	for first use	175
\cGlsformat: added	96	short-desc: fixed typo in	
\cGlsplformat: added	96	\glxtrinlinefullformat and	
\cGlsplformat: added	96	added missing second argument ...	211
\glstdisablehyper: added	80	1.02 (2016-04-25)	
\glstdohyperlink: added	78	\@glsxtr@current@style: new	50
\glstdonohyperlink: added	80	\Glsfmtfull: new	311

\glsfmtfull: new	311	\@GLSdescplural@: set abbreviation and regular format	65
\Glsfmtfullpl: new	312	\@GLSfirst@: set abbreviation format ..	62
\glsfmtfullpl: new	311	\@GLSfirstplural@: set abbreviation and regular format	63
\Glsfmtlong: new	310	\@GLSname@: set abbreviation and regular format	64
\glsfmtlong: new	310	\@GLSplural@: set abbreviation and regular format	62
\Glsfmtlongpl: new	310	\@GLSsymbol@: set regular format	65
\glsfmtlongpl: new	310	\@GLSsymbolplural@: set regular format	66
\Glsxtrheadfull: new	305	\@GLStext@: set abbreviation and regular format	61
\glsxtrheadfull: new	304	\@GLSuseri@: set regular format	66
\Glsxtrheadfullpl: new	305	\@GLSuserii@: set regular format	66
\glsxtrheadfullpl: new	304	\@GLSuseriii@: set regular format	67
\Glsxtrheadlong: new	303	\@GLSuseriv@: set regular format	67
\glsxtrheadlong: new	302	\@GLSuservi@: set regular format	68
\Glsxtrheadlongpl: new	303	\@GLSdesc@: set abbreviation and regular format	64
\glsxtrheadlongpl: new	303	\@GLSdescplural@: set abbreviation and regular format	64
\Glsxtrtitlefull: new	305	\@GLSfirst@: set abbreviation and regular format	62
\glsxtrtitlefull: new	304	\@GLSfirstplural@: set abbreviation and regular format	63
\Glsxtrtitlefullpl: new	305	\@GLSname@: set abbreviation and regular format	64
\glsxtrtitlefullpl: new	305	\@GLSplural@: set abbreviation and regular format	62
\Glsxtrtitlelong: new	303	\@GLSsymbol@: set regular format	65
\glsxtrtitlelong: new	302	\@GLSsymbolplural@: set regular format	66
\Glsxtrtitlelongpl: new	304	\@GLStext@: set abbreviation and regular format	61
\glsxtrtitlelongpl: new	303	\@GLSuseri@: set regular format	66
\ifglsxtrinsertinside: new	200	\@GLSuserii@: set regular format	66
postfootnote: added redef of		\@GLSuseriii@: set regular format	67
\glsxtrsetupfulldefs	207	\@GLSuseriv@: set regular format	67
stylemods: new	21	\@GLSuservi@: set regular format	68
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format	64
\@GLSfirstplural@: bug fix: misspelt cs name	63	\@GLSdescplural@: set abbreviation and regular format	62
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLSsymbol@: set regular format	65
\@GLSfirstplural@: bug fix: misspelt cs name	63	\@GLSsymbolplural@: set regular format	66
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLStext@: set abbreviation and regular format	61
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	62	\@GLSuseri@: set regular format	66
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	303	\@GLSuserii@: set regular format	66
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	297	\@GLSuseriii@: set regular format	67
1.04 (2015-04-30)		\@GLSuseriv@: set regular format	67
short-em-footnote: renamed from “footnote-em”	263	\@GLSuservi@: set regular format	67
1.04 (2016-05-02)		\@gls@preglossaryhook: added check for entry’s existence	173
\@glsxtrpostloctag: new	53	\@GLSdesc@: set abbreviation and regular format	64
\@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	63	\glsxtruserparen: new	267
\@glsname@: set abbreviation and regular format	63	\glsxtrusersuffix: new	268
\@glsplural@: set abbreviation and regular format	62	\GlsXtrWarnDeprecatedAbbrStyle: new	199
\@glsymbol@: set regular format	65	short-em-long-em: new	251
\@glsymbolplural@: set regular format	65	short-em-long-em-desc: new	252
\@glstext@: set abbreviation and regular format	61	short-em-nolong: new	254
\@glsxtr@deprecated@abbrstyle: new	199	short-em-nolong-desc: new	255
\@glsxtr@do@style: new	21	short-em-postfootnote: renamed from “postfootnote-em”	265
\@glsxtr@doloctag: new	53	short-footnote: new	207
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	114	short-long-user: new	274
\@glsxtr@pagetag: new	53	short-long-user-desc: new	275
\@glsxtr@pagetag: new	53	short-nolong: new	210
\@glsxtr@preloctag: new	53	short-nolong-desc: new	212
\@glsxtr@postloctag: new	53	short-postfootnote: new	209
\@glsxtr@preloctag: new	52, 53	short-sc-footnote: renamed from “footnote-sc”	227
\glossentrydesc: added glossdescfont attribute check	159	short-sc-nolong: new	222
\Glossentryname: added glossnamefont attribute check	163	short-sc-nolong-desc: new	223
\glossentryname: added glossnamefont attribute check	160	short-sc-postfootnote: renamed from “postfootnote-sc”	229
moved post name hook inside condition	162	short-sm-footnote: renamed from “footnote-sm”	242
\glsabbrvmfont: new	245	short-sm-nolong: new	236
\glsabbrvuserfont: new	267	short-sm-nolong-desc: new	238
\glsfirstabbrvmfont: new	245	short-sm-postfootnote: renamed from “postfootnote-sm”	243
\glsfirstabbrvuserfont: new	267	\letabbreviationstyle: new	199
\glsfirstlongemfont: new	245	\newabbreviationstyle: bug fix: corrected test for existence	198
\glsfirstlonguserfont: new	268	long-em-noshort-em: new	258
\glsifnotregularcategory: new ...	155	long-em-noshort-em-desc: new	261
\glslongdefaultfont: new	184	long-em-short-em: new	247
\glslongemfont: new	246	long-em-short-em-desc: new	248
\glslongfont: new	184	long-noshort: new	216
\glslonguserfont: new	267	long-noshort-desc: new	215
\glsxtrassignfieldfont: new	60	long-noshort-em: renamed from “long-em”	256
\GlsXtrEnablePreLocationTag: new .	52	long-noshort-em-desc: renamed from “long-desc-em”	260
\glsxtrfirstscfont: new	217	long-noshort-sc: renamed from “long-sc”	224
\glsxtrfirstsmfont: new	231	long-noshort-sc-desc: renamed from “long-desc-sc”	226
\glsxtrlongshortdescsort: new ...	202	long-noshort-sm: renamed from “long-sm”	238
\glsxtrpostnamehook: added category check	164	long-noshort-sm-desc: renamed from \long-desc-sm	240
\glsxtrregularfont: new	54		
\glsxtruserfield: new	267		

long-short-user: new	268	docdef option changed to choice	14
long-short-user-desc: new	274	\glxtr@usesee: new	39
\renewabbreviationstyle: new	198	\glxtrusesee: new	39
style: new	21	\glxtruseseeformat: new	39
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	14
\eglssetwidest: new	331	1.07 (2016-08-15)	
\glFindWidestAnyName: new	334	\@@glxtrp: new	84
\glFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	339	nohyperfirst attribute	62
\glFindWidestAnyNameSymbol: new	337	\@GLSfirstplural@: added check for	
\glFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	63
new	338	\@GLSxtrp: new	85
\glFindWidestLevelTwo: new	335	\@Glsfirst@: added check for	
\glFindWidestUsedAnyName: new ..	334	nohyperfirst attribute	62
\glFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	339	nohyperfirst attribute	63
\glFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	85
new	336	\@gls@preglossaryhook: added	
\glFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	174
new	337	\@glsfirst@: added check for	
\glFindWidestUsedLevelTwo: new .	334	nohyperfirst attribute	61
\glFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new	333	nohyperfirst attribute	63
\glfirstlongfootnotefont: new ..	205	\@glxtrinmark: new	294
\glsgetwidestname: new	333	\@glxtrnotinmark: new	294
\glsgetwidestsubname: new	333	\@glxtrp: new	84
\glslongfootnotefont: new	205	\@glxtrp@opt: new	84
\glxtrAltTreeIndent: new	331	\glossxtrsetpopts: new	84
\glxtralttreeInit: new	331	\glsp: new	87
\glxtrAltTreePar: new	331	\glsp: new	87
\glxtrAltTreeSetHangIndent: new	340	\glxtr@entry@p: new	85
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new	205
new	341	\glxtrchecknohyperfirst: new	61
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new	158
new	331	\glxtrifinmark: new	294
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new	88
new	330	\Glsxtrp: new	87
\glxtrComputeTreeIndent: new ...	340	\glxtrp: new	86
\glxtrComputeTreeSubIndent: new	340	\glxtrsetpopts: new	84
\glxtrtreetopindent: new	331	short-long-desc: added text key	204
short-em-long: fixed incorrect font used		fixed misspelling of \glabbrvfont in	
by long form	250	plural key	204
\xglsetwidest: new	332	long-short-desc: added missing text	
1.06 (2016-06-18)		key	202
\@glsdoifexistsorwarn: new	14	fixed misspelling of \glabbrvfont .	202
\@glxtr@docdefval: new	14	footnote: changed first forms to use	
\@glxtr@usesee: new	39	\glfirstlongfootnotefont ...	205
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	25	from first keys	207

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	208	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	124
\@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	124
\@glsxtr@dorecord: new	10		\glsxtr@texencoding: new	123
\@glsxtr@err@undefaction: new	6		\glsxtr@writefields: new	124
\@glsxtr@record: new	7		\GlsXtrLoadResources: new	123
\@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	128	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	173
\glsdoifexists: now defines \glslabel	37		\@glsxtr@counter@recordhook: new ..	125
\glsxtr@do@wrglossary: new	25		\@glsxtr@display@loc: new	116
\glsxtr@addloc@listfield: new	11		\@glsxtr@docounter@record: new ...	126
\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	76
\printunsrt@glossaries: new	128		\@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	76
\@glsxtr@gettype: new	112		\@printunsrt@glossary: new	127
\@glsxtr@mixed@assign@sortkey: new	112			
\@print@glossary: redefined to save options	111			
\glsxtr@make@glossaries: new	111			

General: added target key to printgloss		\GlsXtrLoadResources: removed	
family	111	restriction on only one per document	123
\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	123
field	79	\glxtrresourcefile: added catcode	
\glsnoidxdisplayloc: added		change for @	123
redefinition	116	\glxtrsetaliasnoindex: new	75
\glissettoctitle: added patch	35	\GlsXtrSetField: new	31
\glxtr@counterrecord: new	125	\glxtrsetfieldifexists: new	31
\glxtr@langtag: new	123	\glxtrunsrtdo: new	131
\glxtr@newabbreviation: new	180	\Glsxtrusefield: new	30
\glxtr@org@newignoredglossary:		\glxtrusefield: new	30
Added check for existence	34	short-postlong-user: new	271
\glxtr@pluralsuffixes: new	123	short-postlong-user-desc: new ...	273
\glxtr@provideignoredglossary:		\longnewglossaryentry: added starred	
new	36	version	33
\glxtr@s@newignoredglossary:		long-postshort-user: new	269
Added check for existence	35	long-postshort-user-desc: new ...	270
\glxtr@s@provideignoredglossary:		postdot: new	15
new	36	\pretoglossarypreamble: new	32
\glxtrabbrvpluralsuffix: new ...	184	\print@noop@unsrtglossaryunit:	
\glxtralias: new	43	new	130
\glxtrcopytoglossary: new	37	\print@op@unsrtglossaryunit: new	130
\glxtrdeffield: new	31	\printunsrtglossary: added starred	
\glxtrdisplayendloc: new	117	form	127
\glxtrdisplayendlochook: new ...	117	\printunsrtglossaryhandler: new .	129
\glxtrdisplaysingleloc: new	116	\printunsrtglossaryunit: new	11
\glxtrdisplaystartloc: new	116	\printunsrtglossaryunitsetup: new	130
\glxtrdeffield: new	31	\provideignoredglossary: new	36
\glxtrentryfmt: new	28	\s@glxtr@provide@storagekey: new	26
\glxtrfielddolistloop: new	29	\s@printunsrtglossary: new	128
\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 ..	114
\GlsXtrFmtField: new	27	\@gls@removespaces: new	117
\glxtrifkeydefined: new	26	\@glxtr@do@automake@err: new ...	125
\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 .	80	bug	114
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	92	check for seealso field	44
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@	100	instead of \csname	131
\glsnavigation: new	115	\@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	125	new	129
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	117	\@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	124
\glsxtrsupphypernumber: new	118	\@glsxtrautoindex: new	167
\ifglsxtrwrglossbefore: new	57	\@glsxtrautoindexentry: new	167
1.15 (2017-05-10)		\@glsxtrautoindexsort: new	168
\@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	251	\@glsxtruseseealso: new	40
short-long: fixed spelling of		\@glsxtruseseealsoformat: new	40
\glsabbrvfont	203	\@seealso: new	40
short-long-user: fixed spelling of		autoseeindex: new	15
\glsabbrvfont	274	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	179
\glsabbrvfont	271	\@glsxtr@markwordseps: new	179
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	273	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	112
\glsabbrvfont	248	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	269	\glsxtrundeftag	113
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	271	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	113
\glsabbrvfont	201	\@glsxtrifhyphenstart: new	276
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	268	in the abbreviation styles	200
footnote: fixed spelling of		\glsabbrvhyphenfont: new	277
\glsabbrvfont	205	\glsabbrvonlyfont: new	290
postfootnote: fixed spelling of		\glsabbrvscfont: new	217
\glsabbrvfont	207	\glsabbrvsmfont: new	231

