

`glossaries-extra.sty v1.23: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2017-11-12

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	32
1.3.1 Existence Checks	37
1.3.2 Document Definitions	44
1.3.3 Existing Glossary Style Modifications	49
1.3.4 Entry Formatting, Hyperlinks and Indexing	54
1.3.5 Entry Counting	89
1.3.6 Acronym Modifications	102
1.3.7 Indexing and Displaying Glossaries	104
1.3.8 Support for bib2gls	132
1.4 Integration with glossaries-accsupp	140
1.5 Categories	150
1.6 Abbreviations	176
1.6.1 Abbreviation Styles Setup	195
1.6.2 Predefined Styles (Default Font)	198
1.6.3 Predefined Styles (Small Capitals)	214
1.6.4 Predefined Styles (Fake Small Capitals)	228
1.6.5 Predefined Styles (Emphasized)	242
1.6.6 Predefined Styles (User Parentheses Hook)	264
1.6.7 Predefined Styles (Hyphen)	273
1.6.8 Predefined Styles (No Short on First Use)	287
1.7 Using Entries in Headings	289
1.8 Multi-Lingual Support	309
2 Style Adjustments (glossaries-extra-stylemods.sty)	310
2.1 Package Initialisation	310
2.2 List-Like Styles	311
2.3 Longtable Styles	314
2.4 Long Ragged Styles	316
2.5 Supertabular Styles	318
2.6 Super Ragged Styles	320
2.7 Inline Style	322
2.8 Tree Styles	322
2.9 Multicolumn Styles	339

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

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/11/12 v1.23 (NLCT)]

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

Requires etoolbox package.
4 \RequirePackage{etoolbox}

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

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

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

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

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

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

```

```

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

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

```

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

`\@glstr@record` Does nothing by default.

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

`\glstr@recordsee` Does nothing by default.

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

`\ultnumberformat`

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

```

`\ultNumberFormat`

```

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

```

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

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

```

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

```

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

```

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

```

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

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

```

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

```


122 \beginngroup

Save the label in case it's needed.

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

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

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

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

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

Check if thevalue has been set.

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

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

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

Save the entry counter.

```
142     \glsxtr@saveentrycounter
```

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

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

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

```
146     \let\theglentrycounter\@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 \glstr@dooption{nopostdot=false}%
363 \ifstrequal{#1}{dot}%
364 {%
365 \renewcommand*{\@glstr@defpostpunc}{%
366 \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
367 }%
368 }%
369 {%
370 \ifstrequal{#1}{comma}%
371 {%
372 \renewcommand*{\@glstr@defpostpunc}{%
373 \renewcommand*{\glspostdescription}{,}%
374 }%
375 }%
376 {%
377 \ifstrequal{#1}{none}%
378 {%
379 \glstr@dooption{nopostdot=true}%
380 \renewcommand*{\@glstr@defpostpunc}{%
381 \renewcommand*{\glspostdescription}{}%
382 }%
383 }%
384 {%
385 \renewcommand*{\@glstr@defpostpunc}{%
386 \renewcommand*{\glspostdescription}{#1}%
387 }%
388 }%
389 }%
390 }%
391 }

```

`glstrabbrvtype` Glossary type for abbreviations.

```

392 \newcommand*{\glstrabbrvtype}{\gldefaulttype}

```

`abbreviationsdef` Set by abbreviations option.

```

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

```

`abbreviationsdef`

```

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

```



```

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

If the acronym option hasn't been used, change \acronymtype to \glstrabbrvtype.

404 \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}{\ifglacrshortcutsacro\else none\fi}%
```

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

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

```

516      \glsacrshortcutstrue

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

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

    Leave none and false as last option.

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

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

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

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

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

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

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

```

stylemods

```

549 \define@key{glossaries-extra.sty}{stylemods}[default]{%
550   \ifstrequal{#1}{default}%
551   {%
552     \renewcommand*{\@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 au-
toseeindex option.
657 \ifdef\@glo@autosee
658 {%
659 \renewcommand*{\@glo@autosee}{%
660 \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
661 }%
662 {}

checkseeallowed Don't prohibit the use of the see key before the indexing files have been opened if the auto-
matic see indexing has been disabled, since it's no longer an issue.
663 \renewcommand*{\gls@checkseeallowed}{%
664 \if@glxtr@autoseeindex\@gls@see@noindex\fi
665 }

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

Setup shortcuts if required.
668 \@glxtr@setupshortcuts

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

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

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

```


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

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

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

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

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

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

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

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

Set up record option if required.

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

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

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

1.2 Extra Utilities

`trifemptyglossary` `\glstrtrifemptyglossary{<type>}{<true>}{<false>}`

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

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

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

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

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

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

vide@storagekey Unstarred version.

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

```

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

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

```

Provide the no-link command if not already defined.

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

vide@storagekey Starred version.

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

```

```

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

```

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

`\GlsXtrFmtField`

```

745 \newcommand{\GlsXtrFmtField}{useri}

```

`\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 \glstrfmtdisplay{@firstofone}{#3}{#4}%
774 }%
775 \endgroup
776 }

```

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

```

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

```

`\glxtrentryfmt` No link or indexing.

```

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

```

`@glxtrentryfmt`

```

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

```

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

```

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

```

trfieldlistgadd Similarly but uses \listcsgadd.

```

801 \newcommand*{\glxtrfieldlistgadd}[3]{%
802   \listcsgadd{glo@\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 }
```

rfieldxifinlist Expands item.

```

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

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.

```

822 \newrobustcmd{\glxtrifhasfield}{%
823   \@ifstar{\s@glxtrifhasfield}{\@glxtrifhasfield}%
824 }
```

`\lsxtrifhasfield` Unstarred version adds grouping.

```
825 \newcommand{\@glxtrifhasfield}[4]{%
826   {\s@glxtrifhasfield{#1}{#2}{#3}{#4}}%
827 }
```

`\lsxtrifhasfield` Starred version omits grouping.

```
828 \newcommand{\s@glxtrifhasfield}[4]{%
829   \letcs{\glscurrentfieldvalue}{glo@glsdetoklabel{#2}@#1}%
830   \ifundef\glscurrentfieldvalue
831     {#4}%
832     {%
833       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
834     }%
835 }
```

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

Just uses `\ifcsundef`.

```
836 \newcommand{\GlsXtrIfFieldUndef}[2]{%
837   \ifcsundef{glo@glsdetoklabel{#2}@#1}%
838 }
```

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

```
839 \newcommand*{\glxtrusefield}[2]{%
840   \@gl@entry@field{#1}{#2}%
841 }
```

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

```
842 \newcommand*{\Glsxtrusefield}[2]{%
843   \@gl@entry@field{#1}{#2}%
844 }
```

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

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

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

```
846 \newcommand*{\glxtrredefield}[2]{\csedef{glo@glsdetoklabel{#1}@#2}{}}
```

`\setfieldifexists`

```
847 \newcommand*{\glxtrsetfieldifexists}[3]{\gl@doifexists{#1}{#3}}
```

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

```
848 \newrobustcmd*{\GlsXtrSetField}[3]{%
```

```

849 \glstrsetfieldifexists{#1}{#2}%
850 {\csdef{glo@glsetoklabel{#1}@#2}{#3}}%
851 }

\GlsXtrLetField Uses \cslet instead. Third argument should be a macro.
852 \newrobustcmd*{\GlsXtrLetField}[3]{%
853 \glstrsetfieldifexists{#1}{#2}%
854 {\cslet{glo@glsetoklabel{#1}@#2}{#3}}%
855 }

\csGlsXtrLetField Uses \csletcs instead. Third argument should be a control sequence name.
856 \newrobustcmd*{\csGlsXtrLetField}[3]{%
857 \glstrsetfieldifexists{#1}{#2}%
858 {\csletcs{glo@glsetoklabel{#1}@#2}{#3}}%
859 }

\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.
860 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
861 \glstrsetfieldifexists{#1}{#2}%
862 {\csletcs{glo@glsetoklabel{#1}@#2}{glo@glsetoklabel{#3}@#4}}%
863 }

\gGlsXtrSetField Allow the user to set a field. First argument entry label, second argument field label, third
argument value.
864 \newrobustcmd*{\gGlsXtrSetField}[3]{%
865 \glstrsetfieldifexists{#1}{#2}%
866 {\csgdef{glo@glsetoklabel{#1}@#2}{#3}}%
867 }

\XGlsXtrSetField
868 \newrobustcmd*{\XGlsXtrSetField}[3]{%
869 \glstrsetfieldifexists{#1}{#2}%
870 {\protected@csxdef{glo@glsetoklabel{#1}@#2}{#3}}%
871 }

\eGlsXtrSetField
872 \newrobustcmd*{\eGlsXtrSetField}[3]{%
873 \glstrsetfieldifexists{#1}{#2}%
874 {\protected@csedef{glo@glsetoklabel{#1}@#2}{#3}}%
875 }

\XtrIfFieldEqStr
876 \newrobustcmd*{\GlsXtrIfFieldEqStr}[5]{%
877 \glstrifhasfield{#1}{#2}%
878 {%
879 \ifdefstring{glscurrentfieldvalue}{#3}{#4}{#5}%
880 }%

```

```

881  {#5}%
882 }

```

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

```

883 \ifglentrycounter
884   \newcommand*{\glxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
885 \else
886   \ifglssubentrycounter
887     \newcommand*{\glxtrpageref}[1]{\pageref{glsentry-\glsdetoklabel{#1}}}
888   \else
889     \newcommand*{\glxtrpageref}[1]{\gls{#1}}
890   \fi
891 \fi

```

`lossarypreamble`

```

892 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
893   \ifcsdef{glolist@#1}%
894   {%
895     \ifcsundef{@glossarypreamble@#1}%
896     {\csdef{@glossarypreamble@#1}{}}%
897   }%
898   \csappto{@glossarypreamble@#1}{#2}%
899 }%
900 {%
901   \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
902 }%
903 }

```

`lossarypreamble`

```

904 \newcommand{\preglossarypreamble}[2][\glsdefaulttype]{%
905   \ifcsdef{glolist@#1}%
906   {%
907     \ifcsundef{@glossarypreamble@#1}%
908     {\csdef{@glossarypreamble@#1}{}}%
909   }%
910   \cspreto{@glossarypreamble@#1}{#2}%
911 }%
912 {%
913   \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
914 }%
915 }

```

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`

```
916 \renewcommand*{\longnewglossaryentry}{%
917   \@ifstar\@glstr@s@longnewglossaryentry\@glstr@longnewglossaryentry
918 }
```

`ewglossaryentry`

Starred version.

```
919 \newcommand{\@glstr@s@longnewglossaryentry}[3]{%
920   \glstoifnoexists{#1}%
921   {%
922     \bgroup
923     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
924     \long\def\@newglossaryentryprehook{%
925       \long\def\@glo@desc{#3}%
926       \@org@newglossaryentryprehook
927     }%
928     \renewcommand*{\glstr@assign@desc}[1]{%
929       \global\cslet{glo@\glstrdetoklabel{#1}@desc}{\@glo@desc}%
930       \global\cslet{glo@\glstrdetoklabel{#1}@descplural}{\@glo@descplural}%
931     }
932     \glstr@defglossaryentry{#1}{#2}%
933   \egroup
934   }%
935 }
```

`ewglossaryentry`

Unstarred version.

```
936 \newcommand{\@glstr@longnewglossaryentry}[3]{%
937   \glstoifnoexists{#1}%
938   {%
939     \bgroup
940     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
941     \long\def\@newglossaryentryprehook{%
942       \long\def\@glo@desc{#3\glstrpostlongdescription}%
943       \@org@newglossaryentryprehook
944     }%
945     \renewcommand*{\glstr@assign@desc}[1]{%
946       \global\cslet{glo@\glstrdetoklabel{#1}@desc}{\@glo@desc}%
947     }
```

The following is different from the base glossaries.sty:

```
947       \global\cslet{glo@\glstrdetoklabel{#1}@descplural}{\@glo@descplural}%
948     }
949     \glstr@defglossaryentry{#1}{#2}%
950   \egroup
951   }%
952 }
```

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

```
953 \newcommand*{\glxtrpostlongdescription}{\leavevmode\unskip\nopostdesc}
```

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

ignoredglossary Redefine to check for star.

```
954 \renewcommand{\newignoredglossary}{%
955 \ifstar\glxtr@s@newignoredglossary\glxtr@org@newignoredglossary
956 }
```

ignoredglossary The original definition is patched to check for existence.

```
957 \newcommand*{\glxtr@org@newignoredglossary}[1]{%
958 \ifcsdef{glolist@#1}
959 {%
960 \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
961 }%
962 {%
963 \ifdefempty\@ignored@glossaries
964 {%
965 \edef\@ignored@glossaries{#1}%
966 }%
967 {%
968 \eappto\@ignored@glossaries{,#1}%
969 }%
970 \csgdef{glolist@#1}{,}%
971 \ifcsundef{gls@#1@entryfmt}%
972 {%
973 \defglsentryfmt[#1]{\glsentryfmt}%
974 }%
975 {}%
976 \ifdefempty\@gls@nohyperlist
977 {%
978 \renewcommand*{\@gls@nohyperlist}{#1}%
979 }%
980 {%
981 \eappto\@gls@nohyperlist{,#1}%
982 }%
983 }%
984 }
```

ignoredglossary Starred form.

```
985 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
986 \ifcsdef{glolist@#1}
987 {%
988 \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
989 }%
990 {%
991 \ifdefempty\@ignored@glossaries
```

```

992   {%
993     \edef\@ignored@glossaries{#1}%
994   }%
995   {%
996     \eappto\@ignored@glossaries{,#1}%
997   }%
998   \csgdef{glolist@#1}{,}%
999   \ifcsundef{gls@#1@entryfmt}%
1000  {%
1001    \defglsentryfmt[#1]{\glsentryfmt}%
1002  }%
1003  {}%
1004 }%
1005 }

```

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

```

1006 \glsifusetranslator
1007 {%
1008   \renewcommand*{\glssettoctitle}[1]{%
1009     \ifcsdef{gls@tr@set@#1@toctitle}%
1010     {%
1011       \csuse{gls@tr@set@#1@toctitle}%
1012     }%
1013     {%
1014       \ifcsdef{@glotype@#1@title}%
1015       {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1016       {\def\glossarytoctitle{\glossarytitle}}%
1017     }%
1018   }%
1019 }
1020 {
1021   \renewcommand*{\glssettoctitle}[1]{%
1022     \ifcsdef{@glotype@#1@title}%
1023     {\def\glossarytoctitle{\csname @glotype@#1@title\endcsname}}%
1024     {\def\glossarytoctitle{\glossarytitle}}%
1025   }
1026 }

```

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

```

1027 \newcommand{\provideignoredglossary}{%
1028   \@ifstar\glstr@s@provideignoredglossary\glstr@provideignoredglossary
1029 }

```

`ignoredglossary` Unstarred version.

```

1030 \newcommand*{\glstr@provideignoredglossary}[1]{%
1031   \ifcsdef{glolist@#1}
1032   {}%
1033   {%

```

```

1034 \ifdefempty\@ignored@glossaries
1035 {%
1036 \edef\@ignored@glossaries{#1}%
1037 }%
1038 {%
1039 \eappto\@ignored@glossaries{,#1}%
1040 }%
1041 \csgdef{glolist@#1}{,}%
1042 \ifcsundef{gls@#1@entryfmt}%
1043 {%
1044 \defglsentryfmt[#1]{\glsentryfmt}%
1045 }%
1046 {}%
1047 \ifdefempty\@gls@nohyperlist
1048 {%
1049 \renewcommand*{\@gls@nohyperlist}{#1}%
1050 }%
1051 {%
1052 \eappto\@gls@nohyperlist{,#1}%
1053 }%
1054 }%
1055 }

```

`ignoredglossary` Starred form.

```

1056 \newcommand*{\glxtr@s@provideignoredglossary}[1]{%
1057 \ifcsdef{glolist@#1}
1058 {}%
1059 {%
1060 \ifdefempty\@ignored@glossaries
1061 {%
1062 \edef\@ignored@glossaries{#1}%
1063 }%
1064 {%
1065 \eappto\@ignored@glossaries{,#1}%
1066 }%
1067 \csgdef{glolist@#1}{,}%
1068 \ifcsundef{gls@#1@entryfmt}%
1069 {%
1070 \defglsentryfmt[#1]{\glsentryfmt}%
1071 }%
1072 {}%
1073 }%
1074 }

```

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

```

1075 \newcommand*{\glxtrcopytoglossary}[2]{%
1076 \glsdofexists{#1}%
1077 {%

```

```

1078 \ifcsdef{glolist@#2}
1079 {%
1080 \cseappto{glolist@#2}{#1,}%
1081 }%
1082 {%
1083 \glstrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1084 }%
1085 }%
1086 }

```

1.3.1 Existence Checks

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

```

1087 \renewcommand{\glstoifexists}[2]{%
1088 \ifglstryexists{#1}{#2}%
1089 {%

```

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

```

1090 \edef\glslabel{\glstetoklabel{#1}}%
1091 \glstrundefaction{Glossary entry ‘\glslabel’
1092 has not been defined}{You need to define a glossary entry before
1093 you can reference it.}%
1094 }%
1095 }

```

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

```

1096 \renewcommand{\glstoifnoexists}[2]{%
1097 \ifglstryexists{#1}{%
1098 \glstrundefaction{Glossary entry ‘\glstetoklabel{#1}’
1099 has already been defined}{}}{#2}%
1100 }

```

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

```

1101 \ifdef\glstoifexistsordo
1102 {%
1103 \renewcommand{\glstoifexistsordo}[3]{%
1104 \ifglstryexists{#1}{#2}%
1105 {%
1106 \glstrundefaction{Glossary entry ‘\glstetoklabel{#1}’
1107 has not been defined}{You need to define a glossary entry
1108 before you can use it.}%
1109 #3%
1110 }%
1111 }%
1112 }
1113 {%
1114 \glstr@warnonexistsordo\glstoifexistsordo

```

```

1115 \newcommand{\glsdoifexistsordo}[3]{%
1116   \ifglsentryexists{#1}{#2}%
1117   {%
1118     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1119     has not been defined}{You need to define a glossary entry
1120     before you can use it.}%
1121     #3%
1122   }%
1123 }%
1124 }

```

`arynoexistsordo` Similarly for `\doifglossarynoexistsordo`.

```

1125 \ifdef\doifglossarynoexistsordo
1126 {%
1127   \renewcommand{\doifglossarynoexistsordo}[3]{%
1128     \ifglossaryexists{#1}%
1129     {%
1130       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1131       #3%
1132     }%
1133     {#2}%
1134   }%
1135 }
1136 {%
1137   \glstr@warnonexistsordo\doifglossarynoexistsordo
1138   \newcommand{\doifglossarynoexistsordo}[3]{%
1139     \ifglossaryexists{#1}%
1140     {%
1141       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1142       #3%
1143     }%
1144     {#2}%
1145   }%
1146 }
1147

```

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.

```

1148 \appto\newglossaryentryposthook{%
1149   \ifdefvoid\@glo@see
1150   {\csxdef{glo@\@glo@label @see}{}}%
1151   {%
1152     \csxdef{glo@\@glo@label @see}{\@glo@see}%
1153     \if@glstr@autoindex
1154       \@glstr@autoindexcrossrefs
1155     \fi

```

```

1156 }%
1157 }
1158 \appto\@gls@keymap{,{see}{see}}

```

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

```

1159 \newcommand*{\glxtrusesee}[1]{%
1160   \glsdoifexists{#1}%
1161   {%
1162     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1163     \ifdefempty\@glo@see
1164     {}%
1165     {%
1166       \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1167     }%
1168   }%
1169 }

```

`\glxtr@usesee`

```

1170 \newcommand*{\glxtr@usesee}[1][\seename]{%
1171   \@glxtr@usesee[#1]%
1172 }

```

`\@glxtr@usesee`

```

1173 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1174   \glxtruseseeformat{#1}{#2}%
1175 }

```

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

```

1176 \newcommand*{\glxtruseseeformat}[2]{%
1177   \glsseeformat{#1}{#2}{}%
1178 }

```

`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.

```

1179 \renewcommand*{\glsseeitemformat}[1]{%
1180   \ifglshashshort{\glslabel}{\glsaccesstext{#1}}{\glsaccessname{#1}}%
1181 }

```

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

```

1182 \newcommand*{\glxtruseseealso}[1]{%
1183   \glsdoifexists{#1}%
1184   {%
1185     \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@seealso}%

```

```

1186 \ifdefempty\@glo@see
1187 {}%
1188 {%
1189 \expandafter\glstruseseealsoformat\expandafter{\@glo@see}%
1190 }%
1191 }%
1192 }

```

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

```

1193 \newcommand*{\glstruseseealsoformat}[1]{%
1194 \glseeformat[\seealso]{#1}{}%
1195 }

```

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

```

1196 \newrobustcmd{\glstrseelist}[1]{%
1197 \edef\@glo@tmp{\noexpand\glseeelist{#1}}\@glo@tmp
1198 }

```

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

```

1199 \providecommand{\seealso}{see also}

```

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

```

1200 \ifdef\@xdycrossrefhook
1201 {
1202   Add the cross-reference class definition to the hook.
1203   \appto\@xdycrossrefhook{%
1204     \write\glswrite{(define-crossref-class \string"seealso\string"
1205       :unverified )}%
1206     \write\glswrite{(markup-crossref-list
1207       :class \string"seealso\string"^^J\space\space\space
1208       :open \string"\string\glstruseseealsoformat\glsoopenbrace\string"
1209       :close \string"\glsclosebrace\string")}%
1210   }

```

Append to class list.

```

1210 \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

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

```

1211 \newrobustcmd*{\glstrindexseealso}[2]{%
1212 \ifx\@glstr@record@setting\@glstr@record@setting@alsoindex
1213 \@glstr@recordsee{#1}{#2}%
1214 \fi
1215 \glsoifexists{#1}%
1216 {%

```



```

1217 \@@glsxtrwrglossmark
1218 \def\@gls@xref{#2}%
1219 \@onelevel@sanitize\@gls@xref
1220 \@gls@checkmkidxchars\@gls@xref
1221 \gls@glossary{\csname glo@#1@type\endcsname}{%
1222   (indexentry
1223     :tkey (\csname glo@#1@index\endcsname)
1224     :xref (\string"\@gls@xref\string")
1225     :attr \string"seealso\string"
1226   )
1227 }%
1228 }%
1229 }
1230 }
1231 {
  xindy not in use or glossaries version too old to support this.
1232 \newrobustcmd*{\glsxtrindexseealso}{\glssee[\seealsoname]}
1233 }

```

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

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

```

1234 \ifdef\gls@set@xr@key
1235 {

```

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

```

1236 \define@key{glossentry}{alias}{%
1237   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1238 }
1239 \define@key{glossentry}{seealso}{%
1240   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1241 }

```

Add to the key mappings.

```

1242 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1243 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1244 \appto\@newglossaryentryposthook{%
1245   \ifdefined\@glo@seealso
1246     {\csxdef{glo@\@glo@label @seealso}{}}%
1247   {%
1248     \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1249     \ifglsxtr@autoseeindex

```

```

1250      \@glxtr@autoindexcrossrefs
1251      \fi
1252    }%

```

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

```

1253    \ifdefvoid\@glo@alias
1254      {\csxdef{glo@\@glo@label @alias}{}}%
1255      {%
1256        \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1257      }%
1258  }

```

Provide user-level commands to access the values.

`\glxtralias`

```

1259  \newcommand*\glxtralias}[1]{\@gls@entry@field{#1}{alias}}

```

`trseealsolabels`

```

1260  \newcommand*\glxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}

```

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

```

1261  \appto\@glo@autoseehook{%
1262    \ifdefvoid\@glo@alias
1263      {%
1264        \ifdefvoid\@glo@seealso
1265          {}%
1266        {%
1267          \edef\@do@glssee{\noexpand\glxtrindexseealso
1268            {\@glo@label}{\@glo@seealso}}%
1269          \@do@glssee
1270        }%
1271      }%
1272    {%

```

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

```

1273      \ifdefvoid\@glo@see
1274        {%
1275          \edef\@do@glssee{\noexpand\glssee{\@glo@label}{\@glo@alias}}%
1276          \@do@glssee
1277        }%
1278      {}%
1279    }%
1280  }%
1281 }
1282 {

```

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

`\glxtralias`

```

1283  \glsaddstoragekey*{alias}{}{\glxtralias}

```

trseealsolabels

```
1284 \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.

```
1285 \appto\@newglossaryentryposthook{%
1286   \ifcsvoid{glo@\@glo@label @alias}%
1287   {%
1288     \ifcsvoid{glo@\@glo@label @seealso}%
1289     {}%
1290     {%
1291       \edef\@do@glsssee{\noexpand\glstrindexseealso
1292         {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1293       \@do@glsssee
1294     }%
1295   }%
1296   {%
```

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

```
1297   \ifdefvoid\@glo@see
1298   {%
1299     \edef\@do@glsssee{\noexpand\glsssee
1300       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1301     \@do@glsssee
1302   }%
1303   {}%
1304 }%
1305 }
1306 }
```

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

```
1307 \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.

```
1308 \newcommand*\glstraddallcrossrefs{%
1309   \forallglossaries{\@glo@type}%
1310   {%
1311     \forglsentries[\@glo@type]{\@glo@label}%
1312     {%
1313       \ifglssused{\@glo@label}%
1314       {\expandafter\@glstr@addunusedxrefs\expandafter{\@glo@label}}{}%
1315     }%
1316   }%
1317 }
```

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

```

1318 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
1319   \letcs{\@glo@see}{glo@glsdetoklabel{#1}@see}%
1320   \ifdefvoid\@glo@see
1321   {}%
1322   {%
1323     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1324   }%
1325   \letcs{\@glo@see}{glo@glsdetoklabel{#1}@seealso}%
1326   \ifdefvoid\@glo@see
1327   {}%
1328   {%
1329     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1330   }%
1331 }

```

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

```

1332 \newcommand*{\glxtr@addunused}[1][]{%
1333   \@glxtr@addunused
1334 }

```

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

```

1335 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1336   \@for\@glxtr@label:=#1\do
1337   {%
1338     \ifglxsused{\@glxtr@label}{}%
1339     {%
1340       \glxadd[format=glxtrunusedformat]{\@glxtr@label}%
1341       \glxunset{\@glxtr@label}%
1342       \expandafter\glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1343     }%
1344   }%
1345 }

```

`xtrunusedformat`

```

1346 \newcommand*{\glxtrunusedformat}[1]{\unskip}

```

1.3.2 Document Definitions

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

```

1347 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
1348 \renewcommand{\makenoidxglossaries}{%
1349   \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off
1350   {%
1351     \glxtr@orgmakenoidxglossaries

```

Add marker to \@do@seeglossary

```

1352   \renewcommand{\@do@seeglossary}[2]{%
1353     \@glsxtrwrglossmark
1354     \edef\@gls@label{\glsdetoklabel{##1}}%
1355     \protected@write\@auxout{}{%
1356       \string\@gls@reference
1357       {\csname glo@\@gls@label @type\endcsname}%
1358       {\@gls@label}%
1359       {%
1360         \string\glsseeformat##2}%
1361       }%
1362     }%
1363   }%

```

Check for docdefs=restricted:

```

1364   \ifglsxtrdocdefrestricted

```

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

```

1365     \renewcommand*\@gls@reference}[3]{%
1366       \ifcsundef{\@glsref@##1}{\csgdef{\@glsref@##1}{}}{}%
1367       \ifinlistcs{##2}{\@glsref@##1}%
1368       {}%
1369       {\listcsgadd{\@glsref@##1}{##2}}%
1370       \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1371       {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1372       {}%
1373       \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1374     }%
1375   \else

```

Disable document definitions.

```

1376     \@glsxtrdocdeffalse
1377     \fi
1378     \disable@keys{glossaries-extra}{docdef}%
1379   }%
1380   {%
1381     \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1382       not permitted\MessageBreak
1383       with record=\@glsxtr@record@setting\space package option}%
1384     {You may only use \string\makenoidxglossaries\space with the
1385       record=off option}%
1386   }%
1387 }

```

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

```

1388 \renewcommand*\@gls@defdocnewglossaryentry{%
1389   \ifcase\@glsxtr@docdefval
1390     docdef=false:

```

```

1390 \renewcommand*{\newglossaryentry}[2]{%
1391 \PackageError{glossaries-extra}{Glossary entries must
1392 be \MessageBreak defined in the preamble with \MessageBreak
1393 package option 'docdef=false'\MessageBreak(consider using
1394 'docdef=restricted')}{Move your glossary definitions to
1395 the preamble. You can also put them in a \MessageBreak separate file
1396 and load them with \string\loadglsentries.}%
1397 }%
1398 \or

```

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

```

1399 \let\gls@checkseeallowed\relax
1400 \let\newglossaryentry\newglossaryentry
1401 \or

```

Restricted mode just needs to allow the see value.

```

1402 \let\gls@checkseeallowed\relax
1403 \fi
1404 }%

```

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`

```

1405 \newcommand*{\GlsXtrEnableOnTheFly}{%
1406 \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1407 }

```

`\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.)

```

1408 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1409 \renewcommand*{\glsdetoklabel}[1]{%
1410 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1411 {%
1412 \expandafter\detokenize\expandafter{##1}%
1413 }%
1414 {\detokenize{##1}}%
1415 }%
1416 \@GlsXtrEnableOnTheFly
1417 }
1418 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1419 \expandafter\if\glsbackslash#1%
1420 #3%
1421 \else
1422 #4%

```

```

1423 \fi
1424 }

```

sxtrstarflywarn

```

1425 \newcommand*{\glxtrstarflywarn}{%
1426   \GlossariesExtraWarning{Experimental starred version of
1427   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1428   read the warnings in the glossaries-extra user manual)}}%
1429 }

```

rEnableOnTheFly

```

1430 \newcommand*{\@GlsXtrEnableOnTheFly}{%

```

Don't redefine \glsdetoklabel 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

```

1431 \newcommand*{\glxtrcat}{general}

```

\glxtr

```

1432 \newcommand*{\glxtr}[1] [] {%
1433   \def\glxtr@keylist{##1}%
1434   \@glxtr
1435 }

```

\@glxtr

```

1436 \newcommand*{\@glxtr}[2] [] {%
1437   \ifglentryexists{##2}%
1438   {%
1439     \ifblank{##1}{ }\{\GlsXtrWarning{##1}{##2}}%
1440   }%
1441   {%
1442     \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1443     description={\nopostdesc},##1}%
1444   }%
1445   \expandafter\gls\expandafter[\glxtr@keylist]{##2}%
1446 }

```

\Glsxtr

```

1447 \newcommand*{\Glsxtr}[1] [] {%
1448   \def\glxtr@keylist{##1}%
1449   \@Glsxtr
1450 }

```

\@Glsxtr

```

1451 \newcommand*{\@Glsxtr}[2] [] {%
1452   \ifglentryexists{##2}%

```

```

1453   {%
1454     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1455   }%
1456   {%
1457     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1458       description={\nopostdesc},##1}%
1459   }%
1460   \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
1461 }

```

\glstrpl

```

1462 \newcommand*{\glstrpl}[1][]{%
1463   \def\glstr@keylist{##1}%
1464   \@glstrpl
1465 }

```

\@glstrpl

```

1466 \newcommand*{\@glstrpl}[2][]{%
1467   \ifglentryexists{##2}%
1468   {%
1469     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1470   }%
1471   {%
1472     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1473       description={\nopostdesc},##1}%
1474   }%
1475   \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
1476 }

```

\Glsxtrpl

```

1477 \newcommand*{\Glsxtrpl}[1][]{%
1478   \def\glstr@keylist{##1}%
1479   \@Glsxtrpl
1480 }

```

\@Glsxtrpl

```

1481 \newcommand*{\@Glsxtrpl}[2][]{%
1482   \ifglentryexists{##2}
1483   {%
1484     \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1485   }%
1486   {%
1487     \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1488       description={\nopostdesc},##1}%
1489   }%
1490   \expandafter\Glspl\expandafter[\glstr@keylist]{##2}%
1491 }

```

\GlsXtrWarning


```

1492 \newcommand*{\GlsXtrWarning}[2]{%
1493   \def\@glxtr@optlist{##1}%
1494   \@onelevel@sanitize\@glxtr@optlist
1495   \GlossariesExtraWarning{The options ‘\@glxtr@optlist’ have
1496   been ignored for entry ‘##2’ as it has already been defined}%
1497 }

```

Disable commands after the glossary:

```

1498 \renewcommand\@printglossary[2]{%
1499   \def\@glxtr@printglossopts{##1}%
1500   \@glxtr@orgprintglossary{##1}{##2}%
1501   \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
1502   \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1503   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1504   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1505 }

```

abledflycommand

```

1506 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1507   \PackageError{glossaries-extra}%
1508   {\string##1\space can’t be used after any of the \MessageBreak
1509   glossaries have been displayed}%
1510   {The on-the-fly commands enabled by
1511   \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1512   before the glossaries. If you want to use any entries \MessageBreak
1513   after any of the glossaries, you must use the standard \MessageBreak
1514   method of first defining the entry and then using the \MessageBreak
1515   entry with commands like \string\glx}%
1516   \@glxtr@disabledflycommand
1517 }%
1518 \newcommand*{\@glxtr@disabledflycommand}[2][\]{##2}

```

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

```

1519 \let\GlsXtrEnableOnTheFly\relax
1520 }
1521 \@onlypreamble\GlsXtrEnableOnTheFly

```

1.3.3 Existing Glossary Style Modifications

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

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

```

1522 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}

```

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

etglossarystyle

```

1523 \renewcommand*{\setglossarystyle}[1]{%

```

```

1524 \ifcsundef{@glsstyle@#1}%
1525 {%
1526   \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1527 }%
1528 {%
1529   \csname @glsstyle@#1\endcsname

```

Only set the current style if it exists.

```

1530   \protected@edef\@glsxtr@current@style{#1}%
1531 }%
1532 \ifx\@glossary@default@style\relax
1533   \protected@edef\@glossary@default@style{#1}%
1534 \fi
1535 }

```

In case we have an old version of glossaries:

```

1536 \ifdef\@glossary@default@style
1537 {}
1538 {%
1539   \let\@glossary@default@style\relax
1540 }

```

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

```

1541 \ifdef\glslistdottedwidth
1542 {%
1543   \ifdim\glslistdottedwidth=.5\hsize
1544     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1545     \AtBeginDocument{%
1546       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1547         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1548       \fi
1549     }%
1550 \fi
1551 }
1552 }%

```

Similarly for `\glsdescwidth`:

`\glsdescwidth`

```

1553 \ifdef\glsdescwidth
1554 {%
1555   \ifdim\glsdescwidth=.6\hsize
1556     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1557     \AtBeginDocument{%
1558       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1559         \setlength{\glsdescwidth}{.6\columnwidth}%
1560       \fi
1561     }%
1562 \fi

```

```

1563 }
1564 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

1565 \ifdef\glspagelistwidth
1566 {%
1567   \ifdim\glspagelistwidth=.1\hsize
1568     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1569     \AtBeginDocument{%
1570       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1571         \setlength{\glspagelistwidth}{.1\columnwidth}%
1572       \fi
1573     }%
1574   \fi
1575 }
1576 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

1577 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1578 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1579   \glsnonnumberlistfalse
1580   \renewcommand*{\glossaryentrynumbers}[1]{%
1581     \ifglentryexists{\glscurrententrylabel}%
1582     {%
1583       \@glstrpreloctag
1584       \GlsXtrFormatLocationList{#1}%
1585       \@glstrpostloctag
1586       \gls@save@numberlist{#1}%
1587     }{}%
1588   }%
1589 \else
1590   \glsnonnumberlisttrue
1591   \renewcommand*{\glossaryentrynumbers}[1]{%
1592     \ifglentryexists{\glscurrententrylabel}%
1593     {%
1594       \gls@save@numberlist{#1}%
1595     }{}%
1596   }%
1597 \fi

```

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

```

1598 \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

```

1599 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1600   \let\@glstrpreloctag\@glstrpreloctag
1601   \let\@glstrpostloctag\@glstrpostloctag
1602   \renewcommand*{\@glstr@pagetag}{#1}%
1603   \renewcommand*{\@glstr@pagetag}{#2}%
1604   \renewcommand*{\@glstr@savepreloctag}[2]{%
1605     \csgdef{\@glstr@preloctag@##1}{##2}%
1606   }%
1607   \renewcommand*{\@glstr@doloctag}{%
1608     \ifcsundef{\@glstr@preloctag\@glscurrententrylabel}%
1609     {%
1610       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
1611         Rerun required}%
1612     }%
1613     {%
1614       \csuse{\@glstr@preloctag\@glscurrententrylabel}%
1615     }%
1616   }%
1617 }
1618 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glstrpreloctag

```

1619 \newcommand*{\@glstrpreloctag}{%
1620   \let\@glstr@org@delimN\delimN
1621   \let\@glstr@org@delimR\delimR
1622   \let\@glstr@org@glsgignore\glsgignore
1623   \gdef is required as the delimiters may occur inside a scope.
1624   \gdef\@glstr@thisloctag{\@glstr@pagetag}%
1625   \renewcommand*{\delimN}{%
1626     \gdef\@glstr@thisloctag{\@glstr@pagetag}%
1627     \@glstr@org@delimN}%
1628   \renewcommand*{\delimR}{%
1629     \gdef\@glstr@thisloctag{\@glstr@pagetag}%
1630     \@glstr@org@delimR}%
1631   \renewcommand*{\glsgignore}[1]{%
1632     \gdef\@glstr@thisloctag{\relax}%
1633     \@glstr@org@glsgignore{##1}}%
1634   \@glstr@doloctag

```

glstrpreloctag

```

1635 \newcommand*{\@glstrpreloctag}{}

```

@glsxtr@pagetag

```
1636 \newcommand*{\@glsxtr@pagetag}{}%
```

glsxtr@pagetag

```
1637 \newcommand*{\@glsxtr@pagetag}{}%
```

lsxtrpostloctag

```
1638 \newcommand*{\@glsxtrpostloctag}{%
1639   \let\delimN\@glsxtr@org@delimN
1640   \let\delimR\@glsxtr@org@delimR
1641   \let\glsignore\@glsxtr@org@glsignore
1642   \protected@write\@auxout{}%
1643     {\string\@glsxtr@savepreloctag{\glscurrententrylabel}\@glsxtr@thisloctag}}%
1644 }
```

lsxtrpostloctag

```
1645 \newcommand*{\@glsxtrpostloctag}{}
```

lsxtr@preloctag

```
1646 \newcommand*{\@glsxtr@savepreloctag}[2]{%
1647 \protected@write\@auxout{}{%
1648   \string\providecommand\string\@glsxtr@savepreloctag[2]{}}
```

glsxtr@doloctag

```
1649 \newcommand*{\@glsxtr@doloctag}{}
```

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

```
1650 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1651   \XKV@plfalse
1652   \XKV@sttrue
1653   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1654   {%
1655     \csname glsnonumberlist\XKV@resa\endcsname
1656     \ifglsnonumberlist
1657       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1658     \else
1659       \def\glossaryentrynumbers##1{%
1660         \@glsxtrpreloctag
1661         \GlsXtrFormatLocationList{##1}%
1662         \@glsxtrpostloctag
1663         \gls@save@numberlist{##1}}%
1664       \fi
1665   }%
1666 }
```

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.

```
1667 \renewcommand*{\glsentryfmt}{%
1668   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}}{}%
1669   \glsifregular{\glslabel}%
1670   {\glstrregularfont{\glsgenentryfmt}}%
1671   {%
1672     \ifglshasshort{\glslabel}%
1673     {\glstrgenabbrvfmt}%
1674     {\glstrregularfont{\glsgenentryfmt}}%
1675   }%
1676 }
```

`glstrregularfont` Font used for regular entries.

```
1677 \newcommand*{\glstrregularfont}[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 `\glstrifwasfirstuse` etc to allow the postlink hook to work better. This now has an optional argument that sets up the defaults.

```
1678 \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).

```
1679 \glstr@record{#2}{#3}{glslink}%
1680 \glstoifexists{#3}%
1681 {%
```

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).

```
1682 \let\glstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1683 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1684 \def\glscustomtext{#4}%
1685 \@glstr@field@linkdefs
1686 #1%
1687 \@gls@link[#2]{#3}{#4}%
1688 \let\ifKV@glslink@hyper\glstrorg@ifKV@glslink@hyper
1689 }%
```

```

1690 \glspostlinkhook
1691 }

```

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

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

```

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

```

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

```

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

```

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

```

1702 \let\@glsxtr@org@Gls@\@Gls@
1703 \def\@Gls@#1#2{%
1704   \@glsxtr@record{#1}{#2}{glslink}%
1705   \@glsxtr@org@Gls@{#1}{#2}%
1706 }%

```

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

```

1707 \let\@glsxtr@org@Glspl@\@Glspl@
1708 \def\@Glspl@#1#2{%
1709   \@glsxtr@record{#1}{#2}{glslink}%
1710   \@glsxtr@org@Glspl@{#1}{#2}%
1711 }%

```

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

```

1712 \let\@glsxtr@org@GLS@\@GLS@
1713 \def\@GLS@#1#2{%
1714   \@glsxtr@record{#1}{#2}{glslink}%
1715   \@glsxtr@org@GLS@{#1}{#2}%
1716 }%

```

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

```

1717 \let\@glsxtr@org@GLSpl@\@GLSpl@
1718 \def\@GLSpl@#1#2{%
1719   \@glsxtr@record{#1}{#2}{glslink}%
1720   \@glsxtr@org@GLSpl@{#1}{#2}%
1721 }%

```

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

```

1722 \renewcommand*{\@glsdisp}[3][\]{%
1723   \@glsxtr@record{#1}{#2}{glslink}%
1724   \glsdoifexists{#2}{%
1725     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
1726     \let\glsifplural\@secondoftwo
1727     \let\glsapscase\@firstofthree
1728     \def\glscustomtext{#3}%
1729     \def\glsinsert{}%
1730     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1731     \@gls@link[#1]{#2}{\@glo@text}%
1732     \ifKV@glslink@local
1733       \glslocalunset{#2}%
1734     \else
1735       \glsunset{#2}%
1736     \fi
1737   }%
1738   \glspostlinkhook
1739 }

```

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

```

1740 \renewcommand*{\@gls@link}[3][\]{%
1741   \@glsxtr@record{#1}{#2}{glslink}%
1742   \glsdoifexistsordo{#2}%
1743   {%
1744     \let\do@gls@link@checkfirsthyper\relax
1745     \@gls@link[#1]{#2}{#3}%
1746   }%
1747   {%
1748     \glstextformat{#3}%
1749   }%
1750   \glspostlinkhook
1751 }

```

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

```

1752 \newcommand*{\glsxtrinitwrgloss}{%
1753   \glsifattribute{\glslabel}{wrgloss}{after}%
1754   {%
1755     \glsxtrinitwrglossbeforefalse
1756   }%
1757   {%
1758     \glsxtrinitwrglossbeforetrue
1759   }%
1760 }

```

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

```

1761 \newif\ifglsxtrinitwrglossbefore
1762 \glsxtrinitwrglossbeforetrue

```


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

```
1763 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1764 {%
1765   \ifcase\nr\relax
1766   \glstrinitwrglossbeforetrue
1767   \or
1768   \glstrinitwrglossbeforefalse
1769   \fi
1770 }
```

```
1771 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}
```

```
1772 \define@key{glslink}{theHvalue}{\def\@glstr@theHvalue{#1}}
```

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

```
1773 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{}
1774 \glstr@hyperoutsidettrue
```

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

```
1775 \newcommand*{\glstrnithyperoutside}{%
1776   \glsifattribute{\glslabel}{hyperoutside}{false}%
1777   {%
1778     \glstr@hyperoutsidfalse
1779   }%
1780   {%
1781     \glstr@hyperoutsidettrue
1782   }%
1783 }
```

`\@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`.

```
1784 \def\@gls@link[#1]#2#3{%
1785   \leavevmode
1786   \edef\glslabel{\glsdetoklabel{#2}}%
1787   \def\@gls@link@opts{#1}%
1788   \let\@gls@link@label\glslabel
1789   \let\@glsnumberformat\@glstr@defaultnumberformat
1790   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1791   \edef\glstype{\csname glo@\glslabel @type\endcsname}%
1792   \let\org@ifKV@glslink@hyper@ifKV@glslink@hyper
```

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

```
1793 \def\@glstr@thevalue{%
1794 \def\@glstr@theHvalue{\@glstr@thevalue}%
```

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

```
1795 \glstrinitwrgloss
```

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

```
1796 \glxtrinithyperoutside
```

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

```
1797 \@gls@setdefault@glslink@opts
1798 \do@glssdisablehyperinlist
1799 \do@gls@link@checkfirsthyper
1800 \setkeys{glslink}{#1}%
1801 \glslinkpostsetkeys
```

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

```
1802 \ifdefempty{\@glxtr@thevalue}%
1803 {%
1804   \@gls@saveentrycounter
1805 }%
1806 {%
1807   \let\theglssentrycounter\@glxtr@thevalue
1808   \def\theHglssentrycounter{\@glxtr@theHvalue}%
1809 }%
1810 \@gls@setsort{\glslabel}%
```

Check textformat attribute (new to v1.21).

```
1811 \glshasattribute{\glslabel}{textformat}%
1812 {%
1813   \edef\@glxtr@attrval{\glssgetattribute{\glslabel}{textformat}}%
1814   \ifcsdef{\@glxtr@attrval}%
1815   {%
1816     \letcs{\@glxtr@textformat}{\@glxtr@attrval}%
1817   }%
1818   {%
1819     \GlossariesExtraWarning{Unknown control sequence name
1820       '\@glxtr@attrval' supplied in textformat attribute
1821       for entry '\glslabel'. Reverting to default \string\glstextformat}%
1822     \let\@glxtr@textformat\glstextformat
1823   }%
1824 }%
1825 {%
1826   \let\@glxtr@textformat\glstextformat
1827 }%
```

Do write if it should occur before the link text:

```
1828 \ifglxtrinitwrglossbefore
1829   \@do@wrglossary{#2}%
1830 \fi
```

Do the link text:

```
1831 \ifKV@glslink@hyper
1832   \ifglxtr@hyperoutside
1833     \@glslink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1834   \else
1835     \@glxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
```

```

1836 \fi
1837 \else
1838 \ifglxtr@hyperoutside
1839 \glsdonohyperlink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1840 \else
1841 \@glxtr@textformat{\glsdonohyperlink{\glolinkprefix\glslabel}{#3}}%
1842 \fi
1843 \fi

```

Do write if it should occur after the link text:

```

1844 \ifglxtrinitwrglossbefore
1845 \else
1846 \@do@wrglossary{#2}%
1847 \fi

```

As the original definition:

```

1848 \let@ifKV@glslink@hyper\org@ifKV@glslink@hyper
1849 }

```

```

1850 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

```

```

1851 \define@key{glossadd}{theHvalue}{\def\@glxtr@theHvalue{#1}}

```

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

```

1852 \renewrobustcmd*{\glsadd}[2][\]{%
1853 \glxtrifinmark
1854 }%
1855 {%
1856 \@gls@adjustmode
1857 \@glxtr@record{#1}{#2}{glossadd}%
1858 \glsoifexists{#2}%
1859 {%
1860 \let\@glsnumberformat\@glxtr@defaultnumberformat
1861 \edef\@gls@counter{\csname glo@\glsetoklabel{#2}@counter\endcsname}%
1862 \def\@glxtr@thevalue{}%
1863 \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1864 \setkeys{glossadd}{#1}%
1865 \ifdefempty{\@glxtr@thevalue}%
1866 {%
1867 \@gls@saveentrycounter
1868 }%
1869 {%
1870 \let\theglentrycounter\@glxtr@thevalue
1871 \def\theHglentrycounter{\@glxtr@theHvalue}%
1872 }%
1873 \@do@wrglossary{#2}%
1874 }%
1875 }%
1876 }

```

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

```
1877 \newcommand*{\@glsxtr@field@linkdefs}{%
1878   \let\glsxtrifwasfirstuse\@secondoftwo
1879   \let\glsifplural\@secondoftwo
1880   \let\glscapscase\@firstofthree
1881   \let\glsinsert\@empty
1882 }
```

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

assignfieldfont

```
1883 \newcommand*{\glsxtrassignfieldfont}[1]{%
1884   \ifglstryexists{#1}%
1885   {%
1886     \ifglshasshort{#1}%
1887     {%
1888       \glsssetabbrvfmt{\glscategory{#1}}%
1889       \glsifregular{#1}%
1890       {\let\@gls@field@font\glsxtrregularfont}%
1891       {\let\@gls@field@font\@firstofone}%
1892     }%
1893     {%
1894       \glsifnotregular{#1}%
1895       {\let\@gls@field@font\@firstofone}%
1896       {\let\@gls@field@font\glsxtrregularfont}%
1897     }%
1898   }%
1899   {%
1900     \let\@gls@field@font\@gobble
1901   }%
1902 }
```

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

```
1903 \def\@glstext@#1#2[#3]{%
1904   \glsxtrassignfieldfont{#2}%
1905   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1906 }
```

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

```
1907 \def\@GLStext@#1#2[#3]{%
1908   \glsxtrassignfieldfont{#2}%
1909   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1910   {\@gls@field@font{\@GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1911 }
```

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

```
1912 \def\@Glstext@#1#2[#3]{%
1913   \glsxtrassignfieldfont{#2}%
1914 }
```

```

1914 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1915 {\@gls@field@font{\Glsaccessstext{#2}#3}}%
1916 }

```

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

`ecknohyperfirst`

```

1917 \newcommand*{\glstrchecknohyperfirst}[1]{%
1918 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1919 }

```

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

```

1920 \def\@glsfirst@#1#2[#3]{%
1921 \glstrassignfieldfont{#2}%
    Ensure that \glsfirst honours the nohyperfirst attribute.
1922 \@gls@field@link
1923 [\let\glstrifwasfirstuse\@firstoftwo
1924 \glstrchecknohyperfirst{#2}%
1925 ]{#1}{#2}%
1926 {\@gls@field@font{\glsaccessfirst{#2}#3}}%
1927 }

```

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

```

1928 \def\@Glsfirst@#1#2[#3]{%
1929 \glstrassignfieldfont{#2}%
    Ensure that \Glsfirst honours the nohyperfirst attribute.
1930 \@gls@field@link
1931 [\let\glstrifwasfirstuse\@firstoftwo
1932 \let\glscapscase\@secondofthree
1933 \glstrchecknohyperfirst{#2}%
1934 ]%
1935 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}%
1936 }

```

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

```

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

```

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

```
1946 \def\@glsplural@#1#2[#3]{%
1947   \glsxtrassignfieldfont{#2}%
1948   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1949   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1950 }
```

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

```
1951 \def\@Glsplural@#1#2[#3]{%
1952   \glsxtrassignfieldfont{#2}%
1953   \@gls@field@link
1954   [\let\glsifplural\@firstoftwo
1955   \let\glscapscase\@secondofthree
1956   ]%
1957   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1958 }
```

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

```
1959 \def\@GLSplural@#1#2[#3]{%
1960   \glsxtrassignfieldfont{#2}%
1961   \@gls@field@link
1962   [\let\glsifplural\@firstoftwo
1963   \let\glscapscase\@thirdofthree
1964   ]%
1965   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1966 }
```

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

```
1967 \def\glsfirstplural@#1#2[#3]{%
1968   \glsxtrassignfieldfont{#2}%
1969   \@gls@field@link
1970   [\let\glsxtrifwasfirstuse\@firstoftwo
1971   \let\glsifplural\@firstoftwo
1972   \glsxtrchecknohyperfirst{#2}%
1973   ]%
1974   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1975 }
```

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

```
1976 \def\Glsfirstplural@#1#2[#3]{%
1977   \glsxtrassignfieldfont{#2}%
1978   \@gls@field@link
1979   [\let\glsxtrifwasfirstuse\@firstoftwo
1980   \let\glsifplural\@firstoftwo
1981   \let\glscapscase\@secondofthree
```

```

1982 \glstrchecknohyperfirst{#2}%
1983 ]%
1984 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1985 }

```

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

```

1986 \def\@GLSfirstplural@#1#2[#3]{%
1987 \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
1988 \@gls@field@link
1989 [\let\glstrifwasfirstuse\@firstoftwo
1990 \let\glsifplural\@firstoftwo
1991 \let\glscapscase\@thirdofthree
1992 \glstrchecknohyperfirst{#2}%
1993 ]%
1994 {#1}{#2}%
1995 {\@gls@field@font{\Glsaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1996 }

```

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

```

1997 \def\@glsname@#1#2[#3]{%
1998 \glstrassignfieldfont{#2}%
1999 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2000 }

```

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

```

2001 \def\@Glsname@#1#2[#3]{%
2002 \glstrassignfieldfont{#2}%
2003 \@gls@field@link
2004 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2005 {\@gls@field@font{\Glsaccessname{#2}#3}}%
2006 }

```

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

```

2007 \def\@GLSname@#1#2[#3]{%
2008 \glstrassignfieldfont{#2}%
2009 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2010 {#1}{#2}%
2011 {\@gls@field@font{\Glsaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2012 }

```

\@glsdesc@

```

2013 \def\@glsdesc@#1#2[#3]{%
2014 \glstrassignfieldfont{#2}%
2015 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2016 }

```

```

\@Glsdesc@    First letter uppercase version.
2017 \def\@Glsdesc@#1#2[#3]{%
2018   \glstrassignfieldfont{#2}%
2019   \@gls@field@link
2020   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2021   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2022 }

\@GLSdesc@    All uppercase version.
2023 \def\@GLSdesc@#1#2[#3]{%
2024   \glstrassignfieldfont{#2}%
2025   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2026   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2027 }

@glsglscapscase@    No case-changing version.
2028 \def\@glsglscapscase@#1#2[#3]{%
2029   \glstrassignfieldfont{#2}%
2030   \@gls@field@link
2031   [\let\glscapscase\@secondoftwo
2032   \let\glsifplural\@firstoftwo
2033   ]{#1}{#2}{\@gls@field@font{\glsglscapscase{#2}#3}}%
2034 }

@Glsdescplural@    First letter uppercase version.
2035 \def\@Glsdescplural@#1#2[#3]{%
2036   \glstrassignfieldfont{#2}%
2037   \@gls@field@link
2038   [\let\glscapscase\@secondoftwo
2039   \let\glsifplural\@firstoftwo
2040   ]{#1}{#2}{\@gls@field@font{\Glsaccessdescplural{#2}#3}}%
2041 }

@GLSdescplural@    All uppercase version.
2042 \def\@GLSdescplural@#1#2[#3]{%
2043   \glstrassignfieldfont{#2}%
2044   \@gls@field@link
2045   [\let\glscapscase\@thirdoftwo
2046   \let\glsifplural\@firstoftwo
2047   ]%
2048   {#1}{#2}%
2049   {\@gls@field@font{\GLSaccessdescplural{#2}\mfirstucMakeUppercase{#3}}}%
2050 }

\@glssymbol@
2051 \def\@glssymbol@#1#2[#3]{%
2052   \glstrassignfieldfont{#2}%
2053   \@gls@field@link{#1}{#2}{\@gls@field@font{\glssymbol{#2}#3}}%
2054 }

```


\@Glssymbol@ First letter uppercase version.

```
2055 \def\@Glssymbol@#1#2[#3]{%
2056   \glstrassignfieldfont{#2}%
2057   \@gls@field@link
2058   [\let\glscapscase\@secondoftwo]%
2059   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2060 }
```

\@GLSsymbol@ All uppercase version.

```
2061 \def\@GLSsymbol@#1#2[#3]{%
2062   \glstrassignfieldfont{#2}%
2063   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2064   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2065 }
```

lssymbolplural@ No case-changing version.

```
2066 \def\@lssymbolplural@#1#2[#3]{%
2067   \glstrassignfieldfont{#2}%
2068   \@gls@field@link
2069   [\let\glscapscase\@secondoftwo
2070   \let\glsifplural\@firstoftwo
2071   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2072 }
```

lssymbolplural@ First letter uppercase version.

```
2073 \def\@lssymbolplural@#1#2[#3]{%
2074   \glstrassignfieldfont{#2}%
2075   \@gls@field@link
2076   [\let\glscapscase\@secondoftwo
2077   \let\glsifplural\@firstoftwo
2078   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2079 }
```

LSsymbolplural@ All uppercase version.

```
2080 \def\@LSsymbol@#1#2[#3]{%
2081   \glstrassignfieldfont{#2}%
2082   \@gls@field@link
2083   [\let\glscapscase\@thirdoftwo
2084   \let\glsifplural\@firstoftwo
2085   ]%
2086   {#1}{#2}%
2087   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2088 }
```

\@Glsuseri@ First letter uppercase version.

```
2089 \def\@Glsuseri@#1#2[#3]{%
2090   \glstrassignfieldfont{#2}%
2091   \@gls@field@link
```

```

2092 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2093 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2094 }

```

\@GLSuseri@ All uppercase version.

```

2095 \def\@GLSuseri@#1#2[#3]{%
2096 \glstrassignfieldfont{#2}%
2097 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2098 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseri{#2}#3}}}%
2099 }