\glsabbrvuserfont: initialised to default font	267	short-long-user-desc: corrected first forms	275
\glsfirstabbrvhyphenfont: new ...	277	short-nolong-desc-noreg: new ...	212
\glsfirstabbrvonlyfont: new	290	short-nolong-noreg: new	210
\glsfirstabbrvscfont: new	217	long-em-noshort-em-desc-noreg: new	263
\glsfirstabbrvsmfont: new	231	long-em-noshort-em-noreg: new ...	259
\glsfirstlonghyphenfont: new ...	277	long-hyphen-noshort-desc-noreg: new	279
\glsfirstlongonlyfont: new	290	long-hyphen-postshort-hyphen: new	282
\glslonghyphenfont: new	277	long-hyphen-postshort-hyphen-desc: new	284
\glslongonlyfont: new	290	long-hyphen-short-hyphen: new ...	277
\glslonguserfont: initialised to default font	267	long-hyphen-short-hyphen-desc: new	278
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong	180	long-noshort-desc-noreg: new ...	215
\GlsXtrDefineAcShortcuts: new	18	long-noshort-noreg: new	216
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside	194	long-only-short-only: new	291
\glsxtrhyphensuffix: new	277	long-only-short-only-desc: new ..	292
\glsxtrifhyphenstart: new	276	long-short-user-desc: corrected first forms	274
\glsxtrlonghyphen: new	281	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	279	stylemods: changed default value to “default”	21
\glsxtrlonghyphenshort: new	276	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	202	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new	292	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref	10
\glsxtronlydescsort: new	292	\@glsxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new	290	\@glsxtr@record@setting: new	12
\glsxtrparen: new	182	\@glsxtr@record@setting@alsoindex: new	12
\glsxtrposthyphenlong: new	287	\@glsxtrifhasfield: new	30
\glsxtrposthyphenshort: new	282	General: added \glslink option theHvalue	57
\glsxtrposthyphensubsequent: new	282	added \glslink option thevalue ...	57
\glsxtrshortdescname: new	211	\glsxtr@writefields: removed double-quotes around \jobname ..	125
\glsxtrshorthyphen: new	287	\glsxtrdoautoindexname: changed format test	167
\glsxtrshorthyphenlong: new	285	\glsxtrhyperlink: new	79
\glsxtrshortlongdescname: new ...	204	\glsxtrifhasfield: new	30
\glsxtrshortlongdescsort: new ...	204	\GlsXtrSetDefaultNumberFormat: new	7
\Glsxtrsubsequentfmt: new	197	\s@glsxtrifhasfield: new	30
\glsxtrsubsequentfmt: new	196		
\Glsxtrsubsequentplfmt: new	197		
\glsxtrsubsequentplfmt: new	196		
\glsxtrword: new	179		
\glsxtrwordsep: new	179		
short-hyphen-long-hyphen: new ...	285		
short-hyphen-long-hyphen-desc: new	286		
short-hyphen-postlong-hyphen: new	287		
short-hyphen-postlong-hyphen-desc: new	289		

headings	345	\glxtrbookindexbetween: new	350
redefined		\glxtrbookindexbookmark: new ...	350
mcoltreenonamehypergroup to		\glxtrbookindexcols: new	349
discourage breaks after group		\glxtrbookindexcolspread: new ..	350
headings	345	\glxtrbookindexfirstmark: new ..	354
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	354
discourage breaks after group		\glxtrbookindexformatheader: new	350
headings	346	\glxtrbookindexgroupskip: new ..	350
redefined mcoltreepspannav to		\glxtrbookindexlastmark: new ...	354
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	354
headings	344	\glxtrbookindexmarkentry: new ..	354
redefined treegroup to discourage		\glxtrbookindexname: new	349
breaks after group headings	328	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new	349
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings	329	new	349
redefined treenonamegroup to		\glxtrbookindexprelocation: new	349
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings	330	new	350
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	350
discourage breaks after group		\glxtrbookindexsubname: new	349
headings	330	\glxtrbookindexsubprelocation:	
debug: new	22	new	349
\gglssetwidest: new	331	\glxtrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new	350
existence	80	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	80	new	350
\glsenablehyper: changed to use \def		\glxtrbookindexthepage: new	353
rather than \let	80	\glxtrdetoklocation: new	135
\Glsfmtname: new	307	\glxtrenablerecordcount: new ...	135
\glsfmtname: new	307	\glxtrglossentry: new	126
\glshex: new	133	\glxtrgroupfield: new	131
\glslstchildpostlocation: new ..	316	\Glsxtrheadname: new	298
\glslstchildprelocation: new ...	316	\glxtrheadname: new	298
\glslstprelocation: new	316	\GlsXtrIfFieldEqStr: new	32
\glsnahyperlink: patched	78	\glxtriflabelinlist: new	130
\glssseeitemformat: new	39	\glxtrifrecordtrigger: new	136
\glsshowsheet: new	23	\glxtrindexseealso: added check	
\glstreechildprelocation: new ...	326	that the entry exists	41
\glstreeprelocation: new	326	\glxtrtrinityhyperoutside: new	57
\glstriggerrecordformat: new	137	\GlsXtrLocationRecordCount: new ..	135
\glseuseabbrvfont: new	194	\glxtrnewgls: new	133, 134
\glseuselongfont: new	194	\glxtrnewGLSlike: new	134
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new	134
new	8	\glxtrnewrgls: new	134
\glxtr@org@@do@wrglossary: new ..	25	\glxtrnewrGLSlike: new	135
\glxtr@org@dohyperlink: new	78	\glxtrnewrglslike: new	134
\glxtr@setbookindexmark: new ...	354	\glxtrprelocation: new	315, 349
\glxtrbookindexatendgroup: new ..	350	\GlsXtrRecordCount: new	135

\glxtrrecordtriggervalue: new ..	135	\glseeitemformat: switched check	
\glxtrresourcefile: now disables		from regular to short	39
record key	122	\glxtr@setaccessdisplay: new ...	164
\glxtrresourceinit: new	133	\glxtr@writefields: provide	
\GlsXtrSetRecordCountAttribute:		\glxtr@record in aux file	124
new	136	\glxtractivatenopost: new	110
\glxtrtitlename: new	298	\glxtrbookindexprelocation:	
\glxtrtitleorpdforheading: new .	294	removed check for no post dot	349
\GlsXtrTotalRecordCount: new	135	\glxtrglossentryother: new	126
\glxtrwrglossmark: new	22	\glxtrnopostpunc: new	110
short-em: new	253	1.23 (2017-11-12)	
short-sc: corrected first letter		\@@glxtrfmt: added check for indexing	28
uppercasing	221	added grouping	27
short-sm: corrected first letter		new	27
uppercasing	236	\@glxtr@nopostpunc@postdesc: new	110
\ifglxtr@hyperoutside: new	57	\@glxtr@restore@postpunc: new ..	111
all: new	314	\@glxtrentryfmt: fixed missing label	
nolong-short: new	212	argument	28
nolong-short-em: new	255	\@glxtrfmt: new	27
nolong-short-noreg: new	213	\eglupdatewidest: new	332
nolong-short-sc: new	223	\gglupdatewidest: new	332
nolong-short-sm: new	238	\glupdatewidest: new	332
nopostdot: new	15	\GlsXtrDefineAbbreviationShortcuts:	
\printunrtglossaryentryprocesshook:		changed \newabbr definition to use	
new	129	\providecommand	17
\printunrtglossarypredoglossary:		\GlsXtrDefineAcShortcuts: changed	
new	129	\newabbr definition to use	
\rGLS: new	139	\providecommand	18
\rGls: new	138	\glxtrfmtdisplay: new	28
\rgls: new	137	\glxtrifcustomdiscardperiod: new	175
\rGLSformat: new	140	\GlsXtrIfFieldUndef: new	30
\rGlsformat: new	140	\glxtrrestorepostpunc: new	111
\rglsformat: new	139	\s@glxtrfmt: new	27
\rGLSpl: new	139	\s@glxtrfmt: new	27
\rGlspl: new	138	\xglupdatewidest: new	332
\rglspl: new	137	1.24 (2017-11-14)	
\rGLSplformat: new	140	\glsadd: added \@gls@setsort	60
\rglsplformat: new	140	\glxtrforcsvfield: new	29
\rglsplformat: new	140	\glxtrlocalsetgrouptitle: new ..	115
\s@glxtrifhasfield: switched from		1.25 (2017-11-14)	
\ifdef to \ifundef	30	\glxtrbookindexmulticolsest: new	350
1.22 (2017-11-08)		1.25 (2017-11-24)	
\@glxtr@nopostpunc: new	110	\glsextrapostnamehook: new	164
\@glxtr@orgprintglossary: changed		\glxtrfootnotename: new	205
explicit \let for \nopostdesc to		\glxtrlongnoshortdescname: new .	214
\glxtractivatenopost	109	\glxtrlongnoshortname: new	216
\@glxtrglossentryother: new	127	\glxtrlongshortname: new	200
\glossentrynameother: new	165	\glxtrlongshortuserdescname: new	270
		\glxtronlyname: new	290

\glstrpostlinkAddDescOnFirstUse:		\glslinkpresetkeys: new	57
changed to use \glstrparen	175	\glstr@inc@linkcount: new	57
\glstrpostlinkAddSymbolOnFirstUse:		\GlsXtrEnableLinkCounting: new ..	142
changed to use \glstrparen	175	\GlsXtrIfLinkCounterDef: new	141
\glstrshortlongname: new	202	\glstrinclinkcounter: new	141
\glstrshortlonguserdescname: new	273	\GlsXtrLinkCounterName: new	141
\glstrshortnolongname: new	209	\GlsXtrLinkCounterValue: new	141
1.26 (2018-01-05)		\GlsXtrTheLinkCounter: new	141
\@glstr@do@inc@linkcount: new ..	140		

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	
\.	<i>15, 16, 175, 326</i>
\@	<i>123</i>
\@cGLS@	<i>92, 100</i>
\@cGLSpl@	<i>92, 100</i>
\@cGls@	<i>92, 100</i>
\@cGlspl@	<i>92, 100</i>
\@cgls@	<i>92, 100</i>
\@cglspl@	<i>92, 100</i>
\@do@wrglossary	<i>8, 9, 106, 107</i>
\@do@wrglossary	<i>11, 13, 25, 60, 76</i>
\@glo@assign@sortkey	<i>112</i>
\@glo@list	<i>6</i>
\@glo@type	<i>128</i>
\@glossarysec	<i>350</i>
\@gls@expand@field	<i>26</i>
\@glslocalreset	<i>90</i>
\@glslocalunset	<i>89</i>
\@glsreset	<i>90</i>
\@glsunset	<i>89</i>
\@glsxtr@autoindex@escspch	<i>169, 170</i>
\@glsxtr@checkspch	<i>168–171</i>
\@glsxtr@disabledflycommand	<i>49</i>
\@glsxtr@org@postdescription	<i>110</i>
\@glsxtr@record	<i>13</i>
\@glsxtr@recordcounter	<i>13, 125</i>
\@glsxtrfmt	<i>27</i>
\@glsxtrp	<i>84, 85</i>
\@glsxtrpostloctag	<i>52</i>
\@glsxtrpreloctag	<i>52</i>
\@glsxtrwrglossmark	<i>8, 11, 23, 25, 41, 45, 106</i>
\@newglossaryentry@defcounters	<i>91</i>
\@newglossaryentry@defunitcounters	<i>98</i>
\@par	<i>331</i>
\@ACRlong	<i>81</i>
\@ACRlongpl	<i>81</i>
\@ACRshort	<i>81</i>
\@ACRshortpl	<i>81</i>
\@ACrlong	<i>81</i>
\@ACrlongpl	<i>81</i>
\@ACrshort	<i>81</i>
\@ACrshortpl	<i>81</i>
\@GLS@	<i>81, 94, 95, 139</i>
\@GLSdesc@	<i>65</i>
\@GLSpl@	<i>81, 94, 95, 139</i>
\@GLSplural@	<i>82</i>
\@GLSsymbol@	<i>66</i>
\@GLStext@	<i>81</i>
\@GLSxtr@full	<i>186</i>
\@GLSxtr@fullpl	<i>187</i>
\@GLSxtr@p@acrlong@	<i>81</i>
\@GLSxtr@p@acrlongpl@	<i>81</i>
\@GLSxtr@p@acrshort@	<i>81</i>
\@GLSxtr@p@acrshortpl@	<i>81</i>
\@GLSxtr@p@long@	<i>81</i>
\@GLSxtr@p@longpl@	<i>81</i>
\@GLSxtr@p@plural@	<i>81</i>
\@GLSxtr@p@short@	<i>81</i>
\@GLSxtr@p@shortpl@	<i>81</i>
\@GLSxtr@p@text@	<i>81</i>
\@GLSxtrlong	<i>81, 190</i>
\@GLSxtrlongpl	<i>81, 193, 194</i>
\@GLSxtrp	<i>88, 89</i>
\@GLSxtrshort	<i>81, 189</i>
\@GLSxtrshortpl	<i>81, 192</i>
\@Gls@	<i>81, 94, 95, 138</i>
\@Gls@acentryname	<i>102</i>
\@Gls@entry@field	<i>72, 87, 88, 165</i>
\@Gls@entryname	<i>102</i>
\@GlsXtrEnableOnTheFly	<i>46, 47</i>
\@Glspl@	<i>81, 94, 95, 139</i>
\@Glsplural@	<i>81</i>
\@Glstext@	<i>81</i>
\@Glsxtr	<i>48, 49</i>

\@Glsxtr@full	185	\@endfortrue	29, 164, 197
\@Glsxtr@fullpl	187	\@firstofone	60, 61, 128, 159, 160, 166, 172
\@Glsxtr@p@acrlong@	81	\@firstofthree	56, 60, 68–71, 77, 78, 185, 186, 188, 189, 191, 193
\@Glsxtr@p@acrlongpl@	81	\@firstoftwo	61–66, 69–72, 74, 77, 104, 164, 165, 176, 177, 185–187, 191–194, 294, 295
\@Glsxtr@p@acrshort@	81	\@for	6, 21, 30, 44, 90, 102, 106, 108, 115, 128, 136, 142, 158, 164, 172
\@Glsxtr@p@acrshortpl@	81	\@glo@alias	41–43
\@Glsxtr@p@long@	81	\@glo@assign@sortkey	108
\@Glsxtr@p@longpl@	81	\@glo@autosee	23
\@Glsxtr@p@plural@	81	\@glo@autoseehook	42
\@Glsxtr@p@short@	81	\@glo@category	96
\@Glsxtr@p@shortpl@	81	\@glo@check@sortallowed	109
\@Glsxtr@p@text@	81	\@glo@counterprefix	10, 117
\@Glsxtrlong	81, 190	\@glo@countunit	96
\@Glsxtrlongpl	81, 193	\@glo@default@sorttype	108
\@Glsxtrp	87, 88	\@glo@desc	33, 34
\@Glsxtrpl	48, 49	\@glo@descplural	33, 34
\@Glsxtrshort	81, 188	\@glo@group	12
\@Glsxtrshortpl	81, 191	\@glo@label	11, 12, 26, 39, 42–44, 72, 80, 333–340
\@acrlong	81	\@glo@location	12
\@acrlongpl	81	\@glo@loclist	11
\@acrshort	81	\@glo@name	167
\@acrshortpl	81	\@glo@no@assign@sortkey	112
\@addtoreset	141	\@glo@parent	335, 336
\@afterheading	317, 318, 327–330, 343–346, 353	\@glo@see	39, 40, 42–44
\@alt@glshyp@opt	77	\@glo@seealso	41, 42
\@auxout	10, 11, 45, 53, 92, 101, 106, 118, 119, 123–126, 354	\@glo@sort	167
\@bibgl@restoreat	123	\@glo@sorttype	108, 115
\@cGLS	95	\@glo@text	56
\@cGLS@	92, 95, 100	\@glo@thislettergrp	131
\@cGLSpl	95	\@glo@thisvalue	267
\@cGLSpl@	92, 95, 100	\@glo@tmp	26, 40, 72
\@cGls@	92, 100	\@glo@type	44, 78, 103, 106, 109, 112, 115, 116, 118, 119, 121, 122, 128
\@cGlspl@	92, 100	\@glo@types	156, 333–339
\@cgls@	92, 100	\@glossary@default@style	50, 109, 348
\@cglspl@	92, 100	\@glossarystyle	109, 110
\@disable@onlypremakeg	106	\@gls@	81, 93, 95, 137
\@do@auxoutstuff	118, 119	\@gls@link	56
\@do@gl@getcounterprefix	10	\@gls@ReturnAfterFi	117
\@do@gl@see	42, 43	\@gls@actualchar	168
\@do@newglossaryentry	103, 182	\@gls@adjustmode	60
\@do@seeglossary	13, 23, 45, 106	\@gls@alt@hyp@opt	78
\@do@wrglossary	59, 137	\@gls@alt@hyp@opt@char	77, 78
\@empty	60, 68–72, 111, 168, 169, 185–194	\@gls@alt@hyp@opt@keys	78
\@end@glxtr@addunused	44	\@gls@automake	108
\@end@glxtr@gettype	108, 112		
\@end@glxtr@useseesee	39		
\@end@glxtr@trifhyphenstart	276		

<code>\@gls@between</code>	115	<code>\@gls@noidxloclist@prev</code>	112
<code>\@gls@checkedmkidx</code>	168–171	<code>\@gls@noidxloclist@sep</code>	112
<code>\@gls@checkmkidxchars</code>	41, 167	<code>\@gls@noref@warn</code>	107, 116
<code>\@gls@codepage</code>	119	<code>\@gls@org@glsnoidxdisplayloc</code>	113
<code>\@gls@counter</code>	9, 10, 58, 60, 76, 136	<code>\@gls@org@glssseeformat</code>	113
<code>\@gls@currentlettergroup</code> ...	115, 128, 131	<code>\@gls@preglossaryhook</code>	110, 172
<code>\@gls@declareoption</code>	5	<code>\@gls@prevlevel</code>	341, 342, 346, 347
<code>\@gls@default@longpl</code>	180, 181	<code>\@gls@quotechar</code>	168
<code>\@gls@doautomake</code>	108, 125	<code>\@gls@reference</code>	45, 106
<code>\@gls@doautomake@err</code>	125	<code>\@gls@saveentrycounter</code> .	13, 25, 58, 60, 136
<code>\@gls@encapchar</code>	168	<code>\@gls@see@noindex</code>	24, 123
<code>\@gls@entry@count</code>	92	<code>\@gls@setdefault@glslink@opts</code>	9, 28, 58, 76
<code>\@gls@entry@field</code>	26, 30, 42, 72, 86–89, 91, 127	<code>\@gls@setsort</code>	58, 60
<code>\@gls@entry@unitcount</code>	100, 101	<code>\@gls@setupsort@none</code>	13
<code>\@gls@field@font</code>	60–68	<code>\@gls@short</code>	181
<code>\@gls@field@link</code>	61–68, 73, 74	<code>\@gls@shortpl</code>	178, 181, 182
<code>\@gls@getcounterprefix</code>	10	<code>\@gls@sort</code>	131
<code>\@gls@getgrouptitle</code>	114, 128	<code>\@gls@thisval</code>	164, 165
<code>\@gls@grptitle</code>	78, 115	<code>\@gls@tmp</code>	115
<code>\@gls@hyp@opt</code>	73, 74, 78, 95, 133, 137–139, 184–193	<code>\@gls@tmpb</code>	170, 171
<code>\@gls@hyp@opt@cs</code>	77, 78	<code>\@gls@type</code>	106–108, 197, 333–340
<code>\@gls@ifinlist</code>	130	<code>\@gls@write@entrycounts</code>	92
<code>\@gls@increment@currcount</code>	91	<code>\@gls@write@entryunitcounts</code>	100
<code>\@gls@increment@currunitcount</code>	99	<code>\@gls@write@entryunitcounts@do</code>	101
<code>\@gls@keymap</code> ..	11, 12, 26, 39, 42, 72, 124, 164	<code>\@gls@xref</code>	11, 41
<code>\@gls@label</code> ..	8–10, 45, 77, 106, 107, 126, 197	<code>\@gls@abbrv@current@abbreviation</code> ..	180, 194
<code>\@gls@levelchar</code>	168	<code>\@gls@acronymlists</code>	102
<code>\@gls@link</code>	27, 55, 56, 68–72, 185–194	<code>\@gls@doifexistsorwarn</code> ..	14, 160, 161, 163, 165
<code>\@gls@link@checkfirsthyper</code>	56, 104	<code>\@gls@entry</code>	92, 101
<code>\@gls@link@label</code>	58, 136	<code>\@gls@link</code>	59, 78, 80
<code>\@gls@link@nocheckfirsthyper</code>	55, 68–72, 185–194	<code>\@gls@nextpages</code>	109
<code>\@gls@link@opts</code>	58	<code>\@gls@nonextpages</code>	109
<code>\@gls@list</code>	115	<code>\@gls@numberformat</code>	9, 10, 58, 60, 76, 136, 164, 167
<code>\@gls@local@increment@currcount</code>	91	<code>\@gls@order</code>	106
<code>\@gls@local@increment@currunitcount</code> ..	99	<code>\@gls@pl@</code>	81, 93, 95, 138
<code>\@gls@location</code>	132	<code>\@gls@plural@</code>	81
<code>\@gls@loclist</code>	112, 113, 132	<code>\@gls@punc@token</code>	177
<code>\@gls@long</code>	180	<code>\@gls@recordlocref</code>	10
<code>\@gls@longpl</code>	178, 180–182	<code>\@gls@showtarget</code>	79
<code>\@gls@map</code>	164	<code>\@gls@style@altlist</code>	316
<code>\@gls@nohyperlist</code>	34, 36	<code>\@gls@style@altlistgroup</code>	317
<code>\@gls@noidx@do</code>	116	<code>\@gls@style@altlisthypergroup</code>	318
<code>\@gls@noidx@getgrouptitle</code>	128	<code>\@gls@style@alttree</code>	330
<code>\@gls@noidx@nosanitizesort</code>	108	<code>\@gls@style@alttreegroup</code>	342
<code>\@gls@noidx@sanitizesort</code>	108	<code>\@gls@style@alttreehypergroup</code>	342
<code>\@gls@noidxloclist@finalsep</code>	112	<code>\@gls@style@index</code>	326
		<code>\@gls@style@indexgroup</code>	327

<code>\@glsstyle@indexhypergroup</code>	327	<code>\@glsxtr@autoindexcrossrefs</code>	13, 14, 39, 42
<code>\@glsstyle@inline</code>	326	<code>\@glsxtr@autoseeindexfalse</code>	13
<code>\@glsstyle@list</code>	316	<code>\@glsxtr@autoseeindextrue</code>	15
<code>\@glsstyle@listdotted</code>	315	<code>\@glsxtr@bookindex@atendgroup</code> ..	351–353
<code>\@glsstyle@listgroup</code>	317	<code>\@glsxtr@bookindex@atsubendgroup</code> ..	352
<code>\@glsstyle@listhypergroup</code>	317	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	352
<code>\@glsstyle@mcolalttree</code>	346	<code>\@glsxtr@bookindex@between</code>	351, 353
<code>\@glsstyle@mcolalttreegroup</code>	346	<code>\@glsxtr@bookindex@sep</code>	351, 352
<code>\@glsstyle@mcolalttreehypergroup</code> ..	347	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcolalttreesspannav</code>	347	351–353
<code>\@glsstyle@mcolindexgroup</code>	343	<code>\@glsxtr@bookindex@subbetween</code> ..	351, 352
<code>\@glsstyle@mcolindexhypergroup</code>	343	<code>\@glsxtr@bookindex@subsep</code>	351, 352
<code>\@glsstyle@mcolindexspannav</code>	343	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcoltreegroup</code>	344	351–353
<code>\@glsstyle@mcoltreehypergroup</code>	344	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcoltreenamegroup</code>	345	351, 352
<code>\@glsstyle@mcoltreenamehypergroup</code>	345	<code>\@glsxtr@bookindex@groupskip</code> ...	351, 353
<code>\@glsstyle@mcoltreenamespannav</code> ..	346	<code>\@glsxtr@cat</code>	90, 91, 102, 136, 172
<code>\@glsstyle@mcoltreesspannav</code>	344	<code>\@glsxtr@checkgroup</code>	129
<code>\@glsstyle@tree</code>	328	<code>\@glsxtr@counterrecordhook</code>	10, 11
<code>\@glsstyle@treegroup</code>	328	<code>\@glsxtr@csname</code>	97–100
<code>\@glsstyle@treehypergroup</code>	329	<code>\@glsxtr@current@style</code>	50, 348
<code>\@glsstyle@treename</code>	329	<code>\@glsxtr@currentunitcount</code>	97–100
<code>\@glsstyle@treenamegroup</code>	330	<code>\@glsxtr@currunitcount</code>	99, 101
<code>\@glsstyle@treenamehypergroup</code> ...	330	<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>	333, 340	7, 9, 58, 60, 76, 164, 167
<code>\@glsxtr</code>	47, 49	<code>\@glsxtr@defpostpunc</code>	15, 16, 23
<code>\@glsxtr@do@wrglossary</code>	106, 107	<code>\@glsxtr@deprecated@abbrstyle</code>	
<code>\@glsxtr@abbreviationsdef</code>	17, 24	226, 227, 229,
<code>\@glsxtr@accessdisplay</code>	164–166	231, 240, 242, 243, 245, 258, 261, 265, 267	
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@disabledflycommand</code>	49
.....	172, 173	<code>\@glsxtr@display@loc</code>	116
<code>\@glsxtr@addunitcounter</code>	96	<code>\@glsxtr@do@windex</code>	77
<code>\@glsxtr@addunused</code>	44	<code>\@glsxtr@do@gl:disablehyperinlist</code>	74, 75
<code>\@glsxtr@addunusedxrefs</code>	44	<code>\@glsxtr@do@inc@linkcount</code>	142
<code>\@glsxtr@attrval</code>		<code>\@glsxtr@do@record@wrglossary</code>	8, 13
.....	58, 59, 159, 160, 162, 163, 165, 167	<code>\@glsxtr@do@redef@forglsentries</code>	7
<code>\@glsxtr@autoindex@at</code>	167–169	<code>\@glsxtr@do@style</code>	21, 313
<code>\@glsxtr@autoindex@doextra@esc</code>	167	<code>\@glsxtr@do@titlecaps@warn</code>	
<code>\@glsxtr@autoindex@encap</code>	167–169	159–162, 165, 172, 173
<code>\@glsxtr@autoindex@esc</code>	168, 170, 171	<code>\@glsxtr@doabbreviationsdef</code>	17
<code>\@glsxtr@autoindex@escat</code>	168, 169	<code>\@glsxtr@doaccsupp</code>	20, 22
<code>\@glsxtr@autoindex@escencap</code> ...	168, 169	<code>\@glsxtr@docdefval</code>	14, 46
<code>\@glsxtr@autoindex@esclevel</code> ...	168, 169	<code>\@glsxtr@docounterrecord</code>	11
<code>\@glsxtr@autoindex@escquote</code> ...	168, 170	<code>\@glsxtr@doglossary</code>	128, 129
<code>\@glsxtr@autoindex@level</code>	168, 169	<code>\@glsxtr@doiflabelinlist</code>	130
<code>\@glsxtr@autoindex@setname</code>	167	<code>\@glsxtr@doloctag</code>	52, 53

\@glxtr@dorecord	8, 9	\@glxtr@noidx@numberlistloop	108
\@glxtr@dorecordnodefer	8, 9	\@glxtr@noop@recordcounter	10, 13
\@glxtr@dosee@alsoindex@glossary ..	13	\@glxtr@nopostpunc	110
\@glxtr@doseeglossary	13, 23	\@glxtr@nopostpunc@postdesc ..	110, 111
\@glxtr@dostylewarn	197	\@glxtr@notfoundinlist	177
\@glxtr@enabletagging	171	\@glxtr@op@recordcounter	13
\@glxtr@end@	47	\@glxtr@optlist	49
\@glxtr@endescpsh	168–171	\@glxtr@org@@starttoc	294
\@glxtr@entrycount@org@localreset	91, 92	\@glxtr@org@GLS@	55
\@glxtr@entrycount@org@localunset ..	91	\@glxtr@org@GLSpl@	56
\@glxtr@entrycount@org@reset	91	\@glxtr@org@Gls@	55
\@glxtr@entrycount@org@unset	91	\@glxtr@org@Glspl@	55
\@glxtr@entryunitcount@org@localreset	100	\@glxtr@org@Glsxtrtitlefirst ..	295, 296
\@glxtr@entryunitcount@org@localunset	99	\@glxtr@org@Glsxtrtitlefirstplural	295, 296
\@glxtr@entryunitcount@org@reset ..	99	\@glxtr@org@Glsxtrtitlefull ..	295, 296
\@glxtr@entryunitcount@org@unset ..	99	\@glxtr@org@Glsxtrtitlefullpl ..	295, 296
\@glxtr@err@undefaction	7, 13	\@glxtr@org@Glsxtrtitlelong ..	295, 296
\@glxtr@field@linkdefs	55	\@glxtr@org@Glsxtrtitlelongpl ..	295, 296
\@glxtr@format@overridefalse	166	\@glxtr@org@Glsxtrtitlename ..	295, 296
\@glxtr@format@overridefalse ..	166, 167	\@glxtr@org@Glsxtrtitleplural ..	295, 296
\@glxtr@foundinlist	177	\@glxtr@org@Glsxtrtitleshort ..	295, 296
\@glxtr@full	184	\@glxtr@org@Glsxtrtitleshortpl ..	295, 296
\@glxtr@fullpl	186	\@glxtr@org@Glsxtrtitletext ..	295, 296
\@glxtr@gettype	108	\@glxtr@org@MakeUppercase	295, 296
\@glxtr@glossdescfont	159, 160	\@glxtr@org@checkfirsthyper ...	74, 104
\@glxtr@glossnamefont	160–166	\@glxtr@org@delimN	52, 53
\@glxtr@gobbleto@endescpsh	170	\@glxtr@org@delimR	52, 53
\@glxtr@groupheading	129, 131	\@glxtr@org@doseeglossary	23, 106
\@glxtr@idx@displaynumberlist	107	\@glxtr@org@gloautosee	24
\@glxtr@idx@entrynumberlist	107	\@glxtr@org@gls@	55
\@glxtr@ifcsstart	47	\@glxtr@org@glsdohypertarget	111
\@glxtr@ifpunctoken	177	\@glxtr@org@glsignore	52, 53
\@glxtr@ifunitcounter	96	\@glxtr@org@glspl@	55
\@glxtr@insert@dots	179	\@glxtr@org@glsxtrtitlefirst ..	295, 296
\@glxtr@insert@dots@next	179	\@glxtr@org@glsxtrtitlefirstplural	295, 296
\@glxtr@insertdots	181	\@glxtr@org@glsxtrtitlefull ..	295, 296
\@glxtr@label	30, 44, 142, 158	\@glxtr@org@glsxtrtitlefullpl ..	295, 296
\@glxtr@loadstyles	314, 315	\@glxtr@org@glsxtrtitlelong ..	295, 296
\@glxtr@longnewglossaryentry	33	\@glxtr@org@glsxtrtitlelongpl ..	295, 296
\@glxtr@mark@wordseps	179	\@glxtr@org@glsxtrtitlename ..	295, 296
\@glxtr@mark@wordseps@next	180	\@glxtr@org@glsxtrtitleorpdforheading	295, 296
\@glxtr@markwordseps	180–182	\@glxtr@org@glsxtrtitleplural ..	295, 296
\@glxtr@mixed@assign@sortkey	108	\@glxtr@org@glsxtrtitleshort ..	295, 296
\@glxtr@noidx@displaynumberlist ..	107	\@glxtr@org@glsxtrtitleshortpl ..	295, 296
\@glxtr@noidx@do	131	\@glxtr@org@glsxtrtitletext ..	295, 296
\@glxtr@noidx@entrynumberlist	107	\@glxtr@org@makeglossaries	105