```

\@Glsuserii@ First letter uppercase version.

```

2100 \def\@Glsuserii@#1#2[#3]{%
2101 \glstrassignfieldfont{#2}%
2102 \@gls@field@link
2103 [\let\glscapscase\@secondoftwo]%
2104 {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2105 }

```

\@GLSuserii@ All uppercase version.

```

2106 \def\@GLSuserii@#1#2[#3]{%
2107 \glstrassignfieldfont{#2}%
2108 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2109 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2110 }

```

\@Glsuseriii@ First letter uppercase version.

```

2111 \def\@Glsuseriii@#1#2[#3]{%
2112 \glstrassignfieldfont{#2}%
2113 \@gls@field@link
2114 [\let\glscapscase\@secondoftwo]%
2115 {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2116 }

```

\@GLSuseriii@ All uppercase version.

```

2117 \def\@GLSuseriii@#1#2[#3]{%
2118 \glstrassignfieldfont{#2}%
2119 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2120 {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2121 }

```

\@Glsuseriv@ First letter uppercase version.

```

2122 \def\@Glsuseriv@#1#2[#3]{%
2123 \glstrassignfieldfont{#2}%
2124 \@gls@field@link
2125 [\let\glscapscase\@secondoftwo]%
2126 {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2127 }

```

\@GLSuseriv@ All uppercase version.

```
2128 \def\@GLSuseriv@#1#2[#3]{%
2129   \glstrassignfieldfont{#2}%
2130   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2131   {#1}{#2}%
2132   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2133 }
```

\@Glsuserv@ First letter uppercase version.

```
2134 \def\@Glsuserv@#1#2[#3]{%
2135   \glstrassignfieldfont{#2}%
2136   \@gls@field@link
2137   [\let\glscapscase\@secondoftwo]%
2138   {#1}{#2}{\@gls@field@font{\Glsentryuserv{#2}#3}}}%
2139 }
```

\@GLSuserv@ All uppercase version.

```
2140 \def\@GLSuserv@#1#2[#3]{%
2141   \glstrassignfieldfont{#2}%
2142   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2143   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserv{#2}#3}}}%
2144 }
```

\@Glsuservi@ First letter uppercase version.

```
2145 \def\@Glsuservi@#1#2[#3]{%
2146   \glstrassignfieldfont{#2}%
2147   \@gls@field@link
2148   [\let\glscapscase\@secondoftwo]%
2149   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
2150 }
```

\@GLSuservi@ All uppercase version.

```
2151 \def\@GLSuservi@#1#2[#3]{%
2152   \glstrassignfieldfont{#2}%
2153   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2154   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuservi{#2}#3}}}%
2155 }
```

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

\@acrshort No case change.

```
2156 \def\@acrshort#1#2[#3]{%
2157   \glsdoifexists{#2}%
2158   {%
2159     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2160     \let\glxtrifwasfirstuse\@secondoftwo
2161     \let\glsifplural\@secondoftwo
```

```

2162 \let\glscapscase\@firstofthree
2163 \let\glsinsert\@empty
2164 \def\glscustomtext{%
2165     \acronymfont{\glsaccesssshort{#2}}#3%
2166 }%
2167 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2168 }%
2169 \glspostlinkhook
2170 }

```

\@Acrshort First letter uppercase.

```

2171 \def\@Acrshort#1#2[#3]{%
2172     \glsdoifexists{#2}%
2173     {%
2174         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2175         \let\glstrifwasfirstuse\@secondoftwo
2176         \let\glsifplural\@secondoftwo
2177         \let\glscapscase\@secondofthree
2178         \let\glsinsert\@empty
2179         \def\glscustomtext{%
2180             \acronymfont{\Glsaccesssshort{#2}}#3%
2181         }%
2182         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2183     }%
2184     \glspostlinkhook
2185 }

```

\@ACRshort All uppercase.

```

2186 \def\@ACRshort#1#2[#3]{%
2187     \glsdoifexists{#2}%
2188     {%
2189         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2190         \let\glstrifwasfirstuse\@secondoftwo
2191         \let\glsifplural\@secondoftwo
2192         \let\glscapscase\@thirdofthree
2193         \let\glsinsert\@empty
2194         \def\glscustomtext{%
2195             \mfirstucMakeUppercase{\acronymfont{\glsaccesssshort{#2}}#3}%
2196         }%
2197         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2198     }%
2199     \glspostlinkhook
2200 }

```

\@acrshortpl No case change.

```

2201 \def\@acrshortpl#1#2[#3]{%
2202     \glsdoifexists{#2}%
2203     {%
2204         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

2205 \let\glxtrifwasfirstuse\@secondoftwo
2206 \let\glsifplural\@firstoftwo
2207 \let\glscapscase\@firstofthree
2208 \let\glsinsert\@empty
2209 \def\glscustomtext{%
2210 \acronymfont{\glsaccessshortpl{#2}}#3%
2211 }%
2212 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2213 }%
2214 \glspostlinkhook
2215 }

```

\@Acrshortpl First letter uppercase.

```

2216 \def\@Acrshortpl#1#2[#3]{%
2217 \glsdoifexists{#2}%
2218 {%
2219 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2220 \let\glxtrifwasfirstuse\@secondoftwo
2221 \let\glsifplural\@firstoftwo
2222 \let\glscapscase\@secondofthree
2223 \let\glsinsert\@empty
2224 \def\glscustomtext{%
2225 \acronymfont{\Glsaccessshortpl{#2}}#3%
2226 }%
2227 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2228 }%
2229 \glspostlinkhook
2230 }

```

\@ACRshortpl All uppercase.

```

2231 \def\@ACRshortpl#1#2[#3]{%
2232 \glsdoifexists{#2}%
2233 {%
2234 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2235 \let\glxtrifwasfirstuse\@secondoftwo
2236 \let\glsifplural\@firstoftwo
2237 \let\glscapscase\@thirdofthree
2238 \let\glsinsert\@empty
2239 \def\glscustomtext{%
2240 \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2241 }%
2242 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2243 }%
2244 \glspostlinkhook
2245 }

```

\@acrlong No case change.

```

2246 \def\@acrlong#1#2[#3]{%
2247 \glsdoifexists{#2}%

```

```

2248 {%
2249   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2250   \let\glxtrifwasfirstuse\@secondoftwo
2251   \let\gl@ifplural\@secondoftwo
2252   \let\gl@scapscase\@firstofthree
2253   \let\gl@insert\@empty
2254   \def\gl@customtext{%
2255     \acronymfont{\gl@saccesslong{#2}}#3%
2256   }%
2257   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2258 }%
2259 \glspostlinkhook
2260 }

```

\@Acrlong First letter uppercase.

```

2261 \def\@Acrlong#1#2[#3]{%
2262   \gl@doifexists{#2}%
2263   {%
2264     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2265     \let\glxtrifwasfirstuse\@secondoftwo
2266     \let\gl@ifplural\@secondoftwo
2267     \let\gl@scapscase\@secondofthree
2268     \let\gl@insert\@empty
2269     \def\gl@customtext{%
2270       \acronymfont{\gl@saccesslong{#2}}#3%
2271     }%
2272     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2273   }%
2274   \glspostlinkhook
2275 }

```

\@ACRlong All uppercase.

```

2276 \def\@ACRlong#1#2[#3]{%
2277   \gl@doifexists{#2}%
2278   {%
2279     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2280     \let\glxtrifwasfirstuse\@secondoftwo
2281     \let\gl@ifplural\@secondoftwo
2282     \let\gl@scapscase\@thirdofthree
2283     \let\gl@insert\@empty
2284     \def\gl@customtext{%
2285       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2286     }%
2287     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2288   }%
2289   \glspostlinkhook
2290 }

```

\@acrlongpl No case change.

```

2291 \def\@acrlongpl#1#2[#3]{%
2292   \glsdoifexists{#2}%
2293   {%
2294     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2295     \let\glxtrifwasfirstuse\@secondoftwo
2296     \let\gl@sifplural\@firstoftwo
2297     \let\glscapscase\@firstofthree
2298     \let\gl$insert\@empty
2299     \def\glscustomtext{%
2300       \acronymfont{\gl@saccesslongpl{#2}}#3%
2301     }%
2302     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2303   }%
2304   \glspostlinkhook
2305 }

```

\@Acrlongpl First letter uppercase.

```

2306 \def\@Acrlongpl#1#2[#3]{%
2307   \glsdoifexists{#2}%
2308   {%
2309     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2310     \let\glxtrifwasfirstuse\@secondoftwo
2311     \let\gl@sifplural\@firstoftwo
2312     \let\glscapscase\@secondofthree
2313     \let\gl$insert\@empty
2314     \def\glscustomtext{%
2315       \acronymfont{\Glsaccesslongpl{#2}}#3%
2316     }%
2317     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2318   }%
2319   \glspostlinkhook
2320 }

```

\@ACrlongpl All uppercase.

```

2321 \def\@ACrlongpl#1#2[#3]{%
2322   \glsdoifexists{#2}%
2323   {%
2324     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2325     \let\glxtrifwasfirstuse\@secondoftwo
2326     \let\gl@sifplural\@firstoftwo
2327     \let\glscapscase\@thirdofthree
2328     \let\gl$insert\@empty
2329     \def\glscustomtext{%
2330       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%
2331     }%
2332     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2333   }%
2334   \glspostlinkhook
2335 }

```

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

\@glsaddkey

```

2336 \renewcommand*{\@glsaddkey}[7]{%
2337   \key@ifundefined{glossentry}{#1}%
2338   {%
2339     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2340     \appto\@gls@keymap{,{#1}{#1}}%
2341     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2342     \appto\@newglossaryentryposthook{%
2343       \letcs{\@glo@tmp}{@glo@#1}%
2344       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2345     }%
2346     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2347     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2348   \ifcsdef{@gls@user@#1@}%
2349   {%
2350     \PackageError{glossaries}%
2351     {Can't define '\string#5' as helper command
2352     '\expandafter\string\csname @gls@user@#1@endcsname' already
2353     exists}%
2354     {}%
2355   }%
2356   {%
2357     \expandafter\newcommand\expandafter*\expandafter
2358     {\csname @gls@user@#1@endcsname}[2][ ]{%
2359       \new@ifnextchar[%
2360         {\csuse{@gls@user@#1@}{##1}{##2}}%
2361         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2362     \csdef{@gls@user@#1@}##1##2[##3]{%
2363       \@gls@field@link{##1}{##2}{#3{##2}##3}%
2364     }%
2365     \newrobustcmd*{#5}{%
2366       \expandafter\@gls@hyp@opt\csname @gls@user@#1@endcsname}%
2367     }%

```

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

```

2368   \ifcsdef{@Gls@user@#1@}%
2369   {%
2370     \PackageError{glossaries}%
2371     {Can't define '\string#6' as helper command
2372     '\expandafter\string\csname @Gls@user@#1@endcsname' already
2373     exists}%
2374     {}%
2375   }%
2376   {%
2377     \expandafter\newcommand\expandafter*\expandafter

```



```

2378     {\csname @Gls@user@#1\endcsname}[2] [] {%
2379         \new@ifnextchar [%
2380             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2381             {\csuse{@Gls@user@#1@}{##1}{##2} [] }}}%
2382     \csdef{@Gls@user@#1@}##1##2[##3]{%
2383         \@gls@field@link[\let\glscapscase\@secondofthree]%
2384         {##1}{##2}{#4{##2}##3}%
2385     }%
2386     \newrobustcmd*{#6}{%
2387         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2388     }%

```

Finally the all caps version (modified):

```

2389     \ifcsdef{@GLS@user@#1@}%
2390     {%
2391         \PackageError{glossaries}%
2392         {Can't define '\string#7' as helper command
2393         '\expandafter\string\csname @GLS@user@#1\endcsname' already
2394         exists}%
2395     }%
2396 }%
2397 {%
2398     \expandafter\newcommand\expandafter*\expandafter
2399     {\csname @GLS@user@#1\endcsname}[2] [] {%
2400         \new@ifnextchar [%
2401             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2402             {\csuse{@GLS@user@#1@}{##1}{##2} [] }}}%
2403     \csdef{@GLS@user@#1@}##1##2[##3]{%
2404         \@gls@field@link[\let\glscapscase\@thirdofthree]%
2405         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2406     }%
2407     \newrobustcmd*{#7}{%
2408         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2409     }%
2410 }%
2411 {%
2412     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2413 }%
2414 }

```

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

```

2415 \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.

```

2416 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2417 \renewcommand*{\@gls@link@checkfirsthyper}{%

```

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

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

```

2418 \ifglssused{\glslabel}%
2419 {\let\glstrifwasfirstuse\@secondoftwo}
2420 {\let\glstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.

2421 \edef\glscategorylabel{\glscategory{\glslabel}}%
2422 \ifglssused{\glslabel}%
2423 {%
2424   \glssifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2425   {\KV@glsslink@hyperfalse}{}}%
2426 }%
2427 {%
2428   \glssifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2429   {\KV@glsslink@hyperfalse}{}}%
2430 }%
2431 \glsslinkcheckfirsthyperhook
2432 }

```

`\glsdisablehyperinlist` 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.

```

2433 \ifdefined\do@glssdisablehyperinlist
2434 {%
2435   \let\@glssxtr\do@glssdisablehyperinlist\do@glssdisablehyperinlist
2436   \renewcommand*{\do@glssdisablehyperinlist}{%
2437     \@glssxtr\do@glssdisablehyperinlist
2438     \glssifattribute{\glslabel}{nohyper}{true}{\KV@glsslink@hyperfalse}{}}%
2439   }
2440 }
2441 {}

```

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

```

2442 \define@boolkey{glsslink}{noindex}[true]{}
2443 \KV@glsslink@noindexfalse

```

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

`\@glsslink@opts`

```

2444 \ifdefined\@glss@setdefault@glsslink@opts
2445 {
2446   \renewcommand*{\@glss@setdefault@glsslink@opts}{%
2447     \KV@glsslink@noindexfalse
2448     \@glssxtrsetaliasnoindex
2449   }
2450 }
2451 {

```

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

```
2452 \newcommand*{\@glsglslink@noindex}{%
2453   \KV@glsglslink@noindexfalse
2454   \@glsglslink@noindex
2455 }
2456 \preto\do@glsglslink@noindex{\@glsglslink@noindex}
2457 }
```

`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.)

```
2458 \providecommand*{\glsglslink@noindex}{%
2459   \KV@glsglslink@noindextrue
2460 }
```

`setaliasindex`

```
2461 \newcommand*{\@glsglslink@noindex}{%
2462   \glsglslink@noindexfalse
2463 }%
2464 \let\glsglslink@noindex\glsglslink@noindex
2465 \glsglslink@noindex
2466 \let\glsglslink@noindex\@no@glsglslink@noindex
2467 }%
2468 {}%
2469 }
```

`xtrindexaliased`

```
2470 \newcommand*{\@glsglslink@noindex}{%
2471   \ifKV@glsglslink@noindex
2472   \else
2473     \begingroup
2474     \let\@glsglslink@noindex\glsglslink@noindex
2475     \edef\@glsglslink@noindex{\csname glo@glsglslink@noindex\endcsname}%
2476     \glsglslink@noindex
2477     \do@glsglslink@noindex{\glsglslink@noindex}%
2478   \endgroup
2479   \fi
2480 }
```

`xtrindexaliased`

```
2481 \newcommand*{\@no@glsglslink@noindex}{%
2482   \PackageError{glossaries-extra}{\string\glsglslink@noindex\space
2483     not permitted outside definition of \string\glsglslink@noindex}%
2484   }%
2485 }
```

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

```
2486 \let\glsglslink@noindex\@no@glsglslink@noindex
```

DefaultGlsOpts Set the default options for \glslink etc.

```
2487 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2488   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2489     \setkeys{glslink}{#1}%
2490     \@glsxtrsetaliasnoindex
2491   }%
2492 }
```

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

```
2493 \newcommand*{\glsxtrifindexing}[2]{%
2494   \ifKV@glslink@noindex #2\else #1\fi
2495 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2496 \renewcommand*{\glswriteentry}[2]{%
2497   \glsxtrifindexing
2498   {%
2499     \ifglsindexonlyfirst
2500       \ifglsused{#1}
2501       {\glsxtrdoautoindexname{#1}{dualindex}}%
2502       {#2}%
2503     \else
2504       \glsifattribute{#1}{indexonlyfirst}{true}%
2505       {\ifglsused{#1}
2506        {\glsxtrdoautoindexname{#1}{dualindex}}%
2507        {#2}}%
2508       {#2}%
2509     \fi
2510   }%
2511   {}%
2512 }
```

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

```
2513 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
2514   \glsxtrdowrglossaryhook{\@gls@label}%
2515 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2516 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
2517   \glsxtrdowrglossaryhook{\@gls@label}%
2518 }
```

xtr@do@@wrindex

```
2519 \newcommand*{\@glsxtr@do@@wrindex}{%
```

```

2520 \glstrdoautoindexname{\@gls@label}{dualindex}%
2521 }

\orglossaryhook Allow user to hook into indexing code. (Always used by \glsadd. Used by \gls when index-
ing, which may or may not occur depending on the indexing settings.)
2522 \newcommand*{\glstrdowrglossaryhook}[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.
2523 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2524 \let\glslinkvar\@firstofthree
2525 \let\@gls@hyp@opt@cs#1\relax
2526 \@ifstar{\s@gls@hyp@opt}%
2527 {\@ifnextchar+%
2528 {\@firstoftwo{\p@gls@hyp@opt}}%
2529 {%
2530 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2531 {\@firstoftwo{\@alt@gls@hyp@opt}}%
2532 {#1}%
2533 }%
2534 }%
2535 }

\alt@gls@hyp@opt User version
2536 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
2537 \let\glslinkvar\@firstofthree
2538 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}

\lt@hyp@opt@char Contains the character used as the command modifier.
2539 \newcommand*{\@gls@alt@hyp@opt@char}{%

\lt@hyp@opt@keys Contains the option list used as the command modifier.
2540 \newcommand*{\@gls@alt@hyp@opt@keys}{%

\SetAltModifier
2541 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2542 \let\@gls@hyp@opt\@gls@alt@hyp@opt
2543 \def\@gls@alt@hyp@opt@char{#1}%
2544 \def\@gls@alt@hyp@opt@keys{#2}%
2545 }

\org@dohyperlink
2546 \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.

```
2547 \ifdef\glshnavhyperlink
2548 {
2549   \renewcommand*{\glshnavhyperlink}[3][\@glo@type]{%
2550     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
```

Scope:

```
2551   {%
2552     \let\glsdohyperlink\glstr@org@dohyperlink
2553     \@glslink{\glshnavhyperlinkname{#1}{#2}}{#3}%
2554   }%
2555 }%
2556 }
2557 {}
```

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

```
2558 \renewcommand*{\glsdohyperlink}[2]{%
2559   \glsattribute{\glslabel}{targeturl}%
2560   {%
2561     \glsattribute{\glslabel}{targetname}%
2562     {%
2563       \glsattribute{\glslabel}{targetcategory}%
2564       {%
2565         \hyperref{\glsattribute{\glslabel}{targeturl}}{%
2566           {\glsattribute{\glslabel}{targetcategory}}%
2567           {\glsattribute{\glslabel}{targetname}}%
2568           {\glstrprotectlinks#2}}%
2569       }%
2570     }%
2571     \hyperref{\glsattribute{\glslabel}{targeturl}}{%
2572       }%
2573       {\glsattribute{\glslabel}{targetname}}%
2574       {\glstrprotectlinks#2}}%
2575   }%
2576 }%
2577 {%
2578   \href{\glsattribute{\glslabel}{targeturl}}{%
2579     {\glstrprotectlinks#2}}%
2580 }%
2581 }%
2582 {}
```

Check for alias.

```

2583 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2584 \ifdefvoid\gloaliaslabel
2585 {%
2586   \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2587 }%
2588 {%

```

Redirect link to the alias target.

```

2589   \glsxtrhyperlink
2590   {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2591   {\glsxtrprotectlinks#2}}%
2592 }%
2593 }%
2594 }

```

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

```

2595 \ifdef\glsshowtarget
2596 {
2597   \newcommand{\glsxtrhyperlink}[2]{%
2598     \@glsshowtarget{#1}%
2599     \hyperlink{#1}{#2}}%
2600 }%
2601 }
2602 {
2603   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2604 }

```

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.

```

2605 \renewrobustcmd*{\glsdohyperlink}[2][\glssentrytext{\@glo@label}]{%
2606   \glsdoifexists{#2}%
2607   {%
2608     \def\@glo@label{#2}%
2609     {\edef\glslabel{#2}%
2610      \@glslink{\glolinkprefix\glslabel}{#1}}%
2611   }%
2612 }

```

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

```

2613 \renewcommand{\glsdisablehyper}{%
2614   \KV@glslink@hyperfalse
2615   \def\@glslink{\glsdohyperlink}%
2616   \let\@glstarget\@secondoftwo
2617 }

```

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

```
2618 \renewcommand{\glsenablehyper}{%
2619   \KV@glslink@hypertrue
2620   \def\@glslink{\glsdohyperlink}%
2621   \def\@glstarget{\glsdohypertarget}%
2622 }
```

`\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.

```
2623 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

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

```
2624 \ifcsundef{hyperlink}%
2625 {%
2626   \def\@glslink{\glsdonohyperlink}
2627 }%
2628 {%
2629   \def\@glslink{\glsdohyperlink}
2630 }
```

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

```
2631 \newcommand*{\glstrprotectlinks}{%
2632   \KV@glslink@hyperfalse
2633   \KV@glslink@noindextrue
2634   \let\@gls@\@glstr@p@text@
2635   \let\@Gls@\@Glsxtr@p@text@
2636   \let\@GLS@\@GLSxtr@p@text@
2637   \let\@glspl@\@glstr@p@plural@
2638   \let\@Glspl@\@Glsxtr@p@plural@
2639   \let\@GLSpl@\@GLSxtr@p@plural@
2640   \let\@glstrshort@\@glstr@p@short@
2641   \let\@Glsxtrshort@\@Glsxtr@p@short@
2642   \let\@GLSxtrshort@\@GLSxtr@p@short@
2643   \let\@glstrlong@\@glstr@p@long@
2644   \let\@Glsxtrlong@\@Glsxtr@p@long@
2645   \let\@GLSxtrlong@\@GLSxtr@p@long@
2646   \let\@glstrshortpl@\@glstr@p@shortpl@
2647   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
2648   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
2649   \let\@glstrlongpl@\@glstr@p@longpl@
2650   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
2651   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
2652   \let\@acrshort@\@glstr@p@acrshort@
2653   \let\@Acrshort@\@Glsxtr@p@acrshort@
2654   \let\@ACRshort@\@GLSxtr@p@acrshort@}
```



```

2655 \let\@acrshortpl\@glsxtrp@acrshortpl@
2656 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
2657 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
2658 \let\@acrlong\@glsxtrp@acrlong@
2659 \let\@Acrlong\@Glsxtrp@acrlong@
2660 \let\@ACRlong\@GLSxtrp@acrlong@
2661 \let\@acrlongpl\@glsxtrp@acrlongpl@
2662 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
2663 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
2664 }

```

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

@glsxtrp@text@

```
2665 \def\@glsxtrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2666 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2667 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2668 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2669 \def\@lsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
2670 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtrp@short@

```

2671 \def\@glsxtrp@short@#1#2[#3]{%
2672 {%
2673   \glssetabbrvfmt{\glscategory{#2}}%
2674   \glsabbrvfont{\glstryshort{#2}}#3%
2675 }%
2676 }

```

Glsxtrp@short@

```

2677 \def\@Glsxtrp@short@#1#2[#3]{%
2678 {%
2679   \glssetabbrvfmt{\glscategory{#2}}%
2680   \glsabbrvfont{\Glsentryshort{#2}}#3%
2681 }%
2682 }

```

GLSxtr@p@short@

```
2683 \def\@GLSxtr@p@short@#1#2[#3]{%
2684   {%
2685     \glsetabbrvfmt{\glscategory{#2}}%
2686     \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
2687   }%
2688 }
```

sxtr@p@shortpl@

```
2689 \def\@glxtr@p@shortpl@#1#2[#3]{%
2690   {%
2691     \glsetabbrvfmt{\glscategory{#2}}%
2692     \glabbrvfont{\glentryshortpl{#2}}#3%
2693   }%
2694 }
```

Sxtr@p@shortpl@

```
2695 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2696   {%
2697     \glsetabbrvfmt{\glscategory{#2}}%
2698     \glabbrvfont{\Glsentryshortpl{#2}}#3%
2699   }%
2700 }
```

Sxtr@p@shortpl@

```
2701 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2702   {%
2703     \glsetabbrvfmt{\glscategory{#2}}%
2704     \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
2705   }%
2706 }
```

@glxtr@p@long@

```
2707 \def\@glxtr@p@long@#1#2[#3]{\{\glentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2708 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2709 \def\@GLSxtr@p@long@#1#2[#3]{%
2710   {\mfirstucMakeUppercase{\glslongfont{\glentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2711 \def\@glxtr@p@longpl@#1#2[#3]{\{\glentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2712 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2713 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2714   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2715 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2716 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2717 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2718   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2719 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2720 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2721 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2722   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2723 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
2724 \def\@GLSxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
2725 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2726   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
2727 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2728 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2729 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2730   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2731 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
2732 \newcommand*{\glsxtrsetpopts}[1]{%
2733   \renewcommand*{\@glsxtrp@opt}{#1}%
2734 }
```

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

```
2735 \newcommand*{\glossxtrsetpopts}{%
2736   \glsxtrsetpopts{noindex}%
2737 }
```

`\@glsxtrp`

```
2738 \newrobustcmd*{\@glsxtrp}[2]{%
```

Add scope.

```
2739   {%
2740     \let\glspostlinkhook\relax
2741     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2742   }%
2743 }
```

`\@glsxtrp`

```
2744 \newrobustcmd*{\@glsxtrp}[2]{%
2745   \ifcsdef{gls#1}%
2746     {%
2747       \@glsxtrp{gls#1}{#2}%
2748     }%
2749     {%
2750       \ifcsdef{glsxtr#1}%
2751         {%
2752           \@glsxtrp{glsxtr#1}{#2}%
2753         }%
2754         {%
2755           \PackageError{glossaries-extra}{‘#1’ not recognised by
2756             \string\glsxtrp}{}%
2757         }%
2758       }%
2759 }
```

`\@Glsxtrp`

```
2760 \newrobustcmd*{\@Glsxtrp}[2]{%
2761   \ifcsdef{Gls#1}%
2762     {%
2763       \@glsxtrp{Gls#1}{#2}%
2764     }%
2765     {%
2766       \ifcsdef{Glsxtr#1}%
2767         {%
2768           \@glsxtrp{Glsxtr#1}{#2}%
2769         }%
2770     }
```

```

2770   {%
2771     \PackageError{glossaries-extra}{‘#1’ not recognised by
2772       \string\Glsxtrp}{}%
2773   }%
2774 }%
2775 }

```

\@GLSxtrp

```

2776 \newrobustcmd*{\@GLSxtrp}[2]{%
2777   \ifcsdef{GLS#1}%
2778   {%
2779     \@@glxtrp{GLS#1}{#2}%
2780   }%
2781   {%
2782     \ifcsdef{GLSxtr#1}%
2783     {%
2784       \@@glxtrp{GLSxtr#1}{#2}%
2785     }%
2786     {%
2787       \PackageError{glossaries-extra}{‘#1’ not recognised by
2788         \string\GLSxtrp}{}%
2789     }%
2790   }%
2791 }

```

\glxtr@entry@p

```

2792 \newrobustcmd*{\glxtr@headentry@p}[2]{%
2793   \gl@ifattribute{#1}{headuc}{true}%
2794   {%
2795     \mfirstucMakeUppercase{\@glx@entry@field{#1}{#2}}%
2796   }%
2797   {%
2798     \@glx@entry@field{#1}{#2}%
2799   }%
2800 }

```

\glxtrp Not robust as it needs to expand somewhat.

```

2801 \ifdef\texorpdfstring
2802 {
2803   \newcommand{\glxtrp}[2]{%
2804     \protect\NoCaseChange
2805     {%
2806       \protect\texorpdfstring
2807       {%
2808         \protect\glxtrifinmark
2809         {%
2810           \ifcsdef{glxtrhead#1}%
2811           {%
2812             {\protect\csuse{glxtrhead#1}{#2}}%

```

```

2813         }%
2814         {%
2815         \glxtr@headentry@p{#2}{#1}%
2816         }%
2817     }%
2818     {%
2819     \@glxtrp{#1}{#2}%
2820     }%
2821 }%
2822 {%
2823 \protect\@gls@entry@field{#2}{#1}%
2824 }%
2825 }%
2826 }
2827 }
2828 {
2829 \newcommand{\glxtrp}[2]{%
2830 \protect\NoCaseChange
2831 {%
2832 \protect\glxtrifinmark
2833 {%
2834 \ifcsdef{glxtrhead#1}%
2835 {%
2836 {\protect\csuse{glxtrhead#1}}%
2837 }%
2838 {%
2839 \glxtr@headentry@p{#2}{#1}%
2840 }%
2841 }%
2842 {%
2843 \@glxtrp{#1}{#2}%
2844 }%
2845 }%
2846 }
2847 }

```

Provide short synonyms for the most common option.

`\glsp`

```
2848 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
2849 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```
2850 \ifdef\teorpdfstring
```

```
2851 {
```

```
2852 \newcommand{\Glsxtrp}[2]{%
```

```

2853 \protect\NoCaseChange
2854 {%
2855 \protect\texorpdfstring
2856 {%
2857 \protect\glxtrifinmark
2858 {%
2859 \ifcsdef{Glsxtrhead#1}%
2860 {%
2861 {\protect\csuse{Glsxtrhead#1}{#2}}%
2862 }%
2863 {%
2864 \protect\@Gls@entry@field{#2}{#1}%
2865 }%
2866 }%
2867 {%
2868 \@Glsxtrp{#1}{#2}%
2869 }%
2870 }%
2871 {%
2872 \protect\@gls@entry@field{#2}{#1}%
2873 }%
2874 }%
2875 }
2876 }
2877 {
2878 \newcommand{\Glsxtrp}[2]{%
2879 \protect\NoCaseChange
2880 {%
2881 \protect\glxtrifinmark
2882 {%
2883 \ifcsdef{Glsxtrhead#1}%
2884 {%
2885 {\protect\csuse{Glsxtrhead#1}}%
2886 }%
2887 {%
2888 \protect\@Gls@entry@field{#2}{#1}%
2889 }%
2890 }%
2891 {%
2892 \@Glsxtrp{#1}{#2}%
2893 }%
2894 }%
2895 }
2896 }

```

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

```

2897 \ifdef\texorpdfstring
2898 {
2899 \newcommand{\GLSxtrp}[2]{%

```

```

2900 \protect\NoCaseChange
2901 {%
2902 \protect\texorpdfstring
2903 {%
2904 \protect\glstrifinmark
2905 {%
2906 \ifcsdef{GLSxtr#1}%
2907 {%
2908 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2909 }%
2910 {%
2911 \protect\mfirstucMakeUppercase
2912 {%
2913 \protect\@gls@entry@field{#2}{#1}%
2914 }%
2915 }%
2916 }%
2917 {%
2918 \@GLSxtrp{#1}{#2}%
2919 }%
2920 }%
2921 {%
2922 \protect\@gls@entry@field{#2}{#1}%
2923 }%
2924 }%
2925 }
2926 }
2927 {
2928 \newcommand{\GLSxtrp}[2]{%
2929 \protect\NoCaseChange
2930 {%
2931 \protect\glstrifinmark
2932 {%
2933 \ifcsdef{GLSxtr#1}%
2934 {%
2935 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2936 }%
2937 {%
2938 \protect\mfirstucMakeUppercase
2939 {%
2940 \protect\@gls@entry@field{#2}{#1}%
2941 }%
2942 }%
2943 }%
2944 {%
2945 \@GLSxtrp{#1}{#2}%
2946 }%
2947 }%
2948 }

```


2949 }

1.3.5 Entry Counting

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

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

\@glsunset Global unset.

```
2950 \renewcommand*{\@glsunset}[1]{%
2951   \@glsunset{#1}%
2952   \glsxtrpostunset{#1}%
2953 }%
```

glsxtrpostunset

```
2954 \newcommand*{\glsxtrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2955   \renewcommand*{\@glslocalunset}[1]{%
2956     \@glslocalunset{#1}%
2957     \glsxtrpostlocalunset{#1}%
2958   }%
```

rpostlocalunset

```
2959 \newcommand*{\glsxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2960 \renewcommand*{\@glsreset}[1]{%
2961   \@glsreset{#1}%
2962   \glsxtrpostreset{#1}%
2963 }%
```

glsxtrpostreset

```
2964 \newcommand*{\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2965 \renewcommand*{\@glslocalreset}[1]{%
2966   \@glslocalreset{#1}%
2967   \glsxtrpostlocalreset{#1}%
2968 }%
```

rpostlocalreset

```
2969 \newcommand*{\glsxtrpostlocalreset}[1]{}
```

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

```
2970 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2971 \glsenableentrycount
```

Redefine \gls etc:

```
2972 \renewcommand*{\gls}{\cglsl}%
2973 \renewcommand*{\Gls}{\cGls}%
2974 \renewcommand*{\glspl}{\cglspl}%
2975 \renewcommand*{\Glspl}{\cGlspl}%
2976 \renewcommand*{\GLS}{\cGLS}%
2977 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2978 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2979 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2980 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2981   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2982     can't be used with \string\GlsXtrEnableEntryCounting}%
2983   {Use one or other but not both commands}}%
2984 }
```

ycountunsetattr

```
2985 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2986   \@for\@glsxtr@cat:=#1\do
2987   {%
2988     \ifdefempty{\@glsxtr@cat}{}%
2989     {%
2990       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2991     }%
2992   }%
2993 }
```

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

nableentrycount

```
2994 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2995 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
2996 \renewcommand*{\gls@defdocnewglossaryentry}{%
2997   \renewcommand*{\newglossaryentry}[2]{%
2998     \PackageError{glossaries}{\string\newglossaryentry\space
2999       may only be used in the preamble when entry counting has
3000       been activated}{If you use \string\glsenableentrycount\space
3001       you must place all entry definitions in the preamble not in
3002       the document environment}%
3003   }%
3004 }
```

New commands to access new fields:

```

3005 \newcommand*{\glsentrycurrcount}[1]{%
3006 \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
3007 {0}{\@gls@entry@field{##1}{currcount}}%
3008 }%
3009 \newcommand*{\glsentryprevcount}[1]{%
3010 \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
3011 {0}{\@gls@entry@field{##1}{prevcount}}%
3012 }%

```

Adjust post unset and reset:

```

3013 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
3014 \renewcommand*{\glsxtrpostunset}[1]{%
3015 \@glsxtr@entrycount@org@unset{##1}%
3016 \@gls@increment@currcount{##1}%
3017 }%
3018 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
3019 \renewcommand*{\glsxtrpostlocalunset}[1]{%
3020 \@glsxtr@entrycount@org@localunset{##1}%
3021 \@gls@local@increment@currcount{##1}%
3022 }%
3023 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3024 \renewcommand*{\glsxtrpostreset}[1]{%
3025 \@glsxtr@entrycount@org@reset{##1}%
3026 \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
3027 }%
3028 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3029 \renewcommand*{\glsxtrpostlocalreset}[1]{%
3030 \@glsxtr@entrycount@org@localreset{##1}%
3031 \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
3032 }%

```

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

```

3033 \let\@cgl@s\@cgl@s
3034 \let\@cgl spl\@cgl spl

3035 \let\@cGls\@cGls
3036 \let\@cGlspl\@cGlspl
3037 \let\@cGLS\@cGLS
3038 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

3039 \AtEndDocument{\@gls@write@entrycounts}%
3040 \renewcommand*{\@gls@entry@count}[2]{%
3041 \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
3042 }%
3043 \let\glsenableentrycount\relax
3044 \renewcommand*{\glsenableentryunitcount}{%
3045 \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space

```

```

3046      can't be used with \string\glsenableentrycount}%
3047      {Use one or other but not both commands}%
3048  }%
3049 }

```

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

```

3050 \renewcommand*{\@gls@write@entrycounts}{%
3051   \immediate\write\@auxout
3052   {\string\providecommand*\string\@gls@entry@count}[2]{}%
3053   \count@=0\relax
3054   \forallglsentries{\@glsentry}{%
3055     \glshasattribute{\@glsentry}{entrycount}%
3056     {%
3057       \ifglsused{\@glsentry}%
3058       {%
3059         \immediate\write\@auxout
3060         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3061       }%
3062     }%
3063     \advance\count@ by \@ne
3064   }%
3065 }%
3066 }%
3067 \ifnum\count@=0
3068   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3069   \MessageBreak with \string\glsenableentrycount\space but the
3070   \MessageBreak attribute 'entrycount' hasn't
3071   \MessageBreak been assigned to any of the defined
3072   \MessageBreak entries}%
3073 \fi
3074 }

```

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

```

3075 \newcommand*\glstrifcounttrigger[3]{%
3076   \glshasattribute{#1}{entrycount}%
3077   {%
3078     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
3079     #3%
3080   \else
3081     #2%
3082   \fi
3083 }%
3084 {#3}%
3085 }

```

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

\@@cgl's@

```

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

\@@cgl'spl@

```

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

\@@cGl's@

```

3106 \def\@@cGl's@#1#2[#3]{%
3107   \gl'sxtrifcounttrigger{#2}%
3108   {%
3109     \cGl'sformat{#2}{#3}%
3110     \gl'sunset{#2}%
3111   }%
3112   {%
3113     \@Gl's@{#1}{#2}[#3]%
3114   }%
3115 }%
```

\@@cGl'spl@

```

3116 \def\@@cGl'spl@#1#2[#3]{%
3117   \gl'sxtrifcounttrigger{#2}%
3118   {%
3119     \cGl'splformat{#2}{#3}%
3120     \gl'sunset{#2}%
3121   }%
3122   {%
3123     \@Gl'spl@{#1}{#2}[#3]%
3124   }%
3125 }%
```

\@@cGLS@

```
3126 \def\@@cGLS@#1#2[#3]{%
3127   \glxtrifcounttrigger{#2}%
3128   {%
3129     \cGLSformat{#2}{#3}%
3130     \glset{#2}%
3131   }%
3132   {%
3133     \@GLS@{#1}{#2}[#3]%
3134   }%
3135 }%
```

\@@cGLSpl@

```
3136 \def\@@cGLSpl@#1#2[#3]{%
3137   \glxtrifcounttrigger{#2}%
3138   {%
3139     \cGLSplformat{#2}{#3}%
3140     \glset{#2}%
3141   }%
3142   {%
3143     \@GLSpl@{#1}{#2}[#3]%
3144   }%
3145 }%
```

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

\@cgl's@

```
3146 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
3147 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
3148 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```

\@cGL'spl@

```
3149 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
3150 \newrobustcmd*{\cGLS}{\@gl'shyp@opt\@cGLS}
```

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

```
3151 \newcommand*{\@cGLS}[2][ ]{%
3152   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}%
3153 }
```

```

\@cGLS@
3154 \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.
3155 \newcommand*\cGLSformat}[2]{%
3156   \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
3157 }

\cGLSpl
3158 \newrobustcmd*\cGLSpl{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
3159 \newcommand*\@cGLSpl}[2][{}]{%
3160   \new@ifnextchar[\@cGLSpl@{#1}{#2}}{\@cGLSpl@{#1}{#2}[{}]}%
3161 }

\@cGLSpl@
3162 \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.
3163 \newcommand*\cGLSplformat}[2]{%
3164   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
3165 }

      Modify the trigger formats to check for the regular attribute.

\cglformat
3166 \renewcommand*\cglformat}[2]{%
3167   \glsifregular{#1}
3168   {\glsentryfirst{#1}}%
3169   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
3170 }

\cGlsformat
3171 \renewcommand*\cGlsformat}[2]{%
3172   \glsifregular{#1}
3173   {\Glsentryfirst{#1}}%
3174   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
3175 }

\cglsplformat
3176 \renewcommand*\cglsplformat}[2]{%
3177   \glsifregular{#1}
3178   {\glsentryfirstplural{#1}}%
3179   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
3180 }

```

\cGlsplformat

```
3181 \renewcommand*{\cGlsplformat}[2]{%
3182   \glsifregular{#1}
3183   {\Glsentryfirstplural{#1}}%
3184   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
3185 }
```

New code similar to above for unit counting.

defunitcounters

```
3186 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3187   \edef\@glo@countunit{\csuse{\glstr@categoryattr@@\@glo@category @unitcount}}%
3188   \ifdefvoid\@glo@countunit
3189     {}%
3190     {%
3191       \@glstr@ifunitcounter{\@glo@countunit}%
3192       {}%
3193       {\expandafter\@glstr@addunitcounter\expandafter{\@glo@countunit}}%
3194     }%
3195 }
```

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

```
3196 \newcommand*{\@glstr@unitcountlist}{{}}
```

@addunitcounter

```
3197 \newcommand*{\@glstr@addunitcounter}[1]{%
3198   \listadd{\@glstr@unitcountlist}{#1}%
3199   \ifcsundef{glstr@theunit@#1}
3200     {%
3201       \ifcsdef{theH#1}%
3202       {\csdef{glstr@theunit@#1}{\csuse{theH#1}}}%
3203       {\csdef{glstr@theunit@#1}{\csuse{the#1}}}%
3204     }%
3205   {}%
3206 }
```

r@ifunitcounter

```
3207 \newcommand*{\@glstr@ifunitcounter}[3]{%
3208   \xifinlist{#1}{\@glstr@unitcountlist}{#2}{#3}%
3209 }
```

urrentunitcount

```
3210 \newcommand*{\@glstr@currentunitcount}[1]{%
3211   glo@\glstetoklabel{#1}@currunit@\glstetattribute{#1}{unitcount}.%
3212   \csuse{glstr@theunit@\glstetattribute{#1}{unitcount}}%
3213 }
```


previousunitcount

```
3214 \newcommand*\@glsxtr@previousunitcount[1]{%
3215   glo@\glsdetoklabel{#1}@prevunit@\glsgetattribute{#1}{unitcount}.%
3216   \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3217 }
```

t@currunitcount

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

t@currunitcount

```
3238 \newcommand*\@gls@local@increment@currunitcount[1]{%
3239   \gls@hasattribute{#1}{unitcount}%
3240   {%
3241     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3242     \ifcsundef{\@glsxtr@csname}%
3243     {%
3244       \csdef{\@glsxtr@csname}{1}%
3245       \listcseadd
3246         {glo@\glsdetoklabel{#1}@unitlist}%
3247         {\glsgetattribute{#1}{unitcount}.%
3248         \csuse{glsxtr@theunit@\glsgetattribute{#1}{unitcount}}}%
3249       }%
3250     }%
3251     {%
3252       \csedef{\@glsxtr@csname}%
3253       {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3254     }%
3255   }%
3256   {}%
3257 }
```

r@currunitcount

```
3258 \newcommand*{\@glxstr@currunitcount}[2]{%
3259   \ifcsundef
3260     {glo@\glstetoklabel{#1}@currunit@#2}%
3261     {0}%
3262     {\csuse{glo@\glstetoklabel{#1}@currunit@#2}}%
3263 }%
```

r@prevunitcount

```
3264 \newcommand*{\@glxstr@prevunitcount}[2]{%
3265   \ifcsundef
3266     {glo@\glstetoklabel{#1}@prevunit@#2}%
3267     {0}%
3268     {\csuse{glo@\glstetoklabel{#1}@prevunit@#2}}%
3269 }%
```

entryunitcount

```
3270 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3271   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3272   \renewcommand*{\gls@defdocnewglossaryentry}{%
3273     \renewcommand*\newglossaryentry[2]{%
3274       \PackageError{glossaries}{\string\newglossaryentry\space
3275         may only be used in the preamble when entry counting has
3276         been activated}{If you use \string\glsenableentryunitcount\space
3277         you must place all entry definitions in the preamble not in
3278         the document environment}%
3279     }%
3280   }%
```

New commands to access new fields:

```
3281   \newcommand*{\glsentrycurrcount}[1]{%
3282     \@glxstr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3283     \csuse{glxstr@theunit@\glsgetattribute{##1}{unitcount}}}%
3284   }%
3285   \newcommand*{\glsentryprevcount}[1]{%
3286     \@glxstr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3287     \csuse{glxstr@theunit@\glsgetattribute{##1}{unitcount}}}%
3288   }%
```

Access total count:

```
3289   \newcommand*{\glsentryprevtotalcount}[1]{%
3290     \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
3291     {0}%
3292     {%
3293       \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
3294     }%
3295   }%
```

Access max value:

```

3296 \newcommand*{\glentryprevmaxcount}[1]{%
3297   \ifcsundef{glo@\glsetoklabel{##1}@prevunitmax}%
3298     {0}%
3299     {%
3300       \number\csuse{glo@\glsetoklabel{##1}@prevunitmax}
3301     }%
3302   }%

```

Adjust post unset and reset:

```

3303 \let\@glxtr@entryunitcount@org@unset\glxtrpostunset
3304 \renewcommand*{\glxtrpostunset}[1]{%
3305   \@glxtr@entryunitcount@org@unset{##1}%
3306   \@glxtr@increment@currunitcount{##1}%
3307 }%
3308 \let\@glxtr@entryunitcount@org@localunset\glxtrpostlocalunset
3309 \renewcommand*{\glxtrpostlocalunset}[1]{%
3310   \@glxtr@entryunitcount@org@localunset{##1}%
3311   \@glxtr@local@increment@currunitcount{##1}%
3312 }%
3313 \let\@glxtr@entryunitcount@org@reset\glxtrpostreset
3314 \renewcommand*{\glxtrpostreset}[1]{%
3315   \glshasattribute{##1}{unitcount}%
3316   {%
3317     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
3318     \ifcsundef{\@glxtr@csname}%
3319       {}%
3320       {\csgdef{\@glxtr@csname}{0}}%
3321     }%
3322   }%
3323 }%
3324 \let\@glxtr@entryunitcount@org@localreset\glxtrpostlocalreset
3325 \renewcommand*{\glxtrpostlocalreset}[1]{%
3326   \@glxtr@entryunitcount@org@localreset{##1}%
3327   \glshasattribute{##1}{unitcount}%
3328   {%
3329     \edef\@glxtr@csname{\@glxtr@currentunitcount{##1}}%
3330     \ifcsundef{\@glxtr@csname}%
3331       {}%
3332       {\csdef{\@glxtr@csname}{0}}%
3333     }%
3334   }%
3335 }%

```

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

```

3336 \let\@cgl@@@cgl@
3337 \let\@cgl@pl@@cgl@pl@
3338 \let\@cGl@@@cGl@

```

```

3339 \let\@cGlspl@\@cGlspl@
3340 \let\@cGLS@\@cGLS@
3341 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3342 \AtEndDocument{\@gls@write@entryunitcounts}%
3343 \renewcommand*{\@gls@entry@unitcount}[3]{%
3344   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
3345   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
3346   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
3347   {%
3348     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
3349       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
3350     }%
3351     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
3352     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
3353     {%
3354       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
3355       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
3356     }%
3357   }%
3358 }%
3359 \let\glsenableentryunitcount\relax
3360 \renewcommand*{\glsenableentrycount}{%
3361   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3362     can't be used with \string\glsenableentryunitcount}%
3363   {Use one or other but not both commands}%
3364 }%
3365 }
3366 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3367 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3368 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3369   \immediate\write\@auxout
3370   {\string\@gls@entry@unitcount
3371     {\@glsentry}%
3372     {\@glsxtr@currunitcount{\@glsentry}{#1}%
3373     }%
3374     {#1}}%
3375 }

```

entryunitcounts

```

3376 \newcommand*{\@gls@write@entryunitcounts}{%
3377   \immediate\write\@auxout
3378   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3379   \count@=0\relax

```

```

3380 \foralllglsentries{\@glsentry}{%
3381   \glshasattribute{\@glsentry}{unitcount}%
3382   {%
3383     \ifglused{\@glsentry}%
3384     {%
3385       \forlistcsloop
3386         {\@gls@write@entryunitcounts@do}%
3387         {glo@\glsdetoklabel{\@glsentry}@unitlist}%
3388     }%
3389   }%
3390   \advance\count@ by \@ne
3391 }%
3392 {}%
3393 }%
3394 \ifnum\count@=0
3395   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3396     \MessageBreak with \string\glsenableentryunitcount\space but the
3397     \MessageBreak attribute ‘unitcount’ hasn’t
3398     \MessageBreak been assigned to any of the defined
3399     \MessageBreak entries}%
3400 \fi
3401 }

```

`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.

```

3402 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3403   \glsenableentryunitcount

```

Redefine \gls etc:

```

3404   \renewcommand*{\gls}{\cgl}%
3405   \renewcommand*{\Gls}{\cGls}%
3406   \renewcommand*{\glspl}{\cglspl}%
3407   \renewcommand*{\Glspl}{\cGlspl}%
3408   \renewcommand*{\GLS}{\cGLS}%
3409   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3410   \@glxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3411   \let\GlsXtrEnableEntryUnitCounting\@glxtr@setentryunitcountunsetattr
3412   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3413     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3414       can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3415     {Use one or other but not both commands}}%
3416 }

```

`tcountunsetattr`

```

3417 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3418   \@for\@glsxtr@cat:=#1\do
3419   {%
3420     \ifdefempty{\@glsxtr@cat}{}%
3421     {%
3422       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3423       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3424     }%
3425   }%
3426 }

```

1.3.6 Acronym Modifications

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

`\SetGenericNewAcronym`

```

3427 \renewcommand*{\SetGenericNewAcronym}{%
3428   \let\@Gls@entryname\@Gls@acrentryname
3429   \renewcommand{\newacronym}[4][]{%
3430     \ifdefempty{\@glsacronymlists}%
3431     {%
3432       \def\@glo@type{\acronymtype}%
3433       \setkeys{glossentry}{##1}%
3434       \DeclareAcronymList{\@glo@type}%
3435     }%
3436   }%
3437   \glskeylisttok{##1}%
3438   \glslabeltok{##2}%
3439   \glsshorttok{##3}%
3440   \glslongtok{##4}%
3441   \newacronymhook
3442   \protected@edef\@do@newglossaryentry{%
3443     \noexpand\newglossaryentry{\the\glslabeltok}%
3444     {%
3445       type=\acronymtype,%
3446       name={\expandonce{\acronymentry{##2}}},%
3447       sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%
3448       text={\the\glsshorttok},%
3449       short={\the\glsshorttok},%
3450       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3451       long={\the\glslongtok},%
3452       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3453       category=acronym,

```

```

3454 \GenericAcronymFields,%
3455 \the\glskeylisttok
3456 }%
3457 }%
3458 \@do@newglossaryentry
3459 }%
3460 \renewcommand*{\acrfullfmt}[3]{%
3461 \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3462 \renewcommand*{\Acrfullfmt}[3]{%
3463 \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3464 \renewcommand*{\ACRfullfmt}[3]{%
3465 \glslink[##1]{##2}{%
3466 \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3467 \renewcommand*{\acrfullplfmt}[3]{%
3468 \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3469 \renewcommand*{\Acrfullplfmt}[3]{%
3470 \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3471 \renewcommand*{\ACRfullplfmt}[3]{%
3472 \glslink[##1]{##2}{%
3473 \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3474 \renewcommand*{\glstryfull}[1]{\genacrfullformat{##1}{}}}%
3475 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3476 \renewcommand*{\glstryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3477 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3478 }

```

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

First save the original definitions:

```

3479 \let\@glstr@org@setacronymstyle\setacronymstyle
3480 \let\@glstr@org@newacronymstyle\newacronymstyle

```

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

```

3481 \newcommand*{\MakeAcronymsAbbreviations}{%
3482 \renewcommand*{\newacronym}[4][]{%
3483 \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3484 }%
3485 \renewcommand*{\firstacronymfont}[1]{\glstr@firstabbrvfont{##1}}%
3486 \renewcommand*{\acronymfont}[1]{\glstr@abbrvfont{##1}}%
3487 \renewcommand*{\setacronymstyle}[1]{%
3488 \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3489 unavailable.
3490 Use \string\setabbreviationstyle\space instead.
3491 The original acronym interface can be restored with
3492 \string\RestoreAcronyms}{}%
3493 }%
3494 \renewcommand*{\newacronymstyle}[1]{%

```

```

3495 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3496 available unless you restore the original acronym interface with
3497 \string\RestoreAcronyms}%
3498 \@glstr@org@newacronymstyle{##1}%
3499 }%
3500 }

```

Switch acronyms to abbreviations:

```
3501 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

3502 \newcommand*{\RestoreAcronyms}{%
3503 \SetGenericNewAcronym
3504 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3505 \renewcommand{\acronymfont}[1]{##1}%
3506 \let\setacronymstyle\@glstr@org@setacronymstyle
3507 \let\newacronymstyle\@glstr@org@newacronymstyle

```

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

```

3508 \renewcommand*\@gls@link@checkfirsthyper{%
3509 \ifglused{\glslabel}%
3510 {\let\glstrifwasfirstuse\@secondoftwo}
3511 {\let\glstrifwasfirstuse\@firstoftwo}%
3512 \@glstr@org@checkfirsthyper
3513 }
3514 \glssetcategoryattribute{acronym}{regular}{false}%
3515 \setacronymstyle{long-short}%
3516 }

```

\glsacspace Allow the user to customise the maximum value.

```

3517 \renewcommand*{\glsacspace}[1]{%
3518 \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3519 \ifdim\dimen@<\glsacspacemax~\else\space\fi
3520 }

```

\glsacspacemax Value used in the above.