<code>\@glxtr@org@markboth</code>	293, 294	<code>\@glxtr@taggingcs</code>	172
<code>\@glxtr@org@markright</code>	293, 294	<code>\@glxtr@textformat</code>	59
<code>\@glxtr@org@newacronymstyle</code>	104	<code>\@glxtr@theHvalue</code> ...	8, 9, 57–60, 136, 137
<code>\@glxtr@org@postdescription</code> ..	111, 173	<code>\@glxtr@thevalue</code> ...	8, 9, 57–60, 136, 137
<code>\@glxtr@org@see@noindex</code>	123	<code>\@glxtr@thisloctag</code>	52, 53
<code>\@glxtr@org@setacronymstyle</code>	104	<code>\@glxtr@titlelabel</code>	114, 115, 130
<code>\@glxtr@org@theHvalue</code>	8, 9	<code>\@glxtr@tmp</code>	21, 117
<code>\@glxtr@orgprefix</code>	10	<code>\@glxtr@type</code>	158
<code>\@glxtr@orgprintglossary</code>	49, 111	<code>\@glxtr@unitcountlist</code>	97
<code>\@glxtr@orgwarndep</code>	178	<code>\@glxtr@unsrt@getgroupitle</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>\@glxtrdocdeffalse</code>	45
<code>\@glxtr@p@long@</code>	81	<code>\@glxtrentryfmt</code>	28
<code>\@glxtr@p@longpl@</code>	81	<code>\@glxtrfmt</code>	27
<code>\@glxtr@p@plural@</code>	81	<code>\@glxtrglossentry</code>	126
<code>\@glxtr@p@short@</code>	81	<code>\@glxtrglossentryother</code>	127
<code>\@glxtr@p@shortpl@</code>	81	<code>\@glxtrhypernameprefix</code>	111, 130
<code>\@glxtr@p@text@</code>	81	<code>\@glxtrifhasfield</code>	29, 30
<code>\@glxtr@pagetag</code>	52	<code>\@glxtrifhyphenstart</code>	276
<code>\@glxtr@pagetag</code>	52	<code>\@glxtrindexaliased</code>	75
<code>\@glxtr@prevunitcount</code>	99	<code>\@glxtrindexcrossrefsfalse</code>	14
<code>\@glxtr@printglossopts</code>	49, 108, 111	<code>\@glxtrindexcrossrefstrue</code>	15
<code>\@glxtr@printunsrtglossaryskipentry</code>	128, 129	<code>\@glxtrinmark</code>	293, 294
<code>\@glxtr@provide@addstoragekey</code>	27	<code>\@glxtrlong</code>	81, 189
<code>\@glxtr@provide@storagekey</code>	26	<code>\@glxtrlongpl</code>	81, 192, 193
<code>\@glxtr@record</code>	13, 54–56, 60	<code>\@glxtrnewgls</code>	134, 135
<code>\@glxtr@record@setting</code>	8, 9, 12, 41, 45, 46, 105	<code>\@glxtrnewgls@inner</code>	133
<code>\@glxtr@record@setting@alsoindex</code>	8, 9, 41, 105	<code>\@glxtrnewgls@innercsname</code> ...	133, 134
<code>\@glxtr@record@setting@off</code>	45	<code>\@glxtrnotinmark</code>	293, 294
<code>\@glxtr@record@setting@only</code>	105	<code>\@glxtrp</code>	86, 87
<code>\@glxtr@recordsee</code>	13, 23, 41	<code>\@glxtrp@opt</code>	84
<code>\@glxtr@redef@forglsentries</code>	7, 24	<code>\@glxtrpl</code>	48, 49
<code>\@glxtr@redefstyles</code>	21, 313	<code>\@glxtrpostloctag</code>	51, 52, 54
<code>\@glxtr@reg@glosslist</code>	105–108, 112	<code>\@glxtrpreloctag</code>	51, 52, 54
<code>\@glxtr@restore@postpunc</code>	110	<code>\@glxtrsetaliasnoindex</code>	75, 76
<code>\@glxtr@rglstrigger@record</code> ...	137–139	<code>\@glxtrshort</code>	81, 188
<code>\@glxtr@s@longnewglossaryentry</code> ...	33	<code>\@glxtrshortpl</code>	81, 191
<code>\@glxtr@savepreloctag</code>	52, 53	<code>\@glxtrundeftag</code>	6, 25
<code>\@glxtr@setentrycountunsetattr</code> ...	90	<code>\@glxtrwrglossmark</code>	22
<code>\@glxtr@setentryunitcountunsetattr</code>	102	<code>\@gobble</code> ..	7, 13, 15, 61, 129, 130, 179, 351–353
<code>\@glxtr@setupshortcuts</code>	19, 20, 24	<code>\@gobbletwo</code>	178
<code>\@glxtr@shortcutsval</code>	19, 124	<code>\@ifnextchar</code>	77
<code>\@glxtr@swaptwo</code>	178	<code>\@ifpackageloaded</code>	5, 16, 124, 142, 158, 160, 163, 164, 166, 312
<code>\@glxtr@tag</code>	172	<code>\@ifstar</code> ..	26, 27, 30, 33, 34, 36, 46, 77, 127, 171
		<code>\@ifundefined</code>	312
		<code>\@ignored@glossaries</code>	34–36

<code>\acfp</code>	18	<code>\as</code>	17
<code>\ACL</code>	18	<code>\ASP</code>	17
<code>\Acl</code>	18	<code>\Asp</code>	17
<code>\acl</code>	18	<code>\asp</code>	17
<code>\ACLP</code>	18	<code>\AtBeginDocument</code>	22, 25, 50, 51, 125
<code>\Aclp</code>	18	<code>\AtEndDocument</code>	43, 92, 100, 118, 119
<code>\aclp</code>	18		
<code>\ACP</code>	18	B	
<code>\Acp</code>	18	babel package	167, 169, 176
<code>\acp</code>	18	<code>\begin</code>	15, 16, 115, 120, 121, 128, 316, 318–325, 344–347, 351
<code>\ACRfullfmt</code>	103	<code>\beginngroup</code>	8, 9, 27, 28, 76, 126–128, 141
<code>\Acrfullfmt</code>	103	<code>\bgroup</code>	33, 109
<code>\acrfullfmt</code>	103	bib2gls	3, 28, 131, 133, 136, 137
<code>\ACRfullplfmt</code>	103		
<code>\Acrfullplfmt</code>	103	C	
<code>\acrfullplfmt</code>	103	<code>\catcode</code>	123
<code>\acronymentry</code>	103	category attributes:	
<code>\acronymfont</code>	68–72, 83, 104	apospplural	181
<code>\acronymname</code>	16	discardperiod	175
<code>\acronymsort</code>	103	entrycount	89–92, 101, 102
<code>\acronymtype</code>	17, 103, 104	firstuc	162
<code>\acrpluralsuffix</code>	103, 124	glossdesc	158
<code>\ACS</code>	18	glossdescfont	159
<code>\Acs</code>	18	glossname	160
<code>\acs</code>	18	glossnamefont	160, 163
<code>\ACSP</code>	18	headuc	296
<code>\Acsp</code>	18	indexname	167
<code>\acsp</code>	18	indexonlyfirst	76
<code>\actualchar</code>	170	insertdots	181
<code>\addtolength</code>	341	linkcount	140
<code>\advance</code>	92, 101, 123, 133	linkcountmaster	141
<code>\AF</code>	17	markshortwords	181
<code>\Af</code>	17	markwords	180, 181, 276, 277, 285
<code>\af</code>	17	nohyper	74
<code>\AFP</code>	17	nohyperfirst	61–63
<code>\Afp</code>	17	noshortplural	181
<code>\afp</code>	17	regular	54, 96, 200–205, 207, 210, 212, 213, 215, 216, 219, 220, 222, 223, 225, 227–229, 233–237, 240–242, 244, 247–253, 255, 257, 259, 261–263, 265, 268, 274, 276, 278, 279, 281, 286, 291, 292
<code>\AL</code>	17	textformat	58
<code>\Al</code>	17	<code>\cdot</code>	22
<code>\al</code>	17	<code>\centering</code>	350
<code>\ALP</code>	17	<code>\cGLS</code>	17, 18, 90, 102
<code>\Alp</code>	17	<code>\cGls</code>	17, 18, 90, 102
<code>\alp</code>	17	<code>\cglS</code>	17, 18, 90, 102
<code>\AnyTrackedLanguages</code>	312	<code>\cGLSformat</code>	94
<code>\appto</code>	11, 12, 21, 26, 39–43, 72, 77, 91, 98, 129, 130, 166, 176, 179, 180, 314	<code>\cGlsformat</code>	94
<code>\arabic</code>	353		
<code>\AS</code>	17		
<code>\As</code>	17		

<code>\end</code>	116,	<code>\Genacrfullformat</code>	103
120, 121, 129, 316, 318–325, 344–347, 351		<code>\genacrfullformat</code>	103
<code>\end@glstr@display@loc</code>	116	<code>\GenericAcronymFields</code>	103
<code>\endcsname</code>	6, 26, 35,	<code>\Genplacrfullformat</code>	103
41, 45, 50, 53, 56, 58, 60, 68–74, 76, 84,		<code>\genplacrfullformat</code>	103
97, 98, 106, 107, 115, 118, 121, 122, 128,		<code>\glo@grabfirst</code>	131
133–136, 141, 158, 178, 185–194, 200, 340		<code>\glo@name</code>	161, 162, 166
<code>\endgroup</code>	8, 10, 28, 76, 126–128, 141	<code>\gloaliaslabel</code>	79
<code>\ensuremath</code>	22	<code>\global</code>	10, 33, 34, 110, 132
entry categories:		<code>\glolinkprefix</code>	59, 79, 80, 125
abbreviation	194	glossaries package 13, 23, 24, 39–41, 43, 109, 315	
general	153, 155	glossaries-accsupp package	20, 22, 142
index	156	glossaries-extra package	2
<code>\epreto</code>	167	glossaries-extra-stylemods package .	20, 174, 313
<code>\equal</code>	121, 122	glossaries-stylemods package	349
etoolbox package	5	glossaries.sty package	34
<code>\expandafter</code>	22, 26,	<code>\GlossariesExtraWarning</code>	6,
27, 30, 39, 40, 44, 47–49, 72–74, 77, 78,		15, 32, 33, 47, 49, 59, 104, 107, 117, 120,	
84, 95, 96, 106–108, 112, 115, 117, 128,		123, 128, 159, 160, 162, 163, 165, 172, 199	
129, 131, 133, 134, 140, 158, 161, 162,		<code>\GlossariesExtraWarningNoLine</code>	15, 92, 101
164, 166, 167, 170, 177, 180–182, 276, 351		<code>\GlossariesWarning</code> 52, 107, 109, 112, 113, 197	
<code>\expandonce</code>	103, 131, 168, 202, 279	<code>\GlossariesWarningNoLine</code>	107, 119
F			
<code>\fi</code> 7–11, 13–15, 17, 19, 20, 22–24, 28, 32, 39,		glossary styles:	
41–43, 45–47, 50–52, 54, 56, 57, 59, 76,		altlist	316
77, 93, 100, 101, 105, 108–111, 114, 116,		altlistgroup	317
117, 119–123, 125, 136, 137, 166–169,		altlisthypergroup	318
171, 177, 179, 180, 182, 188–194, 196,		almtree	330, 331, 341
197, 201, 203, 204, 206–215, 218, 220–		almtreegroup	342
248, 250–266, 268–270, 272, 273, 275–		almtreehypergroup	342
277, 279, 282, 284, 285, 287, 289, 291,		index	326
292, 316, 318–330, 332–342, 348, 350, 353		indexgroup	327
first use	355	indexhypergroup	327
flag	355	inline	326
text	355	list	316
<code>\firstacronymfont</code>	104, 105	listdotted	315
fontspec package	124	listdottedstyle	316
<code>\footnote</code>	205	listgroup	317
<code>\forall glossaries</code>		listhypergroup	317
..... 44, 128, 156, 158, 333, 334, 336–340		mcolalmtree	346
<code>\forall glossentries</code>	92, 101	mcolalmtreegroup	346
<code>\ForEachTrackedDialect</code>	312	mcolalmtreehypergroup	347
<code>\for glossentries</code>	6, 44, 156, 158, 333–340	mcolalmtreespannav	347
<code>\forlistcsloop</code>	29, 101, 116	mcolindexgroup	343
<code>\forlistloop</code>	112, 113, 173	mcolindexhypergroup	343
<code>\futurelet</code>	177	mcolindexspannav	343
G			
<code>\gdef</code>	52, 53, 169, 170	mcoltreegroup	344
		mcoltreehypergroup	344
		mcoltreenonamegroup	345
		mcoltreenonamehypergroup	345