```
3521 \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

```
3522 \newcommand*{\@glstr@reg@glosslist}{}
```


Save the original definition of `\makeglossaries`:

```
3523 \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
```

```
3524 \renewcommand*{\makeglossaries}[1] [] {%
3525   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3526     \PackageError{glossaries-extra}{\string\makeglossaries\space
3527       not permitted\MessageBreak with record=only package option}%
3528     {You may only use \string\makeglossaries\space with
3529       record=off or record=alsoindex options}%
3530   \else
3531     \ifblank{#1}%
3532     {\@glxtr@org@makeglossaries}%
3533     {%
3534       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3535         \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3536           not permitted\MessageBreak with record=alsoindex package option}%
3537         {You may only use the hybrid \string\makeglossaries[...]\space with
3538           record=off option}%
3539       \else
3540         \edef\@glxtr@reg@glosslist{#1}%
3541         \ifundef{\glswrite}{\newwrite\glswrite}{}%
3542         \protected@write\@auxout{}{\string\providecommand
3543           \string\@glxtr@order[1]{}%
3544           \protected@write\@auxout{}{\string\providecommand
3545             \string\@istfilename[1]{}%
3546             \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3547             \protected@write\@auxout{}{\string\@glxtr@order{\glxtr@order}}%
3548             \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}%
3549             \write\@auxout{\string\providecommand\string\@glxtr@reference[3]{}%

```

Iterate through each supplied glossary type and activate it.

```
3550   \@for\@glo@type:=#1\do{%
3551     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3552   }%
```

New glossaries must be created before `\makeglossaries`:

```
3553   \renewcommand*\newglossary[4] [] {%
3554     \PackageError{glossaries}{New glossaries
3555       must be created before \string\makeglossaries}{You need
3556       to move \string\makeglossaries\space after all your
3557       \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
3558   \let\@makeglossary\relax
3559   \let\makeglossary\relax
```

```

3560      \renewcommand\makeglossaries[1][]{}%
      Disable all commands that have no effect after \makeglossaries
3561      \@disable@onlypremakeg
      Allow see key:
3562      \let\gls@checkseeallowed\relax
      Adjust \@do@seeglossary. This needs to check for the entries existence.
3563      \renewcommand*{\@do@seeglossary}[2]{%
3564      \glsdoifexists{##1}%
3565      {%
3566      \edef\gls@label{\glsdetoklabel{##1}}%
3567      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3568      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3569      {\glsxtr@org@doseeglossary{##1}{##2}}%
3570      {%
3571      @@glsxtrwrglossmark
3572      \protected@write\@auxout{}{%
3573      \string\gls@reference
3574      {\gls@type}{\gls@label}{\string\glsseeformat##2{}}%
3575      }%
3576      }%
3577      }%
3578      }%
      Adjust \@do@@wrglossary
3579      \let\glsxtr@@do@@wrglossary\@do@@wrglossary
3580      \def\@do@@wrglossary{%
3581      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3582      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3583      {\glsxtr@@do@@wrglossary}%
3584      {\gls@noidxglossary}%
3585      }%
      Suppress warning about no \makeglossaries
3586      \let\warn@nomakeglossaries\relax
3587      \def\warn@noprintglossary{%
3588      \GlossariesWarningNoLine{No \string\printglossary\space
3589      or \string\printglossaries\space
3590      found.^^J(Remove \string\makeglossaries\space if you don't want
3591      any glossaries.)^^JThis document will not have a glossary}%
3592      }%
      Only warn for glossaries not listed.
3593      \renewcommand{\@gls@noref@warn}[1]{%
3594      \edef\gls@type{##1}%
3595      \expandafter\DTLifinlist\expandafter{\gls@type}{\glsxtr@reg@glosslist}%
3596      {%
3597      \GlossariesExtraWarning{Can't use
3598      \string\printnoidxglossary[type={\gls@type}]
3599      when '\gls@type' is listed in the optional argument of

```

```

3600         \string\makeglossaries}%
3601     }%
3602     {%
3603         \GlossariesWarning{Empty glossary for
3604         \string\printnoidxglossary[type={##1}].
3605         Rerun may be required (or you may have forgotten to use
3606         commands like \string\gls)}%
3607     }%
3608 }%

```

Adjust display number list to check for type:

```

3609     \renewcommand*{\glsdisplaynumberlist}[1]{%
3610         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3611         {\@glxtr@idx@displaynumberlist{##1}}%
3612         {\@glxtr@noidx@displaynumberlist{##1}}%
3613     }%

```

Adjust entry list:

```

3614     \renewcommand*{\glsentrynumberlist}[1]{%
3615         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3616         {\@glxtr@idx@entrynumberlist{##1}}%
3617         {\@glxtr@noidx@entrynumberlist{##1}}%
3618     }%

```

Adjust number list loop

```

3619     \renewcommand*{\glsnumberlistloop}[2]{%
3620         \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3621         {%
3622             \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3623             not available for glossary ‘##1’}{}%
3624         }%
3625         {\@glxtr@noidx@numberlistloop{##1}{##2}}%
3626     }%

```

Only sanitize sort for normal indexing glossaries.

```

3627     \renewcommand*{\glsprestandardsort}[3]{%
3628         \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
3629         {%
3630             \glsdosanitizesort
3631         }%
3632         {%
3633             \ifglssanitizesort
3634                 \@gls@noidx@sanitizesort
3635             \else
3636                 \@gls@noidx@nosanitizesort
3637             \fi
3638         }%
3639     }%

```

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

```

3640 \renewcommand*\new@glossaryentry[2]{%
3641 \PackageError{glossaries-extra}{Glossary entries must be defined
3642 in the preamble\MessageBreak when you use the optional argument
3643 of \string\makeglossaries}{Either move your definitions to the
3644 preamble or don't use the optional argument of
3645 \string\makeglossaries}%
3646 }%

```

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).

```

3647 \let\@glo@assign@sortkey\@glxtr@mixed@assign@sortkey
3648 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3649 \expandafter\@glxtr@gettype\expandafter,\@glxtr@printglossopts,%
3650 type=\glsdefaulttype,\@end@glxtr@gettype
3651 \def\@glo@sorttype{\@glo@default@sorttype}%
3652 }%

```

Check automake setting:

```

3653 \ifglsautomake
3654 \renewcommand*\@gls@doautomake{%
3655 \@for\@gls@type:=\@glxtr@reg@glosslist\do{%
3656 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3657 }%
3658 }%
3659 \fi

```

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

```

3660 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3661 \fi
3662 }%
3663 \fi
3664 }

```

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

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

```

3665 \newcommand{\@glxtr@orgprintglossary}[2]{%
3666 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3667 \def\glossarytitle{%
3668 \ifcsdef{\@glo@type@\@glo@type @title}%
3669 {\csuse{\@glo@type@\@glo@type @title}}%
3670 {\glossaryname}}%
3671 \def\glossarytoctitle{\glossarytitle}%

```

```

3672 \let\org@glossarytitle\glossarytitle
3673 \def\@glossarystyle{%
3674   \ifx\@glossary@default@style\relax
3675     \GlossariesWarning{No default glossary style provided \MessageBreak
3676       for the glossary '\@glo@type'. \MessageBreak
3677       Using deprecated fallback. \MessageBreak
3678       To fix this set the style with \MessageBreak
3679       \string\setglossarystyle\space or use the \MessageBreak
3680       style key=value option}%
3681   \fi
3682 }%
3683 \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
3684 \let\org@glossaryentrynumbers\glossaryentrynumbers
3685 \bgroup
3686   \@printgloss@setsort
3687   \setkeys{printgloss}{#1}%
3688   \ifx\glossarytitle\org@glossarytitle
3689   \else
3690     \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3691   \fi
3692   \let\currentglossary\@glo@type
3693   \let\org@glossaryentrynumbers\glossaryentrynumbers
3694   \let\glsnonextpages\@glsnonextpages
3695   \let\glsnextpages\@glsnextpages

3696   \glsxtractivatenopost
3697   \gls@dotoc@title
3698   \@glossarystyle
3699   \let\gls@org@glossaryentryfield\glossentry
3700   \let\gls@org@glossarysubentryfield\subglossentry
3701   \renewcommand{\glossentry}[1]{%
3702     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3703     \gls@org@glossaryentryfield{##1}%
3704   }%
3705   \renewcommand{\subglossentry}[2]{%
3706     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3707     \gls@org@glossarysubentryfield{##1}{##2}%
3708   }%
3709   \@gls@preglossaryhook
3710   #2%
3711 \egroup
3712 \global\let\glossaryentrynumbers\org@glossaryentrynumbers
3713 \global\let\warn@noprntglossary\relax
3714 }

```

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

```

3715 \newcommand*\glsxtractivatenopost{%
3716   \let\nopostdesc\@nopostdesc
3717   \let\glsxtrnopostpunc\@glsxtr@nopostpunc
3718 }

```

lsxtrnopostpunc

```
3719 \newrobustcmd*{\glxtrnopostpunc}{}
```

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

```
3720 \newcommand{\@glxtr@nopostpunc}{%
3721   \let\@glxtr@org@postdescription\glspostdescription
3722   \ifglsnopostdot
3723     \renewcommand{\glspostdescription}{%
3724       \glsnopostdottrue
3725       \let\glspostdescription\@glxtr@org@postdescription
3726       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3727       \glxtrpostdescription
3728       \@glxtr@nopostpunc@postdesc}%
3729   \else
3730     \renewcommand{\glspostdescription}{%
3731       \let\glspostdescription\@glxtr@org@postdescription
3732       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3733       \glxtrpostdescription
3734       \@glxtr@nopostpunc@postdesc}%
3735   \fi
3736   \glsnopostdotfalse
3737 }
```

stpunc@postdesc

```
3738 \newcommand*{\@glxtr@nopostpunc@postdesc}{}
```

restore@postpunc

```
3739 \newcommand*{\@glxtr@restore@postpunc}{%
3740   \def\@glxtr@nopostpunc@postdesc{%
3741     \@glxtr@org@postdescription
3742     \let\@glxtr@nopostpunc@postdesc\@empty
3743     \let\glxtrrestorepostpunc\@empty
3744   }%
3745 }
```

restorepostpunc Does nothing outside of glossary.

```
3746 \newcommand*{\glxtrrestorepostpunc}{}
```

\@printglossary Redefine.

```
3747 \renewcommand{\@printglossary}[2]{%
3748   \def\@glxtr@printglossopts{#1}%
3749   \@glxtr@orgprintglossary{#1}{#2}%
3750 }
```

Add a key that switches off the entry targets:

```
3751 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3752   \ifcase\nr
```

```

3753 \let\@glstarget\glsdohypertarget
3754 \else
3755 \let\@glstarget\@secondoftwo
3756 \fi
3757 }

```

hypernameprefix

```

3758 \newcommand{\@glxtrhypernameprefix}{}

```

New to v1.20:

```

3759 \define@key{printgloss}{targetnameprefix}{%
3760 \renewcommand{\@glxtrhypernameprefix}{#1}%
3761 }

```

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

```

3762 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3763 \renewcommand{\glsdohypertarget}[2]{%
3764 \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3765 }

```

@makeglossaries For the benefit of makeglossaries

```

3766 \newcommand*{\@glxtr@makeglossaries}[1]{}

```

@glxtr@gettype Get just the type.

```

3767 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3768 \def\@glo@type{#2}%
3769 }

```

@assign@sortkey Assign the sort key.

```

3770 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3771 \edef\@glo@type{\@glo@type}%
3772 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3773 {%
3774 \@glo@no@assign@sortkey{#1}%
3775 }%
3776 {%
3777 \@glo@assign@sortkey{#1}%
3778 }%
3779 }%

```

Display number list for the regular version:

splaynumberlist

```

3780 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3781 \newcommand*{\@glsxtr@noidx@displaynumberlist}[1]{%
3782   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3783   \ifdef\@gls@loclist
3784   {%
3785     \def\@gls@noidxloclist@sep{%
3786       \def\@gls@noidxloclist@sep{%
3787         \def\@gls@noidxloclist@sep{%
3788           \glsnumlistsep
3789         }%
3790         \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
3791       }%
3792     }%
3793     \def\@gls@noidxloclist@finalsep{}%
3794     \def\@gls@noidxloclist@prev{}%
3795     \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
3796     \@gls@noidxloclist@finalsep
3797     \@gls@noidxloclist@prev
3798   }%
3799   {%

3800   \glsxtrundeftag
3801   \glsdoifexists{#1}%
3802   {%
3803     \GlossariesWarning{Missing location list for ‘#1’. Either
3804       a rerun is required or you haven’t referenced the entry.}%
3805   }%
3806 }%
3807 }%
3808
```

And for the number list loop:

@numberlistloop

```

3809 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3810   \letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
3811   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3812   \let\@gls@org@glsseeformat\glsseeformat
3813   \let\glsnoidxdisplayloc#2\relax
3814   \let\glsseeformat#3\relax
3815   \ifdef\@gls@loclist
3816   {%
3817     \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3818   }%
3819   {%

3820   \glsxtrundeftag
3821   \glsdoifexists{#1}%
3822   {%
3823     \GlossariesWarning{Missing location list for ‘##1’. Either
```



```

3824         a rerun is required or you haven't referenced the entry.}%
3825     }%
3826 }%
3827 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3828 \let\glsseeformat\@gls@org@glsseeformat
3829 }%

```

Same for entry number list.

entrynumberlist

```

3830 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3831   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3832   \ifdef\@gls@loclist
3833   {%
3834     \glsnoidxloclist{\@gls@loclist}%
3835   }%
3836   {%
3837     \glsxtrundeftag
3838     \glsdoifexists{#1}%
3839     {%
3840       \GlossariesWarning{Missing location list for '#1'. Either
3841         a rerun is required or you haven't referenced the entry.}%
3842     }%
3843   }%
3844 }%

```

entrynumberlist

```

3845 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3846 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3847   \protected@edef\@glsxtr@titlelabel{#1}%
3848   \ifdefvoid\@glsxtr@titlelabel
3849   {%
3850   {%
3851     \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle{#1}}}%
3852   }%
3853   \ifdefvoid{\@glsxtr@titlelabel}%
3854   {%
3855     \DTLifint{#1}%
3856     {%
3857       \ifnum#1<256\relax
3858       \edef#2{\char#1\relax}%
3859     \else
3860     \edef#2{#1}%
3861     \fi
3862   }%
3863   {%
3864     \ifcsundef{#1groupname}%

```

```

3865     {\def#2{#1}}%
3866     {\letcs#2{#1groupname}}%
3867   }%
3868 }%
3869 {%
3870   \let#2\@glsxtr@titlelabel
3871 }%
3872 }

```

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

```

3873 \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.

```

3874 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3875   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3876   \@onelevel@sanitize\@glsxtr@titlelabel
3877   \ifcsdef{\@glsxtr@titlelabel}
3878     {\letcs{#2}{\@glsxtr@titlelabel}}%
3879     {\glsxtr@org@getgrouptitle{#1}{#2}}%
3880 }
3881 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

trsetgrouptitle Sets the title for the given group label.

```

3882 \newcommand{\glsxtrsetgrouptitle}[2]{%
3883   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3884   \@onelevel@sanitize\@glsxtr@titlelabel
3885   \csxdef{\@glsxtr@titlelabel}{#2}%
3886 }

```

\glsnavigation Redefine to use new user-level command.

```

3887 \renewcommand*{\glsnavigation}{%
3888   \def\@gls@between{}%
3889   \ifcsundef{\@gls@hypergroupelist@\@glo@type}%
3890     {%
3891       \def\@gls@list{}%
3892     }%
3893     {%
3894       \expandafter\let\expandafter\@gls@list
3895       \csname @gls@hypergroupelist@\@glo@type\endcsname
3896     }%
3897     \@for\@gls@tmp:=\@gls@list\do{%
3898       \@gls@between
3899       \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3900       \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3901       \let\@gls@between\glshypernavsep
3902     }%
3903 }

```

@noidx@glossary

```
3904 \renewcommand*{\@print@noidx@glossary}{%
3905   \ifcsdef{@glsref@{\@glo@type}%
3906     {%
3907       \ifcsdef{@glo@sortmacro@{\@glo@sorttype}%
3908         {%
3909           \csuse{@glo@sortmacro@{\@glo@sorttype}}{\@glo@type}%
3910         }%
3911       {%
3912         \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3913       }%
3914     }%
3915     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3916     \glossarypreamble
```

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

```
3916   \def\@gls@currentlettergroup{%
3917     \begin{theglossary}%
3918     \glossaryheader
3919     \glsresetentrylist
3920     \forlistcsloop{\@gls@noidx@do}{\@glsref@{\@glo@type}%
3921     \end{theglossary}%
3922     \glossarypostamble
3923   }%
3924   {%
```

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

```
3925   \glstrifemptyglossary{\@glo@type}%
3926   }%
3927   {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3928   \@gls@noref@warn{\@glo@type}%
3929 }%
3930 }
```

noidxdisplayloc Patch to check for range formations.

```
3931 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3932   \setentrycounter[#1]{#2}%
3933   \@glstr@display@loc#3\empty\end@glstr@display@loc{#4}%
3934 }
```

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

```
3935 \def\@glstr@display@loc#1#2\end@glstr@display@loc#3{%
3936   \ifx#1(\relax
3937     \glstrdisplaystartloc{#2}{#3}%
3938   \else
3939     \ifx#1)\relax
3940       \glstrdisplayendloc{#2}{#3}%
3941     \else
```

```

3942      \glxtrdisplaysingleloc{#1#2}{#3}%
3943      \fi
3944      \fi
3945 }

```

`isplaysingleloc` Single location.

```

3946 \newcommand*{\glxtrdisplaysingleloc}[2]{%
3947   \csuse{#1}{#2}%
3948 }

```

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

`displaystartloc` Start of a location range.

```

3949 \newcommand*{\glxtrdisplaystartloc}[2]{%
3950   \edef\glxtrlocrangefmt{#1}%
3951   \ifx\glxtrlocrangefmt\empty
3952     \def\glxtrlocrangefmt{\glsnnumberformat}%
3953   \fi
3954   \expandafter\glxtrdisplaysingleloc
3955   \expandafter{\glxtrlocrangefmt}{#2}%
3956 }

```

`trdisplayendloc` End of a location range.

```

3957 \newcommand*{\glxtrdisplayendloc}[2]{%
3958   \edef\@glxtr@tmp{#1}%
3959   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{\glsnnumberformat}}{}%
3960   \ifx\glxtrlocrangefmt\@glxtr@tmp
3961     \else
3962       \GlossariesExtraWarning{Mismatched end location range
3963         (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3964     \fi
3965     \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3966     \expandafter\glxtrdisplaysingleloc
3967     \expandafter{\glxtrlocrangefmt}{#2}%
3968   \def\glxtrlocrangefmt{}%
3969 }

```

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

```

3970 \newcommand*{\glxtrdisplayendloohook}[2]{%

```

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

```

3971 \newcommand*{\glxtrlocrangefmt}{}

```

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

```

3972 \def\@gl@removespaces#1 #2\@nil{%
3973   \toks@=\expandafter{\the\toks@#1}%
3974   \ifx\#2\%

```

```

3975 \edef\x{\the\toks@}%
3976 \ifx\x\empty
3977 \else
3978 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3979 \fi
3980 \else
3981 \@gls@ReturnAfterFi{%
3982 \glxtrlocationhyperlink{\glsentrycounter}{\@glo@counterprefix}{\the\toks@}%
3983 }%
3984 \fi
3985 }

```

cationhyperlink

```

3986 \newcommand*{\glxtrlocationhyperlink}[3]{%
3987 \ifdefined\glxtrsupplocationurl
3988 {%
3989 \glxtrhyperlink{#1#2#3}{#3}%
3990 }%
3991 {%
3992 \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
3993 }%
3994 }

```

supphypernumber

```

3995 \newcommand*{\glxtrsupphypernumber}[1]{%
3996 {%
3997 \glshasattribute{\glscurrententrylabel}{externallocation}%
3998 {%
3999 \def\glxtrsupplocationurl{%
4000 \glsglshattribute{\glscurrententrylabel}{externallocation}}%
4001 }%
4002 {%
4003 \def\glxtrsupplocationurl{}%
4004 }%
4005 \glshypernumber{#1}%
4006 }%
4007 }

```

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

@print@glossary

```

4008 \renewcommand{\@print@glossary}{%
4009 \makeatletter
4010 \@input{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4011 \ifFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4012 {}%
4013 {\glxtrNoGlossaryWarning{\@glo@type}}%
4014 \ifglxindy

```

```

4015 \ifcsundef{@xdy@\@glo@type @language}%
4016 {%
4017 \edef\@do@auxoutstuff{%
4018 \noexpand\AtEndDocument{%
4019 \noexpand\immediate\noexpand\write\@auxout{%
4020 \string\providecommand\string\@xdylanguage[2]{}}%
4021 \noexpand\immediate\noexpand\write\@auxout{%
4022 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4023 }%
4024 }%
4025 }%
4026 {%
4027 \edef\@do@auxoutstuff{%
4028 \noexpand\AtEndDocument{%
4029 \noexpand\immediate\noexpand\write\@auxout{%
4030 \string\providecommand\string\@xdylanguage[2]{}}%
4031 \noexpand\immediate\noexpand\write\@auxout{%
4032 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4033 @language\endcsname}}%
4034 }%
4035 }%
4036 }%
4037 \@do@auxoutstuff
4038 \edef\@do@auxoutstuff{%
4039 \noexpand\AtEndDocument{%
4040 \noexpand\immediate\noexpand\write\@auxout{%
4041 \string\providecommand\string\@gls@codepage[2]{}}%
4042 \noexpand\immediate\noexpand\write\@auxout{%
4043 \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4044 }%
4045 }%
4046 \@do@auxoutstuff
4047 \fi
4048 \renewcommand*{\@warn@nomakeglossaries}{%
4049 \GlossariesWarningNoLine{\string\makeglossaries\space
4050 hasn't been used,^^Jthe glossaries will not be updated}%
4051 }%
4052 }

```

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

`oGlsWarningHead` Header message.

```

4053 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4054 This document is incomplete. The external file associated with
4055 the glossary '#1' (which should be called \texttt{#2})
4056 hasn't been created.%
4057 }

```

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

```

4058 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4059   This has probably happened because there are no entries defined
4060   in this glossary.%
4061 }

```

WarningEmptyMain The default “main” glossary is empty.

```

4062 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4063   If you don't want this glossary,
4064   add \texttt{nomain} to your package option list when you load
4065   \texttt{glossaries-extra.sty}. For example:%
4066 }

```

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

```

4067 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4068   Did you forget to use \texttt{type=#1} when you defined your
4069   entries? If you tried to load entries into this glossary with
4070   \texttt{\string\loadglsentries} did you remember to use
4071   \texttt{[#1]} as the optional argument? If you did, check that
4072   the definitions in the file you loaded all had the type set
4073   to \texttt{\string\glsdefaulttype}.%
4074 }

```

WarningCheckFile Advisory message to check the file contents.

```

4075 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4076   Check the contents of the file \texttt{#1}. If
4077   it's empty, that means you haven't indexed any of your entries in this
4078   glossary (using commands like \texttt{\string\gls} or
4079   \texttt{\string\glsadd}) so this list can't be generated.
4080   If the file isn't empty, the document build process hasn't been
4081   completed.%
4082 }

```

WarningAutoMake Message when automake option has been used.

```

4083 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4084   You may need to rerun \LaTeX. If you already have, it may be that
4085   \TeX's shell escape doesn't allow you to run
4086   \ifglxindy xindy\else makeindex\fi. Check the
4087   transcript file \texttt{\jobname.log}. If the shell escape is
4088   disabled, try one of the following:
4089
4090   \begin{itemize}
4091     \item Run the external (Lua) application:
4092
4093       \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4094
4095     \item Run the external (Perl) application:
4096
4097       \texttt{makeglossaries \string"\jobname\string"}
4098   \end{itemize}

```

```

4099
4100 Then rerun \LaTeX\ on this document.
4101 \GlossariesExtraWarning{Rerun required to build the
4102 glossary ‘#1’ or check TeX’s shell escape allows
4103 you to run \ifglxsindy xindy\else makeindex\fi}%
4104 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4105 \newcommand{\GlsXtrNoGlsWarningMismatch}{%
4106 You need to either replace \texttt{\string\makenoidxglossaries}
4107 with \texttt{\string\makeglossaries} or replace
4108 \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4109 \texttt{\string\printnoidxglossary}
4110 (or \texttt{\string\printnoidxglossaries}) and then rebuild
4111 this document.%
4112 }

```

WarningBuildInfo Build advice.

```

4113 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4114 Try one of the following:
4115 \begin{itemize}
4116 \item Add \texttt{automake} to your package option list when you load
4117 \texttt{glossaries-extra.sty}. For example:
4118
4119 \texttt{\string\usepackage[automake]%
4120 \glsopenbrace glossaries-extra\glsclosebrace}
4121
4122 \item Run the external (Lua) application:
4123
4124 \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4125
4126 \item Run the external (Perl) application:
4127
4128 \texttt{makeglossaries \string"\jobname\string"}
4129 \end{itemize}
4130
4131 Then rerun \LaTeX\ on this document.%
4132 }

```

oGlsWarningTail Final paragraph.

```

4133 \newcommand{\GlsXtrNoGlsWarningTail}{%
4134 This message will be removed once the problem has been fixed.%
4135 }

```

GlsWarningNoOut No out file created. Build advice.

```

4136 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4137 The file \texttt{#1} doesn’t exist. This most likely means you haven’t used
4138 \texttt{\string\makeglossaries} or you have used

```



```

4139 \texttt{\string\nofiles}. If this is just a draft version of the
4140 document, you can suppress this message using the
4141 \texttt{nomissingglstext} package option.%
4142 }

```

glossarywarning

```

4143 \newcommand*{\@glstr@defaultnoglossarywarning}[1]{%
4144 \glossarysection[\glossarytoctitle]{\glossarytitle}
4145 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@\@glo@type @in\endcsname}
4146 \par
4147 \glstrifemptyglossary{#1}%
4148 {%
4149 \GlsXtrNoGlsWarningEmptyStart\space
4150 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4151 \medskip
4152 \noindent\texttt{\string\usepackage[nomain\ifglssacronym ,acronym\fi]%
4153 \glsoopenbrace glossaries-extra\glsclosebrace}
4154 \medskip
4155 }%
4156 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4157 }%
4158 {%
4159 \IfFileExists{\jobname.\csname @glotype@\@glo@type @out\endcsname}
4160 {%
4161 \GlsXtrNoGlsWarningCheckFile
4162 {\jobname.\csname @glotype@\@glo@type @out\endcsname}
4163
4164 \ifglssautomake
4165
4166 \GlsXtrNoGlsWarningAutoMake{#1}
4167
4168 \else
4169
4170 \ifthenelse{\equal{#1}{main}}%
4171 {%
4172 \GlsXtrNoGlsWarningEmptyMain\par
4173 \medskip
4174 \noindent\texttt{\string\usepackage[nomain]%
4175 \glsoopenbrace glossaries-extra\glsclosebrace}
4176 \medskip
4177 }%
4178 {}%
4179
4180 \ifdefequal\makeglossaries\@no@makeglossaries
4181 {%
4182 \GlsXtrNoGlsWarningMisMatch
4183 }%
4184 {}%
4185 \GlsXtrNoGlsWarningBuildInfo

```

```

4186     }%
4187     \fi
4188 }%
4189 {%
4190     \GlsXtrNoGlsWarningNoOut
4191     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4192 }%
4193 }%
4194 \par
4195 \GlsXtrNoGlsWarningTail
4196 }

```

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

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

```

4197 \newcommand*{\glxtrresourcefile}[2] [] {%

```

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

```

4198     \disable@keys{glossaries-extra.sty}{record}%
4199     \glxtr@writefields
4200     \protected@write\@auxout{\glxtrresourceinit}{\string\glxtr@resource{#1}{#2}}%
4201     \let\@glxtr@org@see@noindex\@gls@see@noindex
4202     \let\@gls@see@noindex\relax
4203     \IfFileExists{#2.glstex}%
4204     {%

```

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

```

4205     \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4206     \makeatletter
4207     \@input{#2.glstex}%
4208     \@bibgls@restreat
4209 }%
4210 {%
4211     \GlossariesExtraWarning{No file '#2.glstex'}%
4212 }%
4213     \let\@gls@see@noindex\@glxtr@org@see@noindex
4214 }
4215 \@onlypreamble\glxtrresourcefile

```

trresourcecount

```

4216 \newcount\glxtrresourcecount

```

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

```

4217 \newcommand*{\GlsXtrLoadResources}[1] [] {%
4218     \ifnum\glxtrresourcecount=0\relax
4219     \glxtrresourcefile[#1]{\jobname}%
4220     \else
4221     \glxtrresourcefile[#1]{\jobname-\the\glxtrresourcecount}%

```

```

4222 \fi
4223 \advance\glxstrresourcecount by 1\relax
4224 }

glxstr@resource
4225 \newcommand*{\glxstr@resource}[2]{

\glxstr@fields
4226 \newcommand*{\glxstr@fields}[1]{

xtr@texencoding
4227 \newcommand*{\glxstr@texencoding}[1]{

\glxstr@langtag
4228 \newcommand*{\glxstr@langtag}[1]{

@pluralsuffixes
4229 \newcommand*{\glxstr@pluralsuffixes}[4]{

tr@shortcutsval
4230 \newcommand*{\glxstr@shortcutsval}[1]{

sxtr@linkprefix
4231 \newcommand*{\glxstr@linkprefix}[1]{

xtr@writefields This information only needs to be written once, so disable it after it's been used.
4232 \newcommand*{\glxstr@writefields}{%

4233 \protected@write\@auxout{%
4234 {\string\providecommand*{\string\glxstr@fields}[1]{}}%
4235 \protected@write\@auxout{%
4236 {\string\providecommand*{\string\glxstr@resource}[2]{}}%
4237 \protected@write\@auxout{%
4238 {\string\providecommand*{\string\glxstr@pluralsuffixes}[4]{}}%
4239 \protected@write\@auxout{%
4240 {\string\providecommand*{\string\glxstr@shortcutsval}[1]{}}%
4241 \protected@write\@auxout{%
4242 {\string\providecommand*{\string\glxstr@linkprefix}[1]{}}%
4243 \protected@write\@auxout{{\string\glxstr@fields{\@gls@keymap}}}%

4244 \protected@write\@auxout{%
4245 {\string\providecommand*{\string\glxstr@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 indi-
cated in the sort key when using \glxstrresourcefile.

4246 \ifdef\CurrentTrackedLanguageTag
4247 {%

```

```

4248     \protected@write\@auxout{}\{%
4249       \string\glxtr@langtag{\CurrentTrackedLanguageTag}}%
4250   }%
4251   {%
4252   \protected@write\@auxout{}\{\string\glxtr@pluralsuffixes
4253     {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
4254     {\glxtrabbrvpluralsuffix}}%
4255   \ifdef\inputencodingname
4256   {%
4257     \protected@write\@auxout{}\{\string\glxtr@texencoding{\inputencodingname}}%
4258   }%
4259   {%

```

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.)

```

4260     \ifpackageloaded{fontspec}%
4261     {\protected@write\@auxout{}\{\string\glxtr@texencoding{utf8}}}%
4262     {%
4263   }%
4264   \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.

```

4265   \AtBeginDocument
4266   {\protected@write\@auxout{}\{\string\glxtr@linkprefix{\glolinkprefix}}}%
4267   \let\glxtr@writefields\relax

```

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

```

4268   \ifglautomake
4269   \IfFileExists{\jobname.aux}%
4270   {\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`.

```

4271   \ifx\@gls@doautomake\@gls@doautomake@err
4272   \let\@gls@doautomake\relax
4273   \fi
4274   \fi
4275 }

```

`do@automake@err`

```

4276 \newcommand*{\@gls@doautomake@err}{%
4277   \PackageError{glossaries}{You must use
4278     \string\makeglossaries\space with automake=true}
4279   {%
4280     Either remove the automake=true setting or
4281     add \string\makeglossaries\space to your document preamble.%
4282   }%
4283 }

```

Allow locations specific to a particular counter to be recorded.

```

\glxtr@record
4284 \newcommand*{\glxtr@record}[5]{%

r@counterrecord  Aux file command.
4285 \newcommand*{\glxtr@counterrecord}[3]{%
4286   \glxtrfieldlistgadd{#1}{record.#2}{#3}%
4287 }

unterrecordhook  Hook used by \@glxtr@dorecord.
4288 \newcommand*{\@glxtr@counterrecordhook}{%

trRecordCounter  Activate recording for a particular counter (identified in the argument).
4289 \newcommand*{\GlsXtrRecordCounter}[1]{%
4290   \@glxtr@recordcounter{#1}%
4291 }
4292 \@onlypreamble\GlsXtrRecordCounter

docounterrecord
4293 \newcommand*{\@glxtr@docounterrecord}[1]{%
4294   \protected@write\@auxout{}\string\glxtr@counterrecord
4295     {\@glslabel}{#1}{\csuse{the#1}}}%
4296 }

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.
4297 \newcommand*{\glxtrglossentry}[1]{%
4298   \glxtrtitleorpdforheading
4299   {\@glxtrglossentry{#1}}%
4300   {\glsenryname{#1}}%
4301   {\glxtrheadname{#1}}%
4302 }

lsxtrglossentry  Another test is needed in case \@glxtrglossentry has been written to the table of contents.
4303 \newrobustcmd*{\@glxtrglossentry}[1]{%
4304   \glxtrtitleorpdforheading
4305   {%
4306     \glsdofexists{#1}%
4307     {%
4308       \begingroup
4309         \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4310         \edef\currentglossary{\glsenrytype{\glscurrententrylabel}}%

```

```

4311         \ifglshasparent{#1}%
4312         {\glssubentryitem{#1}}%
4313         {\glsenentryitem{#1}}%
4314         \glstarget{#1}{\glossentryname{#1}}%
4315     \endgroup
4316 }%
4317 }%
4318 {\glsenentryname{#1}}%
4319 {\glsxtrheadname{#1}}%
4320 }

```

`glossentryother` As `\glsxtrglossentry` 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.

```

4321 \newcommand*{\glsxtrglossentryother}[3]{%
4322     \ifstrempy{#1}%
4323     {%
4324         \ifcsdef{glsxtrhead#3}%
4325         {%
4326             \glsxtrtitleorpdforheading
4327             {\@glsxtrglossentryother{#2}{#3}{#1}}%
4328             {\@gls@entry@field{#2}{#3}}%
4329             {\csuse{glsxtrhead#3}{#2}}%
4330         }%
4331         {%
4332             \glsxtrtitleorpdforheading
4333             {\@glsxtrglossentryother{#2}{#3}{#1}}%
4334             {\@gls@entry@field{#2}{#3}}%
4335             {\@gls@entry@field{NoCaseChange{#2}}{#3}}%
4336         }%
4337     }%
4338     {%
4339         \glsxtrtitleorpdforheading
4340         {\@glsxtrglossentryother{#2}{#3}{#1}}%
4341         {\@gls@entry@field{#2}{#3}}%
4342         {#1}%
4343     }%
4344 }

```

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

```

4345 \newrobustcmd*{\@glsxtrglossentryother}[3]{%
4346     \glsxtrtitleorpdforheading
4347     {%
4348         \glsdoidexists{#1}%
4349         {%
4350             \begingroup
4351             \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4352             \edef\currentglossary{\glsenentrytype{\glscurrententrylabel}}%

```

```

4353     \ifglshasparent{#1}%
4354     {\glssubentryitem{#1}}}%
4355     {\glsentryitem{#1}}}%
4356     \glstarget{#1}{\glossentrynameother{#1}{#2}}}%
4357   \endgroup
4358 }%
4359 }%
4360 {\@gls@entry@field{#1}{#2}}}%
4361 {#3}}%
4362 }

```

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

```

4363 \newcommand*{\printunsrtglossary}{%
4364   \@ifstar\s@printunsrtglossary\@printunsrtglossary
4365 }

```

`\printunsrtglossary` Unstarred version.

```

4366 \newcommand*{\@printunsrtglossary}[1][]{%
4367   \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4368 }

```

`\printunsrtglossary` Starred version.

```

4369 \newcommand*{\s@printunsrtglossary}[2][]{%
4370   \begingroup
4371     #2%
4372     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4373   \endgroup
4374 }

```

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

```

4375 \newcommand*{\printunsrtglossaries}{%
4376   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4377 }

```

`\@print@unsrt@glossary`

```

4378 \newcommand*{\@print@unsrt@glossary}{%
4379   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4380   \glossarypreamble
4381   check for empty list
4381   \glstrifemptyglossary{\@glo@type}%
4382   {%
4383     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4384   }%
4385   {%

```

```

4386 \key@ifundefined{glossentry}{group}%
4387 {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4388 {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4389 \def\@gls@currentlettergroup{}%

```

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

```

4390 \def\@glsxtr@doglossary{%
4391   \begin{theglossary}%
4392   \glossaryheader
4393   \glsresetentrylist
4394 }%
4395 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4396   :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
4397   \ifdefempty{\glscurrententrylabel}
4398   }{%
4399   {%

```

Provide a hook (for example to measure width).

```

4400 \let\glsxtr@process\@firstofone
4401 \let\printunsrtglossaryskipentry
4402 \@glsxtr@printunsrtglossaryskipentry
4403 \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

4404 \glsxtr@process
4405 {%
4406   \ifglshasparent{\glscurrententrylabel}{}%
4407   {%
4408     \@glsxtr@checkgroup\glscurrententrylabel
4409     \expandafter\appto\expandafter\@glsxtr@doglossary\expandafter
4410       {\@glsxtr@groupheading}%
4411   }%
4412   \eappto\@glsxtr@doglossary{%
4413     \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4414   }%
4415 }%
4416 }%
4417 \appto\@glsxtr@doglossary{\end{theglossary}}%
4418 \printunsrtglossarypredoglossary
4419 \@glsxtr@doglossary
4420 }%
4421 \glossarypostamble
4422 }

```

entryprocesshook

```

4423 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{%

```

entryprocesshook

```

4424 \newcommand*{\printunsrtglossaryskipentry}{%
4425   \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space

```



```

4426 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4427 }

```

entryprocesshook

```

4428 \newcommand*{\@glstr@printunsrtglossaryskipentry}{%
4429   \let\glstr@process\@gobble
4430 }

```

rypredoglossary

```

4431 \newcommand*{\printunsrtglossarypredoglossary}{}

```

lossary@handler

```

4432 \newcommand{\@printunsrtglossary@handler}[1]{%
4433   \xdef\glscurrententrylabel{#1}%
4434   \printunsrtglossaryhandler\glscurrententrylabel
4435 }

```

glossaryhandler

```

4436 \newcommand{\printunsrtglossaryhandler}[1]{%
4437   \glstrunsrtdo{#1}%
4438 }

```

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.

```

4439 \newrobustcmd*{\glstriflabelinlist}[4]{%
4440   \protected@edef\@glstr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
4441   \@glstr@doiflabelinlist{#3}{#4}%
4442 }

```

srtglossaryunit

```

4443 \newcommand{\print@op@unsrtglossaryunit}[2][ ]{%
4444   \s@printunsrtglossary[type=\glstypedefaulttype,#1]{%
4445     \printunsrtglossaryunitsetup{#2}%
4446   }%
4447 }

```

ossaryunitsetup

```

4448 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4449   \renewcommand{\printunsrtglossaryhandler}[1]{%
4450     \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4451     {\glstrunsrtdo{##1}}%
4452   }%
4453 }%

```

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

```

4454 \ifcsundef{theH#1}%

```

```

4455 {%
4456   \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4457 }%
4458 {%
4459   \renewcommand*{\@glxtrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4460 }%
4461 \renewcommand*{\glossarysection}[2][{}]{%
4462   \appto\glossarypostamble{\glspare\medskip\glspare}%
4463 }

```

srtglossaryunit

```

4464 \newcommand{\printnoop@unsrtglossaryunit}[2][{}]{%
4465   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4466     requires the record=only or record=alsoindex package option}{}%
4467 }

```

t@getgrouptitle

```

4468 \newrobustcmd*{\@glxtr@unsrt@getgrouptitle}[2]{%
4469   \protected@edef\@glxtr@titlelabel{glxtr@grouptitle@#1}%
4470   \@onelevel@sanitize\@glxtr@titlelabel
4471   \ifcsdef{\@glxtr@titlelabel}
4472     {\letcs{#2}{\@glxtr@titlelabel}}%
4473     {\def#2{#1}}%
4474 }

```

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

```

4475 \newcommand{\glxtrunsrtdo}{\@glxtr@noidx@do}

```

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

```

4476 \newcommand*{\glxtrgroupfield}{group}

```

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

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

```

4477 \newcommand*{\@glxtr@checkgroup}[1]{%
4478   \def\@glxtr@groupheading{}%
4479   \key@ifundefined{glossentry}{group}%
4480   {%
4481     \letcs{\@gls@sort}{glo@\glstdetoklabel{#1}@sort}%
4482     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4483   }%
4484   {%

```

```

4485 \protected@edef\@glo@thislettergrp{%
4486 \csuse{glo@\glsdetoklabel{#1}@\glsxtrgroupfield}}%
4487 }%
4488 \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4489 {}%
4490 {%
4491 \ifdefempty{\@gls@currentlettergroup}{}%
4492 {\def\@glsxtr@groupheading{\glsgroupskip}}%
4493 \eappto\@glsxtr@groupheading{%
4494 \noexpand\glsgroupheading{\expandonce\@glo@thislettergrp}%
4495 }%
4496 }%
4497 \let\@gls@currentlettergroup\@glo@thislettergrp
4498 }

```

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

```

4499 \newcommand{\@glsxtr@noidx@do}[1]{%
4500 \ifglstryexists{#1}%
4501 {%
4502 \global\letcs{\@gls@loclist}{glo@\glsdetoklabel{#1}@loclist}%
4503 \global\letcs{\@gls@location}{glo@\glsdetoklabel{#1}@location}%
4504 \ifglshasparent{#1}%
4505 {%
4506 \gls@level=\csuse{glo@\glsdetoklabel{#1}@level}\relax
4507 \ifdefvoid{\@gls@location}%
4508 {%
4509 \ifdefvoid{\@gls@loclist}%
4510 {%
4511 \subglossentry{\gls@level}{#1}{}%
4512 }%
4513 {%
4514 \subglossentry{\gls@level}{#1}%
4515 {%
4516 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4517 }%
4518 }%
4519 }%
4520 {%
4521 \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4522 }%
4523 }%
4524 {%
4525 \ifdefvoid{\@gls@location}%
4526 {%
4527 \ifdefvoid{\@gls@loclist}%
4528 {%
4529 \glossentry{#1}{}%
4530 }%

```

```

4531      {%
4532      \glossentry{#1}%
4533      {%
4534      \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4535      }%
4536      }%
4537      }%
4538      {%
4539      \glossentry{#1}%
4540      {%
4541      \glossaryentrynumbers{\@gls@location}%
4542      }%
4543      }%
4544      }%
4545      }%
4546      {}%
4547      }

```

1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4548 \newcommand*{\glshex}{\string\u}
```

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

```
4549 \newcommand*{\glsxtrresourceinit}{}

```

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

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

`\glsxtrnewgls`

```
4550 \newcount\@glsxtrnewgls@inner

```

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

`\@glsxtrnewgls`

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

```

4551 \newcommand*{\@glsxtrnewgls}[4]{%
4552   \ifdef{#3}%
4553   {%
4554     \PackageError{glossaries-extra}{Command \string#3\space already
4555     defined}{}%

```

```

4556 }%
4557 {%
4558   \ifcsdef{@#4like@#2}%
4559   {%
4560     \advance\@glxstrnewgls@inner by \@ne
4561     \def\@glxstrnewgls@innercsname{@#4like\number\@glxstrnewgls@inner @#2}%
4562   }%
4563   {\def\@glxstrnewgls@innercsname{@#4like@#2}}%
4564   \expandafter\newrobustcmd\expandafter*\expandafter
4565   #3\expandafter{\expandafter\@glshyp@opt\csname\@glxstrnewgls@innercsname\endcsname}%
4566   \ifstrempy{#1}%
4567   {%
4568     \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2][]{%
4569       \new@ifnextchar[%
4570         {\csname @#4@\endcsname{##1}{#2##2}}%
4571         {\csname @#4@\endcsname{##1}{#2##2}[]}%
4572       }%
4573     }%
4574   {%
4575     \expandafter\newcommand\expandafter*\csname\@glxstrnewgls@innercsname\endcsname[2][]{%
4576       \new@ifnextchar[%
4577         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4578         {\csname @#4@\endcsname{#1,##1}{#2##2}[]}%
4579       }%
4580     }%
4581   }%
4582 }

```

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

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

```

4583 \newrobustcmd*{\glxstrnewgls}[3][]{%
4584   \@glxstrnewgls{#1}{#2}{#3}{gls}%
4585 }

```

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

```

4586 \newrobustcmd*{\glxstrnewglslike}[6][]{%
4587   \@glxstrnewgls{#1}{#2}{#3}{gls}%
4588   \@glxstrnewgls{#1}{#2}{#4}{glsp1}%
4589   \@glxstrnewgls{#1}{#2}{#5}{Gls}%
4590   \@glxstrnewgls{#1}{#2}{#6}{Glsp1}%
4591 }

```

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

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

```
4592 \newrobustcmd*{\glxtrnewGLSlike}[4] [] {%
4593   \@glxtrnewgls{#1}{#2}{#3}{GLS}%
4594   \@glxtrnewgls{#1}{#2}{#4}{GLSpl}%
4595 }
```

`\glxtrnewrgls` As `\glxtrnewgls` but for `\rgls`.

```
4596 \newrobustcmd*{\glxtrnewrgls}[3] [] {%
4597   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4598 }
```

`sxtrnewrglslike` As `\glxtrnewglslike` but for `\rgls` etc.

```
4599 \newrobustcmd*{\glxtrnewrglslike}[6] [] {%
4600   \@glxtrnewgls{#1}{#2}{#3}{rgls}%
4601   \@glxtrnewgls{#1}{#2}{#4}{rglspl}%
4602   \@glxtrnewgls{#1}{#2}{#5}{rGLs}%
4603   \@glxtrnewgls{#1}{#2}{#6}{rGLspl}%
4604 }
```

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

```
4605 \newrobustcmd*{\glxtrnewrGLSlike}[4] [] {%
4606   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4607   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4608 }
```

Provide easy access to record count fields.

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

```
4609 \newcommand*{\GlsXtrTotalRecordCount}[1] {%
4610   \ifcsdef{glo@glstdetoklabel{#1}@recordcount}%
4611   {\csname glo@glstdetoklabel{#1}@recordcount\endcsname}%
4612   {0}%
4613 }
```

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

```
4614 \newcommand*{\GlsXtrRecordCount}[2] {%
4615   \ifcsdef{glo@glstdetoklabel{#1}@recordcount.#2}%
4616   {\csname glo@glstdetoklabel{#1}@recordcount.#2\endcsname}%
4617   {0}%
4618 }
```

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

```
4619 \newcommand*{\GlsXtrLocationRecordCount}[3] {%
4620   \ifcsdef{glo@glstdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
```

```

4621 {\csname glo@glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
4622 {0}%
4623 }

```

trdetoklocation

```

4624 \newcommand*{\glxtrdetoklocation}[1]{#1}

```

ablerecordcount

```

4625 \newcommand*{\glxtrenablerecordcount}{%
4626   \renewcommand*{\glx}{\rglx}%
4627   \renewcommand*{\Glx}{\rGlx}%
4628   \renewcommand*{\glxpl}{\rglxpl}%
4629   \renewcommand*{\Glxpl}{\rGlxpl}%
4630   \renewcommand*{\GLS}{\rGLS}%
4631   \renewcommand*{\GLSpl}{\rGLSpl}%
4632 }

```

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

```

4633 \newcommand*{\glxtrrecordtriggervalue}[1]{%
4634   \GlxXtrTotalRecordCount{#1}%
4635 }

```

dCountAttribute

```

4636 \newcommand*{\GlxXtrSetRecordCountAttribute}[2]{%
4637   \@for\@glxtr@cat:=#1\do
4638   {%
4639     \ifdefempty{\@glxtr@cat}{}%
4640     {%
4641       \glxsetcategoryattribute{\@glxtr@cat}{recordcount}{#2}%
4642     }%
4643   }%
4644 }

```

rifrecordtrigger

$\backslash\text{glxtrifrecordtrigger}\langle\text{label}\rangle\langle\text{trigger format}\rangle\langle\text{normal}\rangle$

```

4645 \newcommand*{\glxtrifrecordtrigger}[3]{%
4646   \glxhasattribute{#1}{recordcount}%
4647   {%
4648     \ifnum\glxtrrecordtriggervalue{#1}>\glxgetattribute{#1}{recordcount}\relax
4649     #3%
4650   \else
4651     #2%
4652   \fi
4653   }%
4654   {#3}%
4655 }

```

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

```
4656 \newcommand*{\@glsxtr@rglstrigger@record}[3]{%
4657   \edef\glslabel{\glsdetoklabel{#2}}%
4658   \let\@gls@link@label\glslabel
4659   \def\@glsxtr@thevalue{%
4660     \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
4661     \def\@glsnumberformat{glstriggerrecordformat}%
4662     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4663     \edef\@gls@type{\csname glo@\glslabel @type\endcsname}%
4664     \def\@glsxtr@thevalue{%
4665       \def\@glsxtr@theHvalue{\@glsxtr@thevalue}%
4666       \glsxtrinitwrgloss
4667       \setkeys{glslink}{#1}%
4668       \glslinkpostsetkeys
4669       \ifdefempty{\@glsxtr@thevalue}%
4670       {%
4671         \@gls@saveentrycounter
4672       }%
4673       {%
4674         \let\theglsentrycounter\@glsxtr@thevalue
4675         \def\theglsentrycounter{\@glsxtr@theHvalue}%
4676       }%
4677       \ifglsxtrinitwrglossbefore
4678       \do@wrglossary{#2}%
4679       \fi
4680       #3%
4681       \ifglsxtrinitwrglossbefore
4682       \else
4683       \do@wrglossary{#2}%
4684       \fi
4685       \ifKV@glslink@local
4686       \glslocalunset{#2}%
4687       \else
4688       \glsunset{#2}%
4689       \fi
4690 }
```

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

```
4691 \newcommand*{\glstriggerrecordformat}[1]{}
```

\rgls

```
4692 \newrobustcmd*{\rgls}{\@gls@hyp@opt\@rgls}
```

\@rgls

```
4693 \newcommand*{\@rgls}[2][ ]{%
4694   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}[]}%
4695 }
```



```

\@rgls@
4696 \def\@rgls@#1#2[#3]{%
4697   \glxtrifrecordtrigger{#2}%
4698   {%
4699     \@glxtr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4700   }%
4701   {%
4702     \@gls@{#1}{#2}[#3]%
4703   }%
4704 }%

\rglspl
4705 \newrobustcmd*{\rglspl}{\@gls@hyp@opt\@rglspl}

\@rglspl
4706 \newcommand*{\@rglspl}[2][{}]{%
4707   \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
4708 }

\@rglspl@
4709 \def\@rglspl@#1#2[#3]{%
4710   \glxtrifrecordtrigger{#2}%
4711   {%
4712     \@glxtr@rglstrigger@record{#1}{#2}{\rglsplformat{#2}{#3}}%
4713   }%
4714   {%
4715     \@glspl@{#1}{#2}[#3]%
4716   }%
4717 }%

\@rGls
4718 \newrobustcmd*{\@rGls}{\@gls@hyp@opt\@rGls}

\@rGls
4719 \newcommand*{\@rGls}[2][{}]{%
4720   \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4721 }

\@rGls@
4722 \def\@rGls@#1#2[#3]{%
4723   \glxtrifrecordtrigger{#2}%
4724   {%
4725     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4726   }%
4727   {%
4728     \@Gls@{#1}{#2}[#3]%
4729   }%
4730 }%

```

```

\rGlspl
4731 \newrobustcmd*{\rGlspl}{\@gls@hyp@opt\rGlspl}

\@rGlspl
4732 \newcommand*{\@rGlspl}[2][\%
4733   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}]%
4734 }

\@rGlspl@
4735 \def\@rGlspl@#1#2[#3]{%
4736   \glsxtrifrecordtrigger{#2}%
4737   {%
4738     \@glsxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4739   }%
4740   {%
4741     \@Glspl@{#1}{#2}[#3]%
4742   }%
4743 }%

\rGLS
4744 \newrobustcmd*{\rGLS}{\@gls@hyp@opt\rGLS}

\@rGLS
4745 \newcommand*{\@rGLS}[2][\%
4746   \new@ifnextchar[{\@rGLS@{#1}{#2}}{\@rGLS@{#1}{#2}[]}]%
4747 }

\@rGLS@
4748 \def\@rGLS@#1#2[#3]{%
4749   \glsxtrifrecordtrigger{#2}%
4750   {%
4751     \@glsxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
4752   }%
4753   {%
4754     \@GLS@{#1}{#2}[#3]%
4755   }%
4756 }%

\rGLSpl
4757 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}

\@rGLSpl
4758 \newcommand*{\@rGLSpl}[2][\%
4759   \new@ifnextchar[{\@rGLSpl@{#1}{#2}}{\@rGLSpl@{#1}{#2}[]}]%
4760 }

```

```

\@rGLSpl@
4761 \def \@rGLSpl@#1#2[#3]{%
4762   \glxtrifrecordtrigger{#2}%
4763   {%
4764     \@glxtr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
4765   }%
4766   {%
4767     \@GLSpl@{#1}{#2}[#3]%
4768   }%
4769 }%

\rglsformat
4770 \newcommand*{\rglsformat}[2]{%
4771   \glsifregular{#1}
4772   {\glsentryfirst{#1}}%
4773   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4774 }

\rglsplformat
4775 \newcommand*{\rglsplformat}[2]{%
4776   \glsifregular{#1}
4777   {\glsentryfirstplural{#1}}%
4778   {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}}%#2%
4779 }

\rGlsformat
4780 \newcommand*{\rGlsformat}[2]{%
4781   \glsifregular{#1}
4782   {\Glsentryfirst{#1}}%
4783   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
4784 }

\rGlsplformat
4785 \newcommand*{\rGlsplformat}[2]{%
4786   \glsifregular{#1}
4787   {\Glsentryfirstplural{#1}}%
4788   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4789 }

\rGLSformat
4790 \newcommand*{\rGLSformat}[2]{%
4791   \expandafter\mfirstucMakeUppercase\expandafter{\rglsformat{#1}{#2}}%
4792 }

\rGLSplformat
4793 \newcommand*{\rGLSplformat}[2]{%
4794   \expandafter\mfirstucMakeUppercase\expandafter{\rglsplformat{#1}{#2}}%
4795 }

```

1.4 Integration with glossaries-accsupp

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

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

```
4796 \@ifpackageloaded{glossaries-accsupp}
4797 {
```

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

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

```
4798 \newcommand*{\glsaccessname}[1]{%
4799 \glsnameaccessdisplay
4800 {%
4801 \glstryname{#1}%
4802 }%
4803 {#1}%
4804 }
```

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

```
4805 \newcommand*{\Glsaccessname}[1]{%
4806 \glsnameaccessdisplay
4807 {%
4808 \Glsstryname{#1}%
4809 }%
4810 {#1}%
4811 }
```

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

```
4812 \newcommand*{\GLSaccessname}[1]{%
4813 \glsnameaccessdisplay
4814 {%
4815 \mfirstucMakeUppercase{\glstryname{#1}}%
4816 }%
4817 {#1}%
4818 }
```

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

```
4819 \newcommand*{\glsaccesstext}[1]{%
4820 \glstextaccessdisplay
4821 {%
4822 \glstrytext{#1}%
4823 }%
4824 {#1}%
4825 }
```

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

```

4826 \newcommand*{\Glsaccesstext}[1]{%
4827   \glstextaccessdisplay
4828   {%
4829     \Glsentrytext{#1}%
4830   }%
4831   {#1}%
4832 }
```

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

```

4833 \newcommand*{\GLSaccesstext}[1]{%
4834   \glstextaccessdisplay
4835   {%
4836     \mfirstucMakeUppercase{\glstextentry{#1}}%
4837   }%
4838   {#1}%
4839 }
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

4840 \newcommand*{\glsaccessplural}[1]{%
4841   \glspluralaccessdisplay
4842   {%
4843     \glsentryplural{#1}%
4844   }%
4845   {#1}%
4846 }
```

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

```

4847 \newcommand*{\GLSaccessplural}[1]{%
4848   \glspluralaccessdisplay
4849   {%
4850     \Glsentryplural{#1}%
4851   }%
4852   {#1}%
4853 }
```

`GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```

4854 \newcommand*{\GLSaccessplural}[1]{%
4855   \glspluralaccessdisplay
4856   {%
4857     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4858   }%
4859   {#1}%
4860 }
```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

4861 \newcommand*{\glsaccessfirst}[1]{%
4862   \glsfirstaccessdisplay
4863   {%
4864     \glstryfirst{#1}%
4865   }%
4866   {#1}%
4867 }

```

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

```

4868 \newcommand*{\Glsaccessfirst}[1]{%
4869   \glsfirstaccessdisplay
4870   {%
4871     \Glsstryfirst{#1}%
4872   }%
4873   {#1}%
4874 }

```

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

```

4875 \newcommand*{\GLSaccessfirst}[1]{%
4876   \glsfirstaccessdisplay
4877   {%
4878     \mfirstucMakeUppercase{\glstryfirst{#1}}%
4879   }%
4880   {#1}%
4881 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

4882 \newcommand*{\glsaccessfirstplural}[1]{%
4883   \glsfirstpluralaccessdisplay
4884   {%
4885     \glstryfirstplural{#1}%
4886   }%
4887   {#1}%
4888 }

```

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

```

4889 \newcommand*{\Glsaccessfirstplural}[1]{%
4890   \glsfirstpluralaccessdisplay
4891   {%
4892     \Glsstryfirstplural{#1}%
4893   }%
4894   {#1}%
4895 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) converted to upper case.

```

4896 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

4897 \glsfirstpluralaccessdisplay
4898 {%
4899 \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4900 }%
4901 {#1}%
4902 }

```

glsaccesssymbol Display the symbol value (no link and no check for existence).

```

4903 \newcommand*{\glsaccesssymbol}[1]{%
4904 \glssymbolaccessdisplay
4905 {%
4906 \glsentrysymbol{#1}%
4907 }%
4908 {#1}%
4909 }

```

Glsaccesssymbol Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4910 \newcommand*{\Glsaccesssymbol}[1]{%
4911 \glssymbolaccessdisplay
4912 {%
4913 \Glsentrysymbol{#1}%
4914 }%
4915 {#1}%
4916 }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```

4917 \newcommand*{\GLSaccesssymbol}[1]{%
4918 \glssymbolaccessdisplay
4919 {%
4920 \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
4921 }%
4922 {#1}%
4923 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence).

```

4924 \newcommand*{\glsaccesssymbolplural}[1]{%
4925 \glssymbolpluralaccessdisplay
4926 {%
4927 \glsentrysymbolplural{#1}%
4928 }%
4929 {#1}%
4930 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4931 \newcommand*{\Glsaccesssymbolplural}[1]{%
4932 \glssymbolpluralaccessdisplay

```

```

4933   {%
4934       \Glsentrysymbolplural{#1}%
4935   }%
4936   {#1}%
4937 }

```

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

```

4938 \newcommand*{\GLSaccesssymbolplural}[1]{%
4939     \glssymbolpluralaccessdisplay
4940     {%
4941         \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
4942     }%
4943     {#1}%
4944 }

```

`\glsaccessdesc` Display the desc value (no link and no check for existence).

```

4945 \newcommand*{\glsaccessdesc}[1]{%
4946     \glsdescriptionaccessdisplay
4947     {%
4948         \glentrydesc{#1}%
4949     }%
4950     {#1}%
4951 }

```

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

```

4952 \newcommand*{\Glsaccessdesc}[1]{%
4953     \glsdescriptionaccessdisplay
4954     {%
4955         \Glsentrydesc{#1}%
4956     }%
4957     {#1}%
4958 }

```

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

```

4959 \newcommand*{\GLSaccessdesc}[1]{%
4960     \glsdescriptionaccessdisplay
4961     {%
4962         \mfirstucMakeUppercase{\glentrydesc{#1}}%
4963     }%
4964     {#1}%
4965 }

```

`\accessdescplural` Display the descplural value (no link and no check for existence).

```

4966 \newcommand*{\glsaccessdescplural}[1]{%
4967     \glsdescriptionpluralaccessdisplay
4968     {%
4969         \glentrydescplural{#1}%

```



```

4970    }%
4971    {#1}%
4972  }

```

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

```

4973  \newcommand*{\glsaccessdescplural}[1]{%
4974    \glsdescriptionpluralaccessdisplay
4975    {%
4976      \Glsentrydescplural{#1}%
4977    }%
4978    {#1}%
4979  }

```

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

```

4980  \newcommand*{\GLSaccessdescplural}[1]{%
4981    \glsdescriptionpluralaccessdisplay
4982    {%
4983      \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4984    }%
4985    {#1}%
4986  }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

4987  \newcommand*{\glsaccessshort}[1]{%
4988    \glsshortaccessdisplay
4989    {%
4990      \Glsentryshort{#1}%
4991    }%
4992    {#1}%
4993  }

```

`\GLSaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

4994  \newcommand*{\GLSaccessshort}[1]{%
4995    \glsshortaccessdisplay
4996    {%
4997      \Glsentryshort{#1}%
4998    }%
4999    {#1}%
5000  }

```

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

```

5001  \newcommand*{\GLSAccessshort}[1]{%
5002    \glsshortaccessdisplay
5003    {%
5004      \mfirstucMakeUppercase{\glsentryshort{#1}}%
5005    }%

```

```

5006     {#1}%
5007 }

```

`\lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

5008 \newcommand*{\lsaccessshortpl}[1]{%
5009     \glshortpluralaccessdisplay
5010     {%
5011         \glentryshortpl{#1}%
5012     }%
5013     {#1}%
5014 }

```

`\Lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5015 \newcommand*{\Lsaccessshortpl}[1]{%
5016     \glshortpluralaccessdisplay
5017     {%
5018         \Glentryshortpl{#1}%
5019     }%
5020     {#1}%
5021 }

```

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

```

5022 \newcommand*{\LSaccessshortpl}[1]{%
5023     \glshortpluralaccessdisplay
5024     {%
5025         \mfirstucMakeUppercase{\glentryshortpl{#1}}%
5026     }%
5027     {#1}%
5028 }

```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

5029 \newcommand*{\glsaccesslong}[1]{%
5030     \glslongaccessdisplay{\glentrylong{#1}}{#1}%
5031 }

```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```

5032
5033 \newcommand*{\Glsaccesslong}[1]{%
5034     \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
5035 }

```

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

```

5036 \newcommand*{\GLSaccesslong}[1]{%
5037     \glslongaccessdisplay
5038     {%
5039         \mfirstucMakeUppercase{\glentrylong{#1}}%
5040     }%

```

```

5041     {#1}%
5042 }

```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5043 \newcommand*{\glsaccesslongpl}[1]{%
5044     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5045 }

```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5046
5047 \newcommand*{\Glsaccesslongpl}[1]{%
5048     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5049 }

```

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

```

5050 \newcommand*{\GLSaccesslongpl}[1]{%
5051     \glslongpluralaccessdisplay
5052     {%
5053         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5054     }%
5055     {#1}%
5056 }

```

End of if part

```

5057 }
5058 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

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

```

5059 \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.

```

5060 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

5061 \newcommand*{\GLSaccessname}[1]{%
5062     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

```

5063 \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.

```

5064 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

```

5065 \newcommand*{\GLSaccessstext}[1]{%
5066 \protect\mfirstucMakeUppercase{\glstentrytext{#1}}}
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

5067 \newcommand*{\glsaccessplural}[1]{\glstentryplural{#1}}
```

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

```

5068 \newcommand*{\Glsaccessplural}[1]{\Glstentryplural{#1}}
```

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.

```

5069 \newcommand*{\GLSaccessplural}[1]{%
5070 \protect\mfirstucMakeUppercase{\glstentryplural{#1}}}
```

`\glsaccessfirst` Display the first value (no link and no check for existence).

```

5071 \newcommand*{\glsaccessfirst}[1]{\glstentryfirst{#1}}
```

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

```

5072 \newcommand*{\Glsaccessfirst}[1]{\Glstentryfirst{#1}}
```

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

```

5073 \newcommand*{\GLSaccessfirst}[1]{%
5074 \protect\mfirstucMakeUppercase{\glstentryfirst{#1}}}
```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

5075 \newcommand*{\glsaccessfirstplural}[1]{\glstentryfirstplural{#1}}
```

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

```

5076 \newcommand*{\Glsaccessfirstplural}[1]{\Glstentryfirstplural{#1}}
```

`GLScessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.

```

5077 \newcommand*{\GLSaccessfirstplural}[1]{%
5078 \protect\mfirstucMakeUppercase{\glstentryfirstplural{#1}}}
```

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

```

5079 \newcommand*{\glsaccesssymbol}[1]{\glstentrysymbol{#1}}
```

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

```

5080 \newcommand*{\Glsaccesssymbol}[1]{\Glstentrysymbol{#1}}
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

```

5081 \newcommand*{\GLSaccesssymbol}[1]{%
5082 \protect\mfirstucMakeUppercase{\glstentrysymbol{#1}}}
```

esssymbolplural Display the symbolplural value (no link and no check for existence).
5083 \newcommand*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.
5084 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.
5085 \newcommand*{\GLSaccesssymbolplural}[1]{%
5086 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).
5087 \newcommand*{\glsaccessdesc}[1]{\glsentrydesc{#1}}

\Glsaccessdesc Display the desc value (no link and no check for existence) with the first letter converted to upper case.
5088 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.
5089 \newcommand*{\GLSaccessdesc}[1]{%
5090 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).
5091 \newcommand*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence) with the first letter converted to upper case.
5092 \newcommand*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.
5093 \newcommand*{\GLSaccessdescplural}[1]{%
5094 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).
5095 \newcommand*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).
5096 \newcommand*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.
5097 \newcommand*{\GLSaccessshort}[1]{%
5098 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).
5099 \newcommand*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}

`\lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5100 \newcommand*{\GLsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

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

```
5101 \newcommand*{\GLSaccessshortpl}[1]{%
5102 \protect\mfirstucMakeUppercase{\glsentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5103 \newcommand*{\glsaccesslong}[1]{\glsentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5104 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
5105 \newcommand*{\GLSaccesslong}[1]{%
5106 \protect\mfirstucMakeUppercase{\glsentrylong{#1}}}
```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5107 \newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}
```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5108 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
5109 \newcommand*{\GLSaccesslongpl}[1]{%
5110 \protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}
```

End of else part

```
5111 }
```

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5112 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5113 \newcommand{\glsifcategory}[4]{%
5114 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5115 }
```

Categories can have attributes.

categoryattribute

`\glsetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
5116 \newcommand*{\glsetcategoryattribute}[3]{%
5117   \csdef{@glxtr@categoryattr@@#1@#2}{#3}%
5118 }
```

categoryattribute

`\glgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5119 \newcommand*{\glgetcategoryattribute}[2]{%
5120   \cuse{@glxtr@categoryattr@@#1@#2}%
5121 }
```

categoryattribute

`\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5122 \newcommand*{\glshascategoryattribute}[4]{%
5123   \ifcvoid{@glxtr@categoryattr@@#1@#2}{#4}{#3}%
5124 }
```

\glsetattribute

`\glsetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5125 \newcommand*{\glsetattribute}[3]{%
5126   \glsetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5127 }
```

\glgetattribute

`\glgetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```
5128 \newcommand*{\glgetattribute}[2]{%
5129   \glgetcategoryattribute{\glscategory{#1}}{#2}%
5130 }
```

`\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.

```
5131 \newcommand*{\glshasattribute}[4]{%
5132   \ifglentryexists{#1}%
5133   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5134   {#4}%
5135 }
```

`categoryattribute`

`\glcifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```
5136 \newcommand{\glcifcategoryattribute}[5]{%
5137   \ifcsundef{@glxtr@categoryattr@#1@#2}%
5138   {#5}%
5139   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5140 }
```

`\glcifattribute`

`\glcifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

Short cut to determine if the given entry has a category with the given attribute set.

```
5141 \newcommand{\glcifattribute}[5]{%
5142   \ifglentryexists{#1}%
5143   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5144   {#5}%
5145 }
```

Set attributes for the default general category:

```
5146 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5147 \glsetcategoryattribute{acronym}{regular}{true}
```

`regularcategory`

Convenient shortcut to create add the regular attribute.

```
5148 \newcommand*{\glsetregularcategory}[1]{%
5149   \glsetcategoryattribute{#1}{regular}{true}%
5150 }
```


regularcategory

`\glsifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5151 \newcommand{\glsifregularcategory}[3]{%
5152   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
5153 }
```

notregularcategory

`\glsifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5154 \newcommand{\glsifnotregularcategory}[3]{%
5155   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
5156 }
```

\glsifregular

`\glsifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5157 \newcommand{\glsifregular}[3]{%
5158   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
5159 }
```

\glsifnotregular

`\glsifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5160 \newcommand{\glsifnotregular}[3]{%
5161   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
5162 }
```

foreachincategory

`\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.

```
5163 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```

```

5164 \forall glossaries[#1]{#3}%
5165 {%
5166     \forall sentries[#3]{#4}%
5167     {%
5168         \glsifcategory{#4}{#2}{#5}{}%
5169     }%
5170 }%
5171 }

```

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.

```

5172 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5173     \forall glossaries[#1]{#4}%
5174     {%
5175         \forall sentries[#4]{#5}%
5176         {%
5177             \glsifattribute{#5}{#2}{#3}{#6}{}%
5178         }%
5179     }%
5180 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```

5181 \ifdef\newterm
5182 {%

```

`\newterm`

```

5183     \renewcommand*{\newterm}[2][ ]{%
5184         \newglossaryentry{#2}%
5185         {type=index,category=index,name={#2},%
5186          description={\glstrpostdescription\nopostdesc},#1}%
5187     }

```

Indexed terms are regular by default.

```

5188     \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5189     \newcommand*{\glstrpostdescindex}{}

5190 }
5191 {}

```

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.

```
5192 \ifdef\printsymbols
5193 {%
```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```
5194 \newcommand*{\glsxtrnewsymbol}[3] [] {%
5195 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5196 }
```

Symbols are regular by default.

```
5197 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
5198 \newcommand*{\glsxtrpostdescsymbol}{}

5199 }
5200 {}
```

Similar for the numbers option.

```
5201 \ifdef\printnumbers
5202 {%
```

`glsxtrnewnumber`

```
5203 \ifdef\printnumbers
5204 \newcommand*{\glsxtrnewnumber}[3] [] {%
5205 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5206 }
```

Numbers are regular by default.

```
5207 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
5208 \newcommand*{\glsxtrpostdescnumber}{}

5209 }
5210 {}
```

`sxtrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
5211 \newcommand*{\glsxtrsetcategory}[2] {%
5212 \@for\@glsxtr@label:=#1\do
5213 {%
5214 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
5215 }%
5216 }
```

`\categoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```

5217 \newcommand*{\glxtrsetcategoryforall}[2]{%
5218   \forallglossaries[#1]{\@glxtr@type}{%
5219     \forallglsentries[\@glxtr@type]{\@glxtr@label}%
5220     {%
5221       \glsfieldxdef{\@glxtr@label}{category}{#2}%
5222     }%
5223   }%
5224 }

```

`\glxtrfieldtitlecase` `\glxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```

5225 \newcommand*{\glxtrfieldtitlecase}[2]{%
5226   \expandafter\glxtrfieldtitlecasesecs\expandafter
5227   {\csname glo\glsdetoklabel{#1}@#2\endcsname}%
5228 }

```

`\glxtrfieldtitlecasesecs` The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

5229 \newcommand*{\glxtrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}

```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

`\glossentrydesc` If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5230 \ifpackageloaded{glossaries-accsupp}
5231 {
5232   \renewcommand*{\glossentrydesc}[1]{%
5233     \glsdoifexistsorwarn{#1}%
5234     {%
5235       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

5236   \glshasattribute{#1}{glossdescfont}%
5237   {%
5238     \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5239     \ifcsdef{\@glxtr@attrval}%
5240     {%
5241       \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5242     }%
5243     {%
5244       \GlossariesExtraWarning{Unknown control sequence name
5245         '\@glxtr@attrval' supplied in glossdescfont attribute

```

```

5246         for entry '#1'. Ignoring}%
5247         \let\@glxtr@glossdescfont\@firstofone
5248     }%
5249 }%
5250 {\let\@glxtr@glossdescfont\@firstofone}%
5251 \glsifattribute{#1}{glossdesc}{firstuc}%
5252 {%
5253     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5254 }%
5255 {%
5256     \glsifattribute{#1}{glossdesc}{title}%
5257     {%
5258         \@glxtr@do@titlecaps@warn
5259         \glsdescriptionaccessdisplay
5260         {%
5261             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5262         }%
5263         {#1}%
5264     }%
5265     {%
5266         \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5267     }%
5268 }%
5269 }%
5270 }
5271 }
5272 {
5273 \renewcommand*{\glossentrydesc}[1]{%
5274     \glsdoifexistsorwarn{#1}%
5275     {%
5276         \glssetabbrvfmt{\glscategory{#1}}%
5277         \glsattribute{#1}{glossdescfont}%
5278         {%
5279             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5280             \ifcsdef{\@glxtr@attrval}%
5281             {%
5282                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5283             }%
5284             {%
5285                 \GlossariesExtraWarning{Unknown control sequence name
5286                     '\@glxtr@attrval' supplied in glossdescfont attribute
5287                     for entry '#1'. Ignoring}%
5288                 \let\@glxtr@glossdescfont\@firstofone
5289             }%
5290         }%
5291         {\let\@glxtr@glossdescfont\@firstofone}%
5292         \glsifattribute{#1}{glossdesc}{firstuc}%
5293         {%
5294             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```

```

5295     }%
5296     {%
5297         \glsifattribute{#1}{glossdesc}{title}%
5298         {%
5299             \@glsxtr@do@titlecaps@warn
5300             \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5301         }%
5302         {%
5303             \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
5304         }%
5305     }%
5306 }%
5307 }
5308 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5309 \ifpackageloaded{glossaries-accsupp}
5310 {
5311     \renewcommand*{\glossentryname}[1]{%
5312         \@glsdoifexistsorwarn{#1}%
5313         {%
5314             \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

5315     \glsifattribute{#1}{glossnamefont}%
5316     {%
5317         \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5318         \ifcsdef{\@glsxtr@attrval}%
5319         {%
5320             \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5321         }%
5322         {%
5323             \GlossariesExtraWarning{Unknown control sequence name
5324                 ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
5325                 for entry ‘#1’. Reverting to default \string\glsnamefont}%
5326             \let\@glsxtr@glossnamefont\glsnamefont
5327         }%
5328     }%
5329     {\let\@glsxtr@glossnamefont\glsnamefont}%
5330     \glsifattribute{#1}{glossname}{firstuc}%
5331     {%
5332         \glsnameaccessdisplay
5333         {%
5334             \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5335         }%
5336         {#1}%
5337     }%
5338     {%
5339         \glsifattribute{#1}{glossname}{title}%

```

```

5340      {%
5341      \@glstr@do@titlecaps@warn
5342      \glsnameaccessdisplay
5343      {%
5344      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
5345      }%
5346      {#1}%
5347      }%
5348      {%
5349      \glsifattribute{#1}{glossname}{uc}%
5350      {%
5351      \glsnameaccessdisplay
5352      {%

```

Hide the label from the upper-casing command.

```

5353      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5354      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5355      }%
5356      {#1}%
5357      }%
5358      {%
5359      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5360      \glsnameaccessdisplay
5361      {%
5362      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5363      }%
5364      {#1}%
5365      }%
5366      }%
5367      }%

```

Do post-name hook:

```

5368      \glstrpostnamehook{#1}%
5369      }%
5370  }
5371 }
5372 {
5373 \renewcommand*{\glossentryname}[1]{%
5374 \@glsdoifexistsorwarn{#1}%
5375 {%
5376 \glssetabbrvfmt{\glscategory{#1}}%
5377 \glsattribute{#1}{glossnamefont}%
5378 {%
5379 \edef\@glstr@attrval{\glsattribute{#1}{glossnamefont}}%
5380 \ifcsdef{\@glstr@attrval}%
5381 {%
5382 \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5383 }%
5384 {%
5385 \GlossariesExtraWarning{Unknown control sequence name

```

```

5386         ‘\@glstr@attrval’ supplied in glossnamefont attribute
5387         for entry ‘#1’. Reverting to default \string\glstrfont}%
5388         \let\@glstr@glossnamefont\glstrfont
5389     }%
5390 }%
5391 {\let\@glstr@glossnamefont\glstrfont}%
5392 \glstrattribute{#1}{glossname}{firstuc}%
5393 {%
5394     \@glstr@glossnamefont{\Glsentryname{#1}}%
5395 }%
5396 {%
5397     \glstrattribute{#1}{glossname}{title}%
5398     {%
5399         \@glstr@do@titlecaps@warn
5400         \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
5401     }%
5402     {%
5403         \glstrattribute{#1}{glossname}{uc}%
5404         {%

```

Hide the label from the upper-casing command.

```

5405         \letcs{\glo@name}{glo@glstrdetoklabel{#1}@name}%
5406         \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5407     }%
5408     {%

```

This little trick is used by glossaries to allow the user to redefine \glstrfont to use \makefirstuc. Support it even though they can now use the firstuc attribute.

```

5409         \letcs{\glo@name}{glo@glstrdetoklabel{#1}@name}%
5410         \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5411     }%
5412 }%
5413 }%

```

Do post-name hook.

```

5414     \glstrpostnamehook{#1}%
5415 }%
5416 }
5417 }

```

\Glsentryname Redefine to set the abbreviation format and accessibility support.

```

5418 \@ifpackageloaded{glossaries-accsupp}
5419 {
5420     \renewcommand*{\Glsentryname}[1]{%
5421         \@glstrdoifexistsorwarn{#1}%
5422         {%
5423             \glstrsetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5424         \glshasattribute{#1}{glossnamefont}%
5425         {%

```



```

5426     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5427     \ifcsdef{\@glsxtr@attrval}%
5428     {%
5429         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5430     }%
5431     {%
5432         \GlossariesExtraWarning{Unknown control sequence name
5433             '\@glsxtr@attrval' supplied in glossnamefont attribute
5434             for entry '#1'. Reverting to default \string\glsnamefont}%
5435         \let\@glsxtr@glossnamefont\glsnamefont
5436     }%
5437 }%
5438 {\let\@glsxtr@glossnamefont\glsnamefont}%
5439 \glsnameaccessdisplay
5440 {%
5441     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5442 }%
5443 {#1}%

```

Do post-name hook:

```

5444     \glsxtrpostnamehook{#1}%
5445 }%
5446 }
5447 }
5448 {
5449     \renewcommand*{\Glossentryname}[1]{%
5450         \@glsdoifexistsorwarn{#1}%
5451         {%
5452             \glssetabbrvfmt{\glscategory{#1}}%
5453             \glschasattribute{#1}{glossnamefont}%
5454             {%
5455                 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5456                 \ifcsdef{\@glsxtr@attrval}%
5457                 {%
5458                     \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5459                 }%
5460                 {%
5461                     \GlossariesExtraWarning{Unknown control sequence name
5462                         '\@glsxtr@attrval' supplied in glossnamefont attribute
5463                         for entry '#1'. Reverting to default \string\glsnamefont}%
5464                     \let\@glsxtr@glossnamefont\glsnamefont
5465                 }%
5466             }%
5467             {\let\@glsxtr@glossnamefont\glsnamefont}%
5468             \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5469     \glsxtrpostnamehook{#1}%
5470 }%
5471 }

```

5472 }

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`glstrpostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```
5473 \newcommand*{\glstrpostnamehook}[1]{%
5474   \let\@glsnumberformat\@glstr@defaultnumberformat
5475   \glstrdoautoindexname{#1}{indexname}%
```

Allow categories to hook in here.

```
5476   \csuse{glstrpostname\glscategory{#1}}%
5477 }
```

`setaccessdisplay`

```
5478 \@ifpackageloaded{glossaries-accsupp}
5479 {
5480   \newcommand*{\glstr@setaccessdisplay}[1]{%
5481     \ifcsdef{gls#1accessdisplay}%
5482       {\letcs\@glstr@accessdisplay{gls#1accessdisplay}}%
5483       {%
```

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.

```
5484     \edef\@gls@thisval{#1}%
5485     \@for\@gls@map:=\@gls@keymap\do{%
5486       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5487       \ifdefequal{\@this@key}{\@gls@thisval}%
5488       {%
5489         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5490         \@endfortrue
5491       }%
5492     }%
5493   }%
5494   \ifcsdef{gls\@gls@thisval accessdisplay}%
5495     {\letcs\@glstr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5496     {\let\@glstr@accessdisplay\@firstoftwo}%
5497   }%
5498 }
5499 }
5500 {%
5501   \newcommand*{\glstr@setaccessdisplay}[1]{%
5502     \let\@glstr@accessdisplay\@firstoftwo
5503 }
```

`glsentrynameother` Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

5504 \newrobustcmd*{\glossentrynameother}[2]{%
5505   \@glsdoifexistsorwarn{#1}%
5506   {%

```

Accessibility support:

```

5507   \glsxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5508   \glssetabbrvfmt{\glscategory{#1}}%
5509   \glsattribute{#1}{glossnamefont}%
5510   {%
5511     \edef\@glsxtr@attrval{\glsattribute{#1}{glossnamefont}}%
5512     \ifcsdef{\@glsxtr@attrval}%
5513     {%
5514       \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5515     }%
5516     {%
5517       \GlossariesExtraWarning{Unknown control sequence name
5518         '\@glsxtr@attrval' supplied in glossnamefont attribute
5519         for entry '#1'. Reverting to default \string\glsnamefont}%
5520       \let\@glsxtr@glossnamefont\glsnamefont
5521     }%
5522   }%
5523   {\let\@glsxtr@glossnamefont\glsnamefont}%
5524   \glsifattribute{#1}{glossname}{firstuc}%
5525   {%
5526     \@glsxtr@accessdisplay
5527     {\@glsxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5528     {#1}%
5529   }%
5530   {%
5531     \glsifattribute{#1}{glossname}{title}%
5532     {%
5533       \@glsxtr@do@titlecaps@warn
5534       \@glsxtr@accessdisplay
5535       {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
5536       {#1}%
5537     }%
5538     {%
5539       \glsifattribute{#1}{glossname}{uc}%
5540       {%
5541         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
5542         \@glsxtr@accessdisplay
5543         {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5544         {#1}%
5545       }%
5546       {%
5547         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
5548         \@glsxtr@accessdisplay
5549         {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%

```

```

5550         {#1}%
5551     }%
5552 }%
5553 }%

    Do post-name hook.
5554     \glxstrpostnamehook{#1}%
5555 }%
5556 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

5557 \newif\if@glxstr@format@override
5558 \@glxstr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5559 \@ifpackageloaded{hyperref}
5560 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
5561     \ifHy@hyperindex
5562         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5563             \@glxstr@format@override true
5564             \appto\theindex{\let\glshypernumber\@firstofone}%
5565         }
5566     \else
5567         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5568             \@glxstr@format@override true
5569             \appto\theindex{\let\glshypernumber\hyperpage}%
5570         }
5571     \fi
5572 }
5573 {
5574     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5575         \@glxstr@format@override true
5576     }
5577 }
5578 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

5579 \newcommand*{\glxstrdoautoindexname}[2]{%
5580     \glshasattribute{#1}{#2}%
5581     {%

```

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.

```

5582     \@glxstr@autoindex@setname{#1}%

```

If the attribute value is simply “true” don’t add an encap, otherwise use the value as the encap.

```

5583 \protected@edef\@glxtr@attrval{\glsggetattribute{#1}{#2}}%
5584 \if@glxtr@format@override

5585 \ifx\@glsnnumberformat\@glxtr@defaultnumberformat
5586 \else
5587 \let\@glxtr@attrval\@glsnnumberformat
5588 \fi
5589 \fi
5590 \ifdefstring{\@glxtr@attrval}{true}%
5591 {}%
5592 {\eappto\@glo@name{\@glxtr@autoindex@encap\@glxtr@attrval}}%
5593 \expandafter\glxtrautoindex\expandafter{\@glo@name}%
5594 }%
5595 {}%
5596 }

```

glxtrautoindex

```

5597 \newcommand*{\glxtrautoindex}{\index}

```

toindex@setname Assign \@glo@name for use with indexname attribute.

```

5598 \newcommand*{\@glxtr@autoindex@setname}[1]{%
5599 \protected@edef\@glo@name{\glxtrautoindexentry{#1}}%
5600 \glxtrautoindexassignsort{\@glo@sort}{#1}%
5601 \@gls@checkmkidxchars\@glo@sort
5602 \@glxtr@autoindex@doextra@esc\@glo@sort
5603 \epreto\@glo@name{\@glo@sort\@glxtr@autoindex@at}%
5604 }

```

rautoindexentry Command used for the actual part when auto-indexing.

```

5605 \newcommand*{\glxtrautoindexentry}[1]{\string\glstryname{#1}}

```

trautoindexsort Used to assign the sort value when auto-indexing.

```

5606 \newcommand*{\glxtrautoindexassignsort}[2]{%
5607 \glsletentryfield{#1}{#2}{sort}%
5608 }

```

dex@doextra@esc

```

5609 \newcommand*{\@glxtr@autoindex@doextra@esc}[1]{%
    Escape the escape character unless it has already been escaped.
5610 \ifx\@glxtr@autoindex@esc\@gls@quotechar
5611 \else
5612 \def\@gls@checkedmkidx{}%
5613 \edef\@@glxtr@checkspch{%
5614 \noexpand\@glxtr@autoindex@escquote\expandonce{#1}%
5615 \noexpand\@empty\@glxtr@autoindex@esc\noexpand\@nnil
5616 \@glxtr@autoindex@esc\noexpand\@empty\noexpand\@glxtr@endescpch}%
5617 \@@glxtr@checkspch

```

```

5618 \let#1\@gls@checkedmkidx\relax
5619 \fi

```

Escape actual character unless it has already been escaped.

```

5620 \ifx\@glsxtr@autoindex@at\@gls@actualchar
5621 \else
5622 \def\@gls@checkedmkidx{}%
5623 \edef\@glsxtr@checkspch{%
5624 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
5625 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
5626 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5627 @@glsxtr@checkspch
5628 \let#1\@gls@checkedmkidx\relax
5629 \fi

```

Escape level character unless it has already been escaped.

```

5630 \ifx\@glsxtr@autoindex@level\@gls@levelchar
5631 \else
5632 \def\@gls@checkedmkidx{}%
5633 \edef\@glsxtr@checkspch{%
5634 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
5635 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
5636 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5637 @@glsxtr@checkspch
5638 \let#1\@gls@checkedmkidx\relax
5639 \fi

```

Escape encap character unless it has already been escaped.

```

5640 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
5641 \else
5642 \def\@gls@checkedmkidx{}%
5643 \edef\@glsxtr@checkspch{%
5644 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
5645 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
5646 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5647 @@glsxtr@checkspch
5648 \let#1\@gls@checkedmkidx\relax
5649 \fi
5650 }

```

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.

```

5651 \newcommand*{\@glsxtr@autoindex@at}{ }

```

trSetActualChar Set the actual character.

```

5652 \newcommand*{\GlsXtrSetActualChar}[1]{%
5653 \gdef\@glsxtr@autoindex@at{#1}%
5654 \def\@glsxtr@autoindex@escat##1##2##3\@glsxtr@endescspch{%

```

```

5655 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escat}{##1}{##2}{##3}%
5656 }%
5657 }
5658 \onlypreamble\GlsXtrSetActualChar
5659 \makeatother
5660 \GlsXtrSetActualChar{@}
5661 \makeatletter

autoindex@encap Encap character for use with \index.
5662 \newcommand*{\@glsxtr@autoindex@encap}{}}

XtrSetEncapChar Set the encap character.
5663 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5664 \gdef\@glsxtr@autoindex@encap{#1}%
5665 \def\@glsxtr@autoindex@escencap##1#1##2#1##3\@glsxtr@endescspch{%
5666 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escencap}{##1}{##2}{##3}%
5667 }%
5668 }
5669 \GlsXtrSetEncapChar{}}
5670 \onlypreamble\GlsXtrSetEncapChar

autoindex@level Level character for use with \index.
5671 \newcommand*{\@glsxtr@autoindex@level}{}}

XtrSetLevelChar Set the encap character.
5672 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5673 \gdef\@glsxtr@autoindex@level{#1}%
5674 \def\@glsxtr@autoindex@esclevel##1#1##2#1##3\@glsxtr@endescspch{%
5675 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@esclevel}{##1}{##2}{##3}%
5676 }%
5677 }
5678 \GlsXtrSetLevelChar{!}
5679 \onlypreamble\GlsXtrSetLevelChar

r@autoindex@esc Escape character for use with \index.
5680 \newcommand*{\@glsxtr@autoindex@esc}{"}}

lsXtrSetEscChar Set the escape character.
5681 \newcommand*{\GlsXtrSetEscChar}[1]{%
5682 \gdef\@glsxtr@autoindex@esc{#1}%
5683 \def\@glsxtr@autoindex@escquote##1#1##2#1##3\@glsxtr@endescspch{%
5684 \@@glsxtr@autoindex@escspch{#1}{\@glsxtr@autoindex@escquote}{##1}{##2}{##3}%
5685 }%
5686 }
5687 \GlsXtrSetEscChar{"}
5688 \onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
5689 \ifdef\actualchar
5690 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5691 {}
```

Quote character \quotechar:

```
5692 \ifdef\quotechar
5693 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5694 {}
```

Level character \levelchar:

```
5695 \ifdef\levelchar
5696 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5697 {}
```

Encap character \encapchar:

```
5698 \ifdef\encapchar
5699 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5700 {}
```

leto@endescspch

```
5701 \def\@glxstr@gobbleto@endescspch#1\@glxstr@endescspch{}
```

toindex@esc@spch

```
\@glxstr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}
```

```
5702 \newcommand*{\@glxstr@autoindex@escspch}[5]{%
5703   \@glstmpb=\expandafter{\@glsc@checkedmkidx}%
5704   \toks@={#3}%
5705   \ifx\@nnil#3\relax
5706     \def\@glxstr@checkspch{\@glxstr@gobbleto@endescspch#5\@glxstr@endescspch}%
5707   \else
5708     \ifx\@nnil#4\relax
5709       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@}%
5710       \def\@glxstr@checkspch{\@glxstr@gobbleto@endescspch
5711         #4#5\@glxstr@endescspch}%
5712     \else
5713       \edef\@glsc@checkedmkidx{\the\@glstmpb\the\toks@
5714         \@glxstr@autoindex@esc#1}%
5715       \def\@glxstr@checkspch{#2#5#1\@nnil#1\@glxstr@endescspch}%
5716     \fi
5717   \fi
5718   \@glxstr@checkspch
5719 }
```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```
5720 \renewcommand*{\Glossentrydesc}[1]{%
5721   \glsoifexistsorwarn{#1}%
```



```

5722  {%
5723    \glssetabbrvfmt{\glscategory{#1}}%
5724    \Glsaccessdesc{#1}%
5725  }%
5726 }

```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

5727 \renewcommand*{\glossentrysymbol}[1]{%
5728   \glsdoifexistsorwarn{#1}%
5729   {%
5730     \glssetabbrvfmt{\glscategory{#1}}%
5731     \Glsaccesssymbol{#1}%
5732   }%
5733 }

```

`\Glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```

5734 \renewcommand*{\Glossentrysymbol}[1]{%
5735   \glsdoifexistsorwarn{#1}%
5736   {%
5737     \glssetabbrvfmt{\glscategory{#1}}%
5738     \Glsaccesssymbol{#1}%
5739   }%
5740 }

```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\enableInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```

5741 \newcommand*{\GlsXtrEnableInitialTagging}{%
5742   \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
5743 }
5744 \@onlypreamble\GlsXtrEnableInitialTagging

```

`\r@enabletagging` Starred version undefines command.

```

5745 \newcommand*{\s@glxtr@enabletagging}[2]{%
5746   \undef#2%
5747   \@glxtr@enabletagging{#1}{#2}%
5748 }

```

`\r@enabletagging` Internal command.

```

5749 \newcommand*{\@glxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
5750   \@for\@glxtr@cat:=#1\do
5751   {%
5752     \ifdefempty\@glxtr@cat
5753     {}%
5754     {\glssetcategoryattribute{\@glxtr@cat}{tagging}{true}}%

```

```

5755 }%
5756 \newrobustcmd*#2[1]{##1}%
5757 \def\@glxstr@taggingcs{#2}%
5758 \renewcommand*\@glxstr@activate@initialtagging{%
5759   \let#2\@glxstr@tag
5760 }%
5761 \ifundef\@gls@preglossaryhook
5762 {\GlossariesExtraWarning{Initial tagging requires at least
5763   glossaries.sty v4.19 to work correctly}}%
5764 {}%
5765 }

```

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

```

5766 \ifundef\mfu@checkword@do
5767 {
5768   \newcommand*\mfu@checkword@do[1]{%
5769     \ifdefstring{\mfu@checkword@arg}{#1}%
5770     {%
5771       \let\@mfu@domakefirstuc\@firstofone
5772       \listbreak
5773     }%
5774   }%
5775 }

```

\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.

```

5776 \ifundef\mfu@checkword
5777 {
5778   \newcommand{\@glxstr@do@titlecaps@warn}{%
5779     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5780       support not available}%

```

One warning should suffice.

```

5781     \let\@glxstr@do@titlecaps@warn\relax
5782   }
5783 }
5784 {
5785   \renewcommand*\mfu@checkword[1]{%
5786     \def\mfu@checkword@arg{#1}%
5787     \let\@mfu@domakefirstuc\makefirstuc
5788     \forlistloop\mfu@checkword@do\@mfu@nocaplist
5789   }
5790 }
5791 }
5792 {}% no patch required

```

@titlecaps@warn Do warning if title case not supported.

5793 \newcommand*{\@glstr@do@titlecaps@warn}{}

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.

5794 \newcommand*\@glstr@activate@initialtagging{}

\@glstr@tag Definition of tagging command when used in glossary.

5795 \newrobustcmd*{\@glstr@tag}[1]{%

5796 \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%

5797 {\glstrtagfont{#1}}{#1}%

5798 }

\glstrtagfont Used in the glossary.

5799 \newcommand*\glstrtagfont[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.

5800 \ifdef\@glspreglossaryhook

5801 {

5802 \renewcommand*\@glspreglossaryhook{%

5803 \@glstr@activate@initialtagging

Since the glossaries are automatically scoped, \@glstr@org@postdescription shouldn't already be defined, but check anyway just as a precautionary measure.

5804 \ifundef\@glstr@org@postdescription

5805 {%

5806 \let\@glstr@org@postdescription\glspostdescription

5807 \renewcommand*\glspostdescription{%

5808 \ifglseentryexists{\glscurrententrylabel}%

5809 {%

5810 \glstrpostdescription

5811 \@glstr@org@postdescription

5812 }%

5813 {}}%

5814 }%

5815 }%

5816 {}%

Enable the options used by \@glstrp:

5817 \glossxtrsetpopts

5818 }%

5819 }

5820 {}

postdescription This command will only be used if \@glspreglossaryhook 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.

```

5821 \newcommand*{\glxtrpostdescription}{%
5822   \csuse{glxtrpostdesc\glscategory{\glscurrententrylabel}}%
5823 }

postdescgeneral
5824 \newcommand*{\glxtrpostdescgeneral}{}

xtrpostdescterm
5825 \newcommand*{\glxtrpostdescterm}{}

postdescacronym
5826 \newcommand*{\glxtrpostdescacronym}{}

descabbreviation
5827 \newcommand*{\glxtrpostdescabbreviation}{}

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.
5828 \renewcommand*{\glspostlinkhook}{%
5829   \ifglentryexists{\glslabel}{\glxtrpostlinkhook}{}%
5830 }

xtrpostlinkhook  The entry label should already be stored in \glslabel by \@gls@link.
5831 \newcommand*{\glxtrpostlinkhook}{%
5832   \glxtrdiscardperiod{\glslabel}%
5833   {\glxtrpostlinkendsentence}%
5834   {\glxtrifcustomdiscardperiod
5835     {\glxtrifperiod{\glxtrpostlinkendsentence}{\glxtrpostlink}}%
5836     {\glxtrpostlink}%
5837   }%
5838 }

omdiscardperiod  Allow user to provide a custom check. Should expand to #2 if no check is required otherwise
                  expand to #1.
5839 \newcommand*{\glxtrifcustomdiscardperiod}[2]{#2}

\glxtrpostlink
5840 \newcommand*{\glxtrpostlink}{%
5841   \csuse{glxtrpostlink\glscategory{\glslabel}}%
5842 }

linkendsentence  Done by \glxtrpostlinkhook if a full stop is discarded.
5843 \newcommand*{\glxtrpostlinkendsentence}{%
5844   \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
5845   {%
5846     \csuse{glxtrpostlink\glscategory{\glslabel}}%

```

Put the full stop back.

```
5847 .\spacefactor\sffcode'\. \relax
5848 }%
5849 {%
```

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

```
5850 \spacefactor\sffcode'\. \relax
5851 }%
5852 }
```

DescOnFirstUse Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
5853 \newcommand*\glstrpostlinkAddDescOnFirstUse{%
5854 \glstrifwasfirstuse{\space(\glssaccessdesc{\glslabel})}{}%
5855 }
```

SymbolOnFirstUse Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```
5856 \newcommand*\glstrpostlinkAddSymbolOnFirstUse{%
5857 \glstrifwasfirstuse
5858 {%
5859 \ifglshassymbol{\glslabel}{\space(\glssaccesssymbol{\glslabel})}{}%
5860 }%
5861 {}%
5862 }
```

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).

```
5863 \newcommand*\glstrdiscardperiod}[3]{%
5864 \glstrifwasfirstuse
5865 {%
5866 \glssifattribute{#1}{retainfirstuseperiod}{true}%
5867 {#3}%
5868 {%
5869 \glssifattribute{#1}{discardperiod}{true}%
5870 {%
5871 \glssifplural
5872 {%
5873 \glssifattribute{#1}{pluraldiscardperiod}{true}%
5874 {\glstrifperiod{#2}{#3}}%
5875 {#3}%
5876 }%
5877 {%
5878 \glstrifperiod{#2}{#3}%
5879 }%
}
```

```

5880     }%
5881     {#3}%
5882 }%
5883 }%
5884 {%
5885   \glsifattribute{#1}{discardperiod}{true}%
5886   {%
5887     \glsifplural
5888     {%
5889       \glsifattribute{#1}{pluraldiscardperiod}{true}%
5890       {\glsxtrifperiod{#2}{#3}}%
5891       {#3}%
5892     }%
5893     {%
5894       \glsxtrifperiod{#2}{#3}%
5895     }%
5896   }%
5897   {#3}%
5898 }%
5899 }

```

`\glsxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

5900 \newcommand*\glsxtrifperiod[1]{\new@ifnextchar.\@firstoftwo{#1}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glsxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```

5901 \newcommand*\glsxtr@punclist{.,:;?!}

```

`\punctuationmark` Add character to punctuation list.

```

5902 \newcommand*\glsxtraddpunctuationmark[1]{\appto\glsxtr@punclist{#1}}

```

`\punctuationmarks` Reset the punctuation list.

```

5903 \newcommand*\glsxtrsetpunctuationmarks[1]{\def\glsxtr@punclist{#1}}

```

`\glsxtrifpunc` `\glsxtrifnextpunc{<true part>}{<false part>}`

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```

5904 \newcommand*\glsxtrifnextpunc[2]{%
5905   \def\reserved@a{#1}%
5906   \def\reserved@b{#2}%
5907   \futurelet\@glspunc@token\glsxtr@ifnextpunc
5908 }

```

sxtr@ifnextpunc

```
5909 \newcommand*{\glsxtr@ifnextpunc}{%
5910 \glsxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}}%
5911 \reserved@b
5912 }
```

xtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
5913 \newcommand*{\glsxtr@ifpunctoken}[1]{%
5914 \expandafter\@glsxtr@ifpunctoken\expandafter#1\glsxtr@punclist\@nnil
5915 }
```

xtr@ifpunctoken

```
5916 \def\@glsxtr@ifpunctoken#1#2{%
5917 \let\reserved@d=#2%
5918 \ifx\reserved@d\@nnil
5919 \let\glsxtr@next\@glsxtr@notfoundinlist
5920 \else
5921 \ifx#1\reserved@d
5922 \let\glsxtr@next\@glsxtr@foundinlist
5923 \else
5924 \let\glsxtr@next\@glsxtr@ifpunctoken
5925 \fi
5926 \fi
5927 \glsxtr@next#1%
5928 }
```

xtr@foundinlist

```
5929 \def\@glsxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5930 \def\@glsxtr@notfoundinlist#1{\@secondoftwo}
```

glsxtrdopostpunc

```
\glsxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do <code> after the character otherwise do <code> before whatever comes next.

```
5931 \newcommand{\glsxtrdopostpunc}[1]{%
5932 \glsxtr@ifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
5933 }
```

@glsxtr@swaptwo

```
5934 \newcommand{\@glsxtr@swaptwo}[2]{#2#1}
```

1.6 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```
5935 \define@key{glstrabbrv}{category}{%
5936   \edef\glscategorylabel{#1}%
5937   \ifcsdef{@glabbrv@current@#1}%
5938   {%
```

Warning should already have been issued.

```
5939   \let\@glstr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5940   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5941   \glstr@applyabbrvstyle{\csname @glabbrv@current@#1\endcsname}%
5942   \let\GlsXtrWarnDeprecatedAbbrStyle\@glstr@orgwarndep
5943   }%
5944   }%
5945 }
```

Save the short plural form. This may be needed before the entry is defined.

```
5946 \define@key{glstrabbrv}{shortplural}{%
5947   \def\@gls@shortpl{#1}%
5948 }
```

Similarly for the long plural form.

```
5949 \define@key{glstrabbrv}{longplural}{%
5950   \def\@gls@longpl{#1}%
5951 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

`\glsshortpltok`

```
5952 \newtoks\glsshortpltok
```

`\glslongpltok`

```
5953 \newtoks\glslongpltok
```

`\glstr@insertdots` Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to `\newabbreviation`. Note that explicitly using the short or shortplural keys will override this.

```
5954 \newcommand*{\@glstr@insertdots}[2]{%
5955   \def#1{%
5956     \@glstr@insert@dots#1#2\@nnil
5957   }
```


xtr@insert@dots

```
5958 \newcommand*{\@glsxtr@insert@dots}[2]{%
5959   \ifx\@nnil#2\relax
5960   \let\@glsxtr@insert@dots@next\@gobble
5961   \else
5962   \ifx\relax#2\relax
5963   \else
5964     \appto#1{#2.}%
5965   \fi
5966   \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
5967   \fi
5968   \@glsxtr@insert@dots@next#1%
5969 }
```

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be defined:

\glsxtrwordsep

```
5970 \newcommand*{\glsxtrwordsep}{\space}
```

Each word is marked with

\glsxtrword

```
5971 \newcommand*{\glsxtrword}[1]{#1}
```

tr@markwordseps

```
5972 \newcommand*{\@glsxtr@markwordseps}[2]{%
5973   \def#1{}%
5974   \@glsxtr@mark@wordseps#1#2 \@nnil
5975 }
```

r@mark@wordseps

```
5976 \def\@glsxtr@mark@wordseps#1#2 #3{%
5977   \ifdefempty{#1}%
5978   {\def#1{\protect\glsxtrword{#2}}}%
5979   {\appto#1{\protect\glsxtrwordsep\protect\glsxtrword{#2}}}%
5980   \ifx\@nnil#3\relax
5981   \let\@glsxtr@mark@wordseps@next\relax
5982   \else
5983     \def\@glsxtr@mark@wordseps@next{%
5984       \@glsxtr@mark@wordseps#1#3}%
5985   \fi
5986   \@glsxtr@mark@wordseps@next
5987 }
```

newabbreviation Define a new generic abbreviation.

```
5988 \newcommand*{\newabbreviation}[4][ ]{%
5989   \glsxtr@newabbreviation{#1}{#2}{#3}{#4}%
5990 }
```

`\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.)

```
5991 \newcommand*{\glxtr@newabbreviation}[4]{%
5992   \glskeylisttok{#1}%
5993   \glslabeltok{#2}%
5994   \glsshorttok{#3}%
5995   \glslongtok{#4}%
```

Save the original short and long values (before attribute settings modify them).

```
5996   \def\glxtrorgshort{#3}%
5997   \def\glxtrorglong{#4}%
```

Get the category.

```
5998   \def\glscategorylabel{abbreviation}%
5999   \glxtr@applyabbrvstyle{\@glsabbrv@current@abbreviation}%
```

Ignore the shortplural and longplural keys.

```
6000   \setkeys*{\glxtrabbrv}[shortplural,longplural]{#1}%
```

Set the default long plural

```
6001   \def\@gls@longpl{#4\glspluralsuffix}%
6002   \let\@gls@default@longpl\@gls@longpl
```

Has the markwords attribute been set?

```
6003   \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6004   {%
6005     \@glxtr@markwordseps\@gls@long{#4}%
6006     \expandafter\def\expandafter\@gls@longpl\expandafter
6007       {\@gls@long\glspluralsuffix}%
6008     \let\@gls@default@longpl\@gls@longpl
```

Update `\glslongtok`.

```
6009     \expandafter\glslongtok\expandafter{\@gls@long}%
6010   }%
6011   {}%
```

Has the markshortwords attribute been set? (Not compatible with `insertdots`.)

```
6012   \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%
6013   {%
6014     \@glxtr@markwordseps\@gls@short{#3}%
6015   }%
6016   {}%
```

Has the `insertdots` attribute been set?

```
6017   \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%
6018   {%
6019     \@glxtr@insertdots\@gls@short{#3}%
6020     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6021   }%
6022   {\def\@gls@short{#3}}%
6023   }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
6024 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
6025 {%
6026   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6027     '\abbrvpluralsuffix}%
6028 }%
6029 {%
```

Has the noshortplural attribute been set?

```
6030 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
6031 {%
6032   \let\@gls@shortpl\@gls@short
6033 }%
6034 {%
6035   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6036     \abbrvpluralsuffix}%
6037 }%
6038 }%
```

Update \glsshorttok:

```
6039 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
6040 \glsxtrnewabbrevpresetkeyhook{#1}{#2}{#3}%
```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary. Ignore the category key (already obtained).

```
6041 \setkeys*{glxtrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
6042 \ifx\@gls@default@longpl\@gls@longpl
6043 \else
```

Has the markwords attribute been set?

```
6044 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6045 {%
6046   \expandafter\@glxtr@keywordseps\expandafter\@gls@longpl\expandafter
6047     {\@gls@longpl}%
6048 }%
6049 {}%
6050 \fi
```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```
6051 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6052 \expandafter\glslongpltok\expandafter{\@gls@longpl}%
```

Do any extra setup provided by hook:

```
6053 \newabbreviationhook
```

Define this entry:

```
6054 \protected@edef\@do@newglossaryentry{%
6055   \noexpand\newglossaryentry{\the\glslabeltok}%
6056   {%
```

```

6057     type=\glstrabbrvtype,%
6058     category=abbreviation,%
6059     short={\the\glsshorttok},%
6060     shortplural={\the\glsshortpltok},%
6061     long={\the\glslongtok},%
6062     longplural={\the\glslongpltok},%
6063     name={\the\glsshorttok},%
6064     \CustomAbbreviationFields,%
6065     \the\glskeylisttok
6066   }%
6067 }%
6068 \@do@newglossaryentry
6069 \GlsXtrPostNewAbbreviation
6070 }

```

`\evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```
6071 \newcommand*{\glstrnewabbrevpresetkeyhook}[3]{}

```

`\NewAbbreviation` Hook used by abbreviation styles.

```
6072 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`\bbreviationhook` Hook for use with `\newabbreviation`.

```
6073 \newcommand*{\newabbreviationhook}{}

```

`\reivationFields`

```
6074 \newcommand*{\CustomAbbreviationFields}{}

```

`\glstrparen` For the parenthetical styles.

```
6075 \newcommand*{\glstrparen}[1]{(#1)}

```

`\lsxtrfullformat` Full format without case change.

```

6076 \newcommand*{\glxtrfullformat}[2]{%
6077   \glsfirstlongfont{\glsaccesslong{#1}}#2\glxtrfullsep{#1}%
6078   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6079 }

```

`\lsxtrfullformat` Full format with case change.

```

6080 \newcommand*{\Glsxtrfullformat}[2]{%
6081   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glxtrfullsep{#1}%
6082   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6083 }

```

`\xtrfullplformat` Plural full format without case change.

```

6084 \newcommand*{\glxtrfullplformat}[2]{%
6085   \glsfirstlongfont{\glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6086   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{#1}}}%
6087 }

```

`\glstrfullplformat` Plural full format with case change.

```
6088 \newcommand*{\Glsxtrfullplformat}[2]{%
6089   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glstrfullsep{#1}%
6090   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6091 }
```

`\glstrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
6092 \newcommand*{\glstrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\glentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glstrinlinefullformat` Full format without case change.

```
6093 \newcommand*{\Glsxtrinlinefullformat}{\glstrfullformat}
```

`\glstrinlinefullformat` Full format with case change.

```
6094 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glstrfullplformat` Plural full format without case change.

```
6095 \newcommand*{\Glsxtrinlinefullplformat}{\glstrfullplformat}
```

`\glstrinlinefullplformat` Plural full format with case change.

```
6096 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

Redefine `\glentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glstrfull` set of commands instead.

`\glentryfull`

```
6097 \renewcommand*{\glentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
6098 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\glentryfullpl`

```
6099 \renewcommand*{\glentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
6100 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
6101 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
6102 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
6103 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

abbrvdefaultfont

```
6104 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
6105 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

longdefaultfont Default font changing command used for the long form in commands like `\glsxtrlong`.

```
6106 \newcommand*{\glslongdefaultfont}[1]{#1}
```

lsfirstlongfont Font changing command used for the long form on first use or in the full format.

```
6107 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

longdefaultfont

```
6108 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

brvpluralsuffix Default plural suffix. Allow an alternative default suffix for abbreviations.

```
6109 \newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}
```

brvpluralsuffix Default plural suffix.

```
6110 \newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
6111 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\@ns@glsxtrfull}
```

```
6112 \newcommand*{\ns@glsxtrfull}[2][\%
```

```
6113 \new@ifnextchar[\@glsxtr@full{#1}{#2}}%
```

```
6114 \@glsxtr@full{#1}{#2}[]}%
```

```
6115 }
```

`\@glsxtr@full` Low-level macro:

```
6116 \def\@glsxtr@full#1#2[#3]{%
```

```
6117 \glsdoifexists{#2}%
```

```
6118 {%
```

```
6119 \glssetabbrvfmt{\glscategory{#2}}%
```

```
6120 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
```

```
6121 \let\glsifplural\@secondoftwo
```

```
6122 \let\glsupcase\@firstofthree
```

```
6123 \let\glsinsert\@empty
```

```
6124 \def\glscustomtext{\glsxtrinlinefullformat{#2}{#3}}%
```

What should `\glsxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
6125 \glsxtrsetupfulldefs
```

```
6126 \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
```

```
6127 }%
```

```
6128 \glspostlinkhook
```

```
6129 }
```

trsetupfulldefs

```
6130 \newcommand*{\glxtrsetupfulldefs}{%
6131   \let\glxtrifwasfirstuse\@firstoftwo
6132 }
```

\Glsxtrfull Full form (first letter uppercase).

```
6133 \newrobustcmd*{\Glsxtrfull}{\@gls@hyp@opt\@ns@Glsxtrfull}
6134 \newcommand*\ns@Glsxtrfull[2][{}]{%
6135   \new@ifnextchar[{\@Glsxtr@full{#1}{#2}}{%
6136     {\@Glsxtr@full{#1}{#2}[]}%
6137 }
```

\@Glsxtr@full Low-level macro:

```
6138 \def\@Glsxtr@full#1#2[#3]{%
6139   \glsdoifexists{#2}%
6140   {%
6141     \glssetabbrvfmt{\glscategory{#2}}%
6142     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6143     \let\glsifplural\@secondoftwo
6144     \let\glscapscase\@secondofthree
6145     \let\glsinsert\@empty
6146     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6147     \glxtrsetupfulldefs
6148     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6149   }%
6150   \glspostlinkhook
6151 }
```

\GLSxtrfull Full form (all uppercase).

```
6152 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
6153 \newcommand*\ns@GLSxtrfull[2][{}]{%
6154   \new@ifnextchar[{\@GLSxtr@full{#1}{#2}}{%
6155     {\@GLSxtr@full{#1}{#2}[]}%
6156 }
```

\@GLSxtr@full Low-level macro:

```
6157 \def\@GLSxtr@full#1#2[#3]{%
6158   \glsdoifexists{#2}%
6159   {%
6160     \glssetabbrvfmt{\glscategory{#2}}%
6161     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6162     \let\glsifplural\@secondoftwo
6163     \let\glscapscase\@thirdofthree
6164     \let\glsinsert\@empty
6165     \def\glscustomtext{\mfirstucMakeUppercase{\glxtrinlinefullformat{#2}{#3}}}%
6166     \glxtrsetupfulldefs
6167     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6168   }%
6169   \glspostlinkhook
```

6170 }

\glstrfullpl Plural full form (no case-change).

```
6171 \newrobustcmd*{\glstrfullpl}{\@gls@hyp@opt\@ns@glstrfullpl}
6172 \newcommand*\ns@glstrfullpl[2][]{%
6173   \new@ifnextchar[{\@glstr@fullpl{#1}{#2}}%
6174   {\@glstr@fullpl{#1}{#2}[]}%
6175 }
```

\@glstr@fullpl Low-level macro:

```
6176 \def\@glstr@fullpl#1#2[#3]{%
6177   \glstoifexists{#2}%
6178   {%
6179     \glsssetabbrfmt{\glscategory{#2}}%
6180     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6181     \let\gl@ifplural\@firstoftwo
6182     \let\glscapscase\@firstofthree
6183     \let\glinsert\@empty
6184     \def\glscustomtext{\glstrinlinefullplformat{#2}{#3}}%
6185     \glstrsetupfulldefs
6186     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6187   }%
6188   \glspostlinkhook
6189 }
```

\Glsxtrfullpl Plural full form (first letter uppercase).

```
6190 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6191 \newcommand*\ns@Glsxtrfullpl[2][]{%
6192   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6193   {\@Glsxtr@fullpl{#1}{#2}[]}%
6194 }
```

\@Glsxtr@fullpl Low-level macro:

```
6195 \def\@Glsxtr@fullpl#1#2[#3]{%
6196   \glstoifexists{#2}%
6197   {%
6198     \glsssetabbrfmt{\glscategory{#2}}%
6199     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6200     \let\gl@ifplural\@firstoftwo
6201     \let\glscapscase\@secondofthree
6202     \let\glinsert\@empty
6203     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6204     \glstrsetupfulldefs
6205     \@gl@link[#1]{#2}{\csname gls@glstype @entryfmt\endcsname}%
6206   }%
6207   \glspostlinkhook
6208 }
```

\GLSxtrfullpl Plural full form (all upper case).


```

6209 \newrobustcmd*{\GLSxtrfullpl}{\@gls@hyp@opt\ns@GLSxtrfullpl}
6210 \newcommand*\ns@GLSxtrfullpl[2][]{%
6211   \new@ifnextchar[{\@GLSxtr@fullpl{#1}{#2}}]{%
6212     {\@GLSxtr@fullpl{#1}{#2}[]}%
6213 }

```

\@GLSxtr@fullpl Low-level macro:

```

6214 \def\@GLSxtr@fullpl#1#2[#3]{%
6215   \glsdoifexists{#2}%
6216   {%
6217     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6218     \let\glsifplural\@firstoftwo
6219     \let\glscapscase\@thirdofthree
6220     \let\glsinsert\@empty
6221     \def\glscustomtext{%
6222       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6223     \glsxtrsetupfulldefs
6224     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6225   }%
6226   \glspostlinkhook
6227 }

```

The short and long forms work in a similar way to acronyms.