mcoltreenonamespannav	346	\gls@type	106
mcoltreespannav	344	\glsabbrvdefaultfont ...	184, 201, 203, 206, 208, 209, 211, 214, 267, 277, 280, 290
sublistdotted	316	\glsabbrvfont	245–254, 256, 258, 260, 262–266
tree	328	\glsabbrvfont 82, 104, 184, 188, 189, 191,	192, 194, 196, 197, 200–209, 211, 214, 216, 218, 219, 221, 222, 225, 226, 228, 230, 232, 234, 235, 237, 239, 240, 242, 244, 246, 248, 250, 251, 253, 254, 256, 258, 260, 262, 264, 266, 268, 269, 272, 274–276, 278, 280, 282, 283, 286, 288, 291
treegroup	328	\glsabbrvhyphenfont	277, 278, 282–286, 288, 289
treehypergroup	329	\glsabbrvonlyfont	290–292
treenoname	329	\glsabbrvscfont .	217–222, 225, 226, 228–230
treenonamegroup	330	\glsabbrvsmfont	231–235, 237, 239, 240, 242, 244
treenonamehypergroup	330	\glsabbrvuserfont	267–275
glossary-bookindex package	315	\GLSaccessdesc	64
glossary-hypernav package	78	\Glsaccessdesc	64, 159, 171
glossary-long package	320	\glsaccessdesc	64, 159, 175
glossary-longbooktabs package	320	\GLSaccessdescplural	65
\glossaryentrynumbers ...	53, 109, 110, 132	\Glsaccessdescplural	65
\glossaryheader	116, 128, 316–325, 327–330, 341–345, 347, 351	\glsaccessdescplural	64
\glossaryname	109	\GLSaccessfirst	62
\glossarypostamble	116, 129, 130	\Glsaccessfirst	62
\glossarypreamble	115, 128	\glsaccessfirst	61
\glossarysection ...	115, 116, 121, 128, 130	\GLSaccessfirstplural	63
\glossarytitle ...	35, 109, 115, 116, 121, 128	\Glsaccessfirstplural	63
\glossarytoctitle	35, 109, 115, 116, 121, 128	\glsaccessfirstplural	63
\glossentry	110, 132, 315, 316, 318–325, 327–329, 341, 351	\Glsaccesslong	70, 183, 190, 201, 210, 214, 215, 218, 224–227, 232, 238–241, 246, 248, 256–260, 262, 269, 270, 278, 280, 281, 283, 284, 286, 291, 292
\glossentrydesc	315–325, 327–331	\glsaccesslong	70, 71, 183, 189, 190, 201, 203, 204, 206, 208, 209, 211, 213–215, 218, 220, 221, 223–229, 231, 232, 234–243, 245, 246, 248, 250, 252–266, 268, 270, 272, 273, 275, 278, 280, 281, 283, 284, 286, 291
\glossentryname	126, 315–325, 327–329, 341, 342, 349	\Glsaccesslongpl	71, 183, 193, 201, 210, 214, 215, 218, 224, 225, 227, 232, 238–241, 247, 248, 256–262, 269, 270, 278, 280, 281, 284, 286, 291, 292
\glossentrynameother	127	\glsaccesslongpl	71, 72, 183, 193, 194, 201, 203, 204, 206–209, 211–215, 218, 220–229, 231, 232, 234, 236–243, 245, 246, 248,
\glossentrysymbol	319–323, 325, 327–329, 331		
\glossxtrsetpopts	174		
\GLS	90, 102, 135		
\Gls	48, 90, 102, 135		
\gls	32, 48, 49, 90, 102, 107, 120, 135		
\gls@assign@desc	33, 34		
\gls@assign@field	11, 12, 26, 72		
\gls@checkseeallowed	46, 106		
\gls@codepage	119		
\gls@defdocnewglossaryentry	91, 98		
\gls@defglossaryentry	33, 34, 47–49		
\gls@dotocitle	109		
\gls@glossary	41		
\gls@grplabel	78		
\gls@level	132		
\gls@noidxglossary	107		
\gls@org@glossaryentryfield	110		
\gls@org@glossarysubentryfield	110		
\gls@save@numberlist	51, 53, 54		
\gls@set@xr@key	41		
\gls@tmplen	333–342		

250, 252, 253, 255–268, 270, 272, 273, 275, 278, 280, 281, 283, 284, 286, 291, 292	\glscategorylabel
\GLSaccessname 74, 178, 180, 181, 207, 229, 244, 265, 269, 271, 273, 283, 284, 288, 290
\Glsaccessname	\glsclosebrace 41, 121, 122
\glsaccessname 40, 63	\glscurrententrylabel
\GLSaccessplural 51–53, 110, 118, 126–129, 173, 174
\Glsaccessplural	\glscurrentfieldvalue .. 27, 28, 30, 32, 267
\glsaccessplural	\glscustomtext .. 55, 56, 68–72, 185–194, 196
\Glsaccessshort 68, 188, 197, 203, 206, 208, 211, 213, 220, 221, 223, 228– 231, 234, 236, 237, 243–245, 250, 252, 253, 255, 264–266, 272, 275, 283, 288, 289	\glsdefaulttype
\glsaccessshort 68, 69, 183, 188, 189, 196, 201, 203, 206, 208–213, 215, 218, 220– 225, 227–232, 234–246, 248, 250, 251, 253–257, 259–262, 264, 266, 268–270, 272, 275, 278, 280, 283, 286, 288, 289, 292 6, 16, 32, 108, 109, 120, 128, 130
\Glsaccessshortpl	\glsdescriptionaccessdisplay 146, 147, 159
..... 69, 192, 197, 204, 206–208, 212, 213, 220, 222, 223, 229–231, 234, 236, 237, 243, 245, 250, 252, 253, 255, 264–266, 272, 273, 275, 283, 288, 289, 292	\glsdescriptionpluralaccessdisplay 147
\glsaccessshortpl	\glsdescwidth 318–325
..... 69, 70, 183, 191, 192, 196, 201, 203, 206, 208–213, 215, 218, 220– 225, 227–234, 236–241, 243–248, 250, 252–257, 259–262, 264, 266, 269, 270, 272, 275, 278, 280, 283, 286, 288, 289, 292	\glsdetoklabel 8, 9, 28–34, 37–40, 44–46, 58, 60, 76, 79, 91, 92, 97– 101, 106, 110, 112, 113, 126, 127, 131, 132, 135, 136, 158, 161, 162, 166, 335, 336
\GLSaccesssymbol	\glsdisplaynumberlist 107, 112
\Glsaccesssymbol 65, 171	\glsdohyperlink 78, 80
\glsaccesssymbol 65, 171, 175	\glsdohypertarget 80, 111
\GLSaccesssymbolplural	\glsdoifexists
\Glsaccesssymbolplural 66 14, 23, 31, 37, 39–41, 54, 56, 60, 68–72, 80, 106, 112, 113, 126, 127, 185–194
\glsaccesssymbolplural 65	\glsdoifexistsordo 27, 28, 56
\GLSaccessstext 61	\glsdoifexistsorwarn 14, 158, 159, 171
\Glsaccessstext 61	\glsdoifnoexists 33
\glsaccessstext 40, 61	\glsdonohyperlink 59, 80
\glsacrshortcutstrue 19, 20	\glsdosanitizesort 108
\glsacspacemax 105	\glsenableentrycount 90, 92, 100
\glsadd 28, 44, 120	\glsenableentryunitcount 92, 101, 102
\glsadd options	\glsentrycounter 117
theHvalue 9	\glsentrycurrcount 91, 92, 99
thevalue 9	\Glsentrydesc 146, 151, 160
\glsaddstoragekey 43, 153	\glsentrydesc 146, 147, 151, 160
\glsbackslash 47	\Glsentrydescplural 147, 151
\glscapscase 56, 60–74, 185–196	\glsentrydescplural 147, 151, 152
\glscategory	\Glsentryfirst 96, 140, 144, 150
.... 54, 60, 74, 82, 154, 155, 158–160, 162–165, 171, 174, 175, 185–189, 191, 192	\glsentryfirst 96, 139, 144, 150, 309
	\Glsentryfirstplural 96, 140, 145, 150
	\glsentryfirstplural
 96, 140, 145, 150, 151, 309
	\glsentryfmt 34–36
	\Glsentryfull 103
	\glsentryfull 103
	\Glsentryfullpl 103
	\glsentryfullpl 103
	\glsentryitem 126, 127, 315, 316, 318–325, 327–329, 341, 352
	\Glsentrylong 83, 96, 140, 149, 152

<code>\glstrylong</code>	83, 84, 96, 139, 149, 152, 207, 230, 244, 265, 271, 273, 287, 310
<code>\Glsentrylongpl</code>	83, 84, 96, 149, 152
<code>\glstrylongpl</code>	83, 84, 96, 149, 152, 310, 311
<code>\Glsentrylongplural</code>	140
<code>\glstrylongplural</code>	140
<code>\Glsentryname</code>	142, 150, 161–164
<code>\glstryname</code>	126, 142, 143, 149, 150, 167, 307, 333–340, 354
<code>\glstrynumberlist</code>	107, 114, 338–340
<code>\Glsentryplural</code>	144, 150
<code>\glstryplural</code>	143, 144, 150, 308
<code>\glstryprevcount</code>	91, 93, 99
<code>\glstryprevmaxcount</code>	99
<code>\glstryprevtotalcount</code>	99
<code>\Glsentryshort</code>	82, 83, 148, 152
<code>\glstryshort</code>	82, 83, 105, 147, 148, 152, 269, 271, 282, 306
<code>\Glsentryshortpl</code>	82, 83, 148, 152
<code>\glstryshortpl</code> ..	82, 83, 148, 152, 306, 307
<code>\Glsentrysymbol</code>	145, 151
<code>\glstrysymbol</code>	145, 146, 151, 337–339
<code>\Glsentrysymbolplural</code>	146, 151
<code>\glstrysymbolplural</code>	146, 151
<code>\Glsentrytext</code>	143, 150
<code>\glstrytext</code>	80, 143, 150, 307, 308
<code>\glstrytype</code>	126, 127
<code>\Glsentryuseri</code>	66
<code>\glstryuseri</code>	66
<code>\Glsentryuserii</code>	66
<code>\glstryuserii</code>	66
<code>\Glsentryuseriii</code>	67
<code>\glstryuseriii</code>	67
<code>\Glsentryuseriv</code>	67
<code>\glstryuseriv</code>	67
<code>\Glsentryuserv</code>	67
<code>\glstryuserv</code>	67
<code>\Glsentryuservi</code>	68
<code>\glstryuservi</code>	68
<code>\glsextrapostnamehook</code>	164
<code>\glsextraxdef</code>	79
<code>\glsextraxdef</code>	158
<code>\glsextraxdef</code>	333
<code>\GLSfirst</code>	301
<code>\Glsfirst</code>	301
<code>\glsextraxdef</code>	301
<code>\glsextraxdef</code>	184, 201, 203, 206, 208, 209, 211, 214, 280
<code>\glsextraxdef</code>	246–266
<code>\glsextraxdef</code>	104, 183, 201–215, 218, 219, 221, 223, 225, 226, 228, 230, 232, 234, 235, 237, 239, 240, 242, 244, 246, 248, 250, 251, 253, 254, 256, 258, 260, 262, 264, 266, 268, 269, 272, 275, 278, 280, 283, 286, 288, 291
<code>\glsextraxdef</code>	277, 278, 282, 283, 285–289
<code>\glsextraxdef</code>	291, 292
<code>\glsextraxdef</code>	217–231
<code>\glsextraxdef</code>	232–245
<code>\glsextraxdef</code>	268–275
<code>\glsextraxdef</code>	144
<code>\glsextraxdef</code>	201, 203, 209, 211, 214, 217–227, 232– 242, 246, 247, 249, 250, 253–257, 260, 261
<code>\glsextraxdef</code>	247–249, 251, 252, 258, 259, 261–263
<code>\glsextraxdef</code>	183, 201–204, 206, 208–216, 218, 220, 221, 223, 225, 226, 228, 230, 232, 234, 235, 237, 239, 240, 242, 244, 246, 248, 250, 251, 253, 254, 256, 258, 260, 262, 264, 266, 268, 269, 272, 275, 278, 280, 283, 286, 288, 291
<code>\glsextraxdef</code>	205–208, 228–231, 242–245, 263–267
<code>\glsextraxdef</code>	277–288
<code>\glsextraxdef</code>	291, 292
<code>\glsextraxdef</code>	268–276
<code>\glsextraxdef</code>	301, 302
<code>\glsextraxdef</code>	302
<code>\glsextraxdef</code>	302
<code>\glsextraxdef</code> ..	144, 145
<code>\glsextraxdef</code>	197
<code>\glsextraxdef</code>	54
<code>\glsextraxdef</code>	58, 79, 93, 97–99, 118, 136, 141, 159, 160, 162, 163, 165, 167
<code>\glsextraxdef</code>	154
<code>\glsextraxdef</code> ..	317, 318, 327–330, 342–348
<code>\glsextraxdef</code>	331
<code>\glsextraxdef</code>	131, 316–325, 327–330, 341–347, 353
<code>\glsextraxdef</code> ..	131, 316, 318–328, 330, 342, 353
<code>\glsextraxdef</code>	58, 78, 79, 92, 93, 97–101, 118, 136, 141, 159, 160, 162, 163, 165, 167, 201–205, 207, 210, 212, 213, 216, 217, 219, 220, 228, 230, 232–235, 242, 244, 246–249,

251, 252, 259, 263–265, 268, 269, 271– 276, 278, 279, 281, 283, 285–288, 290–292	
\glshascategoryattribute	154
\glshyperlink	80
\glshypernavsep	115
\glshypernumber	118, 166
\glsifattribute 56, 57, 61, 75, 76, 85, 140, 156, 159–162, 165, 166, 173, 176, 296–305	
\glsifcategory	156
\glsifcategoryattribute	74, 154, 155, 180, 181
\glsifnotregular	61
\glsifnotregularcategory	155
\glsifplural 56, 60, 62–66, 68–72, 176, 185–195	
\glsifregular	54, 60, 96, 139, 140
\glsifregularcategory	155
\glsifusetranslator	35
\glsignore	52, 53
\glsinlinedescformat	326
\glsinlinesubdescformat	326
\glsinsert	56, 60, 68–72, 185–196, 276, 283, 284, 288, 290
\glskeylisttok	103, 180, 182
\glslabel	8, 9, 27, 37, 40, 54, 56–59, 74–76, 78–80, 104, 136, 140, 141, 174, 175, 194–196, 207, 230, 244, 265, 269, 271, 273, 283, 284, 288, 290
\glslabeltok	103, 180, 182, 201–205, 207, 209–214, 216–222, 224, 228, 230, 232–235, 237, 239, 242, 244, 246–254, 256, 258, 259, 261, 263– 265, 268–279, 281, 283, 285–288, 290–292
\glsletentryfield	168
\glslink	103
\glslink options	
counter	9
format	166
hyper	293
hyperoutside	57
noindex	8, 75, 293
theHvalue	58
thevalue	58, 133
wrgloss	8, 56, 57
\glslinkcheckfirsthyperhook	74
\glslinkpostsetkeys	58, 136
\glslinkpresetkeys	58, 136
\glslinkvar	77, 78
\glslistchildpostlocation	316
\glslistchildprelocation	316, 317
\glslistdottedwidth	315
\glslistgroupheaderfmt	317, 318
\glslistnavigationitem	317, 318
\glslistprelocation	316, 317
\glslocalunset	56, 137
\glslongaccessdisplay	149
\glslongdefaultfont	184, 201, 203, 205, 209, 211, 214, 218, 220, 221, 223– 227, 232, 234, 235, 237, 239–241, 246, 250, 253, 254, 256, 257, 260, 267, 277, 290
\glslongemfont ..	245, 248, 251, 258, 261, 262
\glslongfont	83, 184, 189, 190, 193, 194, 201–204, 206, 208, 209, 211, 214, 216, 218, 220, 221, 223, 225, 226, 228, 230, 232, 234, 235, 237, 239, 240, 242, 244, 246, 248, 250, 251, 253, 254, 256, 258, 260, 262, 264, 266, 268, 269, 272, 275, 278, 280, 283, 286, 288, 291, 292
\glslongfootnotefont	205, 206, 208, 228, 230, 242, 244, 264, 266
\glslonghyphenfont ..	277–283, 285, 286, 288
\glslongonlyfont	290, 291
\glslongpltok	182, 201–205, 214, 216–220, 224, 228, 232, 233, 235, 239, 242, 246–252, 256, 258, 261, 263, 268, 269, 271, 273, 274, 276–279, 281, 282, 284–286, 291, 292
\glslongpluralaccessdisplay	149
\glslongtok	103, 180, 182, 201–205, 207, 209, 211, 214, 216–222, 224, 228, 229, 232, 233, 235, 237, 239, 242, 244, 246–254, 256, 258, 261, 263, 265, 268–271, 273–275, 277–279, 281, 282, 284–286, 288, 291, 292
\glslonguserfont	268–275
\glsmcols	344–347
\GLSname	298, 299
\Glsname	299
\glsname	298
\glsnameaccessdisplay ..	142, 143, 161, 163
\glsnamefont	160–165
\glsnavhyperlink	115
\glsnavhyperlinkname	78
\glsnavhypertarget	317, 318, 328–330, 343–348
\glsnavigation ..	317, 318, 328–330, 342–347
\glsnextpages	109
\glsnoidxdisplayloc	113
\glsnoidxdisplayloclisthandler	112