\glsxtrshort

```

6228 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6229 \newcommand*{\ns@glsxtrshort}[2][]{%
6230   \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2}[]}%
6231 }

```

Read in the final optional argument:

```

6232 \def\@glsxtrshort#1#2[#3]{%
6233   \glsdoifexists{#2}%
6234   {%

```

Need to make sure \glsabbrvfont is set correctly.

```

6235     \glssetabbrvfmt{\glscategory{#2}}%
6236     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6237     \let\glsxtrifwasfirstuse\@secondoftwo
6238     \let\glsifplural\@secondoftwo
6239     \let\glscapscase\@firstofthree
6240     \let\glsinsert\@empty
6241     \def\glscustomtext{%
6242       \glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
6243       \ifglsxtrininsertinside\else#3\fi
6244     }%
6245     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6246   }%
6247   \glspostlinkhook

```



```

6287     }%
6288     }%
6289     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6290     }%
6291     \glspostlinkhook
6292 }

```

`\glsxtrlong`

```

6293 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\@ns@glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6294 \newcommand*{\ns@glsxtrlong}[2][{}]{%
6295   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2}[]}%
6296 }

    Read in the final optional argument:
6297 \def\@glsxtrlong#1#2[#3]{%
6298   \glsdoifexists{#2}%
6299   {%
6300     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6301     \let\glsxtrifwasfirstuse\@secondoftwo
6302     \let\glsifplural\@secondoftwo
6303     \let\glscapscase\@firstofthree
6304     \let\glsinsert\@empty
6305     \def\glscustomtext{%
6306       \glslongfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
6307       \ifglsxtrininsertinside\else#3\fi
6308     }%
6309     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6310     }%
6311     \glspostlinkhook
6312 }

```

`\Glsxtrlong`

```

6313 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6314 \newcommand*{\ns@Glsxtrlong}[2][{}]{%
6315   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2}[]}%
6316 }

    Read in the final optional argument:
6317 \def\@Glsxtrlong#1#2[#3]{%
6318   \glsdoifexists{#2}%
6319   {%
6320     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6321     \let\glsxtrifwasfirstuse\@secondoftwo
6322     \let\glsifplural\@secondoftwo
6323     \let\glscapscase\@secondofthree
6324     \let\glsinsert\@empty
6325     \def\glscustomtext{%

```

```

6326      \glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6327      \ifglxtrinsertinside\else#3\fi
6328  }%
6329  \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6330  }%
6331  \glspostlinkhook
6332 }

```

\GLSxtrlong

```

6333 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\ns@GLSxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
6334 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6335   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6336 }

  Read in the final optional argument:
6337 \def\@GLSxtrlong#1#2[#3] {%
6338   \glsdoifexists{#2}%
6339   {%
6340     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6341     \let\glxtrifwasfirstuse\@secondoftwo
6342     \let\glsifplural\@secondoftwo
6343     \let\glscapscase\@thirdofthree
6344     \let\glsinsert\@empty
6345     \def\glscustomtext{%
6346       \mfirstucMakeUppercase
6347       {\glslongfont{\Glsaccesslong{#2}\ifglxtrinsertinside#3\fi}%
6348       \ifglxtrinsertinside\else#3\fi
6349     }%
6350   }%
6351   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6352   }%
6353   \glspostlinkhook
6354 }

```

Plural short forms:

\glxtrshortpl

```

6355 \newrobustcmd*{\glxtrshortpl}{\@gls@hyp@opt\ns@glxtrshortpl}

  Define the un-starred form. Need to determine if there is a final optional argument
6356 \newcommand*{\ns@glxtrshortpl}[2] [] {%
6357   \new@ifnextchar[{\@glxtrshortpl{#1}{#2}}{\@glxtrshortpl{#1}{#2} []}%
6358 }

  Read in the final optional argument:
6359 \def\@glxtrshortpl#1#2[#3] {%
6360   \glsdoifexists{#2}%
6361   {%
6362     \glssetabbrvfmt{\glscategory{#2}}%

```

```

6363 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6364 \let\glxtrifwasfirstuse\@secondoftwo
6365 \let\gl@sifplural\@firstoftwo
6366 \let\glscapscase\@firstofthree
6367 \let\gl$insert\@empty
6368 \def\glscustomtext{%
6369 \glabbrvfont{\glaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6370 \ifglxtrininsertinside\else#3\fi
6371 }%
6372 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6373 }%
6374 \glspostlinkhook
6375 }

```

\Glsxtrshortpl

```

6376 \newrobustcmd*{\Glsxtrshortpl}{\@gl@hyp@opt\@ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6377 \newcommand*{\ns@Glsxtrshortpl}[2][{}]{%
6378 \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[]}%
6379 }
    Read in the final optional argument:
6380 \def\@Glsxtrshortpl#1#2[#3]{%
6381 \gl@doifexists{#2}%
6382 {%
6383 \glsetabbrvfmt{\glscategory{#2}}%
6384 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6385 \let\glxtrifwasfirstuse\@secondoftwo
6386 \let\gl@sifplural\@firstoftwo
6387 \let\glscapscase\@secondofthree
6388 \let\gl$insert\@empty
6389 \def\glscustomtext{%
6390 \glabbrvfont{\Glsaccessshortpl{#2}\ifglxtrininsertinside#3\fi}%
6391 \ifglxtrininsertinside\else#3\fi
6392 }%
6393 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6394 }%
6395 \glspostlinkhook
6396 }

```

\GLSxtrshortpl

```

6397 \newrobustcmd*{\GLSxtrshortpl}{\@gl@hyp@opt\@ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6398 \newcommand*{\ns@GLSxtrshortpl}[2][{}]{%
6399 \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2}[]}%
6400 }
    Read in the final optional argument:
6401 \def\@GLSxtrshortpl#1#2[#3]{%

```

```

6402 \glsdoifexists{#2}%
6403 {%
6404   \glssetabbrvfmt{\glscategory{#2}}%
6405   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6406   \let\glsxtrifwasfirstuse\@secondoftwo
6407   \let\glsifplural\@firstoftwo
6408   \let\glsapscase\@thirdofthree
6409   \let\glsinsert\@empty
6410   \def\glscustomtext{%
6411     \mfirstucMakeUppercase
6412     {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6413     \ifglsxtrininsertinside\else#3\fi
6414   }%
6415   }%
6416   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6417 }%
6418 \glspostlinkhook
6419 }

```

Plural long forms:

\glsxtrlongpl

```

6420 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6421 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
6422   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
6423 }

```

Read in the final optional argument:

```

6424 \def\@glsxtrlongpl#1#2[#3]{%
6425   \glsdoifexists{#2}%
6426   {%
6427     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6428     \let\glsxtrifwasfirstuse\@secondoftwo
6429     \let\glsifplural\@firstoftwo
6430     \let\glsapscase\@firstofthree
6431     \let\glsinsert\@empty
6432     \def\glscustomtext{%
6433       \glsfont{\glsaccesslongpl{#2}\ifglsxtrininsertinside#3\fi}%
6434       \ifglsxtrininsertinside\else#3\fi
6435     }%
6436     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6437   }%
6438   \glspostlinkhook
6439 }

```

\Glsxtrlongpl

```

6440 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```
6441 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
6442   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
6443 }
```

Read in the final optional argument:

```
6444 \def\@Glsxtrlongpl#1#2[#3]{%
6445   \glsdoifexists{#2}%
6446   {%
6447     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6448     \let\glxtrifwasfirstuse\@secondoftwo
6449     \let\glsifplural\@firstoftwo
6450     \let\glscapscase\@secondofthree
6451     \let\glsinsert\@empty
6452     \def\glscustomtext{%
6453       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6454       \ifglxtrininsertinside\else#3\fi
6455     }%
6456     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6457   }%
6458   \glspostlinkhook
6459 }
```

\GLSxtrlongpl

```
6460 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6461 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
6462   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
6463 }
```

Read in the final optional argument:

```
6464 \def\@GLSxtrlongpl#1#2[#3]{%
6465   \glsdoifexists{#2}%
6466   {%
6467     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6468     \let\glxtrifwasfirstuse\@secondoftwo
6469     \let\glsifplural\@firstoftwo
6470     \let\glscapscase\@thirdofthree
6471     \let\glsinsert\@empty
6472     \def\glscustomtext{%
6473       \mfirstucMakeUppercase
6474       {\glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6475       \ifglxtrininsertinside\else#3\fi
6476     }%
6477   }%
6478   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6479 }%
6480 \glspostlinkhook
6481 }
```

`\glssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6482 \newcommand*{\glssetabbrvfmt}[1]{%
6483   \ifcsdef{@glsabbrv@current@#1}%
6484     {\glstr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6485     {\glstr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6486 }

```

`\glsuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6487 \newrobustcmd*{\glsuseabbrvfont}[2]{\glssetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\glsuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6488 \newrobustcmd*{\glsuselongfont}[2]{\glssetabbrvfmt{#2}\glsfont{#1}}

```

`\glstrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6489 \newcommand*{\glstrgenabbrvfmt}{%
6490   \ifdefempty\glscustomtext
6491     {%
6492       \ifglsused\glslabel
6493         {%

```

Subsequent use:

```

6494       \glsifplural
6495       {%

```

Subsequent plural form:

```

6496       \glscapscase
6497       {%

```

Subsequent plural form, don't adjust case:

```

6498       \glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6499       }%
6500       {%

```

Subsequent plural form, make first letter upper case:

```

6501       \Glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6502       }%
6503       {%

```

Subsequent plural form, all caps:

```

6504       \mfirstucMakeUppercase
6505       {\glstrsubsequentplfmt{\glslabel}{\glsinsert}}%
6506       }%
6507       }%
6508       {%

```

Subsequent singular form

```

6509       \glscapscase
6510       {%

```


Subsequent singular form, don't adjust case:

```
6511      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6512      }%  
6513      {%
```

Subsequent singular form, make first letter upper case:

```
6514      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6515      }%  
6516      {%
```

Subsequent singular form, all caps:

```
6517      \mfirstucMakeUppercase  
6518      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6519      }%  
6520      }%  
6521      }%  
6522      {%
```

First use:

```
6523      \glsifplural  
6524      {%
```

First use plural form:

```
6525      \glscapscase  
6526      {%
```

First use plural form, don't adjust case:

```
6527      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6528      }%  
6529      {%
```

First use plural form, make first letter upper case:

```
6530      \Glsxtrfullplformat{\glslabel}{\glsinsert}%  
6531      }%  
6532      {%
```

First use plural form, all caps:

```
6533      \mfirstucMakeUppercase  
6534      {\glxtrfullplformat{\glslabel}{\glsinsert}}%  
6535      }%  
6536      }%  
6537      {%
```

First use singular form

```
6538      \glscapscase  
6539      {%
```

First use singular form, don't adjust case:

```
6540      \glxtrfullformat{\glslabel}{\glsinsert}%  
6541      }%  
6542      {%
```

First use singular form, make first letter upper case:

```
6543      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6544      }%
6545      {%
```

First use singular form, all caps:

```
6546      \mfirstucMakeUppercase
6547      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6548      }%
6549      }%
6550      }%
6551      }%
6552      {%
```

User supplied text.

```
6553      \glscustomtext
6554      }%
6555      }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6556 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6557   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6558   \ifglsxtrininsertinside \else#2\fi
6559 }
6560 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6561 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6562   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6563   \ifglsxtrininsertinside \else#2\fi
6564 }
6565 \let\glsxtrdefaultsubsequentplfmt\glsxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
6566 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6567   \glsabbrvfont{\Glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6568   \ifglsxtrininsertinside \else#2\fi
6569 }
6570 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```
6571 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6572   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6573   \ifglsxtrininsertinside \else#2\fi
6574 }
6575 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt
```

1.6.1 Abbreviation Styles Setup

abbreviationstyle

```

6576 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6577   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6578   {%
6579     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6580   }%
6581   {%

```

Have abbreviations already been defined for this category?

```

6582   \ifcsstring{@glsabbrv@current@#1}{#2}%
6583   {%

```

Style already set.

```

6584   }%
6585   {%
6586     \def\@glsxtr@dostylewarn{%
6587       \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6588       {%
6589         \def\@glsxtr@dostylewarn{\GlossariesWarning{Abbreviation
6590           style has been switched \MessageBreak
6591           for category ‘#1’, \MessageBreak
6592           but there have already been entries \MessageBreak
6593           defined for this category. Unwanted \MessageBreak
6594           side-effects may result}}%
6595         \@endfortrue
6596       }%
6597     \@glsxtr@dostylewarn

```

Set up the style for the given category.

```

6598     \csdef{@glsabbrv@current@#1}{#2}%
6599     \glsxtr@applyabbrvstyle{#2}%
6600   }%
6601 }%
6602 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

6603 \newcommand*{\glsxtr@applyabbrvstyle}[1]{%
6604   \csuse{@glsabbrv@dispstyle@setup@#1}%
6605   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6606 }

```

r@applyabbrvfmt Only apply the style formats.

```

6607 \newcommand*{\glsxtr@applyabbrvfmt}[1]{%
6608   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6609 }

```

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.

```

6610 \newcommand*{\newabbreviationstyle}[3]{%
6611   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6612   {%
6613     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6614       defined}{}%
6615   }%
6616   {%
6617     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6618       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6619       #2}%
6620     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6621       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6622       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6623       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6624       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
        Reset \glsxtrsubsequentfmt etc in case a style changes this.
6625       \let\glsxtrsubsequentfmt\glsxtrdefaultsubsequentfmt
6626       \let\glsxtrsubsequentplfmt\glsxtrdefaultsubsequentplfmt
6627       \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6628       \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6629       #3}%
6630     }%
6631 }

```

breivationstyle

```

6632 \newcommand*{\renewabbreviationstyle}[3]{%
6633   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
6634   {%
6635     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6636   }%
6637   {%
6638     \csdef{@glsabbrv@dispstyle@setup@#1}{%
        Initialise hook to do nothing. The style may change this.
6639       \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6640       #2}%
6641     \csdef{@glsabbrv@dispstyle@fmts@#1}{%
        Assume in-line form is the same as first use. The style may change this.
6642       \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6643       \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6644       \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6645       \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6646       #3}%
6647     }%
6648 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```
6649 \newcommand*{\letabbreviationstyle}[2]{%
6650   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6651   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6652 }
```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```
6653 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6654   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6655     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6656     \csuse{@glsabbrv@dispstyle@setup@#2}%
6657   }%
6658   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6659 }
```

deprecatedAbbrStyle Generate warning for deprecated style use.

```
6660 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6661   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6662   use ‘#2’ instead}%
6663 }
```

useAbbrStyleSetup

```
6664 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6665   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
6666   {%
6667     \PackageError{glossaries-extra}%
6668     {Unknown abbreviation style definitions ‘#1’}{}%
6669   }%
6670   {%
6671     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6672   }%
6673 }
```

useAbbrStyleFmt

```
6674 \newcommand*{\GlsXtrUseAbbrStyleFmt}[1]{%
6675   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6676   {%
6677     \PackageError{glossaries-extra}%
6678     {Unknown abbreviation style formats ‘#1’}{}%
6679   }%
6680   {%
6681     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6682   }%
6683 }
```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to "true". If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glstrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```
6684 \newif\ifglstrinsertinside
6685 \glstrinsertinsidefalse
```

`long-short`

```
6686 \newabbreviationstyle{long-short}%
6687 {%
6688   \renewcommand*{\CustomAbbreviationFields}{%
6689     name={\protect\glsabbrvfont{\the\glsshorttok}},
6690     sort={\the\glsshorttok},
6691     first={\protect\glsfirstlongfont{\the\glslongtok}%
6692       \protect\glstrfullsep{\the\glslabeltok}%
6693       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6694     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6695       \protect\glstrfullsep{\the\glslabeltok}%
6696       \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
6697     plural={\protect\glsabbrvfont{\the\glsshortpltok}}},%
6698     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
6699   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6700     \glshasattribute{\the\glslabeltok}{regular}%
6701     {%
6702       \glissetattribute{\the\glslabeltok}{regular}{false}%
6703       }%
6704     {}%
6705   }%
6706 }%
6707 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6708   \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6709   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6710   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6711   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6712   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline form are the same for this style.

```

6713 \renewcommand*{\glxtrfullformat}[2]{%
6714   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6715   \ifglxtrinsertinside\else##2\fi
6716   \glxtrfullsep{##1}%
6717   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6718 }%
6719 \renewcommand*{\glxtrfullplformat}[2]{%
6720   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6721   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6722   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6723 }%
6724 \renewcommand*{\Glsxtrfullformat}[2]{%
6725   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
6726   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6727   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6728 }%
6729 \renewcommand*{\Glsxtrfullplformat}[2]{%
6730   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
6731   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6732   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6733 }%
6734 }

```

Set this as the default style for general abbreviations:

```

6735 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

6736 \newcommand*{\glxtrlongshortdescsort}{%
6737   \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6738 }

```

ngshortdescname

```

6739 \newcommand*{\glxtrlongshortdescname}{%
6740   \protect\glslongfont{\the\glslongtok}
6741   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6742 }

```

long-short-desc User supplies description. The long form is included in the name.

```

6743 \newabbreviationstyle{long-short-desc}%
6744 {%
6745   \renewcommand*{\CustomAbbreviationFields}{%
6746     name={\glxtrlongshortdescname},
6747     sort={\glxtrlongshortdescsort},%
6748     first={\protect\glsfirstlongfont{\the\glslongtok}%
6749       \protect\glxtrfullsep{\the\glslabeltok}%
6750       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6751     firstplural={\protect\glsfirstlongfont{\the\glslongtok}%
6752       \protect\glxtrfullsep{\the\glslabeltok}%

```

```
6753 \glstrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
```

The text key should only have the short form.

```
6754 text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```
6755 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6756 }%
```

Unset the regular attribute if it has been set.

```
6757 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6758 \glshasattribute{\the\glslabeltok}{regular}%
6759 {%
6760 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6761 }%
6762 {}%
6763 }%
6764 }%
6765 {%
6766 \GlsXtrUseAbbrStyleFmts{long-short}%
6767 }
```

short-long Short form followed by long form in parenthesis on first use.

```
6768 \newabbreviationstyle{short-long}%
6769 {%
6770 \renewcommand*{\CustomAbbreviationFields}{%
6771 name={\protect\glsabbrvfont{\the\glsshorttok}},
6772 sort={\the\glsshorttok},
6773 description={\the\glslongtok},%
6774 first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6775 \protect\glstrfullsep{\the\glslabeltok}%
6776 \glstrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6777 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6778 \protect\glstrfullsep{\the\glslabeltok}%
6779 \glstrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6780 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6781 }
```

Unset the regular attribute if it has been set.

```
6781 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6782 \glshasattribute{\the\glslabeltok}{regular}%
6783 {%
6784 \glsssetattribute{\the\glslabeltok}{regular}{false}%
6785 }%
6786 {}%
6787 }%
6788 }%
6789 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6790 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6791 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%
```



```

6792 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6793 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6794 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6795 \renewcommand*{\glsxtrfullformat}[2]{%
6796   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6797   \ifglsxtrininsertinside\else##2\fi
6798   \glsxtrfullsep{##1}%
6799   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6800 }%
6801 \renewcommand*{\glsxtrfullplformat}[2]{%
6802   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6803   \ifglsxtrininsertinside\else##2\fi
6804   \glsxtrfullsep{##1}%
6805   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6806 }%
6807 \renewcommand*{\Glsxtrfullformat}[2]{%
6808   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6809   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6810   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6811 }%
6812 \renewcommand*{\Glsxtrfullplformat}[2]{%
6813   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6814   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6815   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6816 }%
6817 }

```

ortlongdescsort

```

6818 \newcommand*{\glsxtrshortlongdescsort}{\the\glsshorttok}

```

ortlongdescname

```

6819 \newcommand*{\glsxtrshortlongdescname}{%
6820   \protect\glsabbrvfont{\the\glsshorttok}
6821   \glsxtrparen{\protect\glslongfont{\the\glslongtok}}%
6822 }

```

short-long-desc User supplies description. The long form is included in the name.

```

6823 \newabbreviationstyle{short-long-desc}%
6824 {%
6825   \renewcommand*{\CustomAbbreviationFields}{%
6826     name={\glsxtrshortlongdescname},
6827     sort={\glsxtrshortlongdescsort},
6828     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6829       \protect\glsxtrfullsep{\the\glslabeltok}%
6830       \glsxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6831     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6832       \protect\glsxtrfullsep{\the\glslabeltok}%
6833       \glsxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%

```

```

6834   text={\protect\glsabbrvfont{\the\glsshorttok}},%
6835   plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6836 }%
    Unset the regular attribute if it has been set.
6837   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6838     \glshasattribute{\the\glslabeltok}{regular}%
6839     {%
6840       \glissetattribute{\the\glslabeltok}{regular}{false}%
6841     }%
6842   }%
6843 }%
6844 }%
6845 {%
6846   \GlsXtrUseAbbrStyleFmts{short-long}%
6847 }

```

ongfootnotefont Only used by the “footnote” styles.

```
6848 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6849 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote `\glsxtrabbrvfootnote{<label>}{<long>}`

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
6850 \newcommand*{\glsxtrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```

6851 \newabbreviationstyle{footnote}%
6852 {%
6853   \renewcommand*{\CustomAbbreviationFields}{%
6854     name={\protect\glsabbrvfont{\the\glsshorttok}},
6855     sort={\the\glsshorttok},
6856     description={\the\glslongtok},%
6857     first={\protect\glsfirstabbrvfont{\the\glsshorttok}}%
6858     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6859     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6860     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6861     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
6862     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

```

```
6863 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.

```
6864 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6865   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6866   \glsattribute{\the\glslabeltok}{regular}%
6867   {%
6868     \glssetattribute{\the\glslabeltok}{regular}{false}%
6869   }%
6870   {}}%
6871 }%
6872 }%
6873 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6874 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6875 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6876 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6877 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6878 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form followed by the long form as a footnote.

```
6879 \renewcommand*{\glsxtrfullformat}[2]{%
6880   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6881   \ifglsxtrininsertinside\else##2\fi
6882   \protect\glsxtrabbrvfootnote{##1}%
6883   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6884 }%
6885 \renewcommand*{\glsxtrfullplformat}[2]{%
6886   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6887   \ifglsxtrininsertinside\else##2\fi
6888   \protect\glsxtrabbrvfootnote{##1}%
6889   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6890 }%
6891 \renewcommand*{\Glsxtrfullformat}[2]{%
6892   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6893   \ifglsxtrininsertinside\else##2\fi
6894   \protect\glsxtrabbrvfootnote{##1}%
6895   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6896 }%
6897 \renewcommand*{\Glsxtrfullplformat}[2]{%
6898   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
6899   \ifglsxtrininsertinside\else##2\fi
6900   \protect\glsxtrabbrvfootnote{##1}%
6901   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6902 }%
```

The first use full form and the inline full form use the short (long) style.

```
6903 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6904   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```

```

6905     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6906     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
6907 }%
6908 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6909     \glsfirstabbrvfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6910     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6911     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
6912 }%
6913 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6914     \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6915     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6916     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
6917 }%
6918 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6919     \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6920     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6921     \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
6922 }%
6923 }

```

short-footnote

```

6924 \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.

```

6925 \newabbreviationstyle{postfootnote}%
6926 {%
6927     \renewcommand*{\CustomAbbreviationFields}{%
6928         name={\protect\glsabbrvfont{\the\glsshorttok}},
6929         sort={\the\glsshorttok},
6930         description={\the\glslongtok},%
6931         first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6932         firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6933         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.

```

6934 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6935     \csdef{glxtrpostlink\glscategorylabel}{%
6936         \glxtrifwasfirstuse
6937         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6938         \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
6939         {\glsfirstlongfootnotefont{\glentrylong{\glslabel}}}%
6940     }%

```

```

6941     {}%
6942   }%
6943   \glshasattribute{\the\glslabeltok}{regular}%
6944   {%
6945     \glsetattribute{\the\glslabeltok}{regular}{false}%
6946   }%
6947   {}%
6948 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

6949 \renewcommand*{\glxtrsetupfulldefs}{%
6950   \let\glxtrifwasfirstuse\@secondoftwo
6951 }%
6952 }%
6953 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6954 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6955 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6956 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6957 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6958 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6959 \renewcommand*{\glxtrfullformat}[2]{%
6960   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6961   \ifglxtrininsertinside\else##2\fi
6962 }%
6963 \renewcommand*{\glxtrfullplformat}[2]{%
6964   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6965   \ifglxtrininsertinside\else##2\fi
6966 }%
6967 \renewcommand*{\Glsxtrfullformat}[2]{%
6968   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6969   \ifglxtrininsertinside\else##2\fi
6970 }%
6971 \renewcommand*{\Glsxtrfullplformat}[2]{%
6972   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6973   \ifglxtrininsertinside\else##2\fi
6974 }%

```

The first use full form and the inline full form use the short (long) style.

```

6975 \renewcommand*{\glxtrinlinefullformat}[2]{%
6976   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6977   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6978   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6979 }%
6980 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6981   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6982   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

6983 \glstrparen{\glfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6984 }%
6985 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6986 \glfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6987 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6988 \glstrparen{\glfirstlongfootnotefont{\glsaccesslong{##1}}}%
6989 }%
6990 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6991 \glfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6992 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6993 \glstrparen{\glfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6994 }%
6995 }

```

rt-postfootnote

```

6996 \letabbreviationstyle{short-postfootnote}{postfootnote}

```

short Provide a style that only displays the short form on first use, but the short and long form can be displayed with the “full” commands that use the inline format. If the user supplies a description, the long form won’t be displayed in the predefined glossary styles, but the post description hook can be employed to automatically insert it.

```

6997 \newabbreviationstyle{short}{%
6998 {%
6999 \renewcommand*{\CustomAbbreviationFields}{%
7000 name={\protect\glfirstabbrvfont{\the\glsshorttok}},
7001 sort={\the\glsshorttok},
7002 first={\protect\glfirstabbrvfont{\the\glsshorttok}},
7003 firstplural={\protect\glfirstabbrvfont{\the\glsshortpltok}},
7004 text={\protect\glfirstabbrvfont{\the\glsshorttok}},
7005 plural={\protect\glfirstabbrvfont{\the\glsshortpltok}},
7006 description={\the\glslongtok}}%
7007 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7008 \glsetattribute{\the\glslabeltok}{regular}{true}}%
7009 }%
7010 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7011 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7012 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvdefaultfont{##1}}%
7013 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
7014 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
7015 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7016 \renewcommand*{\glxtrinlinefullformat}[2]{%
7017 \protect\glfirstabbrvfont{\glsaccessshort{##1}}%
7018 \ifglxtrinsertinside##2\fi}%
7019 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7020 \glstrparen{\glfirstlongfont{\glsaccesslong{##1}}}%
7021 }%

```

```

7022 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7023   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
7024     \ifglxtrininsertinside##2\fi}%
7025   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7026   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7027 }%
7028 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7029   \protect\glsfirstabbrvfont{\glsaccessshort{##1}%
7030     \ifglxtrininsertinside##2\fi}%
7031   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7032   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7033 }%
7034 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7035   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}%
7036     \ifglxtrininsertinside##2\fi}%
7037   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7038   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7039 }%

```

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.

```

7040 \renewcommand*{\glxtrfullformat}[2]{%
7041   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7042   \ifglxtrininsertinside\else##2\fi
7043 }%
7044 \renewcommand*{\glxtrfullplformat}[2]{%
7045   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7046   \ifglxtrininsertinside\else##2\fi
7047 }%
7048 \renewcommand*{\Glsxtrfullformat}[2]{%
7049   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7050   \ifglxtrininsertinside\else##2\fi
7051 }%
7052 \renewcommand*{\Glsxtrfullplformat}[2]{%
7053   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7054   \ifglxtrininsertinside\else##2\fi
7055 }%
7056 }

```

Set this as the default style for acronyms:

```

7057 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7058 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7059 \newabbreviationstyle{short-nolong-noreg}%
7060 {%
7061   \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7062 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7063   \glshasattribute{\the\glslabeltok}{regular}%
7064   {%
7065     \glsselattribute{\the\glslabeltok}{regular}{false}%
7066   }%
7067   {}%
7068 }%
7069}%
7070{%
7071 \GlsXtrUseAbbrStyleFmts{short-nolong}%
7072}

```

trshortdescname

```

7073 \newcommand*{\glxtrshortdescname}{%
7074   \protect\glsabbrvfont{\the\glsshorttok}%
7075 }

```

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.

```

7076 \newabbreviationstyle{short-desc}%
7077 {%
7078   \renewcommand*{\CustomAbbreviationFields}{%
7079     name={\glxtrshortdescname},
7080     sort={\the\glsshorttok},
7081     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7082     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7083     text={\protect\glsabbrvfont{\the\glsshorttok}},
7084     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7085     description={\the\glslongtok}}%
7086   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7087     \glsselattribute{\the\glslabeltok}{regular}{true}}%
7088 }%
7089 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7090 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7091 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7092 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7093 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7094 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7095 \renewcommand*{\glxtrinlinefullformat}[2]{%
7096   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7097   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7098   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7099 }%
7100 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7101   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```



```

7102 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7103 \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
7104 }%
7105 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7106 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7107 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7108 \glxtrparen{\glsfirstlongfont{\glssaccesslong{##1}}}%
7109 }%
7110 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7111 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7112 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7113 \glxtrparen{\glsfirstlongfont{\glssaccesslongpl{##1}}}%
7114 }%

```

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.

```

7115 \renewcommand*{\glxtrfullformat}[2]{%
7116 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7117 \ifglxtrinsertinside\else##2\fi
7118 }%
7119 \renewcommand*{\glxtrfullplformat}[2]{%
7120 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7121 \ifglxtrinsertinside\else##2\fi
7122 }%
7123 \renewcommand*{\Glsxtrfullformat}[2]{%
7124 \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7125 \ifglxtrinsertinside\else##2\fi
7126 }%
7127 \renewcommand*{\Glsxtrfullplformat}[2]{%
7128 \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7129 \ifglxtrinsertinside\else##2\fi
7130 }%
7131 }

```

ort-nolong-desc

```
7132 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7133 \newabbreviationstyle{short-nolong-desc-noreg}%
7134 {%
7135 \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%

```

Unset the regular attribute if it has been set.

```

7136 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7137 \glshasattribute{\the\glslabeltok}{regular}%
7138 {%
7139 \glissetattribute{\the\glslabeltok}{regular}{false}%
7140 }%
7141 {}%
7142 }%

```

```

7143 }%
7144 {%
7145   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7146 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7147 \newabbreviationstyle{nolong-short}%
7148 {%
7149   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7150 }%
7151 {%
7152   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7153   \renewcommand*{\glxtrinlinefullformat}[2]{%
7154     \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7155     \ifglxtrininsertinside##2\fi}%
7156   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7157   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7158 }%
7159 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7160   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7161   \ifglxtrininsertinside##2\fi}%
7162   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7163   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7164 }%
7165 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7166   \protect\glsfirstlongfont{\glsaccesslong{##1}}%
7167   \ifglxtrininsertinside##2\fi}%
7168   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7169   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7170 }%
7171 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7172   \protect\glsfirstlongfont{\glsaccesslongpl{##1}}%
7173   \ifglxtrininsertinside##2\fi}%
7174   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7175   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7176 }%
7177 }

```

nolong-short-noreg Like nolong-short but doesn't set the regular attribute.

```

7178 \newabbreviationstyle{nolong-short-noreg}%
7179 {%
7180   \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7181   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7182     \glshasattribute{\the\glslabeltok}{regular}%
7183     {%

```

```

7184     \glssetattribute{\the\glslabeltok}{regular}{false}%
7185   }%
7186   {}%
7187 }%
7188 }%
7189 {%
7190   \GlsXtrUseAbbrStyleFmts{nolong-short}%
7191 }

```

`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.

```

7192 \newabbreviationstyle{long-desc}%
7193 {%
7194   \renewcommand*{\CustomAbbreviationFields}{%
7195     name={\protect\protect\glslongfont{\the\glslongtok}},
7196     sort={\the\glslongtok},
7197     first={\protect\glsfirstlongfont{\the\glslongtok}},
7198     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7199     text={\glslongfont{\the\glslongtok}},
7200     plural={\glslongfont{\the\glslongpltok}}}%
7201   }%
7202   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7203     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7204 }%
7205 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7206 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
7207 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7208 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7209 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7210 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7211 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
7212   \glslongfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7213   \ifglsxtrininsertinside \else##2\fi
7214 }%
7215 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
7216   \glslongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7217   \ifglsxtrininsertinside \else##2\fi
7218 }%
7219 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7220   \glslongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7221   \ifglsxtrininsertinside \else##2\fi
7222 }%
7223 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7224   \glslongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7225   \ifglsxtrininsertinside \else##2\fi

```

7226 }%

The inline full form displays the long format followed by the short form in parentheses.

```
7227 \renewcommand*{\glxstrinlinefullformat}[2]{%
7228   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7229   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
7230   \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7231 }%
7232 \renewcommand*{\glxstrinlinefullplformat}[2]{%
7233   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7234   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
7235   \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7236 }%
7237 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7238   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7239   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
7240   \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7241 }%
7242 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7243   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7244   \ifglxtrinsertinside\else##2\fi\glxstrfullsep{##1}%
7245   \glxstrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7246 }%
```

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.

```
7247 \renewcommand*{\glxstrfullformat}[2]{%
7248   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7249   \ifglxtrinsertinside\else##2\fi
7250 }%
7251 \renewcommand*{\glxstrfullplformat}[2]{%
7252   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7253   \ifglxtrinsertinside\else##2\fi
7254 }%
7255 \renewcommand*{\Glsxtrfullformat}[2]{%
7256   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7257   \ifglxtrinsertinside\else##2\fi
7258 }%
7259 \renewcommand*{\Glsxtrfullplformat}[2]{%
7260   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7261   \ifglxtrinsertinside\else##2\fi
7262 }%
7263 }
```

ng-noshort-desc Provide a synonym that matches similar styles.

```
7264 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

hort-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```
7265 \newabbreviationstyle{long-noshort-desc-noreg}%
7266 {}%
```

```

7267 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
      Unset the regular attribute if it has been set.
7268 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7269   \glshasattribute{\the\glslabeltok}{regular}%
7270   {%
7271     \glissetattribute{\the\glslabeltok}{regular}{false}%
7272   }%
7273   {}%
7274 }%
7275}%
7276{%
7277 \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7278 }

```

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.

```

7279 \newabbreviationstyle{long}%
7280 {%
7281   \renewcommand*{\CustomAbbreviationFields}{%
7282     name={\protect\glsabbrvfont{\the\glsshorttok}},
7283     sort={\the\glsshorttok},
7284     first={\protect\glsfirstlongfont{\the\glslongtok}},
7285     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7286     text={\glsfont{\the\glslongtok}},
7287     plural={\glsfont{\the\glslongpltok}},%
7288     description={\the\glslongtok}%
7289   }%
7290   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7291     \glissetattribute{\the\glslabeltok}{regular}{true}}%
7292 }%
7293 {%
7294   \GlsXtrUseAbbrStyleFmts{long-desc}%
7295 }

```

long-noshort Provide a synonym that matches similar styles.

```

7296 \letabbreviationstyle{long-noshort}{long}

```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```

7297 \newabbreviationstyle{long-noshort-noreg}%
7298 {%
7299   \GlsXtrUseAbbrStyleSetup{long-noshort}%
      Unset the regular attribute if it has been set.
7300   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7301     \glshasattribute{\the\glslabeltok}{regular}%
7302     {%
7303       \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

7304 }%
7305 {}%
7306 }%
7307 }%
7308 {%
7309 \GlsXtrUseAbbrStyleFmts{long-noshort}%
7310 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

```

\glxtrscfont    Maintained for backward-compatibility.
7311 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}

\glsabbrvscfont Added for consistent naming.
7312 \newcommand*{\glsabbrvscfont}{\glxtrscfont}

sxtfirstscfont  Maintained for backward-compatibility.
7313 \newcommand*{\glxtrfirstscfont}[1]{\glsabbrvscfont{#1}}

irstabbrvscfont Added for consistent naming.
7314 \newcommand*{\glfirstabbrvscfont}{\glxtrfirstscfont}

```

and for the default short form suffix:

```

\glxtrscsuffix
7315 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabbrvpluralsuffix}}

long-short-sc
7316 \newabbreviationstyle{long-short-sc}%
7317 {%
7318 \renewcommand*{\CustomAbbreviationFields}{%
7319   name={\protect\glsabbrvscfont{\the\glsshorttok}},
7320   sort={\the\glsshorttok},
7321   first={\protect\glfirstlongdefaultfont{\the\glslongtok}%
7322     \protect\glxtrfullsep{\the\glslabeltok}}%
7323   \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshorttok}}},%
7324   firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}%
7325     \protect\glxtrfullsep{\the\glslabeltok}}%
7326   \glxtrparen{\protect\glfirstabbrvscfont{\the\glsshortpltok}}},%
7327   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}},%
7328   description={\the\glslongtok}}%
7329 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7330   \glshasattribute{\the\glslabeltok}{regular}%
7331   {%
7332     \glissetattribute{\the\glslabeltok}{regular}{false}%
7333   }%
7334   {}%

```

```

7335 }%
7336 }%
7337 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7338 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7339 \renewcommand*{\glssabrvfont[1]{\glssabrvscfont{##1}}}%
7340 \renewcommand*{\glssfirstabrvfont[1]{\glssfirstabrvscfont{##1}}}%

```

Use the default long fonts.

```

7341 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
7342 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7343 \renewcommand*{\glxtrfullformat}[2]{%
7344   \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7345   \ifglxtrininsertinside\else##2\fi
7346   \glxtrfullsep{##1}%
7347   \glxtrparen{\glssfirstabrvscfont{\glssaccessshort{##1}}}%
7348 }%
7349 \renewcommand*{\glxtrfullplformat}[2]{%
7350   \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7351   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7352   \glxtrparen{\glssfirstabrvscfont{\glssaccessshortpl{##1}}}%
7353 }%
7354 \renewcommand*{\Glsxtrfullformat}[2]{%
7355   \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7356   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7357   \glxtrparen{\glssfirstabrvscfont{\glssaccessshort{##1}}}%
7358 }%
7359 \renewcommand*{\Glsxtrfullplformat}[2]{%
7360   \glssfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7361   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7362   \glxtrparen{\glssfirstabrvscfont{\glssaccessshortpl{##1}}}%
7363 }%
7364 }

```

g-short-sc-desc

```

7365 \newabbreviationstyle{long-short-sc-desc}%
7366 {%
7367   \renewcommand*{\CustomAbbreviationFields}{%
7368     name={\glsxtrlongshortdescname},
7369     sort={\glsxtrlongshortdescsort},%
7370     first={\protect\glssfirstlongdefaultfont{\the\glsslongtok}%
7371       \protect\glxtrfullsep{\the\glsslabeltok}%
7372       \glxtrparen{\protect\glssfirstabrvscfont{\the\glssshorttok}}},%
7373     firstplural={\protect\glssfirstlongdefaultfont{\the\glsslongpltok}%
7374       \protect\glxtrfullsep{\the\glsslabeltok}%
7375       \glxtrparen{\protect\glssfirstabrvscfont{\the\glssshortpltok}}},%
7376     text={\protect\glssabrvscfont{\the\glssshorttok}},%

```

```

7377 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7378 }%

```

Unset the regular attribute if it has been set.

```

7379 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7380   \glshasattribute{\the\glslabeltok}{regular}%
7381   {%
7382     \glssetattribute{\the\glslabeltok}{regular}{false}%
7383   }%
7384   {}%
7385 }%
7386 }%
7387 {%

```

As long-short-sc style:

```

7388 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7389 }

```

Now the short (long) version

```

7390 \newabbreviationstyle{short-sc-long}%
7391 {%
7392   \renewcommand*{\CustomAbbreviationFields}{%
7393     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7394     sort={\the\glsshorttok},
7395     description={\the\glslongtok},%
7396     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7397     \protect\glsxtrfullsep{\the\glslabeltok}%
7398     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7399     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7400     \protect\glsxtrfullsep{\the\glslabeltok}%
7401     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7402     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7403 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7404   \glshasattribute{\the\glslabeltok}{regular}%
7405   {%
7406     \glssetattribute{\the\glslabeltok}{regular}{false}%
7407   }%
7408   {}%
7409 }%
7410 }%
7411 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7412 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7413 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7414 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7415 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7416 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.


```

7417 \renewcommand*{\glxtrfullformat}[2]{%
7418   \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7419   \ifglxtrinsertinside\else##2\fi
7420   \glxtrfullsep{##1}%
7421   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7422 }%
7423 \renewcommand*{\glxtrfullplformat}[2]{%
7424   \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7425   \ifglxtrinsertinside\else##2\fi
7426   \glxtrfullsep{##1}%
7427   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7428 }%
7429 \renewcommand*{\Glsxtrfullformat}[2]{%
7430   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7431   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7432   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7433 }%
7434 \renewcommand*{\Glsxtrfullplformat}[2]{%
7435   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7436   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7437   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7438 }%
7439 }

```

As before but user provides description

```

7440 \newabbreviationstyle{short-sc-long-desc}%
7441 {%
7442   \renewcommand*{\CustomAbbreviationFields}{%
7443     name={\glxtrshortlongdescname},
7444     sort={\glxtrshortlongdescsort},
7445     first={\protect\glsfirstabbrvscfont{\the\glssshorttok}%
7446       \protect\glxtrfullsep{\the\glslabeltok}%
7447       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7448     firstplural={\protect\glsfirstabbrvscfont{\the\glssshortpltok}%
7449       \protect\glxtrfullsep{\the\glslabeltok}%
7450       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7451     text={\protect\glssabbrvscfont{\the\glssshorttok}},%
7452     plural={\protect\glssabbrvscfont{\the\glssshortpltok}}%
7453   }%

```

Unset the regular attribute if it has been set.

```

7454 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7455   \glshasattribute{\the\glslabeltok}{regular}%
7456   {%
7457     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7458   }%
7459   {}%
7460 }%
7461 }%
7462 {%

```

As short-sc-long style:

```
7463 \GlsXtrUseAbbrStyleFmts{short-sc-long}%  
7464 }
```

short-sc

```
7465 \newabbreviationstyle{short-sc}%  
7466 {%  
7467 \renewcommand*{\CustomAbbreviationFields}{%  
7468 name={\protect\glsabbrvscfont{\the\glsshorttok}},  
7469 sort={\the\glsshorttok},  
7470 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},  
7471 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},  
7472 text={\protect\glsabbrvscfont{\the\glsshorttok}},  
7473 plural={\protect\glsabbrvscfont{\the\glsshortpltok}},  
7474 description={\the\glslongtok}}%  
7475 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7476 \glssetattribute{\the\glslabeltok}{regular}{true}}%  
7477 }%  
7478 }
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7479 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%  
7480 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%  
7481 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%  
7482 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%  
7483 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7484 \renewcommand*{\glxtrinlinefullformat}[2]{%  
7485 \protect\glsfirstabbrvscfont{\glsaccesssshort{##1}}%  
7486 \ifglxtrininsertinside##2\fi}%  
7487 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7488 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%  
7489 }%  
7490 \renewcommand*{\glxtrinlinefullplformat}[2]{%  
7491 \protect\glsfirstabbrvscfont{\glsaccesssshortpl{##1}}%  
7492 \ifglxtrininsertinside##2\fi}%  
7493 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7494 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%  
7495 }%  
  
7496 \renewcommand*{\Glsxtrinlinefullformat}[2]{%  
7497 \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}}%  
7498 \ifglxtrininsertinside##2\fi}%  
7499 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%  
7500 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%  
7501 }%  
7502 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%  
7503 \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}}%  
7504 \ifglxtrininsertinside##2\fi}%  
7505 }
```

```

7505     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7506     \glxtrparen{\glsfirstlongdefaultfont{\glaccesslongpl{##1}}}%
7507 }%

```

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.

```

7508 \renewcommand*{\glxtrfullformat}[2]{%
7509     \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7510     \ifglxtrinsertinside\else##2\fi
7511 }%
7512 \renewcommand*{\glxtrfullplformat}[2]{%
7513     \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7514     \ifglxtrinsertinside\else##2\fi
7515 }%
7516 \renewcommand*{\Glsxtrfullformat}[2]{%
7517     \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7518     \ifglxtrinsertinside\else##2\fi
7519 }%
7520 \renewcommand*{\Glsxtrfullplformat}[2]{%
7521     \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7522     \ifglxtrinsertinside\else##2\fi
7523 }%
7524 }

```

short-sc-nolong

```

7525 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7526 \newabbreviationstyle{short-sc-desc}{%
7527 {%
7528     \renewcommand*{\CustomAbbreviationFields}{%
7529         name={\glxtrshortdescname},
7530         sort={\the\glsshorttok},
7531         first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7532         firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7533         text={\protect\glsabbrvscfont{\the\glsshorttok}},
7534         plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7535         description={\the\glslongtok}}%
7536 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7537     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7538 }%
7539 }%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7540 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7541 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7542 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7543 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7544 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7545 \renewcommand*{\glxtrinlinefullformat}[2]{%
7546   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7547   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7548   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7549 }%
7550 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7551   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7552   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7553   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7554 }%
7555 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7556   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7557   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7558   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7559 }%
7560 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7561   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7562   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7563   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7564 }%

```

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.

```

7565 \renewcommand*{\glxtrfullformat}[2]{%
7566   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7567   \ifglxtrinsertinside\else##2\fi
7568 }%
7569 \renewcommand*{\glxtrfullplformat}[2]{%
7570   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7571   \ifglxtrinsertinside\else##2\fi
7572 }%
7573 \renewcommand*{\Glsxtrfullformat}[2]{%
7574   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7575   \ifglxtrinsertinside\else##2\fi
7576 }%
7577 \renewcommand*{\Glsxtrfullplformat}[2]{%
7578   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7579   \ifglxtrinsertinside\else##2\fi
7580 }%
7581 }

```

-sc-nolong-desc

```

7582 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7583 \newabbreviationstyle{nolong-short-sc}%
7584 {%
7585   \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%

```

```

7586 }%
7587 {%
7588   \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7589   \renewcommand*{\glsxtrinlinefullformat}[2]{%
7590     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}}%
7591     \ifglsxtrininsertinside##2\fi}%
7592   \ifglsxtrininsertinside\else##2\fi\glxstrfullsep{##1}%
7593   \glxstrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7594 }%
7595   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7596     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}%
7597     \ifglsxtrininsertinside##2\fi}%
7598   \ifglsxtrininsertinside\else##2\fi\glxstrfullsep{##1}%
7599   \glxstrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7600 }%
7601   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7602     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}}%
7603     \ifglsxtrininsertinside##2\fi}%
7604   \ifglsxtrininsertinside\else##2\fi\glxstrfullsep{##1}%
7605   \glxstrparen{\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7606 }%
7607   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7608     \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}%
7609     \ifglsxtrininsertinside##2\fi}%
7610   \ifglsxtrininsertinside\else##2\fi\glxstrfullsep{##1}%
7611   \glxstrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7612 }%
7613 }

```

`long-noshort-sc` The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7614 \newabbreviationstyle{long-noshort-sc}%
7615 {%
7616   \renewcommand*{\CustomAbbreviationFields}{%
7617     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7618     sort={\the\glsshorttok},
7619     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7620     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7621     text={\protect\glslongdefaultfont{\the\glslongtok}},
7622     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7623     description={\the\glslongtok}%
7624   }%
7625   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7626     \glssetAttribute{\the\glslabeltok}{regular}{true}}%
7627 }%
7628 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7629 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7630 \renewcommand*{\glabbrvfont}[1]{\glabbrvscfont{##1}}%
7631 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvscfont{##1}}%
7632 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongdefaultfont{##1}}%
7633 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7634 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7635   \glslongdefaultfont{\glaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7636   \ifglxtrininsertinside \else##2\fi
7637 }%
7638 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7639   \glslongdefaultfont{\glaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7640   \ifglxtrininsertinside \else##2\fi
7641 }%
7642 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7643   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7644   \ifglxtrininsertinside \else##2\fi
7645 }%
7646 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7647   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7648   \ifglxtrininsertinside \else##2\fi
7649 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7650 \renewcommand*{\glxtrininlinefullformat}[2]{%
7651   \glfirstlongdefaultfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7652   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7653   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshort{##1}}}%
7654 }%
7655 \renewcommand*{\glxtrininlinefullplformat}[2]{%
7656   \glfirstlongdefaultfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7657   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7658   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshortpl{##1}}}%
7659 }%
7660 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
7661   \glfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7662   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7663   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshort{##1}}}%
7664 }%
7665 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
7666   \glfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7667   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7668   \glxtrparen{\protect\glfirstabbrvscfont{\glaccessshortpl{##1}}}%
7669 }%

```

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.

```

7670 \renewcommand*{\glxtrfullformat}[2]{%
7671   \glfirstlongdefaultfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7672   \ifglxtrininsertinside\else##2\fi

```

```

7673 }%
7674 \renewcommand*{\glxtrfullplformat}[2]{%
7675   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7676   \ifglxtrininsertinside\else##2\fi
7677 }%
7678 \renewcommand*{\Glsxtrfullformat}[2]{%
7679   \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7680   \ifglxtrininsertinside\else##2\fi
7681 }%
7682 \renewcommand*{\Glsxtrfullplformat}[2]{%
7683   \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7684   \ifglxtrininsertinside\else##2\fi
7685 }%
7686 }

```

long-sc Backward compatibility:

```

7687 \@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.

```

7688 \newabbreviationstyle{long-noshort-sc-desc}%
7689 {%
7690   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7691 }%
7692 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7693 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7694 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
7695 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvscfont{##1}}%
7696 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
7697 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7698 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7699   \glsslongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7700   \ifglxtrininsertinside \else##2\fi
7701 }%
7702 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7703   \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7704   \ifglxtrininsertinside \else##2\fi
7705 }%
7706 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7707   \glsslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7708   \ifglxtrininsertinside \else##2\fi
7709 }%
7710 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7711   \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7712   \ifglxtrininsertinside \else##2\fi
7713 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7714 \renewcommand*{\glxtrinlinefullformat}[2]{%
7715   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7716   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7717   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7718 }%
7719 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7720   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7721   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7722   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7723 }%
7724 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7725   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7726   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7727   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7728 }%
7729 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7730   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7731   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7732   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7733 }%

```

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.

```

7734 \renewcommand*{\glxtrfullformat}[2]{%
7735   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7736   \ifglxtrininsertinside\else##2\fi
7737 }%
7738 \renewcommand*{\glxtrfullplformat}[2]{%
7739   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7740   \ifglxtrininsertinside\else##2\fi
7741 }%
7742 \renewcommand*{\Glsxtrfullformat}[2]{%
7743   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7744   \ifglxtrininsertinside\else##2\fi
7745 }%
7746 \renewcommand*{\Glsxtrfullplformat}[2]{%
7747   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7748   \ifglxtrininsertinside\else##2\fi
7749 }%
7750 }

```

long-desc-sc Backward compatibility:

```

7751 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7752 \newabbreviationstyle{short-sc-footnote}%
7753 {%
7754   \renewcommand*{\CustomAbbreviationFields}{%

```



```

7755 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7756 sort={\the\glsshorttok},
7757 description={\the\glslongtok},%
7758 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7759 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7760 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7761 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7762 \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
7763 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7764 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.

```

7765 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7766 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7767 \glsattribute{\the\glslabeltok}{regular}%
7768 {%
7769 \glssetattribute{\the\glslabeltok}{regular}{false}%
7770 }%
7771 {}%
7772 }%
7773 }%
7774 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7775 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7776 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7777 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7778 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7779 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7780 \renewcommand*{\glsxtrfullformat}[2]{%
7781 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7782 \ifglsxtrininsertinside\else##2\fi
7783 \protect\glsxtrabbrvfootnote{##1}%
7784 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7785 }%
7786 \renewcommand*{\glsxtrfullplformat}[2]{%
7787 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7788 \ifglsxtrininsertinside\else##2\fi
7789 \protect\glsxtrabbrvfootnote{##1}%
7790 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7791 }%
7792 \renewcommand*{\Glsxtrfullformat}[2]{%
7793 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7794 \ifglsxtrininsertinside\else##2\fi
7795 \protect\glsxtrabbrvfootnote{##1}%
7796 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7797 }%
7798 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

7799 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7800 \ifglxtrinsertinside\else##2\fi
7801 \protect\glxtrabbrvfootnote{##1}%
7802 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7803 }%

```

The first use full form and the inline full form use the short (long) style.

```

7804 \renewcommand*{\glxtrinlinefullformat}[2]{%
7805 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7806 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7807 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7808 }%
7809 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7810 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7811 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7812 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7813 }%
7814 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7815 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7816 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7817 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7818 }%
7819 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7820 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7821 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7822 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7823 }%
7824 }

```

footnote-sc Backward compatibility:

```

7825 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}

```

sc-postfootnote

```

7826 \newabbreviationstyle{short-sc-postfootnote}%
7827 {%
7828 \renewcommand*{\CustomAbbreviationFields}{%
7829 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7830 sort={\the\glsshorttok},
7831 description={\the\glslongtok},%
7832 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7833 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7834 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.

```

7835 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7836 \csdef{glxtrpostlink\glscategorylabel}{%
7837 \glxtrifwasfirstuse
7838 {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

7839      \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}}%
7840      {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
7841      }%
7842      {}%
7843      }%
7844      \glshasattribute{\the\glslabeltok}{regular}%
7845      {%
7846      \glissetattribute{\the\glslabeltok}{regular}{false}%
7847      }%
7848      {}%
7849      }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7850 \renewcommand*{\glxtrsetupfulldefs}{%
7851   \let\glxtrifwasfirstuse\@secondoftwo
7852 }%
7853 }%
7854 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7855 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7856 \renewcommand*{\glxtrabbrvfont}[1]{\glxtrabbrvscfont{##1}}%
7857 \renewcommand*{\glxtrfirstabbrvfont}[1]{\glxtrfirstabbrvscfont{##1}}%
7858 \renewcommand*{\glxtrfirstlongfont}[1]{\glxtrfirstlongfootnotefont{##1}}%
7859 \renewcommand*{\glxtrlongfont}[1]{\glxtrlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7860 \renewcommand*{\glxtrfullformat}[2]{%
7861   \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7862   \ifglxtrininsertinside\else##2\fi
7863 }%
7864 \renewcommand*{\glxtrfullplformat}[2]{%
7865   \glxtrfirstabbrvscfont{\glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7866   \ifglxtrininsertinside\else##2\fi
7867 }%
7868 \renewcommand*{\Glxtrfullformat}[2]{%
7869   \glxtrfirstabbrvscfont{\Glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7870   \ifglxtrininsertinside\else##2\fi
7871 }%
7872 \renewcommand*{\Glxtrfullplformat}[2]{%
7873   \glxtrfirstabbrvscfont{\Glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7874   \ifglxtrininsertinside\else##2\fi
7875 }%

```

The first use full form and the inline full form use the short (long) style.

```

7876 \renewcommand*{\glxtrininlinefullformat}[2]{%
7877   \glxtrfirstabbrvscfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
7878   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7879 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7880 }%
7881 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7882 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7883 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7884 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7885 }%
7886 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7887 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7888 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7889 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7890 }%
7891 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7892 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7893 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7894 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7895 }%
7896 }

```

postfootnote-sc Backward compatibility:

```
7897 \@glsxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}
```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

```

\glsxtrsmfont Maintained for backward compatibility.
7898 \newcommand*{\glsxtrsmfont}[1]{\textsmaller{#1}}

\glsabbrvsmfont Added for consistent naming.
7899 \newcommand*{\glsabbrvsmfont}{\glsxtrsmfont}

sxtrfirstsmfont Maintained for backward compatibility.
7900 \newcommand*{\glsxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}

irstabbrvsmfont Added for consistent naming.
7901 \newcommand*{\glsfirstabbrvsmfont}{\glsxtrfirstsmfont}

and for the default short form suffix:

\glsxtrsmsuffix
7902 \newcommand*{\glsxtrsmsuffix}{\glsxtrabbrvpluralsuffix}

long-short-sm
7903 \newabbreviationstyle{long-short-sm}%
7904 {%
7905 \renewcommand*{\CustomAbbreviationFields}{%

```

```

7906   name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7907   sort={\the\glsshorttok},
7908   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7909     \protect\glsxtrfullsep{\the\glslabeltok}%
7910     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7911   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7912     \protect\glsxtrfullsep{\the\glslabeltok}%
7913     \glsxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7914   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
7915   description={\the\glslongtok}}%
7916 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7917   \glsattribute{\the\glslabeltok}{regular}%
7918   {%
7919     \glssetattribute{\the\glslabeltok}{regular}{false}%
7920   }%
7921   {}%
7922 }%
7923 }%
7924 {%
7925   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
7926   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
7927   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%

```

Use the default long fonts.

```

7928   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7929   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7930   \renewcommand*\glsxtrfullformat[2]{%
7931     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7932     \ifglsxtrininsertinside\else##2\fi
7933     \glsxtrfullsep{##1}%
7934     \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7935   }%
7936   \renewcommand*\glsxtrfullplformat[2]{%
7937     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7938     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7939     \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7940   }%
7941   \renewcommand*\Glsxtrfullformat[2]{%
7942     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7943     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7944     \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7945   }%
7946   \renewcommand*\Glsxtrfullplformat[2]{%
7947     \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7948     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7949     \glsxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7950   }%
7951 }

```

g-short-sm-desc

```
7952 \newabbreviationstyle{long-short-sm-desc}%
7953 {%
7954   \renewcommand*{\CustomAbbreviationFields}{%
7955     name={\glxtrlongshortdescname},
7956     sort={\glxtrlongshortdescsort},%
7957     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7958       \protect\glxtrfullsep{\the\glslabeltok}%
7959       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7960     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7961       \protect\glxtrfullsep{\the\glslabeltok}%
7962       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7963     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7964     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7965   }%
```

Unset the regular attribute if it has been set.

```
7966   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7967     \glshasattribute{\the\glslabeltok}{regular}%
7968     {%
7969       \glissetattribute{\the\glslabeltok}{regular}{false}%
7970     }%
7971   }%
7972 }%
7973 }%
7974 {%
```

As long-short-sm style:

```
7975   \GlsXtrUseAbbrStyleFmts{long-short-sm}%
7976 }
```

short-sm-long Now the short (long) version

```
7977 \newabbreviationstyle{short-sm-long}%
7978 {%
7979   \renewcommand*{\CustomAbbreviationFields}{%
7980     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7981     sort={\the\glsshorttok},
7982     description={\the\glslongtok},%
7983     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7984       \protect\glxtrfullsep{\the\glslabeltok}%
7985       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7986     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7987       \protect\glxtrfullsep{\the\glslabeltok}%
7988       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7989     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7990   }%
```

Unset the regular attribute if it has been set.

```
7990   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7991     \glshasattribute{\the\glslabeltok}{regular}%
7992     {%
```

```

7993     \glsetattribute{\the\glslabeltok}{regular}{false}%
7994 }%
7995 {}%
7996 }%
7997 }%
7998 {%
7999 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8000 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8001 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8002 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8003 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8004 \renewcommand*\glsxtrfullformat[2]{%
8005   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8006   \ifglsxtrininsertinside\else##2\fi
8007   \glsxtrfullsep{##1}%
8008   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8009 }%
8010 \renewcommand*\glsxtrfullplformat[2]{%
8011   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8012   \ifglsxtrininsertinside\else##2\fi
8013   \glsxtrfullsep{##1}%
8014   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8015 }%
8016 \renewcommand*\Glsxtrfullformat[2]{%
8017   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8018   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8019   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8020 }%
8021 \renewcommand*\Glsxtrfullplformat[2]{%
8022   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8023   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8024   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8025 }%
8026 }

```

rt-sm-long-desc As before but user provides description

```

8027 \newabbreviationstyle{short-sm-long-desc}%
8028 {%
8029 \renewcommand*\CustomAbbreviationFields{%
8030   name={\glsxtrshortlongdescname},
8031   sort={\glsxtrshortlongdescsort},
8032   first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8033     \protect\glsxtrfullsep{\the\glslabeltok}%
8034     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8035   firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8036     \protect\glsxtrfullsep{\the\glslabeltok}%
8037     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8038   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%

```

```

8039 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}%
8040 }%

```

Unset the regular attribute if it has been set.

```

8041 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8042   \glshasattribute{\the\glslabeltok}{regular}%
8043   {%
8044     \glissetattribute{\the\glslabeltok}{regular}{false}%
8045   }%
8046   {}%
8047 }%
8048 }%
8049 {%

```

As short-sm-long style:

```

8050 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8051 }

```

short-sm

```

8052 \newabbreviationstyle{short-sm}%
8053 {%
8054   \renewcommand*{\CustomAbbreviationFields}{%
8055     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8056     sort={\the\glsshorttok},
8057     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8058     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8059     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8060     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8061     description={\the\glslongtok}}%
8062   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8063     \glissetattribute{\the\glslabeltok}{regular}{true}}%
8064 }%
8065 {%
8066   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8067   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8068   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
8069   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8070   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8071 \renewcommand*{\glxtrinlinefullformat}[2]{%
8072   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8073   \ifglxtrininsertinside##2\fi}%
8074   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8075   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8076 }%
8077 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8078   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
8079   \ifglxtrininsertinside##2\fi}%
8080   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%

```



```

8081 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8082 }%

8083 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8084 \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}%
8085 \ifglsxtrinsertinside##2\fi}%
8086 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8087 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8088 }%
8089 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8090 \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}%
8091 \ifglsxtrinsertinside##2\fi}%
8092 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8093 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8094 }%

```

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.

```

8095 \renewcommand*{\glsxtrfullformat}[2]{%
8096 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
8097 \ifglsxtrinsertinside\else##2\fi
8098 }%
8099 \renewcommand*{\glsxtrfullplformat}[2]{%
8100 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
8101 \ifglsxtrinsertinside\else##2\fi
8102 }%
8103 \renewcommand*{\Glsxtrfullformat}[2]{%
8104 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
8105 \ifglsxtrinsertinside\else##2\fi
8106 }%
8107 \renewcommand*{\Glsxtrfullplformat}[2]{%
8108 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
8109 \ifglsxtrinsertinside\else##2\fi
8110 }%
8111 }

```

short-sm-nolong

```

8112 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

8113 \newabbreviationstyle{short-sm-desc}{%
8114 {%
8115 \renewcommand*{\CustomAbbreviationFields}{%
8116 name={\glsxtrshortdescname},
8117 sort={\the\glsshorttok},
8118 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8119 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8120 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8121 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},

```

```

8122     description={\the\glslongtok}}%
8123 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8124     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8125 }%
8126 {%
8127 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8128 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8129 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
8130 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8131 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8132 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8133     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8134     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8135 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8136 }%
8137 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8138     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8139     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8140 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8141 }%
8142 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8143     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8144     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8145 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8146 }%
8147 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8148     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8149     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8150 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8151 }%

```

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.

```

8152 \renewcommand*{\glsxtrfullformat}[2]{%
8153     \glsfirstabbrvsmfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8154     \ifglsxtrininsertinside\else##2\fi
8155 }%
8156 \renewcommand*{\glsxtrfullplformat}[2]{%
8157     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8158     \ifglsxtrininsertinside\else##2\fi
8159 }%
8160 \renewcommand*{\Glsxtrfullformat}[2]{%
8161     \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8162     \ifglsxtrininsertinside\else##2\fi
8163 }%
8164 \renewcommand*{\Glsxtrfullplformat}[2]{%
8165     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8166     \ifglsxtrininsertinside\else##2\fi

```

```

8167 }%
8168 }

```

-sm-nolong-desc

```

8169 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8170 \newabbreviationstyle{nolong-short-sm}%
8171 {%
8172   \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8173 }%
8174 {%
8175   \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8176 \renewcommand*{\glxtrinlinefullformat}[2]{%
8177   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8178     \ifglxtrinsertinside##2\fi}%
8179   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8180   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8181 }%
8182 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8183   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8184     \ifglxtrinsertinside##2\fi}%
8185   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8186   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8187 }%
8188 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8189   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8190     \ifglxtrinsertinside##2\fi}%
8191   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8192   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8193 }%
8194 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8195   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8196     \ifglxtrinsertinside##2\fi}%
8197   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8198   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8199 }%
8200 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8201 \newabbreviationstyle{long-noshort-sm}%
8202 {%
8203   \renewcommand*{\CustomAbbreviationFields}{%
8204     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8205     sort={\the\glsshorttok},
8206     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},

```

```

8207     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8208     text={\protect\glslongdefaultfont{\the\glslongtok}},
8209     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8210     description={\the\glslongtok}%
8211 }%
8212 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8213   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8214 }%
8215 {%
8216   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8217   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8218   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8219   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8220   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8221   \renewcommand*\glsxtrsubsequentfmt}[2]{%
8222     \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8223     \ifglsxtrininsertinside \else##2\fi
8224   }%
8225   \renewcommand*\glsxtrsubsequentplfmt}[2]{%
8226     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8227     \ifglsxtrininsertinside \else##2\fi
8228   }%
8229   \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8230     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8231     \ifglsxtrininsertinside \else##2\fi
8232   }%
8233   \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8234     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8235     \ifglsxtrininsertinside \else##2\fi
8236   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8237   \renewcommand*\glsxtrinlinefullformat}[2]{%
8238     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8239     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8240     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8241   }%
8242   \renewcommand*\glsxtrinlinefullplformat}[2]{%
8243     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8244     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8245     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8246   }%
8247   \renewcommand*\Glsxtrinlinefullformat}[2]{%
8248     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8249     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8250     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8251   }%
8252   \renewcommand*\Glsxtrinlinefullplformat}[2]{%

```

```

8253 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8254 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8255 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8256 }%

```

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.

```

8257 \renewcommand*{\glxtrfullformat}[2]{%
8258 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8259 \ifglxtrinsertinside\else##2\fi
8260 }%
8261 \renewcommand*{\glxtrfullplformat}[2]{%
8262 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8263 \ifglxtrinsertinside\else##2\fi
8264 }%
8265 \renewcommand*{\Glsxtrfullformat}[2]{%
8266 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8267 \ifglxtrinsertinside\else##2\fi
8268 }%
8269 \renewcommand*{\Glsxtrfullplformat}[2]{%
8270 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8271 \ifglxtrinsertinside\else##2\fi
8272 }%
8273 }

```

long-sm Backward compatibility:

```

8274 \@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`.

```

8275 \newabbreviationstyle{long-noshort-sm-desc}%
8276 {%
8277 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8278 }%
8279 {%
8280 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8281 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8282 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8283 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8284 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8285 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8286 \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8287 \ifglxtrinsertinside \else##2\fi
8288 }%
8289 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8290 \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8291 \ifglxtrinsertinside \else##2\fi
8292 }%

```

```

8293 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8294   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8295   \ifglxtrinsertinside \else##2\fi
8296 }%
8297 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8298   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8299   \ifglxtrinsertinside \else##2\fi
8300 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8301 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8302   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8303   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8304   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8305 }%
8306 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8307   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8308   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8309   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8310 }%
8311 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8312   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8313   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8314   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8315 }%
8316 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8317   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8318   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8319   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8320 }%

```

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.

```

8321 \renewcommand*{\Glsxtrfullformat}[2]{%
8322   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8323   \ifglxtrinsertinside\else##2\fi
8324 }%
8325 \renewcommand*{\Glsxtrfullplformat}[2]{%
8326   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8327   \ifglxtrinsertinside\else##2\fi
8328 }%
8329 \renewcommand*{\Glsxtrfullformat}[2]{%
8330   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8331   \ifglxtrinsertinside\else##2\fi
8332 }%
8333 \renewcommand*{\Glsxtrfullplformat}[2]{%
8334   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8335   \ifglxtrinsertinside\else##2\fi
8336 }%
8337 }

```

long-desc-sm Backward compatibility:

```
8338 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}
```

ort-sm-footnote

```
8339 \newabbreviationstyle{short-sm-footnote}%
8340 {%
8341   \renewcommand*{\CustomAbbreviationFields}{%
8342     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8343     sort={\the\glsshorttok},
8344     description={\the\glslongtok},%
8345     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8346       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8347       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8348     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8349       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8350       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8351     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
8352   }
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
8352   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8353     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8354     \glsattribute{\the\glslabeltok}{regular}%
8355     {%
8356       \glssetattribute{\the\glslabeltok}{regular}{false}%
8357     }%
8358   }%
8359 }%
8360 }%
8361 {%
8362   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8363   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8364   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
8365   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8366   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
8367 }
```

The full format displays the short form followed by the long form as a footnote.

```
8367 \renewcommand*\glxtrfullformat[2]{%
8368   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8369   \ifglxtrininsertinside\else##2\fi
8370   \protect\glxtrabbrvfootnote{##1}%
8371   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8372 }%
8373 \renewcommand*\glxtrfullplformat[2]{%
8374   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8375   \ifglxtrininsertinside\else##2\fi
8376   \protect\glxtrabbrvfootnote{##1}%
8377   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8378 }%
8379 \renewcommand*\GlsXtrfullformat[2]{%
8380   \glxtrfullformat{1}{2}%
8381   \glxtrfullplformat{1}{2}%
8382 }
```

```

8380 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8381 \ifglxtrinsertinside\else##2\fi
8382 \protect\glxtrabbrvfootnote{##1}%
8383 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8384 }%
8385 \renewcommand*{\Glsxtrfullplformat}[2]{%
8386 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8387 \ifglxtrinsertinside\else##2\fi
8388 \protect\glxtrabbrvfootnote{##1}%
8389 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8390 }%

```

The first use full form and the inline full form use the short (long) style.

```

8391 \renewcommand*{\glxtrinlinefullformat}[2]{%
8392 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8393 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8394 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8395 }%
8396 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8397 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8398 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8399 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8400 }%
8401 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8402 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8403 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8404 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8405 }%
8406 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8407 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8408 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8409 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8410 }%
8411 }

```

footnote-sm Backward compatibility:

```

8412 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8413 \newabbreviationstyle{short-sm-postfootnote}%
8414 {%
8415 \renewcommand*{\CustomAbbreviationFields}{%
8416 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8417 sort={\the\glsshorttok},
8418 description={\the\glslongtok},%
8419 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8420 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8421 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.

```
8422 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8423   \csdef{glxstrpostlink\glscategorylabel}{%
8424     \glxstrifwasfirstuse
8425     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
8426       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8427       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
8428     }%
8429   }%
8430 }%
8431 \glshasattribute{\the\glslabelltok}{regular}%
8432 {%
8433   \glissetattribute{\the\glslabelltok}{regular}{false}%
8434 }%
8435 {}%
8436 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
8437 \renewcommand*{\glxtrsetupfulldefs}{%
8438   \let\glxstrifwasfirstuse\@secondoftwo
8439 }%
8440}%
8441{%
8442 \renewcommand*\glxabbrvfont[1]{\glxabbrvsmfont{##1}}%
8443 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8444 \renewcommand*\abbrvpluralsuffix{\protect\glxtrsmsuffix}%
8445 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8446 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
8447 \renewcommand*{\glxtrfullformat}[2]{%
8448   \glsfirstabbrvsmfont{\glxaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8449   \ifglxtrinsertinside\else##2\fi
8450 }%
8451 \renewcommand*{\glxtrfullplformat}[2]{%
8452   \glsfirstabbrvsmfont{\glxaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8453   \ifglxtrinsertinside\else##2\fi
8454 }%
8455 \renewcommand*{\Glsxtrfullformat}[2]{%
8456   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8457   \ifglxtrinsertinside\else##2\fi
8458 }%
8459 \renewcommand*{\Glsxtrfullplformat}[2]{%
8460   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8461   \ifglxtrinsertinside\else##2\fi
```

```

8462 }%
      The first use full form and the inline full form use the short (long) style.
8463 \renewcommand*{\glxtrinlinefullformat}[2]{%
8464   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8465   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8466   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8467 }%
8468 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8469   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8470   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8471   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8472 }%
8473 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8474   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8475   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8476   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8477 }%
8478 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8479   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8480   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8481   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8482 }%
8483 }

```

postfootnote-sm Backward compatibility:

```

8484 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

```

\glsabbrvemfont
8485 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%

irstabbrvemfont
8486 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%

      The default short form suffix:

\glxtremsuffix
8487 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

firstlongemfont Only used by the “long-em” styles.
8488 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%

\glslongemfont Only used by the “long-em” styles.
8489 \newcommand*{\glslongemfont}[1]{\emph{##1}}%

```

long-short-em The long form is just set in the default long font.

```
8490 \newabbreviationstyle{long-short-em}%
8491 {%
8492   \renewcommand*{\CustomAbbreviationFields}{%
8493     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8494     sort={\the\glsshorttok},
8495     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8496       \protect\glsxtrfullsep{\the\glslabeltok}%
8497       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8498     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8499       \protect\glsxtrfullsep{\the\glslabeltok}%
8500       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8501     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8502     description={\the\glslongtok}}%
8503   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8504     \glsattribute{\the\glslabeltok}{regular}%
8505     {%
8506       \glssetattribute{\the\glslabeltok}{regular}{false}%
8507     }%
8508   }%
8509 }%
8510 }%
8511 {%
8512   \renewcommand*{\glsabbrvfont[1]{\glsabbrvemfont{##1}}}%
8513   \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}}%
8514   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```
8515 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8516 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```
8517 \renewcommand*{\glsxtrfullformat}[2]{%
8518   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8519   \ifglsxtrininsertinside\else##2\fi
8520   \glsxtrfullsep{##1}%
8521   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8522 }%
8523 \renewcommand*{\glsxtrfullplformat}[2]{%
8524   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8525   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8526   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8527 }%
8528 \renewcommand*{\Glsxtrfullformat}[2]{%
8529   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8530   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8531   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8532 }%
8533 \renewcommand*{\Glsxtrfullplformat}[2]{%
8534   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%

```

```

8535 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8536 \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8537 }%
8538 }

```

g-short-em-desc

```

8539 \newabbreviationstyle{long-short-em-desc}%
8540 {%
8541 \renewcommand*{\CustomAbbreviationFields}{%
8542 name={\glxtrlongshortdescname},
8543 sort={\glxtrlongshortdescsort},%
8544 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8545 \protect\glxtrfullsep{\the\glslabeltok}%
8546 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8547 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8548 \protect\glxtrfullsep{\the\glslabeltok}%
8549 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8550 text={\protect\glssabbrvemfont{\the\glsshorttok}},%
8551 plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%
8552 }%

```

Unset the regular attribute if it has been set.

```

8553 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8554 \glshasattribute{\the\glslabeltok}{regular}%
8555 {%
8556 \glissetattribute{\the\glslabeltok}{regular}{false}%
8557 }%
8558 {}%
8559 }%
8560 }%
8561 {%

```

As long-short-em style:

```

8562 \GlsXtrUseAbbrStyleFmts{long-short-em}%
8563 }

```

ong-em-short-em

```

8564 \newabbreviationstyle{long-em-short-em}%
8565 {%
8566 \renewcommand*{\CustomAbbreviationFields}{%
8567 name={\protect\glssabbrvemfont{\the\glsshorttok}},
8568 sort={\the\glsshorttok},
8569 first={\protect\glsfirstlongemfont{\the\glslongtok}%
8570 \protect\glxtrfullsep{\the\glslabeltok}%
8571 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8572 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8573 \protect\glxtrfullsep{\the\glslabeltok}%
8574 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

```

```

8575 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},%
8576 description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8577 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8578   \glsattribute{\the\glslabeltok}{regular}%
8579   {%
8580     \glssetattribute{\the\glslabeltok}{regular}{false}%
8581   }%
8582   {}}%
8583 }%
8584 }%
8585 {%
8586 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8587 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8588 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8589 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8590 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8591 \renewcommand*{\glsxtrfullformat}[2]{%
8592   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8593   \ifglsxtrininsertinside\else##2\fi
8594   \glsxtrfullsep{##1}%
8595   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8596 }%
8597 \renewcommand*{\glsxtrfullplformat}[2]{%
8598   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8599   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8600   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8601 }%
8602 \renewcommand*{\Glsxtrfullformat}[2]{%
8603   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8604   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8605   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8606 }%
8607 \renewcommand*{\Glsxtrfullplformat}[2]{%
8608   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8609   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8610   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8611 }%
8612 }

```

m-short-em-desc

```

8613 \newabbreviationstyle{long-em-short-em-desc}%
8614 {%
8615 \renewcommand*{\CustomAbbreviationFields}{%
8616   name={\glsxtrlongshortdescname},
8617   sort={\glsxtrlongshortdescsort},%
8618   first={\protect\glsfirstlongemfont{\the\glslongtok}}%

```

```

8619 \protect\glxtrfullsep{\the\glslabeltok}%
8620 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8621 firstplural={\protect\glsfirstlongemfont{\the\gslongpltok}%
8622 \protect\glxtrfullsep{\the\glslabeltok}%
8623 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8624 text={\protect\glsabbrvemfont{\the\glsshorttok}}},%
8625 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8626 }%

```

Unset the regular attribute if it has been set.

```

8627 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8628 \glshasattribute{\the\glslabeltok}{regular}%
8629 {%
8630 \glissetattribute{\the\glslabeltok}{regular}{false}%
8631 }%
8632 {}%
8633 }%
8634 }%
8635 {%
8636 \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8637 }

```

short-em-long Now the short (long) version

```

8638 \newabbreviationstyle{short-em-long}%
8639 {%
8640 \renewcommand*{\CustomAbbreviationFields}{%
8641 name={\protect\glsabbrvemfont{\the\glsshorttok}},
8642 sort={\the\glsshorttok},
8643 description={\the\gslongtok},%
8644 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8645 \protect\glxtrfullsep{\the\glslabeltok}%
8646 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongtok}}},%
8647 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8648 \protect\glxtrfullsep{\the\glslabeltok}%
8649 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\gslongpltok}}},%
8650 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8651 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8652 \glshasattribute{\the\glslabeltok}{regular}%
8653 {%
8654 \glissetattribute{\the\glslabeltok}{regular}{false}%
8655 }%
8656 {}%
8657 }%
8658 }%
8659 {%

```

Mostly as short-long style:

```

8660 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%

```

```

8661 \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8662 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8663 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8664 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8665 \renewcommand*\glsxtrfullformat}[2]{%
8666   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8667   \ifglsxtrininsertinside\else##2\fi
8668   \glsxtrfullsep{##1}%
8669   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8670 }%
8671 \renewcommand*\glsxtrfullplformat}[2]{%
8672   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8673   \ifglsxtrininsertinside\else##2\fi
8674   \glsxtrfullsep{##1}%
8675   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8676 }%
8677 \renewcommand*\Glsxtrfullformat}[2]{%
8678   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8679   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8680   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8681 }%
8682 \renewcommand*\Glsxtrfullplformat}[2]{%
8683   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8684   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8685   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8686 }%
8687 }

```

rt-em-long-desc As before but user provides description

```

8688 \newabbreviationstyle{short-em-long-desc}%
8689 {%
8690   \renewcommand*\CustomAbbreviationFields{%
8691     name={\glsxtrshortlongdescname},
8692     sort={\glsxtrshortlongdescsort},
8693     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8694       \protect\glsxtrfullsep{\the\glslabeltok}%
8695       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8696     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8697       \protect\glsxtrfullsep{\the\glslabeltok}%
8698       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8699     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8700     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8701   }%

```

Unset the regular attribute if it has been set.

```

8702 \renewcommand*\GlsXtrPostNewAbbreviation{%
8703   \glshasattribute{\the\glslabeltok}{regular}%
8704   {%

```

```

8705     \glsetattribute{\the\glslabeltok}{regular}{false}%
8706   }%
8707   {}%
8708 }%
8709}%
8710{%
8711  \GlsXtrUseAbbrStyleFmts{short-em-long}%
8712}

```

hort-em-long-em

```

8713 \newabbreviationstyle{short-em-long-em}%
8714 {%
  \glslongemfont is used in the description since \glsgdesc doesn't set the style.
8715  \renewcommand*{\CustomAbbreviationFields}{%
8716    name={\protect\glsabbrvemfont{\the\glsshorttok}},
8717    sort={\the\glsshorttok},
8718    description={\protect\glslongemfont{\the\glslongtok}},%
8719    first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8720    \protect\glsxtrfullsep{\the\glslabeltok}%
8721    \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8722    firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8723    \protect\glsxtrfullsep{\the\glslabeltok}%
8724    \glsxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8725    plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

  Unset the regular attribute if it has been set.
8726  \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8727    \glshasattribute{\the\glslabeltok}{regular}%
8728    {%
8729      \glsetattribute{\the\glslabeltok}{regular}{false}%
8730    }%
8731    {}%
8732  }%
8733}%
8734{%
8735  \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8736  \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8737  \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8738  \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8739  \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

  The first use full form and the inline full form are the same for this style.
8740  \renewcommand*{\glsxtrfullformat}[2]{%
8741    \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8742    \ifglsxtrininsertinside\else##2\fi
8743    \glsxtrfullsep{##1}%
8744    \glsxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8745  }%
8746  \renewcommand*{\glsxtrfullplformat}[2]{%

```



```

8747 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8748 \ifglxtrinsertinside\else##2\fi
8749 \glxtrfullsep{##1}%
8750 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8751 }%
8752 \renewcommand*{\Glsxtrfullformat}[2]{%
8753 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8754 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8755 \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8756 }%
8757 \renewcommand*{\Glsxtrfullplformat}[2]{%
8758 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8759 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8760 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8761 }%
8762 }

```

em-long-em-desc

```

8763 \newabbreviationstyle{short-em-long-em-desc}%
8764 {%
8765 \renewcommand*{\CustomAbbreviationFields}{%
8766 name={\glxtrshortlongdescname},%
8767 sort={\glxtrshortlongdescsort},%
8768 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8769 \protect\glxtrfullsep{\the\glslabeltok}%
8770 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8771 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8772 \protect\glxtrfullsep{\the\glslabeltok}%
8773 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8774 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8775 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8776 }%

```

Unset the regular attribute if it has been set.

```

8777 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8778 \glshasattribute{\the\glslabeltok}{regular}%
8779 {%
8780 \glissetattribute{\the\glslabeltok}{regular}{false}%
8781 }%
8782 {}%
8783 }%
8784 }%
8785 {%
8786 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8787 }

```

short-em

```

8788 \newabbreviationstyle{short-em}%
8789 {%
8790 \renewcommand*{\CustomAbbreviationFields}{%

```

```

8791     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8792     sort={\the\glsshorttok},
8793     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8794     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8795     text={\protect\glsabbrvemfont{\the\glsshorttok}},
8796     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8797     description={\the\glslongtok}}%
8798 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8799   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8800 }%
8801 {%
8802 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8803 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8804 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8805 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8806 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8807 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8808   \protect\glsfirstabbrvemfont{\glsaccessshort{##1}}%
8809   \ifglsxtrininsertinside##2\fi}%
8810 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8811 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8812 }%
8813 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8814   \protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}%
8815   \ifglsxtrininsertinside##2\fi}%
8816 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8817 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8818 }%
8819 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8820   \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}}%
8821   \ifglsxtrininsertinside##2\fi}%
8822 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8823 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8824 }%
8825 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8826   \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}%
8827   \ifglsxtrininsertinside##2\fi}%
8828 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8829 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8830 }%

```

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.

```

8831 \renewcommand*{\glsxtrfullformat}[2]{%
8832   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8833   \ifglsxtrininsertinside\else##2\fi
8834 }%

```

```

8835 \renewcommand*{\glxtrfullplformat}[2]{%
8836   \glsfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8837   \ifglxtrininsertinside\else##2\fi
8838 }%
8839 \renewcommand*{\Glsxtrfullformat}[2]{%
8840   \glsfirstabbrvemfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8841   \ifglxtrininsertinside\else##2\fi
8842 }%
8843 \renewcommand*{\Glsxtrfullplformat}[2]{%
8844   \glsfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8845   \ifglxtrininsertinside\else##2\fi
8846 }%
8847 }

```

short-em-nolong

```

8848 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8849 \newabbreviationstyle{short-em-desc}%
8850 {%
8851   \renewcommand*{\CustomAbbreviationFields}{%
8852     name={\glxtrshortdescname},
8853     sort={\the\glsshorttok},
8854     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8855     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8856     text={\protect\glsabbrvemfont{\the\glsshorttok}},
8857     plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8858     description={\the\glslongtok}}%
8859   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8860     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
8861 }%
8862 {%
8863   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
8864   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8865   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8866   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8867   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8868 \renewcommand*{\glxtrininlinefullformat}[2]{%
8869   \glsfirstabbrvemfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8870   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8871   \glxtrparen{\glsfirstlongdefaultfont{\glaccesslong{##1}}}%
8872 }%
8873 \renewcommand*{\glxtrininlinefullplformat}[2]{%
8874   \glsfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8875   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8876   \glxtrparen{\glsfirstlongdefaultfont{\glaccesslongpl{##1}}}%
8877 }%
8878 \renewcommand*{\Glsxtrininlinefullformat}[2]{%

```

```

8879 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8880 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8881 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8882 }%
8883 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8884 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8885 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8886 \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8887 }%

```

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.

```

8888 \renewcommand*{\glxtrfullformat}[2]{%
8889 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8890 \ifglxtrinsertinside\else##2\fi
8891 }%
8892 \renewcommand*{\glxtrfullplformat}[2]{%
8893 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8894 \ifglxtrinsertinside\else##2\fi
8895 }%
8896 \renewcommand*{\Glsxtrfullformat}[2]{%
8897 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8898 \ifglxtrinsertinside\else##2\fi
8899 }%
8900 \renewcommand*{\Glsxtrfullplformat}[2]{%
8901 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8902 \ifglxtrinsertinside\else##2\fi
8903 }%
8904 }

```

-em-nolong-desc

```

8905 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8906 \newabbreviationstyle{nolong-short-em}%
8907 {%
8908 \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8909 }%
8910 {%
8911 \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8912 \renewcommand*{\glxtrinlinefullformat}[2]{%
8913 \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8914 \ifglxtrinsertinside##2\fi}%
8915 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8916 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8917 }%
8918 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8919 \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%

```

```

8920     \ifglxtrinsertinside##2\fi}%
8921     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8922     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8923 }%
8924 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8925     \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8926         \ifglxtrinsertinside##2\fi}%
8927     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8928     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8929 }%
8930 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8931     \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8932         \ifglxtrinsertinside##2\fi}%
8933     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8934     \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8935 }%
8936 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8937 \newabbreviationstyle{long-noshort-em}%
8938 {%
8939     \renewcommand*{\CustomAbbreviationFields}{%
8940         name={\protect\glsabbrvemfont{\the\glsshorttok}},
8941         sort={\the\glsshorttok},
8942         first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8943         firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8944         text={\protect\glslongdefaultfont{\the\glslongtok}},
8945         plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8946         description={\the\glslongtok}%
8947     }%
8948     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8949         \glssetattribute{\the\glslabeltok}{regular}{true}}%
8950 }%
8951 {%
8952     \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8953     \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8954     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8955     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8956     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8957 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8958     \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8959     \ifglxtrinsertinside \else##2\fi
8960 }%
8961 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8962     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8963     \ifglxtrinsertinside \else##2\fi
8964 }%
8965 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%

```

```

8966 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8967 \ifglxtrinsertinside \else##2\fi
8968 }%
8969 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8970 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8971 \ifglxtrinsertinside \else##2\fi
8972 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8973 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8974 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8975 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8976 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8977 }%
8978 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8979 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8980 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8981 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8982 }%
8983 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8984 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8985 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8986 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8987 }%
8988 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8989 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8990 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8991 \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8992 }%

```

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.

```

8993 \renewcommand*{\glsxtrfullformat}[2]{%
8994 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8995 \ifglxtrinsertinside\else##2\fi
8996 }%
8997 \renewcommand*{\glsxtrfullplformat}[2]{%
8998 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8999 \ifglxtrinsertinside\else##2\fi
9000 }%
9001 \renewcommand*{\Glsxtrfullformat}[2]{%
9002 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9003 \ifglxtrinsertinside\else##2\fi
9004 }%
9005 \renewcommand*{\Glsxtrfullplformat}[2]{%
9006 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9007 \ifglxtrinsertinside\else##2\fi
9008 }%
9009 }

```

long-em Backward compatibility:

```
9010 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like `\glsshort`.

```
9011 \newabbreviationstyle{long-em-noshort-em}%
9012 {%
9013   \renewcommand*{\CustomAbbreviationFields}{%
9014     name={\protect\glsabbrvemfont{\the\glsshorttok}},
9015     sort={\the\glsshorttok},
9016     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9017     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9018     text={\protect\glslongemfont{\the\glslongtok}},
9019     plural={\protect\glslongemfont{\the\glslongpltok}},%
9020     description={\protect\glslongemfont{\the\glslongtok}}%
9021   }%
9022   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9023     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9024 }%
9025 {%
9026   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9027   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9028   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9029   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
9030   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9031 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9032   \glslongemfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9033   \ifglxtrininsertinside \else##2\fi
9034 }%
9035 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9036   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9037   \ifglxtrininsertinside \else##2\fi
9038 }%
9039 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9040   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9041   \ifglxtrininsertinside \else##2\fi
9042 }%
9043 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9044   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9045   \ifglxtrininsertinside \else##2\fi
9046 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
9047 \renewcommand*{\glxtrinlinefullformat}[2]{%
9048   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9049   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9050   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9051 }%
9052 \renewcommand*{\glxtrinlinefullplformat}[2]{%
```

```

9053 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9054 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9055 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9056 }%
9057 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9058 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9059 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9060 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9061 }%
9062 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9063 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9064 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9065 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9066 }%

```

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.

```

9067 \renewcommand*{\glxtrfullformat}[2]{%
9068 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9069 \ifglxtrinsertinside\else##2\fi
9070 }%
9071 \renewcommand*{\glxtrfullplformat}[2]{%
9072 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9073 \ifglxtrinsertinside\else##2\fi
9074 }%
9075 \renewcommand*{\Glsxtrfullformat}[2]{%
9076 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9077 \ifglxtrinsertinside\else##2\fi
9078 }%
9079 \renewcommand*{\Glsxtrfullplformat}[2]{%
9080 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9081 \ifglxtrinsertinside\else##2\fi
9082 }%
9083 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9084 \newabbreviationstyle{long-em-noshort-em-noreg}%
9085 {%
9086 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9087 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9088 \glshasattribute{\the\glslabeltok}{regular}%
9089 {%
9090 \glssetattribute{\the\glslabeltok}{regular}{false}%
9091 }%
9092 {}%
9093 }%
9094 }%
9095 {%

```



```

9096 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9097 }

```

noshort-em-desc The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

9098 \newabbreviationstyle{long-noshort-em-desc}%
9099 {%
9100 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9101 }%
9102 {%
9103 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9104 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
9105 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvemfont{##1}}%
9106 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
9107 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9108 \renewcommand*{\glssxtrsubsequentfmt}[2]{%
9109 \glsslongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9110 \ifglssxtrininsertinside \else##2\fi
9111 }%
9112 \renewcommand*{\glssxtrsubsequentplfmt}[2]{%
9113 \glsslongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9114 \ifglssxtrininsertinside \else##2\fi
9115 }%
9116 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9117 \glsslongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside ##2\fi}%
9118 \ifglssxtrininsertinside \else##2\fi
9119 }%
9120 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9121 \glsslongdefaultfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside ##2\fi}%
9122 \ifglssxtrininsertinside \else##2\fi
9123 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9124 \renewcommand*{\glssxtrinlinefullformat}[2]{%
9125 \glssfirstlongdefaultfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
9126 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9127 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
9128 }%
9129 \renewcommand*{\glssxtrinlinefullplformat}[2]{%
9130 \glssfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
9131 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9132 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshortpl{##1}}}%
9133 }%
9134 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9135 \glssfirstlongdefaultfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
9136 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
9137 \glssxtrparen{\protect\glssfirstabrvemfont{\glssaccessshort{##1}}}%
9138 }%

```

```

9139 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9140   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9141   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9142   \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9143 }%

```

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.

```

9144 \renewcommand*{\glxtrfullformat}[2]{%
9145   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9146   \ifglxtrinsertinside\else##2\fi
9147 }%
9148 \renewcommand*{\glxtrfullplformat}[2]{%
9149   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9150   \ifglxtrinsertinside\else##2\fi
9151 }%
9152 \renewcommand*{\Glsxtrfullformat}[2]{%
9153   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9154   \ifglxtrinsertinside\else##2\fi
9155 }%
9156 \renewcommand*{\Glsxtrfullplformat}[2]{%
9157   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9158   \ifglxtrinsertinside\else##2\fi
9159 }%
9160 }

```

long-desc-em Backward compatibility:

```

9161 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like \glsshort. The long form is emphasized.

```

9162 \newabbreviationstyle{long-em-noshort-em-desc}%
9163 {%
9164   \renewcommand*{\CustomAbbreviationFields}{%
9165     name={\protect\protect\glslongemfont{\the\glslongtok}},
9166     sort={\the\glslongtok},
9167     first={\protect\glsfirstlongemfont{\the\glslongtok}},
9168     firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
9169     text={\glslongemfont{\the\glslongtok}},
9170     plural={\glslongemfont{\the\glslongpltok}}}%
9171   }%
9172   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9173     \glssetattribute{\the\glslabeltok}{regular}{true}}%
9174 }%
9175 {%
9176   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9177   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9178   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9179   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
9180   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9181 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9182   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9183   \ifglxtrinsertinside \else##2\fi
9184 }%
9185 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9186   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9187   \ifglxtrinsertinside \else##2\fi
9188 }%
9189 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9190   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9191   \ifglxtrinsertinside \else##2\fi
9192 }%
9193 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9194   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9195   \ifglxtrinsertinside \else##2\fi
9196 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9197 \renewcommand*{\glxtrinlinefullformat}[2]{%
9198   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9199   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9200   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9201 }%
9202 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9203   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9204   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9205   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9206 }%
9207 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9208   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9209   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9210   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9211 }%
9212 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9213   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9214   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9215   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9216 }%

```

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.

```

9217 \renewcommand*{\glxtrfullformat}[2]{%
9218   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9219   \ifglxtrinsertinside\else##2\fi
9220 }%
9221 \renewcommand*{\glxtrfullplformat}[2]{%
9222   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9223   \ifglxtrinsertinside\else##2\fi
9224 }%

```

```

9225 \renewcommand*{\GlsXtrFullFormat}[2]{%
9226   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9227   \ifglxtrinsertinside\else##2\fi
9228 }%
9229 \renewcommand*{\GlsXtrFullPlFormat}[2]{%
9230   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9231   \ifglxtrinsertinside\else##2\fi
9232 }%
9233 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9234 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9235 {%
9236   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
   Unset the regular attribute if it has been set.
9237   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9238     \glshasattribute{\the\glslabeltok}{regular}%
9239     {%
9240       \glissetattribute{\the\glslabeltok}{regular}{false}%
9241     }%
9242   }%
9243 }%
9244 }%
9245 {%
9246   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9247 }

```

ort-em-footnote

```

9248 \newabbreviationstyle{short-em-footnote}%
9249 {%
9250   \renewcommand*{\CustomAbbreviationFields}{%
9251     name={\protect\glsabbrvemfont{\the\glsshorttok}},
9252     sort={\the\glsshorttok},
9253     description={\the\glslongtok},%
9254     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
9255       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9256       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9257     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
9258       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9259       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9260     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
   Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute
   if it has been set.
9261   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9262     \glissetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9263     \glshasattribute{\the\glslabeltok}{regular}%
9264     {%
9265       \glissetattribute{\the\glslabeltok}{regular}{false}%

```

```

9266 }%
9267 {}%
9268 }%
9269 }%
9270 {%
9271 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
9272 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9273 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9274 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9275 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9276 \renewcommand*{\glsxtrfullformat}[2]{%
9277   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9278   \ifglsxtrininsertinside\else##2\fi
9279   \protect\glsxtrabbrvfootnote{##1}%
9280   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9281 }%
9282 \renewcommand*{\glsxtrfullplformat}[2]{%
9283   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9284   \ifglsxtrininsertinside\else##2\fi
9285   \protect\glsxtrabbrvfootnote{##1}%
9286   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9287 }%
9288 \renewcommand*{\Glsxtrfullformat}[2]{%
9289   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9290   \ifglsxtrininsertinside\else##2\fi
9291   \protect\glsxtrabbrvfootnote{##1}%
9292   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9293 }%
9294 \renewcommand*{\Glsxtrfullplformat}[2]{%
9295   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9296   \ifglsxtrininsertinside\else##2\fi
9297   \protect\glsxtrabbrvfootnote{##1}%
9298   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9299 }%

```

The first use full form and the inline full form use the short (long) style.

```

9300 \renewcommand*{\glsxtrininlinefullformat}[2]{%
9301   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9302   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9303   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9304 }%
9305 \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9306   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9307   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9308   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9309 }%
9310 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9311   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%

```

```

9312     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9313     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9314 }%
9315 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9316     \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9317     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9318     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9319 }%
9320 }

```

footnote-em Backward compatibility:

```

9321 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9322 \newabbreviationstyle{short-em-postfootnote}%
9323 {%
9324     \renewcommand*{\CustomAbbreviationFields}{%
9325         name={\protect\glssabbrvemfont{\the\glssshorttok}},
9326         sort={\the\glssshorttok},
9327         description={\the\glslongtok},%
9328         first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},%
9329         firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},%
9330         plural={\protect\glssabbrvemfont{\the\glssshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

9331 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9332     \csdef{glxtrpostlink\glscategorylabel}{%
9333         \glxtrifwasfirstuse
9334         {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9335         \glxtrdopostpunc{\protect\glxtrabbrvf footnote{\glslabel}%
9336             {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9337     }%
9338     {}%
9339 }%
9340 \glshasattribute{\the\glslabeltok}{regular}%
9341 {%
9342     \glissetattribute{\the\glslabeltok}{regular}{false}%
9343 }%
9344 {}%
9345 }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

9346 \renewcommand*{\glxtrsetupfulldefs}{%
9347     \let\glxtrifwasfirstuse\@secondoftwo
9348 }%

```

```

9349 }%
9350 {%
9351   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9352   \renewcommand*{\glssabrvfont}[1]{\glssabrvfont{##1}}%
9353   \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvfont{##1}}%
9354   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
9355   \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9356   \renewcommand*{\glsxtrfullformat}[2]{%
9357     \glssfirstabrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9358     \ifglsxtrininsertinside\else##2\fi
9359   }%
9360   \renewcommand*{\glsxtrfullplformat}[2]{%
9361     \glssfirstabrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9362     \ifglsxtrininsertinside\else##2\fi
9363   }%
9364   \renewcommand*{\Glsxtrfullformat}[2]{%
9365     \glssfirstabrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9366     \ifglsxtrininsertinside\else##2\fi
9367   }%
9368   \renewcommand*{\Glsxtrfullplformat}[2]{%
9369     \glssfirstabrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9370     \ifglsxtrininsertinside\else##2\fi
9371   }%

```

The first use full form and the inline full form use the short (long) style.

```

9372   \renewcommand*{\glsxtrininlinefullformat}[2]{%
9373     \glssfirstabrvfont{\glssaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9374     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9375     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
9376   }%
9377   \renewcommand*{\glsxtrininlinefullplformat}[2]{%
9378     \glssfirstabrvfont{\glssaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9379     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9380     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9381   }%
9382   \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9383     \glssfirstabrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9384     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9385     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
9386   }%
9387   \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
9388     \glssfirstabrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9389     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9390     \glsxtrparen{\glssfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9391   }%
9392 }

```

postfootnote-em Backward compatibility:

9393 \@glxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

1.6.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

glxtruserfield Default is the useri field.

9394 \newcommand*{\glxtruserfield}{useri}

glxtruserparen The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If \glscurrentfieldvalue has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```
9395 \ifdef\glscurrentfieldvalue
9396 {
9397   \newcommand*{\glxtruserparen}[2]{%
9398     \glxtrfullsep{#2}%
9399     \glxtrparen
9400     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9401   }
9402 }
9403 {
9404   \newcommand*{\glxtruserparen}[2]{%
9405     \glxtrfullsep{#2}%
9406     \glxtrparen
9407     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9408   }
9409 }
```

Font used for short form:

lsabbrvuserfont

9410 \newcommand*{\glssabbrvuserfont}[1]{\glssabbrvdefaultfont{#1}}

Font used for short form on first use:

stabbrvuserfont

9411 \newcommand*{\glssfirstabbrvuserfont}[1]{\glssabbrvuserfont{#1}}

Font used for long form:

glslonguserfont

9412 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}

Font used for long form on first use:

rstlonguserfont

9413 \newcommand*{\glssfirstlonguserfont}[1]{\glslonguserfont{#1}}

The default short form suffix:

lsxtrusersuffix

```
9414 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

long-short-user

```
9415 \newabbreviationstyle{long-short-user}%
9416 {%
9417   \renewcommand*{\CustomAbbreviationFields}{%
9418     name={\protect\glssabbrvuserfont{\the\glssshorttok}},
9419     sort={\the\glssshorttok},
9420     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9421       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glssshorttok}}}%
9422     {\the\glslabeltok}},%
9423     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9424       \protect\glxtruserparen
9425       {\protect\glsfirstabbrvuserfont{\the\glssshortpltok}}{\the\glslabeltok}},%
9426     plural={\protect\glssabbrvuserfont{\the\glssshortpltok}},%
9427     description={\protect\glslonguserfont{\the\glslongtok}}}%
9428   }
```

Unset the regular attribute if it has been set.

```
9428 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9429   \glshasattribute{\the\glslabeltok}{regular}%
9430   {%
9431     \glissetattribute{\the\glslabeltok}{regular}{false}%
9432   }%
9433   {}%
9434 }%
9435 }%
9436 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9437 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9438 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvuserfont{##1}}%
9439 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9440 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9441 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9442 \renewcommand*{\glxtrfullformat}[2]{%
9443   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9444   \ifglxtrinsertinside\else##2\fi
9445   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9446 }%
9447 \renewcommand*{\glxtrfullplformat}[2]{%
9448   \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9449   \ifglxtrinsertinside\else##2\fi
9450   \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9451 }%
9452 \renewcommand*{\GlsXtrfullformat}[2]{%
9453   \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9454 }
```

```

9454 \ifglxtrinsertinside\else##2\fi
9455 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9456 }%
9457 \renewcommand*{\Glsxtrfullplformat}[2]{%
9458 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9459 \ifglxtrinsertinside\else##2\fi
9460 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9461 }%
9462 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9463 \newabbreviationstyle{long-postshort-user}%
9464 {%
9465 \renewcommand*{\CustomAbbreviationFields}{%
9466 name={\protect\glssabbrvuserfont{\the\glssshorttok}},
9467 sort={\the\glssshorttok},
9468 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9469 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9470 plural={\protect\glssabbrvuserfont{\the\glssshortpltok}},%
9471 description={\protect\glslonguserfont{\the\glslongtok}}}%
9472 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9473 \csdef{glxtrpostlink\glscategorylabel}{%
9474 \glxtrifwasfirstuse
9475 {%
9476 \glxtruserparen
9477 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9478 {\glslabel}%
9479 }%
9480 }%
9481 }%
9482 \glshasattribute{\the\glslabeltok}{regular}%
9483 {%
9484 \glissetattribute{\the\glslabeltok}{regular}{false}%
9485 }%
9486 {}%
9487 }%
9488 }%
9489 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9490 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9491 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvuserfont{##1}}%
9492 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9493 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9494 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9495 \renewcommand*{\glxtrfullformat}[2]{%
9496 \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%

```

```

9497 \ifglxtrinsertinside\else##2\fi
9498 }%
9499 \renewcommand*{\glxtrfullplformat}[2]{%
9500 \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9501 \ifglxtrinsertinside\else##2\fi
9502 }%
9503 \renewcommand*{\Glsxtrfullformat}[2]{%
9504 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9505 \ifglxtrinsertinside\else##2\fi
9506 }%
9507 \renewcommand*{\Glsxtrfullplformat}[2]{%
9508 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9509 \ifglxtrinsertinside\else##2\fi
9510 }%

```

In-line format:

```

9511 \renewcommand*{\glxtrinlinefullformat}[2]{%
9512 \glsfirstlonguserfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9513 \ifglxtrinsertinside\else##2\fi
9514 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9515 }%
9516 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9517 \glsfirstlonguserfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9518 \ifglxtrinsertinside\else##2\fi
9519 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9520 }%
9521 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9522 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9523 \ifglxtrinsertinside\else##2\fi
9524 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9525 }%
9526 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9527 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9528 \ifglxtrinsertinside\else##2\fi
9529 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9530 }%
9531 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9532 \newabbreviationstyle{long-postshort-user-desc}%
9533 {%
9534 \renewcommand*{\CustomAbbreviationFields}{%
9535 name={\protect\glslonguserfont{\the\glslongtok}%
9536 \protect\glxtruserparen
9537 {\protect\glsabbrvuserfont{\the\glssshorttok}}{\the\glslabeltok}},
9538 sort={\the\glslongtok},
9539 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9540 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9541 text={\protect\glsabbrvuserfont{\the\glssshorttok}},%

```

```

9542 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9543 }%
9544 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9545   \csdef{glxtrpostlink\glscategorylabel}{%
9546     \glxtrifwasfirstuse
9547     {%
9548       \glxtruserparen
9549       {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9550       {\glslabel}%
9551     }%
9552   }%
9553 }%
9554 \glsattribute{\the\glslabeltok}{regular}%
9555 {%
9556   \glssetattribute{\the\glslabeltok}{regular}{false}%
9557 }%
9558 {}%
9559 }%
9560 }%
9561 {%
9562   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9563 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9564 \newabbreviationstyle{short-postlong-user}%
9565 {%
9566   \renewcommand*{\CustomAbbreviationFields}{%
9567     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9568     sort={\the\glsshorttok},
9569     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9570     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9571     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9572     description={\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9573 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9574   \csdef{glxtrpostlink\glscategorylabel}{%
9575     \glxtrifwasfirstuse
9576     {%
9577       \glxtruserparen
9578       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9579       {\glslabel}%
9580     }%
9581   }%
9582 }%
9583 \glsattribute{\the\glslabeltok}{regular}%
9584 {%
9585   \glssetattribute{\the\glslabeltok}{regular}{false}%
9586 }%
9587 {}%
9588 }%

```

```
9589 }%
9590 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9591 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9592 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9593 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9594 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9595 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

First use full form:

```
9596 \renewcommand*{\glxtrfullformat}[2]{%
9597   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9598   \ifglxtrininsertinside\else##2\fi
9599 }%
9600 \renewcommand*{\glxtrfullplformat}[2]{%
9601   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9602   \ifglxtrininsertinside\else##2\fi
9603 }%
9604 \renewcommand*{\Glsxtrfullformat}[2]{%
9605   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9606   \ifglxtrininsertinside\else##2\fi
9607 }%
9608 \renewcommand*{\Glsxtrfullplformat}[2]{%
9609   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9610   \ifglxtrininsertinside\else##2\fi
9611 }%
```

In-line format:

```
9612 \renewcommand*{\glxtrinlinefullformat}[2]{%
9613   \glsfirstabbrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9614   \ifglxtrininsertinside\else##2\fi
9615   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9616 }%
9617 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9618   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9619   \ifglxtrininsertinside\else##2\fi
9620   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9621 }%
9622 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9623   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9624   \ifglxtrininsertinside\else##2\fi
9625   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9626 }%
9627 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9628   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9629   \ifglxtrininsertinside\else##2\fi
9630   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9631 }%
9632 }
```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```
9633 \newabbreviationstyle{short-postlong-user-desc}%
9634 {%
9635   \renewcommand*{\CustomAbbreviationFields}{%
9636     name={\protect\glsabbrvuserfont{\the\glsshorttok}%
9637       \protect\glsxtruserparen
9638         {\protect\glslonguserfont{\the\glslongpltok}}}%
9639       {\the\glslabeltok}},
9640     sort={\the\glsshorttok},
9641     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9642     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9643     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9644     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}%
9645   }%
9646   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9647     \csdef{glsxtrpostlink\glscategorylabel}{%
9648       \glsxtrifwasfirstuse
9649       {%
9650         \glsxtruserparen
9651           {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9652           {\glslabel}}%
9653       }%
9654     }%
9655   }%
9656   \glsattribute{\the\glslabeltok}{regular}%
9657   {%
9658     \glssetattribute{\the\glslabeltok}{regular}{false}%
9659   }%
9660   {}%
9661 }%
9662 }%
9663 {%
9664   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9665 }
```

short-user-desc

```
9666 \newabbreviationstyle{long-short-user-desc}%
9667 {%
9668   \renewcommand*{\CustomAbbreviationFields}{%
9669     name={\glsxtrlongshortdescname},
9670     sort={\glsxtrlongshortdescsort},%

9671     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9672       \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9673     {\the\glslabeltok}},%
9674     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9675       \protect\glsxtruserparen
9676         {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9677     text={\protect\glsabbrvfont{\the\glsshorttok}},%
```

```

9678 plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
9679 }%

```

Unset the regular attribute if it has been set.

```

9680 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9681   \glshasattribute{\the\glslabeltok}{regular}%
9682   {%
9683     \glissetattribute{\the\glslabeltok}{regular}{false}%
9684   }%
9685   {}%
9686 }%
9687 }%
9688 {%
9689   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9690 }

```

short-long-user

```

9691 \newabbreviationstyle{short-long-user}%
9692 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

9693 \renewcommand*{\CustomAbbreviationFields}{%
9694   name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9695   sort={\the\glsshorttok},
9696   description={\protect\glslonguserfont{\the\glslongtok}},%
9697   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9698   \protect\glsextruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9699   {\the\glslabeltok}},%
9700   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9701   \protect\glsextruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
9702   {\the\glslabeltok}},%

```

```

9703   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9704 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9705   \glshasattribute{\the\glslabeltok}{regular}%
9706   {%
9707     \glissetattribute{\the\glslabeltok}{regular}{false}%
9708   }%
9709   {}%
9710 }%
9711 }%
9712 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9713 \renewcommand*{\abbrvpluralsuffix}{\glsextrusersuffix}%
9714 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9715 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9716 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9717 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline form are the same for this style.

```

9718 \renewcommand*{\glxtrfullformat}[2]{%
9719   \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9720   \ifglxtrinsertinside\else##2\fi
9721   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9722 }%
9723 \renewcommand*{\glxtrfullplformat}[2]{%
9724   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9725   \ifglxtrinsertinside\else##2\fi
9726   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9727 }%
9728 \renewcommand*{\Glsxtrfullformat}[2]{%
9729   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9730   \ifglxtrinsertinside\else##2\fi
9731   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9732 }%
9733 \renewcommand*{\Glsxtrfullplformat}[2]{%
9734   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9735   \ifglxtrinsertinside\else##2\fi
9736   \glxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9737 }%
9738 }

```

-long-user-desc

```

9739 \newabbreviationstyle{short-long-user-desc}%
9740 {%
9741   \renewcommand*{\CustomAbbreviationFields}{%
9742     name={\glxtrshortlongdescname},
9743     sort={\glxtrshortlongdescsort},%
9744     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9745     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9746     {\the\glslabeltok}},%
9747     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9748     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9749     {\the\glslabeltok}},%
9750     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9751     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9752   }%

```

Unset the regular attribute if it has been set.

```

9753 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9754   \glshasattribute{\the\glslabeltok}{regular}%
9755   {%
9756     \glssetattribute{\the\glslabeltok}{regular}{false}%
9757   }%
9758   {}%
9759 }%
9760 }%
9761 {%

```



```

9762 \GlsXtrUseAbbrStyleFmts{short-long-user}%
9763 }

```

1.6.7 Predefined Styles (Hyphen)

These styles are designed to work with the `markwords` attribute. They check if the inserted material (provided by the final optional argument of commands like `\gls`) starts with a hyphen. If it does, the insert is added to the parenthetical material. Note that commands like `\glsxtrlong` 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.

```

9764 \newrobustcmd*{\glsxtrifhyphenstart}[3]{%
9765   \ifx\glsinsert#1\relax
9766   \expandafter\@glsxtrifhyphenstart#1\relax\relax
9767   \@end@glsxtrifhyphenstart{#2}{#3}%
9768   \else
9769   \@glsxtrifhyphenstart#1\relax\relax\@end@glsxtrifhyphenstart{#2}{#3}%
9770   \fi
9771 }

```

`\trifhyphenstart`

```

9772 \def\@glsxtrifhyphenstart#1#2\@end@glsxtrifhyphenstart#3#4{%
9773   \ifx-#1\relax#3\else #4\fi
9774 }

```

`\longhyphenshort`

```
\glsxtrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}
```

The `<long>` and `<short>` arguments may be the plural form. The `<long>` argument may also be the first letter uppercase form.

```

9775 \newcommand*{\glsxtrlonghyphenshort}[4]{%

```

Grouping is needed to localise the redefinitions.

```

9776 {%

```

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.

```

9777   \glsxtrifhyphenstart{#4}{\def\glsxtrwordsep{-}}{}%
9778   \glsfirstlonghyphenfont{#2\ifglsxtrininsertinside{#4}\fi}%
9779   \ifglsxtrininsertinside\else{#4}\fi
9780   \glsxtrfullsep{#1}%
9781   \glsxtrparen{\glsfirstabbrvhyphenfont{#3\ifglsxtrininsertinside{#4}\fi}%
9782   \ifglsxtrininsertinside\else{#4}\fi}%
9783 }%
9784 }

```

abbrvhyphenfont

```
9785 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
9786 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
9787 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%
```

tlonghyphenfont

```
9788 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
9789 \newcommand*{\glsxtrhyphensuffix}{\glsxtrabbrvpluralsuffix}
```

en-short-hyphen Designed for use with the markwords attribute.

```
9790 \newabbreviationstyle{long-hyphen-short-hyphen}%
9791 {%
9792   \renewcommand*{\CustomAbbreviationFields}{%
9793     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9794     sort={\the\glsshorttok},
9795     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
9796     \protect\glsxtrfullsep{\the\glslabeltok}%
9797     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9798     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
9799     \protect\glsxtrfullsep{\the\glslabeltok}%
9800     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9801     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9802     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
9803   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9804     \glshasattribute{\the\glslabeltok}{regular}%
9805     {%
9806       \glssetattribute{\the\glslabeltok}{regular}{false}%
9807     }%
9808   }%
9809 }%
9810 }%
9811 {%
9812   \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9813   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9814   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9815   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9816   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

9817 \renewcommand*{\glxstrfullformat}[2]{%
9818   \glxstrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9819 }%
9820 \renewcommand*{\glxstrfullplformat}[2]{%
9821   \glxstrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
9822   {\glsaccessshortpl{##1}}{##2}%
9823 }%
9824 \renewcommand*{\Glsxstrfullformat}[2]{%
9825   \glxstrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9826 }%
9827 \renewcommand*{\Glsxstrfullplformat}[2]{%
9828   \glxstrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9829   {\glsaccessshortpl{##1}}{##2}%
9830 }%
9831 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9832 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9833 {%
9834   \renewcommand*{\CustomAbbreviationFields}{%
9835     name={\glxstrlongshortdescname},
9836     sort={\glxstrlongshortdescsort},
9837     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}}%
9838     \protect\glxstrfullsep{\the\glslabeltok}}%
9839     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9840     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}}%
9841     \protect\glxstrfullsep{\the\glslabeltok}}%
9842     \glxstrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9843     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}}},%
9844     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
9845 }%

```

Unset the regular attribute if it has been set.

```

9846 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9847   \glshasattribute{\the\glslabeltok}{regular}%
9848   {%
9849     \glissetattribute{\the\glslabeltok}{regular}{false}%
9850   }%
9851   {}%
9852 }%
9853 }%
9854 {%
9855   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9856 }

```

onghyphennoshort `\glxstrlonghyphennoshort{<label>}{<long>}{<insert>}`

```
9857 \newcommand*{\glxtrlonghyphennoshort}[3]{%
```

Grouping is needed to localise the redefinitions.

```
9858 {%
```

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.

```
9859 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{}%
```

```
9860 \glsfirslonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
```

```
9861 \ifglxtrininsertinside\else{#3}\fi
```

```
9862 }%
```

```
9863 }
```

`short-desc-noreg` This version doesn't show the short form (except explicitly with `\glxtrshort`). Since `\glxtrshort` doesn't support the hyphen switch, the short form just uses the default short-form font command. This style won't work with the regular as the regular form isn't flexible enough.

```
9864 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
```

```
9865 {%
```

```
9866 \renewcommand*{\CustomAbbreviationFields}{%
```

```
9867   name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
```

```
9868   sort={\expandonce\glxtrorglong},
```

```
9869   first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
```

```
9870   firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
```

```
9871   plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
```

```
9872 }%
```

Unset the regular attribute if it has been set.

```
9873 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
9874   \glshasattribute{\the\glslabeltok}{regular}%
```

```
9875   {%
```

```
9876     \glissetattribute{\the\glslabeltok}{regular}{false}%
```

```
9877   }%
```

```
9878   {}%
```

```
9879 }%
```

```
9880 }%
```

```
9881 {%
```

```
9882   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9883 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
```

```
9884 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
```

```
9885 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```

```
9886 \renewcommand*{\glsfirslongfont}[1]{\glsfirslonghyphenfont{##1}}%
```

```
9887 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9888 \renewcommand*{\glxtrsubsequentfmt}[2]{%
```

```
9889   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
```

```
9890 }%
```

```

9891 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9892   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9893 }%
9894 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9895   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9896 }%
9897 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9898   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9899 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9900 \renewcommand*{\glxtrinlinefullformat}[2]{%
9901   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
9902   \glxtrfullsep{##1}%
9903   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
9904 }%
9905 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9906   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9907   \glxtrfullsep{##1}%
9908   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{##1}}}%
9909 }%
9910 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9911   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9912   \glxtrfullsep{##1}%
9913   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
9914 }%
9915 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9916   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9917   \glxtrfullsep{##1}%
9918   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
9919 }%

```

The first use full form only displays the long form.