<code>\glsnoidxloclist</code>	113, 132	<code>\glsshorttok</code>	103, 180–182, 200–
<code>\glsnoidxnumberlistloophandler</code>	113		205, 207, 209, 211, 216–222, 224, 228,
<code>\glsnonexpages</code>	109		229, 232–235, 237, 239, 242–244, 246,
<code>\glsnonnumberlistfalse</code>	51		247, 249–254, 256, 258, 263, 265, 268–
<code>\glsnonnumberlisttrue</code>	51		271, 273–278, 281, 282, 284–286, 288–292
<code>\glsnopostdotfalse</code>	110	<code>\glssubentryitem</code>	
<code>\glsnopostdottrue</code>	110	 126, 127, 315–325, 327–329, 341, 352
<code>\glsnumberlistloop</code>	107, 108	<code>\glssymbolaccessdisplay</code>	145
<code>\glsnumlistlastsep</code>	112	<code>\glssymbolpluralaccessdisplay</code>	146
<code>\glsnumlistsep</code>	112	<code>\glstarget</code>	126,
<code>\glsopenbrace</code>	40, 121, 122		127, 315–325, 327–329, 341, 342, 352, 353
<code>\glsorder</code>	106	<code>\GLStext</code>	299
<code>\glspagelistwidth</code>	319, 321, 323, 325	<code>\Glstext</code>	299, 300
<code>\glspar</code>	130	<code>\glstext</code>	299
<code>\GLSpl</code>	90, 102, 135	<code>\glstextaccessdisplay</code>	143
<code>\Glspl</code>	49, 90, 102, 135	<code>\glstextformat</code>	56, 59
<code>\glspl</code>	48, 90, 102, 135	<code>\glstextup</code>	217
<code>\GLSplural</code>	300	<code>\glstreechildpredesc</code>	327, 328
<code>\Glsplural</code>	300	<code>\glstreechildprelocation</code> ...	327, 328, 330
<code>\glsplural</code>	300	<code>\glstreegroupheaderfmt</code>	
<code>\glspluralaccessdisplay</code>	143, 144	 327–330, 342–348, 350
<code>\glspluralsuffix</code>	124, 180, 184	<code>\glstreeindent</code>	328, 329, 340–342
<code>\glspostdescription</code>		<code>\glstreeitem</code>	326, 344, 352
 15, 16, 110, 173, 315–325, 327–331	<code>\glstreenamebox</code>	341, 342
<code>\glspostinline</code>	326	<code>\glstreenamefmt</code>	327–329, 331, 333–342
<code>\glspostlinkhook</code> .	55, 56, 68–72, 84, 185–194	<code>\glstreenavigationfmt</code> ..	328–330, 342–347
<code>\glsprestandardsort</code>	108	<code>\glstreepredesc</code>	327–329
<code>\glsresetentrylist</code>	116, 128	<code>\glstreeprelocation</code>	326–329, 331
<code>\glssee</code>	41, 43	<code>\glstreesubitem</code>	326, 352
<code>\glsseeformat</code>	39, 40, 45, 106, 113	<code>\glstreesubsubitem</code>	327, 353
<code>\glsseelist</code>	40	<code>\GLstrLetField</code>	31
<code>\glssetabbrvfmt</code>	54, 60, 82, 158–160,	<code>\glstype</code>	56, 58, 68–72, 136, 185–194
	162, 163, 165, 171, 185–189, 191, 192, 194	<code>\glunset</code>	44, 56, 93, 94, 137
<code>\glssetattribute</code> 201–205, 207, 209–214,		<code>\glswrite</code>	40, 105
216–222, 224, 228, 230, 232–235, 237,		<code>\glswriteentry</code>	8, 9
239, 242, 244, 246–249, 251–254, 256,		<code>\Glsxtr</code>	49
258, 259, 261, 263–265, 268, 269, 271–		<code>\glsxtr</code>	49
276, 278, 279, 281, 283, 285–288, 290–292		<code>\glsxtr@do@wrglossary</code>	8, 9, 11, 13
<code>\glssetcategoryattribute</code>		<code>\glsxtr@addloclistfield</code>	13
. 91, 102, 104, 136, 142, 154, 155, 157, 172		<code>\glsxtr@addunused</code>	44
<code>\glssetnoexpandfield</code>	11, 12	<code>\glsxtr@applyabbrvfmt</code>	194
<code>\glssettocitle</code>	109	<code>\glsxtr@applyabbrvstyle</code>	178, 180, 197
<code>\glsshortaccessdisplay</code>	147, 148	<code>\glsxtr@counterrecord</code>	126
<code>\glsshortpltok</code>		<code>\glsxtr@do@alsoindex@wrglossary</code>	13
182, 201–205, 207, 209, 211, 217, 219–		<code>\glsxtr@doooption</code>	5, 15, 16, 22, 24
222, 228, 229, 232, 233, 235, 237, 242,		<code>\glsxtr@fields</code>	124
244, 246–254, 263, 265, 268, 269, 271,		<code>\glsxtr@headentry@p</code>	85, 86
273–278, 282, 284–286, 288, 289, 291, 292		<code>\glsxtr@hyperoutsidefalse</code>	57
<code>\glsshortpluralaccessdisplay</code>	148	<code>\glsxtr@hyperoutsidettrue</code>	57

<code>\glxtr@ifnextpunc</code>	177	<code>\glxtralttreeSubSymbolDescLocation</code>	342
<code>\glxtr@ifpunctoken</code>	177	<code>\glxtralttreeSymbolDescLocation</code> ..	
<code>\glxtr@inc@linkcount</code>	58, 142	331, 341
<code>\glxtr@indexonly@saveentrycounter</code>		<code>\glxtrassignfieldfont</code>	61–68
.....	13, 25	<code>\glxtrautoindex</code>	167
<code>\glxtr@keylist</code>	47–49	<code>\glxtrautoindexassignsort</code>	167, 168
<code>\glxtr@label</code>	354	<code>\glxtrautoindexentry</code>	167
<code>\glxtr@langtag</code>	124	<code>\glxtrbookindexatendgroup</code>	352
<code>\glxtr@linkprefix</code>	124, 125	<code>\glxtrbookindexatsubendgroup</code>	352
<code>\glxtr@makeglossaries</code>	106	<code>\glxtrbookindexatsubsubendgroup</code> ..	352
<code>\glxtr@newabbreviation</code>	104, 180	<code>\glxtrbookindexbetween</code>	352
<code>\glxtr@next</code>	177	<code>\glxtrbookindexbookmark</code>	353
<code>\glxtr@org@do@wrglossary</code>	25	<code>\glxtrbookindexcols</code>	351
<code>\glxtr@org@dohyperlink</code>	78	<code>\glxtrbookindexcolspread</code>	351
<code>\glxtr@org@getgrouptitle</code>	114	<code>\glxtrbookindexfirstmarkfmt</code>	354
<code>\glxtr@org@newignoredglossary</code>	34	<code>\glxtrbookindexformatheader</code>	353
<code>\glxtr@org@makenoidxglossaries</code>	45	<code>\glxtrbookindexgroupskip</code>	353
<code>\glxtr@pluralsuffixes</code>	124	<code>\glxtrbookindexlastmarkfmt</code>	354
<code>\glxtr@process</code>	128, 129	<code>\glxtrbookindexmulticolseenv</code>	351
<code>\glxtr@provideignoredglossary</code>	36	<code>\glxtrbookindexname</code>	349, 352
<code>\glxtr@punctlist</code>	176, 177	<code>\glxtrbookindexparentchildsep</code>	349, 351
<code>\glxtr@record</code>	10, 124	<code>\glxtrbookindexparentschildsep</code> .	
<code>\glxtr@recordsee</code>	11	351, 352
<code>\glxtr@resource</code>	123, 124	<code>\glxtrbookindexprelocation</code> ...	349, 352
<code>\glxtr@s@newignoredglossary</code>	34	<code>\glxtrbookindexsubbetween</code>	352
<code>\glxtr@s@provideignoredglossary</code> ...	36	<code>\glxtrbookindexsubname</code>	353
<code>\glxtr@saveentrycounter</code>	8, 9, 11, 76	<code>\glxtrbookindexsubprelocation</code>	353
<code>\glxtr@setaccessdisplay</code>	165	<code>\glxtrbookindexsubsubbetween</code>	352
<code>\glxtr@setbookindexmark</code>	354	<code>\glxtrbookindexthepage</code>	354
<code>\glxtr@setup@record</code>	12, 13, 24, 25	<code>\glxtrcat</code>	47–49
<code>\glxtr@shortcutsval</code>	124	<code>\glxtrchecknohyperfirst</code>	61–63
<code>\glxtr@texencoding</code>	124	<code>\glxtrComputeTreeIndent</code>	341
<code>\glxtr@usesee</code>	39	<code>\glxtrComputeTreeSubIndent</code>	341
<code>\glxtr@warnonexistsordo</code>	7, 13, 38	<code>\GlsXtrdefaultsubsequentfmt</code> ...	197, 198
<code>\glxtr@writefields</code>	123	<code>\glxtrdefaultsubsequentfmt</code> ...	196, 198
<code>\glxtrabbrvfootnote</code>		<code>\GlsXtrdefaultsubsequentplfmt</code> .	197, 198
.....	205–207, 228–230, 242–244, 263–265	<code>\glxtrdefaultsubsequentplfmt</code> .	197, 198
<code>\glxtrabbrvpluralsuffix</code>	124,	<code>\GlsXtrDefineAbbreviationShortcuts</code>	
184, 201, 203, 206, 208, 209, 211, 214,		19, 20
217, 231, 245, 268, 277, 280, 283, 288, 290		<code>\GlsXtrDefineAcShortcuts</code>	19, 20
<code>\glxtrabbrvtype</code>	16, 17, 182	<code>\GlsXtrDefineOtherShortcuts</code>	19, 20
<code>\glxtractivatenopost</code>	109	<code>\glxtrdetoklocation</code>	135
<code>\glxtraddallcrossrefs</code>	43	<code>\glxtrdiscardperiod</code>	174
<code>\glxtralias</code>	76	<code>\glxtrdisplayendloc</code>	116
<code>\glxtrAltTreeIndent</code>	331	<code>\glxtrdisplayendloohook</code>	117
<code>\glxtralttreeInit</code>	341, 346, 347	<code>\glxtrdisplayingleloc</code>	116, 117
<code>\glxtrAltTreePar</code>	331	<code>\glxtrdisplaystartloc</code>	116
<code>\glxtrAltTreeSetHangIndent</code> ...	331, 341	<code>\glxtrdoautoindexname</code>	76, 77, 164
<code>\glxtrAltTreeSetSubHangIndent</code> ...	342	<code>\glxtrdopostpunc</code>	207, 230, 244, 265