```

9920 \renewcommand*{\glxtrfullformat}[2]{%
9921   \glxtrlonghyphennoshort{##1}{\glssaccesslong{##1}}{##2}%
9922 }%
9923 \renewcommand*{\glxtrfullplformat}[2]{%
9924   \glxtrlonghyphennoshort{##1}{\glssaccesslongpl{##1}}{##2}%
9925 }%
9926 \renewcommand*{\Glsxtrfullformat}[2]{%
9927   \glxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9928 }%
9929 \renewcommand*{\Glsxtrfullplformat}[2]{%
9930   \glxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9931 }%
9932 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a descriptionless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

9933 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9934 {%
9935   \renewcommand*{\CustomAbbreviationFields}{%
9936     name={\protect\glsabbrvfont{\the\glsshorttok}},
9937     sort={\the\glsshorttok},
9938     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9939     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9940     text={\protect\glslonghyphenfont{\the\glslongtok}},%
9941     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
9942     description={\the\glslongtok}%
9943   }%
9944   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9945     \glshasattribute{\the\glslabeltok}{regular}%
9946     {%
9947       \glssetattribute{\the\glslabeltok}{regular}{false}%
9948     }%
9949   }%
9950 }%
9951 }%
9952 {%
9953   \GlsXtrUseAbbrStyleFmts{long-desc}%
9954 }

```

glsxtrlonghyphen `\glsxtrlonghyphen{<long>}{<label>}{<insert>}`

Used by long-hyphen-postshort-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

9955 \newcommand*{\glsxtrlonghyphen}[3]{%
9956   {%
9957     \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9958     \glsfirstlonghyphenfont{#1}%
9959   }%
9960 }

```

rposthyphenshort `\glsxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glsxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```

9961 \newcommand*{\glsxtrposthyphenshort}[2]{%
9962   {%

```

```

9963 \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
9964 \ifglxtrinsertinside{\glsfirslonghyphenfont{#2}}\else{#2}\fi
9965 \glxtrfullsep{#1}%
9966 \glxtrparen
9967 {\glsfirabbrvhyphenfont{\glsentryshort{#1}}\ifglxtrinsertinside{#2}\fi}%
9968 \ifglxtrinsertinside\else{#2}\fi
9969 }%
9970 }%
9971 }

```

hyphensubsequent

```
\glxtrposthyphensubsequent{<label>}{<insert>}
```

Format in the post-link hook for subsequent use. The label is ignored by default.

```

9972 \newcommand*{\glxtrposthyphensubsequent}[2]{%
9973 \glssabbrvfont{\ifglxtrinsertinside {#2}\fi}%
9974 \ifglxtrinsertinside \else{#2}\fi
9975 }

```

postshort-hyphen

Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9976 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
9977 {%
9978 \renewcommand*{\CustomAbbreviationFields}{%
9979 name={\protect\glssabbrvhyphenfont{\the\glssshorttok}},
9980 sort={\the\glssshorttok},
9981 first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
9982 firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
9983 plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}},%
9984 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9985 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9986 \csdef{glxtrpostlink\glscategorylabel}{%
9987 \glxtrifwasfirstuse
9988 {%
9989 \glxtrposthyphenshort{\glslabel}{\glssinsert}%
9990 }%
9991 }%

```

Put the insertion into the post-link:

```

9992 \glxtrposthyphensubsequent{\glslabel}{\glssinsert}%
9993 }%
9994 }%
9995 \glshasattribute{\the\glslabeltok}{regular}%
9996 {%
9997 \glsssetattribute{\the\glslabeltok}{regular}{false}%
9998 }%
9999 {}%
10000 }%

```

10001 }%
 10002 {%

In case the user wants to mix and match font styles, these are redefined here.

10003 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
 10004 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
 10005 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
 10006 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
 10007 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

Subsequent use needs to omit the insertion:

10008 \renewcommand*{\glxtrsubsequentfmt}[2]{%
 10009 \glsabbrvfont{\glsaccessshort{##1}}%
 10010 }%
 10011 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
 10012 \glsabbrvfont{\glsaccessshortpl{##1}}%
 10013 }%
 10014 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
 10015 \glsabbrvfont{\Glsaccessshort{##1}}%
 10016 }%
 10017 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
 10018 \glsabbrvfont{\Glsaccessshortpl{##1}}%
 10019 }%

First use full form:

10020 \renewcommand*{\glxtrfullformat}[2]{%
 10021 \glxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
 10022 }%
 10023 \renewcommand*{\glxtrfullplformat}[2]{%
 10024 \glxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
 10025 }%
 10026 \renewcommand*{\Glsxtrfullformat}[2]{%
 10027 \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
 10028 }%
 10029 \renewcommand*{\Glsxtrfullplformat}[2]{%
 10030 \glxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
 10031 }%

In-line format.

10032 \renewcommand*{\glxtrinlinefullformat}[2]{%
 10033 \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
 10034 \ifglxtrininsertinside{##2}\fi}%
 10035 \ifglxtrininsertinside \else{##2}\fi
 10036 }%
 10037 \renewcommand*{\glxtrinlinefullplformat}[2]{%
 10038 \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%
 10039 \ifglxtrininsertinside{##2}\fi}%
 10040 \ifglxtrininsertinside \else{##2}\fi
 10041 }%
 10042 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
 10043 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%


```

10044     \ifglxtrinsertinside{##2}\fi}%
10045     \ifglxtrinsertinside \else{##2}\fi
10046 }%
10047 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10048     \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10049     \ifglxtrinsertinside{##2}\fi}%
10050     \ifglxtrinsertinside \else{##2}\fi
10051 }%
10052 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10053 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10054 {%
10055     \renewcommand*{\CustomAbbreviationFields}{%
10056         name={\glxtrlongshortdescname},
10057         sort={\glxtrlongshortdescsort},%
10058         first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10059         firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10060         text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10061         plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10062 }%
10063 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10064     \csdef{glxtrpostlink\glscategorylabel}{%
10065         \glxtrifwasfirstuse
10066         {%
10067             \glxtrposthyphenshort{\glslabel}{\glsinsert}%
10068         }%
10069         {%

```

Put the insertion into the post-link:

```

10070         \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10071     }%
10072 }%
10073 \glshasattribute{\the\glslabeltok}{regular}%
10074 {%
10075     \glissetattribute{\the\glslabeltok}{regular}{false}%
10076 }%
10077 {}%
10078 }%
10079 }%
10080 {%
10081     \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10082 }

```

`rshorthyphenlong`

```
\glxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be

the first letter uppercase form.

```
10083 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10084 {%
```

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.)

```
10085 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
```

```
10086 \glsfirtabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
```

```
10087 \ifglxtrininsertinside\else{#4}\fi
```

```
10088 \glxtrfullsep{#1}%
```

```
10089 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
```

```
10090 \ifglxtrininsertinside\else{#4}\fi}%
```

```
10091 }%
```

```
10092 }
```

hen-long-hyphen Designed for use with the markwords attribute.

```
10093 \newabbreviationstyle{short-hyphen-long-hyphen}%
```

```
10094 {%
```

```
10095 \renewcommand*{\CustomAbbreviationFields}{%
```

```
10096 name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
```

```
10097 sort={\the\glsshorttok},
```

```
10098 first={\protect\glsfirtabbrvhyphenfont{\the\glsshorttok}}%
```

```
10099 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
10100 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
```

```
10101 firstplural={\protect\glsfirtabbrvhyphenfont{\the\glsshortpltok}}%
```

```
10102 \protect\glxtrfullsep{\the\glslabeltok}}%
```

```
10103 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
```

```
10104 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}},%
```

```
10105 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
10106 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
```

```
10107 \glshasattribute{\the\glslabeltok}{regular}}%
```

```
10108 {%
```

```
10109 \glissetattribute{\the\glslabeltok}{regular}{false}}%
```

```
10110 }%
```

```
10111 {}%
```

```
10112 }%
```

```
10113 }%
```

```
10114 {%
```

```
10115 \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
```

```
10116 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
```

```
10117 \renewcommand*{\glsfirtabbrvfont}[1]{\glsfirtabbrvhyphenfont{##1}}%
```

```
10118 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
```

```
10119 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```

10120 \renewcommand*\glstrfullformat}[2]{%
10121   \glstrshorthyphenlong{##1}{\glssaccessshort{##1}}{\glssaccesslong{##1}}{##2}%
10122 }%
10123 \renewcommand*\glstrfullplformat}[2]{%
10124   \glstrshorthyphenlong{##1}%
10125   {\glssaccessshortpl{##1}}{\glssaccesslongpl{##1}}{##2}%
10126 }%
10127 \renewcommand*\Glsxtrfullformat}[2]{%
10128   \glstrshorthyphenlong{##1}{\glssaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10129 }%
10130 \renewcommand*\Glsxtrfullplformat}[2]{%
10131   \glstrshorthyphenlong{##1}%
10132   {\glssaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10133 }%
10134 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10135 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10136 {%
10137   \renewcommand*\CustomAbbreviationFields{%
10138     name={\glstrshortlongdescname},
10139     sort={\glstrshortlongdescsort},
10140     first={\protect\glssfirstabbrvhyphenfont{\the\glssshorttok}%
10141       \protect\glstrfullsep{\the\glslabeltok}%
10142       \glstrparen{\protect\glssfirstlonghyphenfont{\the\glslongtok}}},%
10143     firstplural={\protect\glssfirstabbrvhyphenfont{\the\glssshortpltok}%
10144       \protect\glstrfullsep{\the\glslabeltok}%
10145       \glstrparen{\protect\glssfirstlonghyphenfont{\the\glslongpltok}}},%
10146     text={\protect\glssabbrvhyphenfont{\the\glssshorttok}},%
10147     plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}}}%
10148 }%

```

Unset the regular attribute if it has been set.

```

10149 \renewcommand*\GlsXtrPostNewAbbreviation{%
10150   \glshasattribute{\the\glslabeltok}{regular}%
10151   {%
10152     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10153   }%
10154   {}%
10155 }%
10156 }%
10157 {%
10158   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10159 }

```

lsxtrshorthyphen `\glstrshorthyphen{<short>}{<label>}{<insert>}`

Used by short-hyphen-postlong-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```
10160 \newcommand*{\glxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```
10161  {%
10162   \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}}%
10163   \glsfirstabbrvhyphenfont{#1}%
10164  }%
10165 }
```

trposthyphenlong `\glxtrposthyphenlong{<label>}{<insert>}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glxtrshorthyphenlong` but omits the *<short>* part. This always uses the singular long form.

```
10166 \newcommand*{\glxtrposthyphenlong}[2]{%
10167  {%
10168   \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}}%
10169   \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10170   \glxtrfullsep{#1}%
10171   \glxtrparen
10172   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
10173   \ifglxtrininsertinside\else{#2}\fi
10174  }%
10175 }%
10176 }
```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
10177 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10178 {%
10179   \renewcommand*{\CustomAbbreviationFields}{%
10180     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
10181     sort={\the\glsshorttok},
10182     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10183     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10184     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10185     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10186   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10187     \csdef{glxtrpostlink\glscategorylabel}{%
10188       \glxtrifwasfirstuse
10189       {%
10190         \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10191       }%
10192     }%
```

Put the insertion into the post-link:

```

10193      \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10194    }%
10195  }%
10196  \glshasattribute{\the\glslabeltok}{regular}%
10197  {%
10198    \glsetattribute{\the\glslabeltok}{regular}{false}%
10199  }%
10200  {}%
10201 }%
10202 }%
10203 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10204 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10205 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10206 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10207 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10208 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10209 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10210   \glsabbrvfont{\glsaccessshort{##1}}%
10211 }%
10212 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10213   \glsabbrvfont{\glsaccessshortpl{##1}}%
10214 }%
10215 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10216   \glsabbrvfont{\Glsaccessshort{##1}}%
10217 }%
10218 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10219   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10220 }%

```

First use full form:

```

10221 \renewcommand*{\glxtrfullformat}[2]{%
10222   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10223 }%
10224 \renewcommand*{\glxtrfullplformat}[2]{%
10225   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10226 }%
10227 \renewcommand*{\Glsxtrfullformat}[2]{%
10228   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10229 }%
10230 \renewcommand*{\Glsxtrfullplformat}[2]{%
10231   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10232 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10233 \renewcommand*{\glxtrinlinefullformat}[2]{%

```

```

10234 \glsfirstabbrvhyphenfont{\glsaccessshort{##1}%
10235 \ifglxtrinsertinside{##2}\fi}%
10236 \ifglxtrinsertinside \else{##2}\fi
10237 }%
10238 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10239 \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}%
10240 \ifglxtrinsertinside{##2}\fi}%
10241 \ifglxtrinsertinside \else{##2}\fi
10242 }%
10243 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10244 \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}%
10245 \ifglxtrinsertinside{##2}\fi}%
10246 \ifglxtrinsertinside \else{##2}\fi
10247 }%
10248 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10249 \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}%
10250 \ifglxtrinsertinside{##2}\fi}%
10251 \ifglxtrinsertinside \else{##2}\fi
10252 }%
10253 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```

10254 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10255 {%
10256 \renewcommand*{\CustomAbbreviationFields}{%
10257 name={\glxtrshortlongdescname},
10258 sort={\glxtrshortlongdescsort},%
10259 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10260 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10261 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10262 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10263 }%
10264 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10265 \csdef{glxtrpostlink\glscategorylabel}{%
10266 \glxtrifwasfirstuse
10267 {%
10268 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10269 }%
10270 }%

```

Put the insertion into the post-link:

```

10271 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10272 }%
10273 }%
10274 \glshasattribute{\the\glslabeltok}{regular}%
10275 {%
10276 \glissetattribute{\the\glslabeltok}{regular}{false}%
10277 }%
10278 {}%
10279 }%

```

```

10280 }%
10281 {%
10282   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10283 }

```

1.6.8 Predefined Styles (No Short on First Use)

These styles show only the long form on first use and only the short form on subsequent use.

lsabbrvonlyfont

```

10284 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%

```

stabbrvonlyfont

```

10285 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%

```

glslongonlyfont

```

10286 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%

```

rstlongonlyfont

```

10287 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%

```

The default short form suffix:

lsxtronlysuffix

```

10288 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}

```

only-short-only

```

10289 \newabbreviationstyle{long-only-short-only}%
10290 {%
10291   \renewcommand*{\CustomAbbreviationFields}{%
10292     name={\protect\glsabbrvonlyfont{\the\glsshorttok}},
10293     sort={\the\glsshorttok},
10294     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10295     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10296     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10297     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10298   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10299     \glshasattribute{\the\glslabeltok}{regular}%
10300     {%
10301       \glssetattribute{\the\glslabeltok}{regular}{false}%
10302     }%
10303   }%
10304 }%
10305 }%
10306 {%
10307   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtronlysuffix}%
10308   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%

```

```

10309 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10310 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10311 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%

```

The first use full form doesn't show the short form.

```

10312 \renewcommand*{\glsxtrfullformat}[2]{%
10313   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10314   \ifglsxtrininsertinside\else##2\fi
10315 }%
10316 \renewcommand*{\glsxtrfullplformat}[2]{%
10317   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10318   \ifglsxtrininsertinside\else##2\fi
10319 }%
10320 \renewcommand*{\Glsxtrfullformat}[2]{%
10321   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10322   \ifglsxtrininsertinside\else##2\fi
10323 }%
10324 \renewcommand*{\Glsxtrfullplformat}[2]{%
10325   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10326   \ifglsxtrininsertinside\else##2\fi
10327 }%

```

The inline full form does show the short form.

```

10328 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10329   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10330   \ifglsxtrininsertinside\else##2\fi
10331   \glsxtrfullsep{##1}%
10332   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10333 }%
10334 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10335   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10336   \ifglsxtrininsertinside\else##2\fi
10337   \glsxtrfullsep{##1}%
10338   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10339 }%
10340 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10341   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
10342   \ifglsxtrininsertinside\else##2\fi
10343   \glsxtrfullsep{##1}%
10344   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10345 }%
10346 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10347   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
10348   \ifglsxtrininsertinside\else##2\fi
10349   \glsxtrfullsep{##1}%
10350   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10351 }%
10352 }

```

xtronlydescsort


```

10353 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

xtronlydescname
10354 \newcommand*{\glxtronlydescname}{%
10355   \protect\glslongfont{\the\glslongtok}%
10356 }

short-only-desc
10357 \newabbreviationstyle{long-only-short-only-desc}%
10358 {%
10359   \renewcommand*{\CustomAbbreviationFields}{%
10360     name={\glxtronlydescname},
10361     sort={\glxtronlydescsort},%
10362     first={\protect\glslongfont{\the\glslongtok}},%
10363     firstplural={\protect\glslongfont{\the\glslongpltok}},%
10364     text={\protect\glsshortfont{\the\glsshorttok}},%
10365     plural={\protect\glsshortfont{\the\glsshortpltok}}%
10366   }%

   Unset the regular attribute if it has been set.
10367   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10368     \glshasattribute{\the\glslabeltok}{regular}%
10369     {%
10370       \glissetattribute{\the\glslabeltok}{regular}{false}%
10371     }%
10372   }%
10373 }%
10374 }%
10375 {%
10376   \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10377 }

```

1.7 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glstryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glssabrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to false, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads textcase, so the label can be protected from case change with textcase's `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
10378 \let\@glxstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
10379 \renewcommand*{\markright}[1]{%
10380   \glxstrmarkhook
10381   \@glxstr@org@markright{\@glxstrinmark#1\@glxstrnotinmark}%
10382   \glxstrrestoremarkhook
10383 }
```

`\markboth` Save original definition:

```
10384 \let\@glxstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
10385 \renewcommand*{\markboth}[2]{%
10386   \glxstrmarkhook
10387   \@glxstr@org@markboth
10388   {\@glxstrinmark#1\@glxstrnotinmark}%
10389   {\@glxstrinmark#2\@glxstrnotinmark}%
10390   \glxstrrestoremarkhook
10391 }
```

Also do this for `\@starttoc`

`\@starttoc` Save original definition:

```
10392 \let\@glxstr@org@@starttoc\@starttoc
```

Redefine:

```
10393 \renewcommand*{\@starttoc}[1]{%
10394   \glxstrmarkhook
10395   \@glxstrinmark
10396   \@glxstr@org@@starttoc{#1}%
10397   \@glxstrnotinmark
10398   \glxstrrestoremarkhook
10399 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

`gxstrRevertMarks`

```
10400 \newcommand*{\gxstrRevertMarks}{%
10401   \let\markright\@glxstr@org@markright
```

```

10402 \let\markboth\@glxtr@org@markboth
10403 \let\@starttoc\@glxtr@org\@starttoc
10404 }

```

\glxtrifinmark

```

10405 \newcommand*\glxtrifinmark}[2]{#2}

```

\@glxtrinmark

```

10406 \newrobustcmd*\@glxtrinmark}{%
10407 \let\glxtrifinmark\@firstoftwo
10408 }

```

glxtrnotinmark

```

10409 \newrobustcmd*\glxtrnotinmark}{%
10410 \let\glxtrifinmark\@secondoftwo
10411 }

```

eorpdforheading

```

10412 \ifdef\texorpdfstring
10413 {
10414 \newcommand*\glxtrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10415 }
10416 {
10417 \newcommand*\glxtrtitleorpdforheading}[3]{#1}
10418 }

```

\glxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

10419 \newcommand*\glxtrmarkhook}{%

```

Save current definitions:

```

10420 \let\@glxtr@org@MakeUppercase\MakeUppercase
10421 \let\@glxtr@org@glxtrtitleorpdforheading\glxtrtitleorpdforheading
10422 \let\@glxtr@org@glxtrtitleshort\glxtrtitleshort
10423 \let\@glxtr@org@glxtrtitleshortpl\glxtrtitleshortpl
10424 \let\@glxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10425 \let\@glxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl
10426 \let\@glxtr@org@glxtrtitlename\glxtrtitlename
10427 \let\@glxtr@org@Glsxtrtitlename\Glsxtrtitlename
10428 \let\@glxtr@org@glxtrtitletext\glxtrtitletext
10429 \let\@glxtr@org@Glsxtrtitletext\Glsxtrtitletext
10430 \let\@glxtr@org@glxtrtitleplural\glxtrtitleplural
10431 \let\@glxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10432 \let\@glxtr@org@glxtrtitlefirst\glxtrtitlefirst
10433 \let\@glxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10434 \let\@glxtr@org@glxtrtitlefirstplural\glxtrtitlefirstplural
10435 \let\@glxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10436 \let\@glxtr@org@glxtrtitlelong\glxtrtitlelong
10437 \let\@glxtr@org@glxtrtitlelongpl\glxtrtitlelongpl

```

```

10438 \let\@glxstr@org@Glsxtrtitlelong\Glsxtrtitlelong
10439 \let\@glxstr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10440 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
10441 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
10442 \let\@glxstr@org@Glsxtrtitlefull\Glsxtrtitlefull
10443 \let\@glxstr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

New definitions

```

10444 \let\glxstrifinmark\@firstoftwo
10445 \let\MakeUppercase\MakeTextUppercase
10446 \let\glxstrtitleorpdforheading\@thirdofthree
10447 \let\glxstrtitleshort\glxstrheadshort
10448 \let\glxstrtitleshortpl\glxstrheadshortpl
10449 \let\Glsxstrtitleshort\Glsxtrheadshort
10450 \let\Glsxstrtitleshortpl\Glsxtrheadshortpl
10451 \let\glxstrtitlename\glxstrheadname
10452 \let\Glsxstrtitlename\Glsxtrheadname
10453 \let\glxstrtitletext\glxstrheadtext
10454 \let\Glsxstrtitletext\Glsxtrheadtext
10455 \let\glxstrtitleplural\glxstrheadplural
10456 \let\Glsxstrtitleplural\Glsxtrheadplural
10457 \let\glxstrtitlefirst\glxstrheadfirst
10458 \let\Glsxstrtitlefirst\Glsxtrheadfirst
10459 \let\glxstrtitlefirstplural\glxstrheadfirstplural
10460 \let\Glsxstrtitlefirstplural\Glsxtrheadfirstplural
10461 \let\glxstrtitlelong\glxstrheadlong
10462 \let\glxstrtitlelongpl\glxstrheadlongpl
10463 \let\Glsxstrtitlelong\Glsxtrheadlong
10464 \let\Glsxstrtitlelongpl\Glsxtrheadlongpl
10465 \let\glxstrtitlefull\glxstrheadfull
10466 \let\glxstrtitlefullpl\glxstrheadfullpl
10467 \let\Glsxstrtitlefull\Glsxtrheadfull
10468 \let\Glsxstrtitlefullpl\Glsxtrheadfullpl
10469 }

```

`\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.)

```

10470 \newcommand*{\glxstrrestoremarkhook}{%
10471 \let\glxstrifinmark\@secondoftwo
10472 \let\MakeUppercase\@glxstr@org@MakeUppercase
10473 \let\glxstrtitleorpdforheading\@glxstr@org@glxstrtitleorpdforheading
10474 \let\glxstrtitleshort\@glxstr@org@glxstrtitleshort
10475 \let\glxstrtitleshortpl\@glxstr@org@glxstrtitleshortpl
10476 \let\Glsxstrtitleshort\@glxstr@org@Glsxtrtitleshort
10477 \let\Glsxstrtitleshortpl\@glxstr@org@Glsxtrtitleshortpl
10478 \let\glxstrtitlename\@glxstr@org@glxstrtitlename
10479 \let\Glsxstrtitlename\@glxstr@org@Glsxtrtitlename

```

```

10480 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
10481 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
10482 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
10483 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
10484 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
10485 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
10486 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
10487 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
10488 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
10489 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
10490 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
10491 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
10492 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
10493 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
10494 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
10495 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
10496 }

```

Instead of using one document-wide conditional, use `headuc` attribute to determine whether or not to use the all upper case form.

glsxtrheadshort Command used to display short form in the page header.

```

10497 \newcommand*{\glsxtrheadshort}[1]{%
10498   \protect\NoCaseChange
10499   {%
10500     \glsifattribute{#1}{headuc}{true}%
10501     {%
10502       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10503     }%
10504     {%
10505       \glsxtrshort[noindex,hyper=false]{#1}[]%
10506     }%
10507   }%
10508 }

```

lgsxtrtitleshort Command to display short form of abbreviation in section title and table of contents.

```

10509 \newrobustcmd*{\lgsxtrtitleshort}[1]{%
10510   \glsxtrshort[noindex,hyper=false]{#1}[]%
10511 }

```

glsxtrheadshortpl Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10512 \newcommand*{\glsxtrheadshortpl}[1]{%
10513   \protect\NoCaseChange
10514   {%
10515     \glsifattribute{#1}{headuc}{true}%
10516     {%
10517       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%

```

```

10518 }%
10519 {%
10520 \glsxtrshortpl [noindex,hyper=false] {#1} []%
10521 }%
10522 }%
10523 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents.

```

10524 \newrobustcmd*{\glsxtrtitleshortpl} [1] {%
10525 \glsxtrshortpl [noindex,hyper=false] {#1} []%
10526 }

```

Glsxtrheadshort Command used to display short form in the page header with the first letter converted to upper case.

```

10527 \newcommand*{\Glsxtrheadshort} [1] {%
10528 \protect\NoCaseChange
10529 {%
10530 \glsifattribute{#1}{headuc}{true}%
10531 {%
10532 \GLSxtrshort [noindex,hyper=false] {#1} []%
10533 }%
10534 {%
10535 \Glsxtrshort [noindex,hyper=false] {#1} []%
10536 }%
10537 }%
10538 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10539 \newrobustcmd*{\Glsxtrtitleshort} [1] {%
10540 \Glsxtrshort [noindex,hyper=false] {#1} []%
10541 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

10542 \newcommand*{\Glsxtrheadshortpl} [1] {%
10543 \protect\NoCaseChange
10544 {%
10545 \glsifattribute{#1}{headuc}{true}%
10546 {%
10547 \GLSxtrshortpl [noindex,hyper=false] {#1} []%
10548 }%
10549 {%
10550 \Glsxtrshortpl [noindex,hyper=false] {#1} []%
10551 }%
10552 }%
10553 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10554 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10555   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10556 }
```

`\glxtrheadname` As above but for the name value.

```
10557 \newcommand*{\glxtrheadname}[1]{%
10558   \protect\NoCaseChange
10559   {%
10560     \glsifattribute{#1}{headuc}{true}%
10561     {%
10562       \GLSname[noindex,hyper=false]{#1}[]%
10563     }%
10564     {%
10565       \glsname[noindex,hyper=false]{#1}[]%
10566     }%
10567   }%
10568 }
```

`glxtrtitlename` Command to display name value in section title and table of contents.

```
10569 \newrobustcmd*{\glxtrtitlename}[1]{%
10570   \glsname[noindex,hyper=false]{#1}[]%
10571 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10572 \newcommand*{\Glsxtrheadname}[1]{%
10573   \protect\NoCaseChange
10574   {%
10575     \glsifattribute{#1}{headuc}{true}%
10576     {%
10577       \GLSname[noindex,hyper=false]{#1}[]%
10578     }%
10579     {%
10580       \Glsname[noindex,hyper=false]{#1}[]%
10581     }%
10582   }%
10583 }
```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
10584 %\changes{1.21}{2017-11-03}{new}
10585 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10586   \Glsname[noindex,hyper=false]{#1}[]%
10587 }
```

`\glxtrheadtext` As above but for the text value.

```
10588 \newcommand*{\glxtrheadtext}[1]{%
```

```

10589 \protect\NoCaseChange
10590 {%
10591   \glsifattribute{#1}{headuc}{true}%
10592   {%
10593     \GLStext[noindex,hyper=false]{#1}[]%
10594   }%
10595   {%
10596     \glstext[noindex,hyper=false]{#1}[]%
10597   }%
10598 }%
10599 }

```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```

10600 \newrobustcmd*{\glsxtrtitletext}[1]{%
10601   \glstext[noindex,hyper=false]{#1}[]%
10602 }

```

`\Glsxtrheadtext` First letter converted to upper case

```

10603 \newcommand*{\Glsxtrheadtext}[1]{%
10604   \protect\NoCaseChange
10605   {%
10606     \glsifattribute{#1}{headuc}{true}%
10607     {%
10608       \GLStext[noindex,hyper=false]{#1}[]%
10609     }%
10610     {%
10611       \Glstext[noindex,hyper=false]{#1}[]%
10612     }%
10613   }%
10614 }

```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```

10615 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10616   \Glstext[noindex,hyper=false]{#1}[]%
10617 }

```

`lslxtrheadplural` As above but for the plural value.

```

10618 \newcommand*{\glsxtrheadplural}[1]{%
10619   \protect\NoCaseChange
10620   {%
10621     \glsifattribute{#1}{headuc}{true}%
10622     {%
10623       \GLSplural[noindex,hyper=false]{#1}[]%
10624     }%
10625     {%
10626       \glsplural[noindex,hyper=false]{#1}[]%
10627     }%
10628   }%

```


10629 }

sxtrtitleplural Command to display plural value in section title and table of contents.

```
10630 \newrobustcmd*{\glsxtrtitleplural}[1]{%
10631   \glsplural[noindex,hyper=false]{#1}[]%
10632 }
```

lsxtrheadplural Convert first letter to upper case.

```
10633 \newcommand*{\Glsxtrheadplural}[1]{%
10634   \protect\NoCaseChange
10635   {%
10636     \glusifattribute{#1}{headuc}{true}%
10637     {%
10638       \GLSplural[noindex,hyper=false]{#1}[]%
10639     }%
10640     {%
10641       \Glsplural[noindex,hyper=false]{#1}[]%
10642     }%
10643   }%
10644 }
```

sxtrtitleplural Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
10645 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10646   \Glsplural[noindex,hyper=false]{#1}[]%
10647 }
```

glsxtrheadfirst As above but for the first value.

```
10648 \newcommand*{\glsxtrheadfirst}[1]{%
10649   \protect\NoCaseChange
10650   {%
10651     \glusifattribute{#1}{headuc}{true}%
10652     {%
10653       \GLSfirst[noindex,hyper=false]{#1}[]%
10654     }%
10655     {%
10656       \glsfirst[noindex,hyper=false]{#1}[]%
10657     }%
10658   }%
10659 }
```

lsxtrtitlefirst Command to display first value in section title and table of contents.

```
10660 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10661   \glsfirst[noindex,hyper=false]{#1}[]%
10662 }
```

Glsxtrheadfirst First letter converted to upper case

```
10663 \newcommand*{\Glsxtrheadfirst}[1]{%
```

```

10664 \protect\NoCaseChange
10665 {%
10666   \glsifattribute{#1}{headuc}{true}%
10667   {%
10668     \GLSfirst[noindex,hyper=false]{#1}[]%
10669   }%
10670   {%
10671     \Glsfirst[noindex,hyper=false]{#1}[]%
10672   }%
10673 }%
10674 }

```

lsxtrtitlefirst Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10675 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10676   \GLSfirst[noindex,hyper=false]{#1}[]%
10677 }

```

headfirstplural As above but for the firstplural value.

```

10678 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10679   \protect\NoCaseChange
10680   {%
10681     \glsifattribute{#1}{headuc}{true}%
10682     {%
10683       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10684     }%
10685     {%
10686       \glsfirstplural[noindex,hyper=false]{#1}[]%
10687     }%
10688   }%
10689 }

```

itlefirstplural Command to display firstplural value in section title and table of contents.

```

10690 \newrobustcmd*{\Glsxtritlefirstplural}[1]{%
10691   \glsfirstplural[noindex,hyper=false]{#1}[]%
10692 }

```

headfirstplural First letter converted to upper case

```

10693 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10694   \protect\NoCaseChange
10695   {%
10696     \glsifattribute{#1}{headuc}{true}%
10697     {%
10698       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10699     }%
10700     {%
10701       \Glsfirstplural[noindex,hyper=false]{#1}[]%
10702     }%
10703   }%

```

10704 }

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10705 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10706   \Glsfirstplural[noindex,hyper=false]{#1}[]%
10707 }
```

`\glxtrheadlong` Command used to display long form in the page header.

```
10708 \newcommand*{\glxtrheadlong}[1]{%
10709   \protect\NoCaseChange
10710   {%
10711     \glsifattribute{#1}{headuc}{true}%
10712     {%
10713       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10714     }%
10715     {%
10716       \glxtrlong[noindex,hyper=false]{#1}[]%
10717     }%
10718   }%
10719 }
```

`\glxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
10720 \newrobustcmd*{\glxtrtitlelong}[1]{%
10721   \glxtrlong[noindex,hyper=false]{#1}[]%
10722 }
```

`\glxtrheadlongpl` 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.

```
10723 \newcommand*{\glxtrheadlongpl}[1]{%
10724   \protect\NoCaseChange
10725   {%
10726     \glsifattribute{#1}{headuc}{true}%
10727     {%
10728       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10729     }%
10730     {%
10731       \glxtrlongpl[noindex,hyper=false]{#1}[]%
10732     }%
10733   }%
10734 }
```

`\glxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
10735 \newrobustcmd*{\glxtrtitlelongpl}[1]{%
10736   \glxtrlongpl[noindex,hyper=false]{#1}[]%
10737 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
10738 \newcommand*{\Glsxtrheadlong}[1]{%
10739   \protect\NoCaseChange
10740   {%
10741     \glsifattribute{#1}{headuc}{true}%
10742     {%
10743       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10744     }%
10745     {%
10746       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10747     }%
10748   }%
10749 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10750 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10751   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10752 }
```

`lgsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```
10753 \newcommand*{\Glsxtrheadlongpl}[1]{%
10754   \protect\NoCaseChange
10755   {%
10756     \glsifattribute{#1}{headuc}{true}%
10757     {%
10758       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10759     }%
10760     {%
10761       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10762     }%
10763   }%
10764 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10765 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10766   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10767 }
```

`\glsxtrheadfull` Command used to display full form in the page header.

```
10768 \newcommand*{\glsxtrheadfull}[1]{%
10769   \protect\NoCaseChange
10770   {%
10771     \glsifattribute{#1}{headuc}{true}%
10772     {%
```

```

10773 \GLSxtrfull[noindex,hyper=false]{#1}[]%
10774 }%
10775 {%
10776 \glsxtrfull[noindex,hyper=false]{#1}[]%
10777 }%
10778 }%
10779 }

```

glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents.

```

10780 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10781 \glsxtrfull[noindex,hyper=false]{#1}[]%
10782 }

```

glsxtrheadfullpl Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10783 \newcommand*{\glsxtrheadfullpl}[1]{%
10784 \protect\NoCaseChange
10785 {%
10786 \glsifattribute{#1}{headuc}{true}%
10787 {%
10788 \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10789 }%
10790 {%
10791 \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10792 }%
10793 }%
10794 }

```

glsxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

10795 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
10796 \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10797 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

10798 \newcommand*{\Glsxtrheadfull}[1]{%
10799 \protect\NoCaseChange
10800 {%
10801 \glsifattribute{#1}{headuc}{true}%
10802 {%
10803 \GLSxtrfull[noindex,hyper=false]{#1}[]%
10804 }%
10805 {%
10806 \Glsxtrfull[noindex,hyper=false]{#1}[]%
10807 }%
10808 }%
10809 }

```

`\Glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10810 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10811   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10812 }
```

`\Glsxtrheadfullpl` Command used to display plural full form in the page header with the first letter converted to upper case.

```
10813 \newcommand*{\Glsxtrheadfullpl}[1]{%
10814   \protect\NoCaseChange
10815   {%
10816     \glsifattribute{#1}{headuc}{true}%
10817     {%
10818       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10819     }%
10820     {%
10821       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10822     }%
10823   }%
10824 }
```

`\Glsxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10825 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10826   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10827 }
```

`\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.

```
10828 \ifdef\texorpdfstring
10829 {
10830   \newcommand*{\glsfmtshort}[1]{%
10831     \texorpdfstring
10832       {\glsxtrtitleshort{#1}}%
10833       {\glsentryshort{#1}}%
10834   }
10835 }
10836 {
10837   \newcommand*{\glsfmtshort}[1]{%
10838     \glsxtrtitleshort{#1}}
10839 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
10840 \ifdef\texorpdfstring
10841 {
10842   \newcommand*{\glsfmtshortpl}[1]{%
```

```

10843 \texorpdfstring
10844     {\glxtrtitleshortpl{#1}}}%
10845     {\glsentryshortpl{#1}}}%
10846 }
10847 }
10848 {
10849 \newcommand*{\glsfmtshortpl}[1]{%
10850 \glxtrtitleshortpl{#1}}
10851 }

```

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).

```

10852 \ifdef\texorpdfstring
10853 {
10854 \newcommand*{\Glsfmtshort}[1]{%
10855 \texorpdfstring
10856     {\Glsxtrtitleshort{#1}}}%
10857     {\glsentryshort{#1}}}%
10858 }
10859 }
10860 {
10861 \newcommand*{\Glsfmtshort}[1]{%
10862 \Glsxtrtitleshort{#1}}
10863 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10864 \ifdef\texorpdfstring
10865 {
10866 \newcommand*{\Glsfmtshortpl}[1]{%
10867 \texorpdfstring
10868     {\Glsxtrtitleshortpl{#1}}}%
10869     {\glsentryshortpl{#1}}}%
10870 }
10871 }
10872 {
10873 \newcommand*{\Glsfmtshortpl}[1]{%
10874 \Glsxtrtitleshortpl{#1}}
10875 }

```

`\glsfmtname` As above but for the name value.

```

10876 \ifdef\texorpdfstring
10877 {
10878 \newcommand*{\glsfmtname}[1]{%
10879 \texorpdfstring
10880     {\glsxtrtitlename{#1}}}%
10881     {\glsentryname{#1}}}%
10882 }

```

```

10883 }
10884 {
10885   \newcommand*{\Glsfmtname}[1]{%
10886     \Glsxtrtitlename{#1}}
10887 }

```

`\Glsfmtname` First letter converted to upper case.

```

10888 \ifdef\texorpdfstring
10889 {
10890   \newcommand*{\Glsfmtname}[1]{%
10891     \texorpdfstring
10892     {\Glsxtrtitlename{#1}}%
10893     {\Glsentryname{#1}}%
10894   }
10895 }
10896 {
10897   \newcommand*{\Glsfmtname}[1]{%
10898     \Glsxtrtitlename{#1}}
10899 }

```

`\glsfmttext` As above but for the text value.

```

10900 \ifdef\texorpdfstring
10901 {
10902   \newcommand*{\glsfmttext}[1]{%
10903     \texorpdfstring
10904     {\Glsxtrtitletext{#1}}%
10905     {\Glsentrytext{#1}}%
10906   }
10907 }
10908 {
10909   \newcommand*{\glsfmttext}[1]{%
10910     \Glsxtrtitletext{#1}}
10911 }

```

`\Glsfmttext` First letter converted to upper case.

```

10912 \ifdef\texorpdfstring
10913 {
10914   \newcommand*{\Glsfmttext}[1]{%
10915     \texorpdfstring
10916     {\Glsxtrtitletext{#1}}%
10917     {\Glsentrytext{#1}}%
10918   }
10919 }
10920 {
10921   \newcommand*{\Glsfmttext}[1]{%
10922     \Glsxtrtitletext{#1}}
10923 }

```

`\glsfmtplural` As above but for the plural value.


```

10924 \ifdef\texorpdfstring
10925 {
10926   \newcommand*\glsfmtplural}[1]{%
10927     \texorpdfstring
10928     {\glsxtrtitleplural{#1}}%
10929     {\glsentryplural{#1}}%
10930   }
10931 }
10932 {
10933   \newcommand*\glsfmtplural}[1]{%
10934     \glsxtrtitleplural{#1}}
10935 }

```

`\Glsfmtplural` First letter converted to upper case.

```

10936 \ifdef\texorpdfstring
10937 {
10938   \newcommand*\Glsfmtplural}[1]{%
10939     \texorpdfstring
10940     {\Glsxtrtitleplural{#1}}%
10941     {\glsentryplural{#1}}%
10942   }
10943 }
10944 {
10945   \newcommand*\Glsfmtplural}[1]{%
10946     \Glsxtrtitleplural{#1}}
10947 }

```

`\glsfmtfirst` As above but for the first value.

```

10948 \ifdef\texorpdfstring
10949 {
10950   \newcommand*\glsfmtfirst}[1]{%
10951     \texorpdfstring
10952     {\glsxtrtitlefirst{#1}}%
10953     {\glsentryfirst{#1}}%
10954   }
10955 }
10956 {
10957   \newcommand*\glsfmtfirst}[1]{%
10958     \glsxtrtitlefirst{#1}}
10959 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

10960 \ifdef\texorpdfstring
10961 {
10962   \newcommand*\Glsfmtfirst}[1]{%
10963     \texorpdfstring
10964     {\Glsxtrtitlefirst{#1}}%
10965     {\glsentryfirst{#1}}%
10966   }

```

```

10967 }
10968 {
10969   \newcommand*{\Glsfmtfirst}[1]{%
10970     \Glsxtrtitlefirst{#1}}
10971 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

10972 \ifdef\texorpdfstring
10973 {
10974   \newcommand*{\glsfmtfirstpl}[1]{%
10975     \texorpdfstring
10976     {\Glsxtrtitlefirstplural{#1}}%
10977     {\glsentryfirstplural{#1}}%
10978   }
10979 }
10980 {
10981   \newcommand*{\glsfmtfirstpl}[1]{%
10982     \Glsxtrtitlefirstplural{#1}}
10983 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

10984 \ifdef\texorpdfstring
10985 {
10986   \newcommand*{\Glsfmtfirstpl}[1]{%
10987     \texorpdfstring
10988     {\Glsxtrtitlefirstplural{#1}}%
10989     {\glsentryfirstplural{#1}}%
10990   }
10991 }
10992 {
10993   \newcommand*{\Glsfmtfirstpl}[1]{%
10994     \Glsxtrtitlefirstplural{#1}}
10995 }

```

`\glsfmtlong` As above but for the long value.

```

10996 \ifdef\texorpdfstring
10997 {
10998   \newcommand*{\glsfmtlong}[1]{%
10999     \texorpdfstring
11000     {\Glsxtrtitlelong{#1}}%
11001     {\glsentrylong{#1}}%
11002   }
11003 }
11004 {
11005   \newcommand*{\glsfmtlong}[1]{%
11006     \Glsxtrtitlelong{#1}}
11007 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11008 \ifdef\teorpdfstring
11009 {
11010   \newcommand*{\Glsfmtlong}[1]{%
11011     \teorpdfstring
11012     {\Glsxtrtitlelong{#1}}%
11013     {\glsentrylong{#1}}%
11014   }
11015 }
11016 {
11017   \newcommand*{\Glsfmtlong}[1]{%
11018     \Glsxtrtitlelong{#1}}
11019 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11020 \ifdef\teorpdfstring
11021 {
11022   \newcommand*{\glsfmtlongpl}[1]{%
11023     \teorpdfstring
11024     {\glsxtrtitlelongpl{#1}}%
11025     {\glsentrylongpl{#1}}%
11026   }
11027 }
11028 {
11029   \newcommand*{\glsfmtlongpl}[1]{%
11030     \glsxtrtitlelongpl{#1}}
11031 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11032 \ifdef\teorpdfstring
11033 {
11034   \newcommand*{\Glsfmtlongpl}[1]{%
11035     \teorpdfstring
11036     {\Glsxtrtitlelongpl{#1}}%
11037     {\glsentrylongpl{#1}}%
11038   }
11039 }
11040 {
11041   \newcommand*{\Glsfmtlongpl}[1]{%
11042     \Glsxtrtitlelongpl{#1}}
11043 }

```

`\glsfmtfull` In-line full format.

```

11044 \ifdef\teorpdfstring
11045 {
11046   \newcommand*{\glsfmtfull}[1]{%
11047     \teorpdfstring
11048     {\glsxtrtitlefull{#1}}%
11049     {\glsxtrinlinefullformat{#1}{}}%
11050   }

```

```

11051 }
11052 {
11053   \newcommand*{\glsfmtfull}[1]{%
11054     \glsxtrtitlefull{#1}}
11055 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11056 \ifdef\teorpdfstring
11057 {
11058   \newcommand*{\Glsfmtfull}[1]{%
11059     \teorpdfstring
11060     {\Glsxtrtitlefull{#1}}%
11061     {\Glsxtrinlinefullformat{#1}{}}%
11062   }
11063 }
11064 {
11065   \newcommand*{\Glsfmtfull}[1]{%
11066     \Glsxtrtitlefull{#1}}
11067 }

```

`\glsfmtfullpl` In-line full plural format.

```

11068 \ifdef\teorpdfstring
11069 {
11070   \newcommand*{\glsfmtfullpl}[1]{%
11071     \teorpdfstring
11072     {\glsxtrtitlefullpl{#1}}%
11073     {\glsxtrinlinefullplformat{#1}{}}%
11074   }
11075 }
11076 {
11077   \newcommand*{\glsfmtfullpl}[1]{%
11078     \glsxtrtitlefullpl{#1}}
11079 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11080 \ifdef\teorpdfstring
11081 {
11082   \newcommand*{\Glsfmtfullpl}[1]{%
11083     \teorpdfstring
11084     {\Glsxtrtitlefullpl{#1}}%
11085     {\Glsxtrinlinefullplformat{#1}{}}%
11086   }
11087 }
11088 {
11089   \newcommand*{\Glsfmtfullpl}[1]{%
11090     \Glsxtrtitlefullpl{#1}}
11091 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

sariesExtraLang

```
11092 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11093   \@ifundefined{ver@glossariesxtr-#1.1df}{\input{glossariesxtr-#1.1df}}{}%
11094 }
```

sariesExtraLang

```
11095 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11096   \ProvidesFile{glossariesxtr-#1.1df}%
11097 }
```

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.)

```
11098 \@ifpackageloaded{tracklang}
11099 {%
11100   \AnyTrackedLanguages
11101   {%
11102     \ForEachTrackedDialect{\this@dialect}{%
11103       \IfTrackedLanguageFileExists{\this@dialect}%
11104       {glossariesxtr-}% prefix
11105       {.1df}%
11106       {%
11107         \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11108       }%
11109     }%
11110   }%
11111 }%
11112 }%
11113 {}%
11114 }
11115 {}
```

Load `glossaries-extra-stylemods` if required.

```
11116 \@glsxtr@redefstyles
```

and set the style:

```
11117 \@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:

```
11118 \NeedsTeXFormat{LaTeX2e}
11119 \ProvidesPackage{glossaries-extra-stylemods}[2017/11/12 v1.23 (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`

```
11120 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
11121 \DeclareOption{all}{%
11122   \appto\@glxtr@loadstyles{%
11123     \RequirePackage{glossary-inline}%
11124     \RequirePackage{glossary-list}%
11125     \RequirePackage{glossary-tree}%
11126     \RequirePackage{glossary-mcols}%
11127     \RequirePackage{glossary-long}%
11128     \RequirePackage{glossary-longragged}%
11129     \RequirePackage{glossary-longbooktabs}%
11130     \RequirePackage{glossary-super}%
11131     \RequirePackage{glossary-superragged}%
11132     \RequirePackage{glossary-bookindex}%
11133   }
11134 }

11135 \DeclareOption*{%
11136   \IfFileExists{glossary-\CurrentOption.sty}
11137   {\eappto\@glxtr@loadstyles{%
11138     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
11139   }%
11140   {%
11141     \PackageError{glossaries-extra-styles}%

```

```

11142     {Unknown option ‘\CurrentOption’}{}%
11143   }%
11144 }

```

Process the package options:

```
11145 \ProcessOptions
```

Load the required packages:

```
11146 \@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`.

```
11147 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

11148 \providecommand{\renewglossarystyle}[2]{%
11149   \ifcsundef{@glsstyle@#1}%
11150   {%
11151     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
11152   }%
11153   {%
11154     \csdef{@glsstyle@#1}{#2}%
11155   }%
11156 }

```

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.

```

11157 \ifdef{@glsstyle@listdotted}
11158 {%
11159   \renewglossarystyle{listdotted}{%
11160     \setglossarystyle{list}%
11161     \renewcommand*{\glossentry}[2]{%
11162       \item[]\makebox[\glslistdottedwidth][l]{%
11163         \glsentryitem{##1}%
11164         \glstarget{##1}{\glossentryname{##1}}%
11165         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11166         \glossentrydesc{##1}\glspostdescription}%
11167     \renewcommand*{\subglossentry}[3]{%
11168       \item[]\makebox[\glslistdottedwidth][l]{%
11169         \glssubentryitem{##2}%
11170         \glstarget{##2}{\glossentryname{##2}}%
11171         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11172         \glossentrydesc{##2}\glspostdescription}%
11173   }

```

```
11174 }
11175 {%
```

Assume the style isn't required if it hasn't already been defined.

```
11176 }
```

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.

```
11177 \ifdef{\@glsstyle@list}
11178 {%
```

listprelocation Space before number list for top-level entries.

```
11179 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11180 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11181 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11182 \renewglossarystyle{list}{%
11183 \renewenvironment{theglossary}%
11184 {\begin{description}}{\end{description}}%
11185 \renewcommand*\glossaryheader{}%
11186 \renewcommand*\glsgroupheading}[1]{%
11187 \renewcommand*\glossentry}[2]{%
11188 \item[\glsentryitem{##1}%
11189 \glstarget{##1}{\glossentryname{##1}}]
11190 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11191 \renewcommand*\subglossentry}[3]{%
11192 \glssubentryitem{##2}%
11193 \glstarget{##2}{\strut}\space
11194 \glossentrydesc{##2}\glspostdescription
11195 \glslistchildprelocation ##3\glslistchildpostlocation}%
11196 \renewcommand*\glsgroupskip{\ifglsnogroupskip\else\indexspace\fi}%
11197 }
11198 }
11199 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11200 \ifdef{\@glsstyle@altlist}
11201 {%
11202 \renewglossarystyle{altlist}{%
11203 \setglossarystyle{list}%
11204 \renewcommand*\glossentry}[2]{%
11205 \item[\glsentryitem{##1}%
```



```

11206      \glstarget{##1}{\glossentryname{##1}}}%
11207      \mbox{}\par\nobreak\@afterheading
11208      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11209      \renewcommand{\subglossentry}[3]{%
11210        \par
11211        \glssubentryitem{##2}%
11212        \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11213        \glslistchildprelocation ##3}%
11214    }
11215 }
11216 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11217 \ifdef{\@glsstyle@listgroup}
11218 {%
11219   \renewglossarystyle{listgroup}{%
11220     \setglossarystyle{list}%
11221     \renewcommand*\glsgroupheading[1]{%
11222       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11223       \mbox{}\par\nobreak\@afterheading
11224     }%
11225   }
11226 }
11227 {}

```

Similarly for listhypergroup.

```

11228 \ifdef{\@glsstyle@listhypergroup}
11229 {%
11230   \renewglossarystyle{listhypergroup}{%
11231     \setglossarystyle{list}%
11232     \renewcommand*\glossaryheader{%
11233       \glslistnavigationitem{\glsnavigation}}%
11234     \renewcommand*\glsgroupheading[1]{%
11235       \item[\glslistgroupheaderfmt
11236         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11237       \mbox{}\par\nobreak\@afterheading
11238     }%
11239   }
11240 }
11241 {}

```

Similarly for altlistgroup.

```

11242 \ifdef{\@glsstyle@altlistgroup}
11243 {%
11244   \renewglossarystyle{altlistgroup}{%
11245     \setglossarystyle{altlist}%
11246     \renewcommand*\glsgroupheading[1]{%
11247       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11248       \mbox{}\par\nobreak\@afterheading
11249     }%
11250   }

```

```

11251 }
11252 {}

    Similarly for altlisthypergroup.
11253 \ifdef{\@glsstyle@altlisthypergroup}
11254 {%
11255   \renewglossarystyle{altlisthypergroup}{%
11256     \setglossarystyle{altlist}%
11257     \renewcommand*\{glossaryheader\}{%
11258       \glslistnavigationitem{\glsnavigation}}%
11259     \renewcommand*\{glsgroupheading}[1]{%
11260       \item[\glslistgroupheaderfmt
11261         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11262       \mbox{}\par\nobreak\@afterheading
11263     }%
11264   }
11265 }
11266 {}

```

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`.

```

11267 \ifcsdef{@glsstyle@long}
11268 {%
11269   \renewglossarystyle{long}{%
11270     \renewenvironment{theglossary}%
11271       {\begin{longtable}{lp{\glsdescwidth}}}%
11272       {\end{longtable}}%
11273     \renewcommand*\{glossaryheader\}{}%
11274     \renewcommand*\{glsgroupheading}[1]{}%
11275     \renewcommand{\glossentry}[2]{%
11276       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11277       \glossentrydesc{##1}\glspostdescription
11278       \glstrprelocation ##2\tabularnewline
11279     }%
11280     \renewcommand{\subglossentry}[3]{%
11281       &
11282       \glssubentryitem{##2}%
11283       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11284       \glstrprelocation ##3\tabularnewline
11285     }%
11286     \ifglsnogroupskip
11287       \renewcommand*\{glsgroupskip\}{}%
11288     \else
11289       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11290     \fi
11291   }

```

```
11292 }
11293 {}
```

Three column style:

```
11294 \ifcsdef{@glsstyle@long3col}
11295 {%
11296   \renewglossarystyle{long3col}{%
11297     \renewenvironment{theglossary}%
11298       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
11299       {\end{longtable}}}%
11300   \renewcommand*{\glossaryheader}{}%
11301   \renewcommand*{\glsgroupheading}[1]{}%
11302   \renewcommand{\glossentry}[2]{%
11303     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11304     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11305   }%
11306   \renewcommand{\subglossentry}[3]{%
11307     &
11308     \glssubentryitem{##2}%
11309     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11310     ##3\tabularnewline
11311   }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11312   \ifglsnogroupskip
11313     \renewcommand*{\glsgroupskip}{}%
11314   \else
11315     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11316   \fi
11317 }
11318 }
11319 {}
```

Four column style:

```
11320 \ifcsdef{@glsstyle@long4col}
11321 {%
11322   \renewglossarystyle{long4col}{%
11323     \renewenvironment{theglossary}%
11324       {\begin{longtable}{llll}}%
11325       {\end{longtable}}}%
11326   \renewcommand*{\glossaryheader}{}%
11327   \renewcommand*{\glsgroupheading}[1]{}%
11328   \renewcommand{\glossentry}[2]{%
11329     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11330     \glossentrydesc{##1}\glspostdescription &
11331     \glossentrysymbol{##1} &
11332     ##2\tabularnewline
11333   }%
11334   \renewcommand{\subglossentry}[3]{%
11335     &
```

```

11336     \glssubentryitem{##2}%
11337     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11338     \glossentrysymbol{##2} & ##3\tabularnewline
11339 }%

11340 \ifglsgroupskip
11341     \renewcommand*{\glsgroupskip}{}%
11342 \else
11343     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11344 \fi
11345 }
11346 }
11347 {}

```

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`.

```

11348 \ifcsdef{@glstyle@longragged}
11349 {%
11350     \renewglossarystyle{longragged}{%
11351         \renewenvironment{theglossary}%
11352             {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}]}%
11353             {\end{longtable}}%
11354         \renewcommand*{\glossaryheader}{}%
11355         \renewcommand*{\glsgroupheading}[1]{}%
11356         \renewcommand{\glossentry}[2]{%
11357             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11358             \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11359             \tabularnewline
11360         }%
11361         \renewcommand{\subglossentry}[3]{%
11362             &
11363             \glssubentryitem{##2}%
11364             \glstarget{##2}{\strut}\glossentrydesc{##2}%
11365             \glspostdescription\glxtrprelocation ##3%
11366             \tabularnewline
11367         }%
11368         \ifglsgroupskip
11369             \renewcommand*{\glsgroupskip}{}%
11370         \else
11371             \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11372         \fi
11373     }
11374 }

```

11375 {}

Three and four column styles don't use \glxstrprelocation since the number list is in its own column.

```
11376 \ifcsdef{@glsstyle@longragged3col}
11377 {%
11378   \renewglossarystyle{longragged3col}{%
11379     \renewenvironment{theglossary}%
11380       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11381         >{\raggedright}p{\glspagelistwidth}}}%
11382       {\end{longtable}}}%
11383     \renewcommand*{\glossaryheader}{}%
11384     \renewcommand*{\glsgroupheading}[1]{}%
11385     \renewcommand{\glossentry}[2]{%
11386       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11387       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11388     }%
11389     \renewcommand{\subglossentry}[3]{%
11390       &
11391       \glssubentryitem{##2}%
11392       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11393       ##3\tabularnewline
11394     }%
11395   \ifglsgroupskip
11396     \renewcommand*{\glsgroupskip}{}%
11397   \else
11398     \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11399   \fi
11400 }
11401 }
11402 {}
```

Four column style:

```
11403 \ifcsdef{@glsstyle@altlongragged4col}
11404 {%
11405   \renewglossarystyle{altlongragged4col}{%
11406     \renewenvironment{theglossary}%
11407       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11408         >{\raggedright}p{\glspagelistwidth}}}%
11409       {\end{longtable}}}%
11410     \renewcommand*{\glossaryheader}{}%
11411     \renewcommand*{\glsgroupheading}[1]{}%
11412     \renewcommand{\glossentry}[2]{%
11413       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11414       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11415       ##2\tabularnewline
11416     }%
11417     \renewcommand{\subglossentry}[3]{%
11418       &
```

```

11419     \glssubentryitem{##2}%
11420     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11421     \glossentrysymbol{##2} & ##3\tabularnewline
11422 }%

11423 \ifglsnogroupskip
11424     \renewcommand*{\glsgroupskip}{}%
11425 \else
11426     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11427 \fi
11428 }
11429 }
11430 {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glstrprelocation`.