<code>\glxtrdowrglossaryhook</code>	77	223, 226–228, 230, 232, 234, 236, 237,
<code>\glxxtremsuffix</code>	246, 248, 250,	240–242, 244, 246, 248, 250, 252, 254,
	251, 253, 254, 256, 258, 260, 261, 264, 266	255, 257, 259, 261, 263, 264, 266, 268,
<code>\GlsXtrEnableEntryCounting</code>	102	270, 272, 275, 278, 281, 283, 286, 289, 291
<code>\GlsXtrEnableEntryUnitCounting</code>	90	<code>\glxtrfullsep</code>
<code>\GlsXtrEnableOnTheFly</code>	47, 49	. 183, 201–204, 206–213, 215, 217–225,
<code>\glxxtrendfor</code>	29	227, 229, 231–241, 243, 245–262, 264–
<code>\glxtrfieldlistgadd</code>	125	267, 277, 278, 280, 282, 285–287, 291, 292
<code>\glxtrfieldtitlecase</code>	159–162, 166	<code>\glxtrngenabbrvfmt</code>
<code>\glxtrfieldtitlecasecs</code>	158	54
<code>\glxtrfieldxifinlist</code>	130	<code>\glxtrgetgrouptitle</code>
<code>\glxtrfirstscfont</code>	217	115, 353
<code>\glxtrfirstsmfont</code>	231	<code>\glxtrgroupfield</code>
<code>\GlsXtrFmtDefaultOptions</code>	27, 28	131
<code>\glxtrfmtdisplay</code>	27, 28	<code>\Glsxtrheadfirst</code>
<code>\GlsXtrFmtField</code>	27, 28	295
<code>\glxtrfootnotename</code>		<code>\glxtrheadfirst</code>
	205, 207, 228, 229, 242, 243, 263, 265	295
<code>\GlsXtrFormatLocationList</code>	51, 54, 338–340	<code>\Glsxtrheadfirstplural</code>
<code>\GLSxtrfull</code>	17, 18, 304, 305	295
<code>\Glsxtrfull</code>	17, 18, 305	<code>\glxtrheadfull</code>
<code>\glxtrfull</code>	17, 18, 304	296
<code>\Glsxtrfullformat</code>		<code>\glxtrheadfull</code>
	183, 196, 198, 199, 201, 203, 206,	296
	208, 210, 212, 215, 218, 220, 222, 223,	<code>\Glsxtrheadfullpl</code>
	226–228, 230, 232, 234, 236, 237, 240,	296
	241, 243, 244, 246, 248, 250, 252, 254,	<code>\Glsxtrheadlong</code>
	255, 257, 259, 261, 263, 264, 266, 269,	296
	270, 272, 275, 278, 281, 283, 286, 289, 291	<code>\glxtrheadlong</code>
<code>\glxtrfullformat</code>		295
	183, 196, 198, 199, 201, 203,	<code>\Glsxtrheadlongpl</code>
	206, 208, 210, 212, 215, 218, 220, 222,	296
	223, 226–228, 230, 232, 234, 236, 237,	<code>\glxtrheadlongpl</code>
	240–242, 244, 246, 248, 250, 251, 253,	295
	255, 257, 259, 261, 262, 264, 266, 268,	<code>\Glsxtrheadname</code>
	270, 272, 275, 278, 281, 283, 286, 289, 291	295
<code>\GLSxtrfullpl</code>	17, 18, 304, 305	<code>\glxtrheadname</code>
<code>\Glsxtrfullpl</code>	17, 18, 305	126, 295
<code>\glxtrfullpl</code>	17, 18, 304, 305	<code>\Glsxtrheadplural</code>
<code>\Glsxtrfullplformat</code>		295
	183, 196, 198, 199, 201, 204, 206, 208,	<code>\glxtrheadplural</code>
	210, 212, 215, 218, 220, 222, 223, 226,	295
	227, 229, 230, 232, 234, 236, 238, 240,	<code>\Glsxtrheadshort</code>
	242, 243, 245, 247, 248, 250, 252, 254,	295
	255, 257, 259, 261, 263, 264, 266, 269,	<code>\glxtrheadshort</code>
	270, 272, 275, 278, 281, 284, 286, 289, 291	295
<code>\glxtrfullplformat</code>		<code>\Glsxtrheadshortpl</code>
	195, 196, 198, 199, 201, 203,	295
	206, 208, 210, 212, 215, 218, 220, 222,	<code>\glxtrheadshortpl</code>
		295
		<code>\Glsxtrheadtext</code>
		295
		<code>\glxtrheadtext</code>
		295
		<code>\glxtrhyperlink</code>
		79, 117
		<code>\glxtrhyphensuffix</code>
		278, 286
		<code>\glxtrtrifcounttrigger</code>
		93, 94
		<code>\glxtrtrifcustomdiscardperiod</code>
		174
		<code>\glxtrtrifemptyglossary</code>
		116, 121, 128
		<code>\glxtrtrifhasfield</code>
		32, 75, 349
		<code>\glxtrtrifhyphenstart</code>
		277, 279, 282, 285, 287
		<code>\glxtrtrifindexing</code>
		76
		<code>\glxtrtrifinmark</code>
		59, 86–89, 294–296
		<code>\glxtrtrifnextpunc</code>
		177, 178
		<code>\glxtrtrifperiod</code>
		174, 176
		<code>\glxtrtrifrecordtrigger</code>
		137–139
		<code>\glxtrtrifwasfirstuse</code>
		60–63, 68–72, 74, 104,
		175, 176, 185, 188–194, 207, 230, 244,
		265, 266, 269, 271, 273, 283, 284, 288, 290

<code>\glxtrinclinkcounter</code>	141	<code>\glxtrlongshortname</code>	200, 217, 232, 246, 247, 268, 269, 277, 282
<code>\glxtrindexaliased</code>	75, 76	<code>\glxtrlongshortuserdesname</code> ..	271, 274
<code>\glxtrindexseealso</code>	42, 43	<code>\glxtrmarkhook</code>	293, 294
<code>\glxtrinithyperoutside</code>	58	<code>\glxtrnewabbrevpresetkeyhook</code>	181
<code>\glxtrnitwrgloss</code>	58, 136	<code>\glxtrnewnumber</code>	18
<code>\glxtrnitwrglossbeforefalse</code> ...	56, 57	<code>\glxtrnewsymbol</code>	18
<code>\glxtrnitwrglossbeforetrue</code>	57	<code>\glxtrNoGlossaryWarning</code>	20, 118
<code>\Glsxtrinlinefullformat</code> 183, 185, 198,		<code>\GlsXtrNoGlsWarningAutoMake</code>	122
199, 206, 208, 209, 211, 213, 215, 221,		<code>\GlsXtrNoGlsWarningBuildInfo</code>	122
223–225, 227, 229, 231, 236–239, 241,		<code>\GlsXtrNoGlsWarningCheckFile</code>	122
243, 245, 253, 255–257, 259, 260, 262,		<code>\GlsXtrNoGlsWarningEmptyMain</code> ..	121, 122
265, 266, 270, 272, 280, 284, 289, 292, 311		<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...	122
<code>\glxtrinlinefullformat</code>		<code>\GlsXtrNoGlsWarningEmptyStart</code>	121
..... 183, 185, 186, 198, 199,		<code>\GlsXtrNoGlsWarningHead</code>	121
206, 208, 209, 211, 213, 215, 221, 223–		<code>\GlsXtrNoGlsWarningMisMatch</code>	122
225, 227, 229, 231, 235, 237–239, 241,		<code>\GlsXtrNoGlsWarningNoOut</code>	122
243, 245, 253–255, 257, 258, 260, 262,		<code>\GlsXtrNoGlsWarningTail</code>	122
264, 266, 270, 272, 280, 284, 289, 291, 311		<code>\glxtrnopostpunc</code>	110
<code>\Glsxtrinlinefullplformat</code> 184, 187, 198,		<code>\glxtronlydesname</code>	292
199, 207, 208, 210, 212, 213, 215, 222–		<code>\glxtronlydescsort</code>	292
225, 227, 229, 231, 236–238, 240, 241,		<code>\glxtronlyname</code>	291
243, 245, 253, 255–257, 259, 261, 262,		<code>\glxtronlysuffix</code>	291
265, 266, 270, 273, 280, 284, 289, 292, 312		<code>\glxtrorg@ifKV@glslink@hyper</code>	55
<code>\glxtrinlinefullplformat</code>		<code>\glxtrorglong</code>	180, 202, 279
..... 183, 184, 186, 187, 198, 199,		<code>\glxtrorgshort</code>	180, 202
206, 208, 209, 211, 213, 215, 221, 223–		<code>\GLSxtrp</code>	85
225, 227, 229, 231, 236–239, 241, 243,		<code>\Glsxtrp</code>	85
245, 253, 254, 256, 257, 259, 260, 262,		<code>\glxtrp</code>	85, 87
264, 266, 270, 272, 280, 284, 289, 292, 311		<code>\glxtrparen</code>	175, 183,
<code>\glxtrininsertinsidefalse</code>	200	201–204, 206–213, 215, 217–225, 227,	
<code>\glxtrlocationhyperlink</code>	117	229, 231–241, 243, 245–257, 259–262,	
<code>\glxtrlocrangefmt</code>	116, 117	264–267, 277, 278, 280, 282, 285–287, 292	
<code>\GLSxtrlong</code>	17, 18, 302, 303	<code>\Glsxtrpl</code>	49
<code>\Glsxtrlong</code>	17, 18, 303	<code>\glxtrpl</code>	49
<code>\glxtrlong</code>	17, 18, 302	<code>\glxtrpostdescription</code> .	110, 157, 173, 326
<code>\glxtrlonghyphen</code>	283, 284	<code>\glxtrposthyphenlong</code>	288, 290
<code>\glxtrlonghyphennoshort</code>	280, 281	<code>\glxtrposthyphenshort</code>	283, 284
<code>\glxtrlonghyphenshort</code>	278	<code>\glxtrposthyphensubsequent</code>	
<code>\glxtrlongnoshortdesname</code> .	214, 261, 279	283, 284, 288, 290
<code>\glxtrlongnoshortname</code>		<code>\glxtrpostlink</code>	174
..... 216, 224, 239, 256, 258, 281		<code>\glxtrpostlinkendsentence</code>	174
<code>\GLSxtrlongpl</code>	17, 18, 303, 304	<code>\glxtrpostlinkhook</code>	174
<code>\Glsxtrlongpl</code>	17, 18, 304	<code>\glxtrpostlocalreset</code>	90–92, 100
<code>\glxtrlongpl</code>	17, 18, 303	<code>\glxtrpostlocalunset</code>	89, 91, 99
<code>\glxtrlongshortdesname</code>		<code>\glxtrpostlongdescription</code>	33
..... 202, 218, 233, 247, 249, 278, 284		<code>\glxtrpostnamehook</code>	161–164, 166
<code>\glxtrlongshortdescsort</code>		<code>\GlsXtrPostNewAbbreviation</code>	
.... 202, 218, 233, 247, 249, 274, 278, 284	 182, 198, 199, 201–205,	

207, 209–214, 216, 217, 219–222, 224, 228, 229, 232–235, 237, 239, 242, 244, 246–249, 251–254, 256, 258, 259, 261, 263, 265, 268, 269, 271, 273, 274, 276, 278, 279, 281, 282, 284, 286–288, 290–292	\glxtrrmsuffix
\glxtrpostreset 90, 91, 99 232, 234, 235, 237, 239, 240, 242, 244
\glxtrpostunset 89, 91, 99	\Glsxtrsubsequentfmt
\glxtrprelocation 195, 198, 214, 225, 226, 239, 241, 257, 258, 260, 262, 280, 283, 288
..... 316, 318, 320, 322, 324, 326, 349	\glxtrsubsequentfmt
\glxtrprotectlinks 79, 80 195, 198, 214, 225, 226, 239, 241, 257, 258, 260, 262, 280, 283, 288
\GlsXtrRecordCounter 11	\Glsxtrsubsequentplfmt
\glxtrrecordtriggervalue 136 195, 198, 214, 225, 226, 239, 241, 257, 258, 260, 262, 280, 283, 288
\glxtrregularfont 54, 60, 61	\glxtrsubsequentplfmt
\glxtrresourcecount 123 195, 198, 214, 225, 226, 239, 241, 256, 258, 260, 262, 280, 283, 288
\glxtrresourcefile 123	\glxtrsupplocationurl 117, 118
\glxtrresourceinit 123	\glxtrtagfont 173
\glxtrrestoremarkhook 293, 294	\Glsxtrtitlefirst 295, 296, 309
\glxtrrestorepostpunc 110, 111	\glxtrtitlefirst 295, 296, 309
\glxtrscfont 217	\Glsxtrtitlefirstplural 295, 296, 309, 310
\glxtrscsuffix	\glxtrtitlefirstplural 295, 296, 309
.... 218, 219, 221, 222, 225, 226, 228, 230	\Glsxtrtitlefull 295, 296, 311
\GlsXtrSetActualChar 170	\glxtrtitlefull 295, 296, 311
\glxtrsetaliasnoindex 13, 75, 76	\Glsxtrtitlefullpl 295, 296, 312
\GlsXtrSetEncapChar 170	\glxtrtitlefullpl 295, 296, 311, 312
\GlsXtrSetEscChar 170	\Glsxtrtitlelong 295, 296, 310
\glxtrsetfieldifexists 31, 32	\glxtrtitlelong 295, 296, 310
\GlsXtrSetLevelChar 170	\Glsxtrtitlelongpl 295, 296, 311
\glxtrsetpopts 84	\glxtrtitlelongpl 295, 296, 310
\glxtrsetupfulldefs	\Glsxtrtitlename 295, 296, 307
..... 185–187, 207, 230, 244, 266	\glxtrtitlename 295, 296, 307
\GLSxtrshort 17, 18, 88, 89, 297	\glxtrtitleorpdforheading
\Glsxtrshort 17, 18, 297, 298 23, 126, 127, 295, 296
\glxtrshort 17, 18, 297	\Glsxtrtitleplural 295, 296, 308
\glxtrshortdescname ... 211, 222, 236, 254	\glxtrtitleplural 295, 296, 308
\glxtrshorthyphen 289	\Glsxtrtitleshort 295, 296, 306
\glxtrshorthyphenlong 286	\glxtrtitleshort 295, 296, 306
\glxtrshortlongdescname	\Glsxtrtitleshortpl 295, 296, 307
..... 204, 220, 234, 250, 252, 286, 289	\glxtrtitleshortpl 295, 296, 306
\glxtrshortlongdescsort	\Glsxtrtitletext 295, 296, 308
.... 204, 220, 234, 250, 252, 275, 286, 289	\glxtrtitletext 295, 296, 307, 308
\glxtrshortlongname	\GlsXtrTotalRecordCount 136
203, 219, 233, 249, 251, 271, 274, 285, 288	\glxtrtreetopindent 331, 340
\glxtrshortlonguserdescname .. 273, 275	\glxtrundefaction .. 7, 13, 25, 34, 35, 37, 38
\glxtrshortnolongname . 209, 221, 235, 253	\glxtrundeftag 25, 112, 113
\GLSxtrshortpl 17, 18, 297, 298	\glxtrunsrtdo 129, 130
\Glsxtrshortpl 17, 18, 298	\GlsXtrUseAbbrStyleFmts
\glxtrshortpl 17, 18, 297 202, 204, 211–213, 216, 217, 219, 221, 224, 233, 235, 238,
\glxtrsmfont 231	

247, 249, 251, 252, 255, 260, 263, 271, 273, 274, 276, 279–281, 285, 287, 290, 293	\ifdefempty 7–9, 30, 34–36, 39, 40, 58, 60, 90, 102, 106, 108, 117, 128, 131, 136, 172, 179, 194, 351
\GlsXtrUseAbbrStyleSetup 210, 212, 213, 215, 216, 224, 226, 238, 240, 255, 259, 260, 263	\ifdefequal 45, 122, 131, 164
\glsxtruserfield 267	\ifdefstring 6, 32, 167, 172, 350
\glsxtruserparen 268–276	\ifdefvoid ... 39, 42–44, 79, 96, 114, 117, 132
\glsxtrusersuffix 268, 269, 272, 275	\ifdim 50, 51, 105, 332–340
\glsxtruseealsoformat 40	\IfFileExists 21, 118, 122, 123, 125, 314
\glsxtruseeeformat 39	\ifglossaryexists 38
\GlsXtrWarnDeprecatedAbbrStyle 178, 199	\ifglssacronym 17, 121
\GlsXtrWarning 47–49	\ifglssacrshortcuts 19
\glsxtrword 180	\ifglssautomake 108, 122, 125
\glsxtrwordsep .. 180, 277, 279, 282, 285, 287	\ifglssentrycounter 32
\glsxtrwrglossmark 22	\ifglssentryexists 8, 37, 38, 47–49, 51, 60, 131, 154, 173, 174
H	\ifglssfieldeq 153
\hangindent . 328, 329, 331, 340–342, 346, 347	\ifglsshasfield 27, 28, 267
\hbox 315	\ifglsshaslong 96, 139, 140
\hfill 315	\ifglsshasparent 126, 127, 129, 132, 333, 335, 336
\href 79	\ifglsshasshort 40, 54, 60
\hsize 50, 51	\ifglsshasymbol 175, 327–329, 331
\hss 315	\ifglssindexonlyfirst 76
\hyperlink 79	\ifglssnogroupskip 316, 318–328, 330, 342, 350
\hyperpage 166	\ifglssnonumberlist 53
\hyperref 79, 118	\ifglssnopostdot 15, 110
hyperref package 80, 166, 293, 306	\ifglssanitizesort 108
I	\ifglssubentrycounter 32
\if 47	\ifglssused 44, 74, 76, 92, 101, 104, 194, 333–335, 337–339
\if@glsxtr@autoseeindex 23, 24, 39, 42	\ifglssxindy 118, 120
\if@glsxtr@format@override 167	\ifglsxtr@hyperoutside 59
\if@glsxtr@docdefrestricted 45	\ifglsxtr@trinitwrglossbefore ... 57, 59, 137
\if@glsxtr@indexcrossrefs 14, 43	\ifglsxtr@insertinside .. 188–194, 196, 197, 201, 203, 204, 206–215, 218, 220– 248, 250–266, 268–270, 272, 273, 275, 277, 279, 282, 284, 285, 287, 289, 291, 292
\ifblank 26, 47–49, 105	\ifHy@hyperindex 166
\ifcase .. 7, 12, 19, 20, 22, 46, 57, 111, 327, 352	\ifinlistcs 29, 45
\ifcsdef 25, 32, 34–37, 58, 72, 73, 84–89, 97, 109, 114, 115, 126, 130, 133, 135, 141, 159, 160, 162–165, 175, 178, 194, 198, 318–325	\ifinner 23
\ifcsstring 25, 154, 197	\ifKV@glsslink@hyper 55, 58, 59
\ifcsundef 30, 32, 34–36, 45, 50, 52, 80, 91, 97–100, 114, 115, 118, 130, 141, 142, 154, 197–200, 315, 332, 333, 340, 354	\ifKV@glsslink@local 56, 137
\ifcsvoid 43, 153	\ifKV@glsslink@noindex 8, 9, 11, 28, 76
\ifdef 13, 18, 24, 28, 37, 38, 40, 41, 50, 51, 74, 75, 78, 79, 86–88, 109, 112, 113, 124, 133, 156, 157, 170, 173, 267, 294, 306–312, 315–318, 326–330, 342–347, 350, 353, 354	\ifmmode 23
	\ifnum 14, 92, 93, 100, 101, 114, 123, 136, 328, 329, 341
	\ifstrempy 126, 133, 142
	\ifstrequal 16, 21
	\ifthenelse 121, 122

<code>\IfTrackedLanguageFileExists</code>	312	<code>\makeatother</code>	169
<code>\ifundef</code>	30, 105, 172, 173	<code>\makebox</code>	315, 341, 342
<code>\ifx</code> ...	8–10, 41, 50, 51, 105, 109, 116, 117, 125, 167, 168, 170, 177, 179–181, 276, 348	<code>\makefirstuc</code>	173
<code>\immediate</code>	92, 101, 118, 119, 125	<code>makeglossaries</code>	111
<code>\index</code>	167	<code>\makeglossaries</code>	105, 119–122, 125
<code>\indexspace</code> .	316, 327–330, 342–348, 350, 353	<code>\makeglossary</code>	106
<code>\input</code>	312	<code>makeindex</code>	355
<code>\inputencodingname</code>	124	<code>makeindex</code>	105
<code>\istfilename</code>	106	<code>\makenoidxglossaries</code>	120
<code>\item</code>	120, 121, 315–318, 326–328, 343, 344	<code>\MakeTextUppercase</code>	295
J			
<code>\jobname</code>	118, 120–123, 125	<code>\MakeUppercase</code>	295, 296
K			
<code>\key@ifundefined</code>	11, 12, 26, 72, 128, 131	<code>\marginpar</code>	23
<code>\KV@glslink@hyperfalse</code>	61, 74, 75, 80	<code>\markboth</code>	294
<code>\KV@glslink@hypertrue</code>	80	<code>\markright</code>	294
<code>\KV@glslink@noindexfalse</code>	75	<code>\maxdimen</code>	50, 51
<code>\KV@glslink@noindextrue</code>	75, 81	<code>\mbox</code>	317, 318, 341
L			
<code>\LaTeX</code>	120, 121	<code>\medskip</code>	121, 122, 130
<code>\leaders</code>	315	<code>\MessageBreak</code>	46, 49, 92, 93, 101, 105, 108, 109, 197
<code>\leavevmode</code>	34, 58	<code>mfirstuc package</code>	172
<code>\let</code>	5, 7–13, 15, 17–19, 23–25, 27, 29, 33, 45, 46, 49, 50, 52, 53, 55, 56, 58–78, 80, 81, 84, 90–92, 99, 100, 102, 104–115, 123, 125, 128, 129, 131, 136, 137, 142, 159–169, 172, 173, 177–181, 185–194, 196–198, 207, 230, 244, 266, 293–296, 326, 327, 331, 333, 344, 351–354	<code>\mfirstucMakeUppercase</code>	61–72, 74, 82–84, 86, 88, 89, 95, 103, 140, 143–152, 161, 162, 166, 186, 187, 189, 190, 192, 194–196
<code>\letabbreviationstyle</code> ..	207, 209, 210, 212, 215, 216, 222, 223, 236, 238, 254, 255	<code>\mfu@checkword@arg</code>	172, 173
<code>\letcs</code>	26, 30, 39, 40, 44, 59, 72, 112–114, 130–132, 159–166, 354	<code>\mfu@checkword@do</code>	173
<code>\levelchar</code>	170	N	
<code>\listadd</code>	97	<code>\NeedsTeXFormat</code>	5, 314, 349
<code>\listbreak</code>	172	<code>\new@glossaryentry</code>	46, 108
<code>\listcsadd</code>	28	<code>\new@ifnextchar</code>	27, 73, 95, 133, 134, 137–139, 176, 184–193
<code>\listcseadd</code>	29, 98	<code>\newabbr</code>	17, 18
<code>\listcsgadd</code>	29, 45	<code>\newabbreviation</code>	17, 18
<code>\listcsxadd</code>	29, 97	<code>\newabbreviationhook</code>	182
<code>\loadglsentries</code>	46, 119	<code>\newabbreviationstyle</code>	200, 202–205, 207, 209–222, 224, 226, 228, 229, 232–236, 238, 240, 242, 243, 246–256, 258–261, 263, 265, 268, 269, 271, 273–275, 277– 279, 281, 282, 284–286, 288, 289, 291, 292
<code>\long</code>	33	<code>\newacronym</code>	102, 104
M			
<code>\MakeAcronymsAbbreviations</code>	104	<code>\newacronymhook</code>	103
<code>\makeatletter</code>	118, 123, 169	<code>\newacronymstyle</code>	104
		<code>\newcommand</code>	5–8, 10–12, 14–40, 42, 44–50, 52–54, 56, 57, 60, 61, 72, 73, 75–80, 84, 86–91, 93, 95–99, 101, 102, 104, 105, 109–131, 133–158, 164– 180, 182–194, 196–200, 202, 204, 205, 209, 211, 214, 216, 217, 231, 245, 246, 267, 268, 270, 273, 277, 279, 281, 282,

285, 287, 290, 292, 294–312, 314, 316, 326, 330–333, 340, 341, 349, 350, 353, 354	\nr 7, 12, 14, 19, 20, 22, 57, 111
\newcount 14, 123, 133	\ns@GLSxtrfull 186
\newcounter 141	\ns@Glsxtrfull 185
\newentry 18	\ns@glxtrfull 184
\newglossary 16, 106	\ns@GLSxtrfullpl 187
\newglossaryentry 18, 46, 91, 98, 103, 156, 157, 182	\ns@Glsxtrfullpl 186
\newglossaryentry options	\ns@glxtrfullpl 186
alias 15, 38, 41–44	\ns@GLSxtrlong 190
desc 146, 147, 151	\ns@Glsxtrlong 190
descplural 147, 151, 152	\ns@glxtrlong 189
first 78, 144, 150, 200, 301, 302, 308, 355	\ns@GLSxtrlongpl 193
firstplural 144, 145, 150, 151, 200, 301, 302, 309, 355	\ns@Glsxtrlongpl 193
group 131	\ns@glxtrlongpl 192
loclist 28	\ns@GLSxtrshort 189
long 149, 152, 310	\ns@Glsxtrshort 188
longplural 149, 152, 310	\ns@glxtrshort 187, 188
name 39, 40, 142, 143, 149, 150, 167, 298, 299, 307	\ns@GLSxtrshortpl 192
plural 143, 144, 150, 200, 300, 308	\ns@Glsxtrshortpl 191
see 15, 24, 38, 39, 41, 44, 46, 106	\ns@glxtrshortpl 191
seealso 15, 38, 40, 41, 43, 44, 366	\null 20
short 148, 152, 179	\number 97–100, 123, 133
shortplural 148, 152, 179	\numexpr 97, 98, 100
symbol 145, 151	
symbolplural 146, 151	O
text 78, 143, 150, 200, 202, 299, 300, 307	\or 7, 13, 19, 20, 22, 46, 57, 327, 352
\newglossarystyle 351	\org@glossaryentrynumbers 51, 109
\newif 57, 166, 200	\org@glossarytitle 109
\newlength 331	\org@ifKV@glslink@hyper 58, 59
\newnum 18	
\newrobustcmd 27, 28, 30–32, 40, 41, 73, 74, 84, 85, 95, 110, 114, 126, 127, 130, 133–135, 137–139, 165, 172, 173, 184–194, 276, 294, 297–305, 333–339	P
\newsym 18	\p@glsl@hyp@opt 77
\newterm 156	package options:
\newtoks 178, 179	abbreviations 16, 17
\newwrite 105	accsupp 20, 142
\nobreak 317, 318, 327–330, 343–346	acronym 17
\NoCaseChange 86–89, 127, 296–305	automake 108, 120, 125
\noexpand 10, 11, 21, 40, 42, 43, 103, 118, 119, 123, 129–131, 141, 168, 169, 182, 314, 352	true 125
\nofiles 121	autoseeindex 24
\noindent 121, 122, 329, 330, 344–347	false 23
\nopagebreak 327–330, 342–349, 353	debug
\nopostdesc 34, 47–49, 110, 157	showtargets 79
	docdef 14, 45, 46, 91, 98
	false 46
	restricted 14
	true 46
	docdefs
	restricted 45
	nonumberlist 51
	nopostdot 15
	false 15

<code>\rGLS</code>	135	<code>\subitem</code>	326, 327
<code>\rGls</code>	135	<code>\subsubitem</code>	327
<code>\rgls</code>	135		
<code>\rGLSformat</code>	139		
<code>\rGlsformat</code>	138		
<code>\rglsformat</code>	137, 140		
<code>\rGLSpl</code>	135		
<code>\rGlspl</code>	135		
<code>\rglspl</code>	135		
<code>\rGLSplformat</code>	139		
<code>\rGlsplformat</code>	138		
<code>\rglsplformat</code>	138, 140		
<code>\romannumeral</code>	331–333, 340		
	S		
<code>\s@glxtrfmt</code>	27		
<code>\s@glshyp@opt</code>	77		
<code>\s@glxtr@enabletagging</code>	171, 172		
<code>\s@glxtrfmt</code>	27		
<code>\s@glxtrifhasfield</code>	30		
<code>\s@printunsrtglossary</code>	127, 130		
<code>\seealso</code>	40, 41		
<code>\seename</code>	39		
<code>\setabbreviationstyle</code>	104, 202, 210		
<code>\setacronymstyle</code>	104, 105		
<code>\setentrycounter</code>	116		
<code>\SetGenericNewAcronym</code>	104		
<code>\setglossarystyle</code>	22, 109, 315–318, 327, 329, 330, 342–348, 351		
<code>\setkeys</code>	9, 21, 24, 28, 58, 60, 76, 103, 109, 136, 180, 181		
<code>\setlength</code>	50, 51, 326, 328, 329, 331, 341, 342, 344–346, 351		
<code>\settowidth</code>	105, 331–340		
<code>\setupglossaries</code>	5, 24		
<code>\sfcode</code>	15, 16, 175, 326		
<code>\small</code>	23		
<code>\space</code>	6, 11, 40, 41, 46, 47, 49, 76, 90–92, 98–102, 104–109, 119, 121, 125, 129, 130, 133, 175, 179, 183, 202, 315, 316, 326–329, 331, 340, 349		
<code>\spacefactor</code>	15, 16, 175, 181, 326		
<code>\stepcounter</code>	141		
<code>\string</code>	6, 10, 11, 40, 41, 45–47, 49, 53, 59, 72, 73, 76, 85, 90–92, 98–102, 104–109, 118– 126, 129, 130, 133, 160, 162–165, 167, 354		
<code>\strut</code>	315–325, 329		
<code>\subglossentry</code>	110, 132, 315–325, 327–329, 341, 352		
		T	
		<code>\tablehead</code>	322–325
		<code>\tabletail</code>	322–325
		<code>\tabularnewline</code>	318–326
		<code>\TeX</code>	120
		<code>\texorpdfstring</code>	28, 86–88, 294, 306–312
		textcase package	293
		<code>\textsc</code>	217
		<code>\textsmaller</code>	231
		<code>\texttt</code>	23, 119–122
		<code>\the</code>	103, 117, 123, 170, 171, 182, 200–205, 207, 209–214, 216–222, 224, 228–230, 232– 235, 237, 239, 242–244, 246–254, 256, 258, 259, 261, 263–265, 268–279, 281–292
		<code>\theglentrycounter</code>	8–10, 58, 60, 137
		<code>\theHglentrycounter</code>	8–10, 58, 60, 137
		<code>\theindex</code>	166
		<code>\this@dialect</code>	312
		<code>\thisgrptitle</code>	353
		<code>\toks@</code>	117, 170, 171
		tracklang package	124
		U	
		<code>\u</code>	133
		<code>\undef</code>	13, 172
		<code>\underline</code>	173
		<code>\unskip</code>	34, 45, 315
		<code>\usepackage</code>	121, 122
		V	
		<code>\val</code>	7, 12, 14, 19, 20, 22, 57, 111
		W	
		<code>\warn@nomakeglossaries</code>	107
		<code>\warn@noprntglossary</code>	107, 110
		<code>\write</code>	40, 92, 101, 106, 118, 119, 125
		X	
		<code>\x</code>	117, 141
		<code>\xcapitalisewords</code>	158
		<code>\xdef</code>	110, 129
		<code>\xifinlist</code>	97
		<code>\xifinlistcs</code>	29
		xindy	355
		xindy	105
		xkeyval package	5

\XKV@checkchoice	53	\XKV@resa	53
\XKV@plfalse	53	\XKV@sttrue	53