```

11431 \ifcsdef{@glstyle@super}
11432 {%
11433     \renewglossarystyle{super}{%
11434         \renewenvironment{theglossary}%
11435             {\tablehead{}\tabletail}%
11436             \begin{supertabular}{lp{\glsdescwidth}}%
11437             {\end{supertabular}}%
11438     \renewcommand*{\glossaryheader}{}%
11439     \renewcommand*{\glsgroupheading}[1]{}%
11440     \renewcommand{\glossentry}[2]{%
11441         \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11442         \glossentrydesc{##1}\glspostdescription
11443         \glstrprelocation ##2\tabularnewline
11444     }%
11445     \renewcommand{\subglossentry}[3]{%
11446         &
11447         \glssubentryitem{##2}%
11448         \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11449         \glstrprelocation ##3\tabularnewline
11450     }%
11451     \ifglsnogroupskip
11452         \renewcommand*{\glsgroupskip}{}%
11453     \else
11454         \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11455     \fi
11456 }
11457 }
11458 {}

```

Three column style:

```

11459 \ifcsdef{@glstyle@super3col}

```

```

11460 {%
11461   \renewglossarystyle{super3col}{%
11462     \renewenvironment{theglossary}%
11463       {\tablehead{}\tabletail{}}%
11464       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
11465     {\end{supertabular}}%
11466     \renewcommand*{\glossaryheader}{}%
11467     \renewcommand*{\glsgroupheading}[1]{}%
11468     \renewcommand{\glossentry}[2]{%
11469       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11470       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11471     }%
11472     \renewcommand{\subglossentry}[3]{%
11473       &
11474       \glssubentryitem{##2}%
11475       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11476       ##3\tabularnewline
11477     }%

11478     \ifglsgnogroupskip
11479       \renewcommand*{\glsgroupskip}{}%
11480     \else
11481       \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11482     \fi
11483   }
11484 }
11485 {}

```

Four column styles:

```

11486 \ifcsdef{@glstyle@super4col}
11487 {%
11488   \renewglossarystyle{super4col}{%
11489     \renewenvironment{theglossary}%
11490       {\tablehead{}\tabletail{}}%
11491       \begin{supertabular}{llll}}{%
11492     \end{supertabular}}%
11493     \renewcommand*{\glossaryheader}{}%
11494     \renewcommand*{\glsgroupheading}[1]{}%
11495     \renewcommand{\glossentry}[2]{%
11496       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11497       \glossentrydesc{##1}\glspostdescription &
11498       \glossentrysymbol{##1} & ##2\tabularnewline
11499     }%
11500     \renewcommand{\subglossentry}[3]{%
11501       &
11502       \glssubentryitem{##2}%
11503       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11504       \glossentrysymbol{##2} & ##3\tabularnewline
11505     }%

```

```

11506 \ifglsgroupskip
11507 \renewcommand*{\glsgroupskip}{}%
11508 \else
11509 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11510 \fi
11511 }
11512 }
11513 {}

```

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`.

```

11514 \ifcsdef{@glstyle@superragged}
11515 {%
11516 \renewglossarystyle{superragged}{%
11517 \renewenvironment{theglossary}%
11518 {\tablehead{}}\tabletail{}}%
11519 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11520 {\end{supertabular}}%
11521 \renewcommand*{\glossaryheader}{}%
11522 \renewcommand*{\glsgroupheading}[1]{}%
11523 \renewcommand{\glossentry}[2]{%
11524 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11525 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11526 \tabularnewline
11527 }%
11528 \renewcommand{\subglossentry}[3]{%
11529 &
11530 \glssubentryitem{##2}%
11531 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11532 \glstrprelocation ##3%
11533 \tabularnewline
11534 }%
11535 \ifglsgroupskip
11536 \renewcommand*{\glsgroupskip}{}%
11537 \else
11538 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11539 \fi
11540 }
11541 }
11542 {}

```

Three column style:

```

11543 \ifcsdef{@glstyle@superragged3col}
11544 {%
11545 \renewglossarystyle{superragged3col}{%

```



```

11546 \renewenvironment{theglossary}%
11547   {\tablehead{}\tabletail{}}%
11548   \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}%
11549     >\raggedright}p{\glspagelistwidth}}}%
11550   {\end{supertabular}}%
11551 \renewcommand*{\glossaryheader}{}%
11552 \renewcommand*{\glsgroupheading}[1]{}%
11553 \renewcommand{\glossentry}[2]{%
11554   \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11555   \glossentrydesc{##1}\glspostdescription &
11556   ##2\tabularnewline
11557 }%
11558 \renewcommand{\subglossentry}[3]{%
11559   &
11560   \glssubentryitem{##2}%
11561   \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11562   ##3\tabularnewline
11563 }%

11564 \ifglsgroupskip
11565   \renewcommand*{\glsgroupskip}{}%
11566 \else
11567   \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
11568 \fi
11569 }
11570 }
11571 {}

```

Four columns:

```

11572 \ifcsdef{@glstyle@altsuperragged4col}
11573 {%
11574   \renewglossarystyle{altsuperragged4col}{%
11575     \renewenvironment{theglossary}%
11576       {\tablehead{}\tabletail{}}%
11577       \begin{supertabular}{1>\raggedright}p{\glsgdescwidth}l%
11578         >\raggedright}p{\glspagelistwidth}}}%
11579       {\end{supertabular}}%
11580   \renewcommand*{\glossaryheader}{}%
11581   \renewcommand{\glossentry}[2]{%
11582     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11583     \glossentrydesc{##1}\glspostdescription &
11584     \glossentrysymbol{##1} & ##2\tabularnewline
11585   }%
11586   \renewcommand{\subglossentry}[3]{%
11587     &
11588     \glssubentryitem{##2}%
11589     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11590     \glossentrysymbol{##2} & ##3\tabularnewline
11591   }%

```

```

11592 \ifglsnogroupskip
11593 \renewcommand*{\glsgroupskip}{}%
11594 \else
11595 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11596 \fi
11597 }
11598 }
11599 {}

```

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`.

```

11600 \ifdef{\@glsstyle@inline}
11601 {%
11602 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11603 \renewcommand*{\glsinlinedescformat}[3]{%
11604 \space#1\glxtrpostdescription}
11605 \renewcommand*{\glsinlinesubdescformat}[3]{%
11606 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11607 }
11608 {}

```

2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11609 \ifdef{\@glsstyle@index}
11610 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11611 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11612 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

11613 \renewglossarystyle{index}{%
11614 \renewenvironment{theglossary}%
11615 {\setlength{\parindent}{0pt}%
11616 \setlength{\parskip}{0pt plus 0.3pt}%
11617 \let\item\glstreeitem
11618 \let\subitem\glstreesubitem

```

```

11619     \let\subsubitem\glstreesubsubitem
11620   }%
11621   {\par}%
11622   \renewcommand*\{\glossaryheader}{}%
11623   \renewcommand*\{\glsgroupheading}[1]{}%
11624   \renewcommand*\{\glossentry}[2]{%
11625     \item\glstentryitem{##1}%
11626     \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11627     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11628     \glstreepredesc \glossentrydesc{##1}\glspostdescription
11629     \glstreeprelocation ##2%
11630   }%
11631   \renewcommand{\subglossentry}[3]{%
11632     \ifcase##1\relax
11633       \item
11634     \or
11635       \subitem
11636       \glssubentryitem{##2}%
11637     \else
11638       \subsubitem
11639     \fi
11640     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11641     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11642     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11643     \glstreechildprelocation ##3%
11644   }%
11645   \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11646 }
11647 }
11648 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11649 \ifdef{\@glsstyle@indexgroup}
11650 {%
11651   \renewglossarystyle{indexgroup}{%
11652     \setglossarystyle{index}%
11653     \renewcommand*\{\glsgroupheading}[1]{%
11654       \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}%
11655       \nopagebreak\indexspace
11656       \nobreak\@afterheading
11657     }%
11658   }
11659 }
11660 {}

```

Similarly for indexhypergroup.

```

11661 \ifdef{\@glsstyle@indexhypergroup}
11662 {%
11663   \renewglossarystyle{indexhypergroup}{%
11664     \setglossarystyle{index}%

```

```

11665 \renewcommand*{\glossaryheader}{%
11666 \item\glstreenavigationfmt{\glsnavigation}%
11667 \nobreak\@afterheading\indexspace}%
11668 \renewcommand*{\glsgroupheading}[1]{%
11669 \item\glstreegroupheaderfmt
11670 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
11671 \nopagebreak\indexspace
11672 \nobreak\@afterheading}%
11673 }%
11674 }
11675 {}

```

Adjust tree style to remove hard coded space before number list.

```

11676 \ifdef{\@glsstyle@tree}
11677 {%
11678 \renewglossarystyle{tree}{%
11679 \renewenvironment{theglossary}%
11680 {\setlength{\parindent}{0pt}%
11681 \setlength{\parskip}{0pt plus 0.3pt}}%
11682 {}%
11683 \renewcommand*{\glossaryheader}{}%
11684 \renewcommand*{\glsgroupheading}[1]{}%
11685 \renewcommand{\glossentry}[2]{%
11686 \hangindent0pt\relax
11687 \parindent0pt\relax
11688 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11689 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11690 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11691 \glstreeprelocation##2\par
11692 }%
11693 \renewcommand{\subglossentry}[3]{%
11694 \hangindent##1\glstreeindent\relax
11695 \parindent##1\glstreeindent\relax
11696 \ifnum##1=1\relax
11697 \glssubentryitem{##2}%
11698 \fi
11699 \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11700 \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11701 \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11702 \glstreechildprelocation ##3\par
11703 }%
11704 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11705 }%
11706 }
11707 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11708 \ifdef{\@glsstyle@treegroup}
11709 {%
11710 \renewglossarystyle{treegroup}{%

```

```

11711 \setglossarystyle{tree}%
11712 \renewcommand{\glsgroupheading}[1]{\par
11713 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11714 \nopagebreak\indexspace\nobreak\@afterheading}%
11715 }
11716 }
11717 {}

```

Similarly for treehypergroup

```

11718 \ifdef{\@glsstyle@treehypergroup}
11719 {%
11720 \renewglossarystyle{treehypergroup}{%
11721 \setglossarystyle{tree}%
11722 \renewcommand*\glossaryheader{%
11723 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11724 \nobreak\@afterheading\indexspace}%
11725 \renewcommand*\glsgroupheading}[1]{%
11726 \par\noindent
11727 \glstreegroupheaderfmt
11728 {\glssnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
11729 \nopagebreak\indexspace\nobreak\@afterheading}%
11730 }
11731 }
11732 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11733 \ifdef{\@glsstyle@treenoname}
11734 {%
11735 \renewglossarystyle{treenoname}{%
11736 \renewenvironment{theglossary}%
11737 {\setlength{\parindent}{0pt}%
11738 \setlength{\parskip}{0pt plus 0.3pt}}%
11739 {}%
11740 \renewcommand*\glossaryheader{}%
11741 \renewcommand*\glsgroupheading}[1]{}%
11742 \renewcommand{\glossentry}[2]{%
11743 \hangindent0pt\relax
11744 \parindent0pt\relax
11745 \glssentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11746 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11747 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11748 \glstreeprelocation##2\par
11749 }%
11750 \renewcommand{\subglossentry}[3]{%
11751 \hangindent##1\glstreeindent\relax
11752 \parindent##1\glstreeindent\relax
11753 \ifnum##1=1\relax
11754 \glssubentryitem{##2}%
11755 \fi
11756 \glstarget{##2}{\strut}%

```

```

11757     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11758 }%
11759 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11760 }
11761 }
11762 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11763 \ifdef{\@glsstyle@treenonamegroup}
11764 {%
11765   \renewglossarystyle{treenonamegroup}{%
11766     \setglossarystyle{treenoname}%
11767     \renewcommand{\glsgroupheading}[1]{\par
11768       \noindent\glstreegroupheaderfmt
11769       {\glsgrouptitle{##1}}}%
11770     \nopagebreak\indexspace\nobreak\@afterheading
11771   }%
11772 }
11773 }
11774 {}

```

Similarly for `treenonamehypergroup`

```

11775 \ifdef{\@glsstyle@treenonamehypergroup}
11776 {%
11777   \renewglossarystyle{treenonamehypergroup}{%
11778     \setglossarystyle{treenoname}%
11779     \renewcommand*{\glossaryheader}{%
11780       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11781       \nobreak\@afterheading\indexspace}%
11782     \renewcommand*{\glsgroupheading}[1]{%
11783       \par\noindent
11784       \glstreegroupheaderfmt
11785       {\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}%
11786     \nopagebreak\indexspace\nobreak\@afterheading}%
11787   }
11788 }
11789 {}

```

The `almtree` style is redefined to make it easier to make minor adjustments.

```

11790 \ifdef{\@glsstyle@almtree}
11791 {%

```

Only redefine this style if it's already been defined.

symbolDescLocation

```
\glxtralttreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

11792 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```

```

11793   {%
11794       \let\par\glxstrAltTreePar
11795       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11796       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11797   }%
11798 }

```

trAltTreeIndent Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11799 \newlength\glxstrAltTreeIndent

```

lsxtrAltTreePar Multi-paragraph descriptions need to keep the hanging indent.

```

11800 \newcommand{\glxstrAltTreePar}{%
11801     @@par
11802     \glxstrAltTreeSetHangIndent
11803     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11804 }

```

symbolDescLocation `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11805 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11806     \glxstralttreeSymbolDescLocation{#2}{#3}%
11807 }

```

trtreetopindent The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

11808 \newlength\glxstrtreetopindent

```

sxtralttreeInit User-level initialisation for the alttree style.

```

11809 \newcommand*{\glxstralttreeInit}{%
11810     \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
11811     \glxstrAltTreeIndent=\parindent
11812 }

```

\glsglissetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\gdef`.

```

11813 \newcommand*{\glsglissetwidest}[2][0]{%
11814     \csgdef{@glswidestname\romannumeral#1}{#2}%
11815 }

```

\eglssetwidest The original `\glsglissetwidest` only uses `\def`. This uses `\protected@csedef`.

```

11816 \newcommand*{\eglssetwidest}[2][0]{%
11817     \protected@csedef{@glswidestname\romannumeral#1}{#2}%
11818 }

```

`\xglissetwidest` Like the above but uses `\protected@csxdef`.

```
11819 \newcommand*\xglissetwidest}[2][0]{%
11820   \protected@csxdef{#1}{#2}%
11821 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
11822 \newcommand*\glsupdatewidest}[2][0]{%
11823   \ifcsundef{#1}%
11824     {\csdef{#1}{#2}}%
11825     {%
11826       \settowidth{#1}{\csuse{#1}}%
11827       \settowidth{#2}%
11828       \ifdim#1>#2%
11829         \csdef{#1}{#2}%
11830       \fi
11831     }%
11832 }
```

`glsupdatewidest` As above but global definition.

```
11833 \newcommand*\glsupdatewidest}[2][0]{%
11834   \ifcsundef{#1}%
11835     {\csgdef{#1}{#2}}%
11836     {%
11837       \settowidth{#1}{\csuse{#1}}%
11838       \settowidth{#2}%
11839       \ifdim#1>#2%
11840         \csgdef{#1}{#2}%
11841       \fi
11842     }%
11843 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
11844 \newcommand*\eglsupdatewidest}[2][0]{%
11845   \ifcsundef{#1}%
11846     {\protected@csedef{#1}{#2}}%
11847     {%
11848       \settowidth{#1}{\csuse{#1}}%
11849       \settowidth{#2}%
11850       \ifdim#1>#2%
11851         \protected@csedef{#1}{#2}%
11852       \fi
11853     }%
11854 }
```

`glsupdatewidest` As above but global.

```
11855 \newcommand*\xglsupdatewidest}[2][0]{%
11856   \ifcsundef{#1}%
11857     {\protected@csxdef{#1}{#2}}%
11858     {%
```



```

11859     \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
11860     \settowidth{\dimen@ii}{#2}%
11861     \ifdim\dimen@ii>\dimen@
11862         \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
11863     \fi
11864 }%
11865 }

```

glswidestname Provide a user-level macro to obtain the widest top-level name.

```

11866 \newcommand*{\glswidestname}{\@glswidestname}

```

glswidestsubname Provide a user-level macro to obtain the widest sub-entry name.

```

11867 \newcommand*{\glswidestsubname}[1]{%
11868     \ifcsundef{@glswidestname\romannumeral#1}%
11869     {\@glswidestname}%
11870     {\csuse{@glswidestname\romannumeral#1}}%
11871 }

```

glswidestTopLevelName CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

11872 \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.

```

11873 \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
11874     \dimen@=0pt\relax
11875     \glstmpplen=0pt\relax
11876     \foralllglossaries[#1]{\@gls@type}%
11877     {%
11878         \forglsentries[\@gls@type]{\@glo@label}%
11879         {%
11880             \ifglswidest{\@glo@label}%
11881             {%
11882                 \ifglshasparent{\@glo@label}%
11883                 {}%
11884                 {%
11885                     \settowidth{\dimen@}{%
11886                         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
11887                     }%
11888                     \ifdim\dimen@>\glstmpplen
11889                         \glstmpplen=\dimen@
11890                     \eglssetwidest{\glstentryname{\@glo@label}}%
11891                 }%
11892             }%
11893         }%
11894     }%
11895 }%
11896 }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11897 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11898   \dimen@=0pt\relax
11899   \gls@tmplen=0pt\relax
11900   \forallglossaries[#1]{\@gls@type}%
11901   {%
11902     \forallgsentries[\@gls@type]{\@glo@label}%
11903     {%
11904       \ifglsused{\@glo@label}%
11905       {%
11906         \settowidth{\dimen@}%
11907           {\glstreenamefmt{\glstryname{\@glo@label}}}%
11908         \ifdim\dimen@>\gls@tmplen
11909           \gls@tmplen=\dimen@
11910           \eglssetwidest{\glstryname{\@glo@label}}%
11911         \fi
11912       }%
11913     }%
11914   }%
11915 }%
11916 }

```

`ndWidestAnyName` Like the above but doesn't check is the entry has been used.

```

11917 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
11918   \dimen@=0pt\relax
11919   \gls@tmplen=0pt\relax
11920   \forallglossaries[#1]{\@gls@type}%
11921   {%
11922     \forallgsentries[\@gls@type]{\@glo@label}%
11923     {%
11924       \settowidth{\dimen@}%
11925         {\glstreenamefmt{\glstryname{\@glo@label}}}%
11926       \ifdim\dimen@>\gls@tmplen
11927         \gls@tmplen=\dimen@
11928         \eglssetwidest{\glstryname{\@glo@label}}%
11929       \fi
11930     }%
11931   }%
11932 }

```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

11933 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
11934   \dimen@=0pt\relax
11935   \dimen@i=0pt\relax
11936   \dimen@ii=0pt\relax
11937   \forallglossaries[#1]{\@gls@type}%
11938   {%

```

```

11939 \forglsentries[\@gls@type]{\@glo@label}%
11940 {%
11941 \ifglsused{\@glo@label}%
11942 {%
11943 \ifglsashasparent{\@glo@label}%
11944 {%
11945 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
11946 \ifglsashasparent{\@glo@parent}%
11947 {%
11948 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
11949 \ifglsashasparent{\@glo@parent}%
11950 {}%
11951 {%
11952 \settowidth{\gls@tmplen}%
11953 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11954 \ifdim\gls@tmplen>\dimen@ii
11955 \dimen@ii=\gls@tmplen
11956 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
11957 \fi
11958 }%
11959 }%
11960 {%
11961 \settowidth{\gls@tmplen}%
11962 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11963 \ifdim\gls@tmplen>\dimen@i
11964 \dimen@i=\gls@tmplen
11965 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
11966 \fi
11967 }%
11968 }%
11969 {%
11970 \settowidth{\gls@tmplen}%
11971 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11972 \ifdim\gls@tmplen>\dimen@
11973 \dimen@=\gls@tmplen
11974 \eglssetwidest{\glsentryname{\@glo@label}}%
11975 \fi
11976 }%
11977 }%
11978 {}%
11979 }%
11980 }%
11981 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

11982 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
11983 \dimen@=0pt\relax
11984 \dimen@i=0pt\relax
11985 \dimen@ii=0pt\relax

```

```

11986 \forall glossaries[#1]{\@gls@type}%
11987 {%
11988 \forall glsentries[\@gls@type]{\@glo@label}%
11989 {%
11990 \ifglshasparent{\@glo@label}%
11991 {%
11992 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}%
11993 \ifglshasparent{\@glo@parent}%
11994 {%
11995 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}%
11996 \ifglshasparent{\@glo@parent}%
11997 {}%
11998 {%
11999 \settowidth{\gls@tmplen}%
12000 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12001 \ifdim\gls@tmplen>\dimen@ii
12002 \dimen@ii=\gls@tmplen
12003 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
12004 \fi
12005 }%
12006 }%
12007 {%
12008 \settowidth{\gls@tmplen}%
12009 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12010 \ifdim\gls@tmplen>\dimen@i
12011 \dimen@i=\gls@tmplen
12012 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
12013 \fi
12014 }%
12015 }%
12016 {%
12017 \settowidth{\gls@tmplen}%
12018 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12019 \ifdim\gls@tmplen>\dimen@
12020 \dimen@=\gls@tmplen
12021 \eglssetwidest{\glsentryname{\@glo@label}}%
12022 \fi
12023 }%
12024 }%
12025 }%
12026 }

```

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.

```

12027 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
12028 \dimen@=0pt\relax
12029 \gls@tmplen=0pt\relax
12030 #2=0pt\relax
12031 \forall glossaries[#1]{\@gls@type}%

```

```

12032  {%
12033    \forglsentries[\@gls@type]{\@glo@label}%
12034  {%
12035    \ifglsused{\@glo@label}%
12036    {%
12037      \settowidth{\dimen@}%
12038        {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12039      \ifdim\dimen@>\gls@tmplen
12040        \gls@tmplen=\dimen@
12041        \eglssetwidest{\glsentryname{\@glo@label}}%
12042      \fi
12043      \settowidth{\dimen@}%
12044        {\glsentrysymbol{\@glo@label}}%
12045      \ifdim\dimen@>#2\relax
12046        #2=\dimen@
12047      \fi
12048    }%
12049  }%
12050 }%
12051 }%
12052 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

12053  \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
12054    \dimen@=0pt\relax
12055    \gls@tmplen=0pt\relax
12056    #2=0pt\relax
12057    \forallglossaries[#1]{\@gls@type}%
12058  {%
12059    \forglsentries[\@gls@type]{\@glo@label}%
12060  {%
12061    \settowidth{\dimen@}%
12062      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
12063    \ifdim\dimen@>\gls@tmplen
12064      \gls@tmplen=\dimen@
12065      \eglssetwidest{\glsentryname{\@glo@label}}%
12066    \fi
12067    \settowidth{\dimen@}%
12068      {\glsentrysymbol{\@glo@label}}%
12069    \ifdim\dimen@>#2\relax
12070      #2=\dimen@
12071    \fi
12072  }%
12073 }%
12074 }

```

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.

```

12075 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
12076   \dimen@=0pt\relax
12077   \gls@tmplen=0pt\relax
12078   #2=0pt\relax
12079   #3=0pt\relax
12080   \foralllglossaries[#1]{\@gls@type}%
12081   {%
12082     \forglsentries[\@gls@type]{\@glo@label}%
12083     {%
12084       \ifglsused{\@glo@label}%
12085       {%
12086         \settowidth{\dimen@}%
12087           {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12088         \ifdim\dimen@>\gls@tmplen
12089           \gls@tmplen=\dimen@
12090           \eglssetwidest{\glsentryname{\@glo@label}}%
12091         \fi
12092         \settowidth{\dimen@}%
12093           {\glsentrysymbol{\@glo@label}}%
12094         \ifdim\dimen@>#2\relax
12095           #2=\dimen@
12096         \fi
12097         \settowidth{\dimen@}%
12098           {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12099         \ifdim\dimen@>#3\relax
12100           #3=\dimen@
12101         \fi
12102       }%
12103     }%
12104   }%
12105 }%
12106 }
```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

12107 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
12108   \dimen@=0pt\relax
12109   \gls@tmplen=0pt\relax
12110   #2=0pt\relax
12111   #3=0pt\relax
12112   \foralllglossaries[#1]{\@gls@type}%
12113   {%
12114     \forglsentries[\@gls@type]{\@glo@label}%
12115     {%
12116       \settowidth{\dimen@}%
12117         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12118       \ifdim\dimen@>\gls@tmplen
12119         \gls@tmplen=\dimen@
12120         \eglssetwidest{\glsentryname{\@glo@label}}%
```

```

12121      \fi
12122      \settowidth{\dimen@}%
12123      {\glstrysymbol{\@glo@label}}}%
12124      \ifdim\dimen@>#2\relax
12125      #2=\dimen@
12126      \fi
12127      \settowidth{\dimen@}%
12128      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12129      \ifdim\dimen@>#3\relax
12130      #3=\dimen@
12131      \fi
12132    }%
12133  }%
12134 }

```

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.

```

12135 \newrobustcmd*{\glstryFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
12136   \dimen@=0pt\relax
12137   \glstry@tmplen=0pt\relax
12138   #2=0pt\relax
12139   \forallglossaries[#1]{\@gls@type}%
12140   {%
12141     \forallglsentries[\@gls@type]{\@glo@label}%
12142     {%
12143       \ifglstryused{\@glo@label}%
12144       {%
12145         \settowidth{\dimen@}%
12146         {\glstrynamefmt{\glstryname{\@glo@label}}}%
12147         \ifdim\dimen@>\glstry@tmplen
12148         \glstry@tmplen=\dimen@
12149         \glstrysetwidest{\glstryname{\@glo@label}}%
12150       \fi
12151       \settowidth{\dimen@}%
12152       {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12153       \ifdim\dimen@>#2\relax
12154       #2=\dimen@
12155     \fi
12156   }%
12157   }%
12158 }%
12159 }%
12160 }

```

AnyNameLocation Like the `\glstryFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

12161 \newrobustcmd*{\glstryFindWidestAnyNameLocation}[2][\@glo@types]{%
12162   \dimen@=0pt\relax
12163   \glstry@tmplen=0pt\relax

```

```

12164      #2=Opt\relax
12165      \forallglossaries[#1]{\@gls@type}%
12166      {%
12167        \forglsentries[\@gls@type]{\@glo@label}%
12168        {%
12169          \settowidth{\dimen@}%
12170            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12171          \ifdim\dimen@>\gls@tmplen
12172            \gls@tmplen=\dimen@
12173            \eglssetwidest{\glsentryname{\@glo@label}}%
12174          \fi
12175          \settowidth{\dimen@}%
12176            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12177          \ifdim\dimen@>#2\relax
12178            #2=\dimen@
12179          \fi
12180        }%
12181      }%
12182    }

```

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`.

```

12183    \newcommand*{\glsxtrComputeTreeIndent}[1]{%
12184      \glstreeindent=\glsxtrtreetopindent\relax
12185    }

```

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.

```

12186    \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
12187      \ifcsundef{@glswidestname\romannumeral#1}%
12188      {%
12189        \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
12190      }%
12191      {%
12192        \settowidth{#3}{\glstreenamefmt{%
12193          \csname @glswidestname\romannumeral#1\endcsname\space}}%
12194      }%
12195    }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

12196    \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```


etSubHangIndent Set \hangindent for sub-entries:

```
12197 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
12198 \renewglossarystyle{alttree}{%
12199   \renewenvironment{theglossary}%
12200     {%
12201       \glxtralttreeInit
12202       \def\@gls@prevlevel{-1}%
12203       \mbox{}\par}%
12204     {\par}%
12205   \renewcommand*{\glossaryheader}{}%
12206   \renewcommand*{\glsgroupheading}[1]{}%
12207   \renewcommand{\glossentry}[2]{%
12208     \ifnum\@gls@prevlevel=0\relax
12209     \else
12210       \glxtrComputeTreeIndent{##1}%
12211     \fi
12212     \parindent\glstreeindent
12213     \glxtrAltTreeSetHangIndent
12214     \makebox[0pt][r]{%
12215       {%
12216         \glstreenamebox{\glstreeindent}%
12217         {%
12218           \glssentryitem{##1}%
12219           \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
12220         }%
12221       }%
12222       \glxtralttreeSymbolDescLocation{##1}{##2}%
12223       \def\@gls@prevlevel{0}%
12224     }
12225   \renewcommand{\subglossentry}[3]{%
12226     \ifnum##1=1\relax
12227       \glssubentryitem{##2}%
12228     \fi
12229     \ifnum\@gls@prevlevel=##1\relax
12230     \else
12231       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12232       \ifnum\@gls@prevlevel<##1\relax
12233         \setlength\glstreeindent\gls@tmplen
12234         \addtolength\glstreeindent\parindent
12235         \parindent\glstreeindent
12236       \else
12237         \ifnum\@gls@prevlevel=0\relax
12238           \glxtrComputeTreeIndent{##2}%
12239         \else
12240           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12241         \fi
12242         \addtolength\parindent{-\glstreeindent}%

```

```

12243         \setlength\glstreeindent\parindent
12244     \fi
12245 \fi
12246 \glstrAltTreeSetSubHangIndent{##1}%
12247 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
12248     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12249 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12250 \def\@gls@prevlevel{##1}%
12251 }%
12252 \renewcommand*\@gls@groupskip{\ifglsnogroupskip\else\indexspace\fi}%
12253 }
12254 }%
12255 {%
12256 }

```

Redefine `alttreegroup` so that it discourages a break after group headings. Can't use `\@afterheading` here as it messes with the first item of the group.

```

12257 \ifdef{\@glsstyle@alttreegroup}
12258 {%
12259     \renewglossarystyle{alttreegroup}{%
12260         \setglossarystyle{alttree}%
12261         \renewcommand{\gls@groupheading}[1]{\par
12262             \def\@gls@prevlevel{-1}%
12263             \hangindentOpt\relax
12264             \parindentOpt\relax
12265             \glstreegroupheaderfmt{\gls@getgrouptitle{##1}}%
12266             \nopagebreak\indexspace\nopagebreak
12267         }%
12268     }%
12269 }%
12270 {%
12271 }

```

Similarly for `alttreehypergroup`.

```

12272 \ifdef{\@glsstyle@alttreehypergroup}
12273 {%
12274     \renewglossarystyle{alttreehypergroup}{%
12275         \setglossarystyle{alttree}%
12276         \renewcommand*\@gls@groupheader{%
12277             \par
12278             \def\@gls@prevlevel{-1}%
12279             \hangindentOpt\relax
12280             \parindentOpt\relax
12281             \glstreenavigationfmt{\gls@navigation}\par\indexspace
12282         }%
12283         \renewcommand*\@gls@groupheading}[1]{%
12284             \par
12285             \def\@gls@prevlevel{-1}%
12286             \hangindentOpt\relax
12287             \parindentOpt\relax

```

```

12288      \glstreegroupheaderfmt
12289      {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12290      \nopagebreak\indexspace\nopagebreak
12291    }%
12292  }
12293 }%
12294 {%
12295 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12296 \ifdef{\@glsstyle@mcolindexgroup}
12297 {%
12298   \renewglossarystyle{mcolindexgroup}{%
12299     \setglossarystyle{mcolindex}%
12300     \renewcommand*{\glsgroupheading}[1]{%
12301       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12302       \nopagebreak\indexspace\nobreak\@afterheading
12303     }%
12304   }
12305 }%
12306 {%
12307 }

```

Similarly for mcolindexhypergroup.

```

12308 \ifdef{\@glsstyle@mcolindexhypergroup}
12309 {%
12310   \renewglossarystyle{mcolindexhypergroup}{%
12311     \setglossarystyle{mcolindex}%
12312     \renewcommand*{\glossaryheader}{%
12313       \item\glstreenavigationfmt{\glsnavigation}%
12314       \indexspace
12315     }%
12316     \renewcommand*{\glsgroupheading}[1]{%
12317       \item\glstreegroupheaderfmt
12318         {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12319       \nopagebreak\indexspace\nobreak\@afterheading
12320     }%
12321   }
12322 }%
12323 {%
12324 }

```

Similarly for mcolindexspannav.

```

12325 \ifdef{\@glsstyle@mcolindexspannav}
12326 {%
12327   \renewglossarystyle{mcolindexspannav}{%
12328     \setglossarystyle{index}%

```

```

12329 \renewenvironment{theglossary}%
12330 {%
12331 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12332 \setlength{\parindent}{0pt}%
12333 \setlength{\parskip}{0pt plus 0.3pt}%
12334 \let\item\glstreeitem}%
12335 {\end{multicols}}}%
12336 \renewcommand*{\glsgroupheading}[1]{%
12337 \item\glstreegroupheaderfmt
12338 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12339 \nopagebreak\indexspace\nobreak\@afterheading
12340 }%
12341 }
12342 }%
12343 {%
12344 }

```

Similarly for mcoltreegroup.

```

12345 \ifdef{\@glstyle@mcoltreegroup}
12346 {%
12347 \renewglossarystyle{mcoltreegroup}{%
12348 \setglossarystyle{mcoltree}%
12349 \renewcommand{\glsgroupheading}[1]{\par
12350 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12351 \nopagebreak\indexspace\nobreak\@afterheading
12352 }%
12353 }
12354 }%
12355 {%
12356 }

```

Similarly for mcoltreehypergroup.

```

12357 \ifdef{\@glstyle@mcoltreehypergroup}
12358 {%
12359 \renewglossarystyle{mcoltreehypergroup}{%
12360 \setglossarystyle{mcoltree}%
12361 \renewcommand*{\glossaryheader}{%
12362 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12363 }%
12364 \renewcommand*{\glsgroupheading}[1]{%
12365 \par\noindent
12366 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12367 \nopagebreak\indexspace\nobreak\@afterheading
12368 }%
12369 }
12370 }%
12371 {%
12372 }

```

Similarly for mcoltreespannav.

```

12373 \ifdef{\@glstyle@mcoltreespannav}

```

```

12374 {%
12375   \renewglossarystyle{mcoltreesspannav}{%
12376     \setglossarystyle{tree}%
12377     \renewenvironment{theglossary}%
12378     {%
12379       \begin{multicols}{\glsmcols}%
12380       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12381       \setlength{\parindent}{0pt}%
12382       \setlength{\parskip}{0pt plus 0.3pt}%
12383     }%
12384     {\end{multicols}}}%
12385   \renewcommand*{\glsgroupheading}[1]{%
12386     \par\noindent
12387     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12388     \nopagebreak\indexspace\nobreak\@afterheading
12389   }%
12390 }
12391 }%
12392 {%
12393 }

```

Similarly for mcoltreenonamegroup.

```

12394 \ifdef{\@glsstyle@mcoltreenonamegroup}
12395 {%
12396   \renewglossarystyle{mcoltreenonamegroup}{%
12397     \setglossarystyle{mcoltreenoname}%
12398     \renewcommand*{\glsgroupheading}[1]{\par
12399       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12400     \nopagebreak\indexspace\nobreak\@afterheading
12401   }%
12402 }
12403 }%
12404 {%
12405 }

```

Similarly for mcoltreenonamehypergroup.

```

12406 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12407 {%
12408   \renewglossarystyle{mcoltreenonamehypergroup}{%
12409     \setglossarystyle{mcoltreenoname}%
12410     \renewcommand*{\glossaryheader}{%
12411       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12412     \renewcommand*{\glsgroupheading}[1]{%
12413       \par\noindent
12414       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12415       \nopagebreak\indexspace\nobreak\@afterheading}%
12416   }
12417 }%
12418 {%
12419 }

```

Similarly for mcoltreenonamespannav.

```

12420 \ifdef{\@glsstyle@mcoltreenonamespannav}
12421 {%
12422   \renewglossarystyle{mcoltreenonamespannav}{%
12423     \setglossarystyle{treenoname}%
12424     \renewenvironment{theglossary}%
12425     {%
12426       \begin{multicols}{\glsmcols}%
12427       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12428       \setlength{\parindent}{0pt}%
12429       \setlength{\parskip}{0pt plus 0.3pt}%
12430     }%
12431     {\end{multicols}}}%
12432   \renewcommand*{\glsgroupheading}[1]{%
12433     \par\noindent
12434     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12435     \nopagebreak\indexspace\nobreak\@afterheading}%
12436   }
12437 }%
12438 {%
12439 }

```

mcolalmtree needs adjusting so that it uses \glxtralttreeInit This doesn't use \mbox{} \par which would unbalance the top of the columns.

```

12440 \ifdef{\@glsstyle@mcolalmtree}
12441 {%
12442   \renewglossarystyle{mcolalmtree}{%
12443     \setglossarystyle{almtree}%
12444     \renewenvironment{theglossary}%
12445     {%
12446       \glxtralttreeInit
12447       \def\@gls@prevlevel{-1}%
12448       \begin{multicols}{\glsmcols}%
12449     }%
12450     {\par\end{multicols}}%
12451   }
12452 }%
12453 {%
12454 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12455 \ifdef{\@glsstyle@mcolalmtreegroup}
12456 {%
12457   \renewglossarystyle{mcolalmtreegroup}{%
12458     \setglossarystyle{mcolalmtree}%
12459     \renewcommand{\glsgroupheading}[1]{\par
12460       \def\@gls@prevlevel{-1}%
12461       \hangindent0pt\relax
12462       \parindent0pt\relax
12463       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

12464     \nopagebreak\indexspace\nopagebreak
12465 }%
12466 }
12467 }%
12468 {%
12469 }

```

Similarly for mcolalmtreehypergroup.

```

12470 \ifdef{\@glsstyle@mcolalmtreehypergroup}
12471 {%
12472   \renewglossarystyle{mcolalmtreehypergroup}{%
12473     \setglossarystyle{mcolalmtree}%
12474     \renewcommand*{\glossaryheader}{%
12475       \par
12476       \def\@gls@prevlevel{-1}%
12477       \hangindent0pt\relax
12478       \parindent0pt\relax
12479       \glstreenavigationfmt{\glsnavigation}%
12480       \par\indexspace
12481     }%
12482     \renewcommand*{\glsgroupheading}[1]{%
12483       \par
12484       \def\@gls@prevlevel{-1}%
12485       \hangindent0pt\relax
12486       \parindent0pt\relax
12487       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12488       \nopagebreak\indexspace\nopagebreak
12489     }%
12490   }
12491 }%
12492 {%
12493 }

```

Similarly for mcolalttreespannav.

```

12494 \ifdef{\@glsstyle@mcolalttreespannav}
12495 {%
12496   \renewglossarystyle{mcolalttreespannav}{%
12497     \setglossarystyle{alttree}%
12498     \renewenvironment{theglossary}%
12499     {%
12500       \glsxtralmtreeInit
12501       \def\@gls@prevlevel{-1}%
12502       \begin{multicols}{\glsncols}%
12503         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12504     }%
12505     {\par\end{multicols}}%
12506     \renewcommand*{\glsgroupheading}[1]{%
12507       \par
12508       \def\@gls@prevlevel{-1}%
12509       \hangindent0pt\relax

```

```

12510      \parindent0pt\relax
12511      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12512      \nopagebreak\indexspace\nopagebreak
12513  }%
12514  }
12515 }%
12516 {%
12517 }

      Reset the default style
12518 \ifx\@glossary@default@style\relax
12519 \else
12520   \setglossarystyle{\@glxtr@current@style}
12521 \fi

```


3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
12522 \NeedsTeXFormat{LaTeX2e}
12523 \ProvidesPackage{glossary-bookindex}[2017/11/12 v1.23 (NLCT)]
```

Load required packages.

```
12524 \RequirePackage{multicol}
12525 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12526 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12527 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`bookindexsubname` Format used for sub entries.

```
12528 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`glstrprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12529 \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.

```
12530 \newcommand*{\glstrbookindexprelocation}[1]{%
12531   \glstrifhasfield{location}{#1}%
12532   {,\glstrprelocation}%
12533   {\glstrprelocation}%
12534 }
```

`glstrsubprelocation` Separator used before location list for sub-entries.

```
12535 \newcommand*{\glstrbookindexsubprelocation}[1]{%
12536   \glstrbookindexprelocation{#1}%
12537 }
```

`glstrparentchildsep` Separator used between top-level parent and child entry.

```
12538 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`glstrparentsubchildsep` Separator used between sub-level parent and child entry.

```
12539 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
12540 \newcommand{\glxstrbookindexbetween}[2] {}
```

`indexsubbetween` Between two level 1 entries identified by the labels in the arguments.

```
12541 \newcommand{\glxstrbookindexsubbetween}[2] {}
```

`exsubsubbetween` Between two level 2 entries identified by the labels in the arguments.

```
12542 \newcommand{\glxstrbookindexsubsubbetween}[2] {}
```

`indexatendgroup` At the end of a letter group. The argument is the index of the last top-level entry.

```
12543 \newcommand{\glxstrbookindexatendgroup}[1] {}
```

`exsubatendgroup` At the end of a letter group. The argument is the index of the last level 1 entry.

```
12544 \newcommand{\glxstrbookindexsubatendgroup}[1] {}
```

`subsubatendgroup` At the end of a letter group. The argument is the index of the last level 2 entry.

```
12545 \newcommand{\glxstrbookindexsubsubatendgroup}[1] {}
```

`bookindexgroupskip` Group separator.

```
12546 \newcommand{\glxstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

`indexformatheader` Group separator.

```
12547 \newcommand*{\glxstrbookindexformatheader}[1] {%
12548   \par{\centering\glstreegroupheaderfmt{#1}\par}%
12549 }
```

`bookindexbookmark` Book mark group heading if supported.

```
12550 \ifdef\pdfbookmark
12551 {%
12552   \newcommand*{\glxstrbookindexbookmark}[2] {%
12553     \ifdefstring{\@glossarysec}{chapter}%
12554     {\pdfbookmark[1]{#1}{#2}}%
12555     {\pdfbookmark[2]{#1}{#2}}%
12556   }
12557 }
12558 {%
12559   \newcommand*{\glxstrbookindexbookmark}[2] {}
12560 }
```

`bookindexcolspread`

```
12561 \newcommand*{\glxstrbookindexcolspread} {}
```

Define the style.

```
12562 \newglossarystyle{bookindex}{%
12563   \setglossarystyle{index}%
12564   \renewenvironment{theglossary}%
```

```

12565 {%
12566   \ifdefempty\glxstrbookindexcolspread
12567   {\begin{multicols}{\glxstrbookindexcols}}%
12568   {\begin{multicols}{\glxstrbookindexcols}[\glxstrbookindexcolspread]}%
12569   \setlength{\parindent}{0pt}%
12570   \setlength{\parskip}{0pt plus 0.3pt}%
12571   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12572   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12573   \let\@glxstr@bookindex@between\@gobble
12574   \let\@glxstr@bookindex@subbetween\@gobble
12575   \let\@glxstr@bookindex@subsubbetween\@gobble
12576   \let\@glxstr@bookindex@atendgroup\relax
12577   \let\@glxstr@bookindex@subatendgroup\relax
12578   \let\@glxstr@bookindex@subsubatendgroup\relax
12579   \let\@glxstr@bookindex@groupskip\relax
12580 }%
12581 {%

```

Do end group hooks.

```

12582   \@glxstr@bookindex@subsubatendgroup
12583   \@glxstr@bookindex@subatendgroup
12584   \@glxstr@bookindex@atendgroup

```

End multicols environment.

```

12585   \end{multicols}%
12586 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12587 \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12588 \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12589   \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12590   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12591   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12592   \let\@glxstr@bookindex@subbetween\@gobble
12593   \let\@glxstr@bookindex@subsubbetween\@gobble
12594   \edef\@glxstr@bookindex@between{%
12595     \noexpand\glxstrbookindexbetween{##1}%
12596   }%
12597   \edef\@glxstr@bookindex@atendgroup{%
12598     \noexpand\glxstrbookindexatendgroup{##1}%
12599   }%
12600   \let\@glxstr@bookindex@subatendgroup\relax
12601   \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12602   \glstreeitem

```

```

12603     \glstryitem{##1}%
12604     \glstarget{##1}{\glxtrbookindexname{##1}}%
12605     \glxtrbookindexprelocation{##1}##2%
12606 }%
12607 \renewcommand{\subglossentry}[3]{%
12608     \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12609     \glstreeitem
12610     \or

```

Level 1.

```

12611     \@glxtr@bookindex@sep
12612     \@glxtr@bookindex@subbetween{##2}%
12613     \let\@glxtr@bookindex@sep\relax

```

Update separators.

```

12614     \let\@glxtr@bookindex@subsubbetween\@gobble
12615     \let\@glxtr@bookindex@subsep\glxtrbookindexparentschildsep
12616     \edef\@glxtr@bookindex@subbetween{%
12617         \noexpand\glxtrbookindexsubbetween{##2}%
12618     }%
12619     \edef\@glxtr@bookindex@atsubendgroup{%
12620         \noexpand\glxtrbookindexatsubendgroup{##1}%
12621     }%

```

Start sub-item.

```

12622     \glstreesubitem
12623     \glssubentryitem{##2}%
12624     \else

```

All other levels.

```

12625     \@glxtr@bookindex@subsep
12626     \@glxtr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12627     \let\@glxtr@bookindex@subsep\relax
12628     \edef\@glxtr@bookindex@subsubbetween{%
12629         \noexpand\glxtrbookindexsubsubbetween{##2}%
12630     }%
12631     \edef\@glxtr@bookindex@atsubsubendgroup{%
12632         \noexpand\glxtrbookindexatsubsubendgroup{##1}%
12633     }%

```

Start sub-sub-item.

```

12634     \glstreesubsubitem
12635     \fi

```

Format entry.

```

12636     \glstarget{##2}{\glxtrbookindexsubname{##2}}%
12637     \glxtrbookindexsubprelocation{##2}##3%
12638 }%

```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12639 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12640 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12641 \@glstr@bookindex@subsubatendgroup
```

```
12642 \@glstr@bookindex@subatendgroup
```

```
12643 \@glstr@bookindex@atendgroup
```

```
12644 \@glstr@bookindexgroupskip
```

Update separators.

```
12645 \let\@glstr@bookindexgroupskip\glstrbookindexgroupskip
```

```
12646 \let\@glstr@bookindex@between\@gobble
```

```
12647 \let\@glstr@bookindex@atendgroup\relax
```

```
12648 \let\@glstr@bookindex@subatendgroup\relax
```

```
12649 \let\@glstr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12650 \glstrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12651 \glstrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12652 \glstrbookindexformatheader{\thisgrptitle}%
```

```
12653 \nopagebreak\indexspace\nopagebreak\@afterheading
```

```
12654 }%
```

```
12655 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glstrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`\glstrbookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
12656 \newcommand{\glstrbookindexthepage}{%
```

```
12657 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
```

```
12658 }
```

`\glstrbookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
12659 \newcommand*{\glstrbookindexmarkentry}[1]{%
```

```
12660 \protected@write\@auxout
```

```
12661 {\let\glstrbookindexthepage\relax}%
```

```
12662 {\string\glstr@setbookindexmark{\glstrbookindexthepage}{#1}}%
```

```
12663 }
```

etbookindexmark

```
12664 \newcommand*{\glxtr@setbookindexmark}[2]{%
12665   \ifcsundef{glxtr@idxfirstmark@#1}%
12666   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
12667   {}%
12668   \csgdef{glxtr@idxlastmark@#1}{#2}%
12669 }
```

dexfirstmarkfmt

```
12670 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
12671   \glentryname{#1}%
12672 }
```

kindexfirstmark

```
12673 \newcommand*{\glxtrbookindexfirstmark}{%
12674   \letcs{glxtr@label}{glxtr@idxfirstmark@glxtrbookindexthepage}%
12675   \ifdefglxtr@label
12676   {\glxtrbookindexfirstmarkfmt{glxtr@label}}%
12677   {}%
12678 }
```

ndexlastmarkfmt

```
12679 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
12680   \glentryname{#1}%
12681 }
```

okindexlastmark

```
12682 \newcommand*{\glxtrbookindexlastmark}{%
12683   \letcs{glxtr@label}{glxtr@idxlastmark@glxtrbookindexthepage}%
12684   \ifdefglxtr@label
12685   {\glxtrbookindexlastmarkfmt{glxtr@label}}%
12686   {}%
12687 }
```

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	303
\glsfmtshort: new	302
\Glsfmtshortpl: new	303
\glsfmtshortpl: new	302
short: switched inline full form to short (long)	206
0.3 (2015-12-02)	
\@ACRlong: added redefinition	70
\@ACRlongpl: added redefinition	71
\@ACRshort: added redefinition	68
\@ACRshortpl: added redefinition	69
\@Acrlong: added redefinition	70
\@Acrlongpl: added redefinition	71
\@Acrshort: added redefinition	68
\@Acrshortpl: added redefinition	69
\@GLSdesc: added redefinition	64
\@GLSdescplural@: added redefinition	64
\@GLSfirst@: added redefinition	61
\@GLSfirstplural@: added redefinition	63
\@GLSname@: added redefinition	63
\@GLSplural@: added redefinition	62
\@GLSsymbol@: added redefinition	65
\@GLSsymbolplural@: added redefinition	65
\@GLStext@: added redefinition	60
\@GLSuseri@: added redefinition	66
\@GLSuserii@: added redefinition	66
\@GLSuseriii@: added redefinition	66
\@GLSuseriv@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@GLSdesc@: added redefinition	64
\@GLSdescplural@: added redefinition	64
\@GLSfirst@: added redefinition	61
\@GLSfirstplural@: added redefinition	62
\@GLSname@: added redefinition	63
\@GLSplural@: added redefinition	62
\@GLsymbol@: added redefinition	65
\@GLsymbolplural@: added redefinition	65
\@GLstext@: added redefinition	60
\@GLSuseri@: added redefinition	65
\@GLSuserii@: added redefinition	66
\@GLSuseriii@: added redefinition	66
\@GLSuseriv@: added redefinition	66
\@GLSuseriv@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@GLSuseriv@: added redefinition	67
\@acrlong: added redefinition	69
\@acrlongpl: added redefinition	70
\@acrshort: added redefinition	67
\@acrshortpl: added redefinition	68
\@gls@field@link: added optional argument	54
\@glsdescplural@: added redefinition	64
\@glsfirst@: added redefinition	61
\@glsfirstplural@: added redefinition	62
\@glsplural@: added redefinition	62
\@glssymbolplural@: added redefinition	65
\@glsxtr@defaultnoglossarywarning: new	121
\@glsxtr@field@linkdefs: new	60
\@glsxtr@insertdots: new	176
\@print@glossary: added redefinition	117
\glsabbrvdefaultfont: renamed from \abbrvdefaultfont	182
\glsaccessdesc: new	144
\glsaccessdescplural: new	144
\glsaccessfirst: new	141
\glsaccessfirstplural: new	142
\glsaccesslong: new	146
\glsaccesslong: new	146
\glsaccessname: new	140
\glsaccessplural: new	141
\glsaccessshort: new	145
\glsaccessshort: new	145
\glsaccessshortpl: new	146

\glsaccessshortpl: new	146	\cGLSpl: new	95
\glsaccesssymbol: new	143	\cGLSpl@: new	95
\glsaccesssymbolplural: new	143	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	140	new	90
\glstentryfmt: added check for short ..	54	\cGLS: new	94
\glslongpltok: new	176	\cGLSformat: new	95
\glsshortpltok: new	176	\cGLSpl: new	95
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	95
name in \setkeys	178	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	15
for plural	173	\glsenableentrycount: new	90
\GLSxtrlongpl: new	191	\glsfirstabbrvdefaultfont: new ..	181
\Glsxtrlongpl: new	190	\glsfirstlongdefaultfont: new ...	182
\glsxtrlongpl: new	190	\Glsfmtfirst: new	305
\glsxtrNoGlossaryWarning: new	20	\glsfmtfirst: new	305
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	306
new	173	\glsfmtfirstpl: new	306
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	305
new	173	\glsfmtplural: new	304
\glsxtrpostlinkendsentence: new ..	172	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	189	\Glsxtrtitleshort	303
\Glsxtrshortpl: new	189	renamed from \glstentryfmtshort ..	303
\glsxtrshortpl: new	188	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	302
\glslabeltok	201	renamed from \glstentryfmtshort ..	302
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	199	\Glsxtrtitleshortpl	303
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl	303
redefinition of \acronymtype	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	302
\Glsxtrshort	303	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl	302
\glsxtrshort	302	\Glsfmttext: new	304
\Glsfmtshortpl: changed to use		\glsfmttext: new	304
\glsxtrshortpl	303	\glshasattribute: new	152
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	151
\glsxtrshortpl	302	\glsxtremsuffix: new	242
\glsxtrifemptyglossary: new	25	\GlsXtrEnableEntryCounting: new ..	89
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	92
argument	155	\glsxtrscfont: new	214
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	214
argument	155	\glsxtrsmfont: new	228
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	228
default type to \acronymtype	103	short-em: new	249
\newterm: fixed name argument	154	short-em-desc: new	251
0.5 (2015-12-07)		short-em-footnote: new	260
\@cGLS: new	94	short-em-long: new	246
\@cGLS@: new	95	short-em-long-desc: new	247

short-em-postfootnote: new	262	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	224	attribute	293
short-sc-postfootnote: new	226	\Glsxtrheadtext: now uses headuc	
short-sm: new	232	attribute	296
short-sm-desc: new	233	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	239	attribute	295
short-sm-long: new	230	short-em-footnote: switch off regular	
short-sm-long-desc: new	231	attribute if set	260
short-sm-postfootnote: new	240	short-long: switch off regular attribute	
long-noshort-em: new	253	if set	200
long-noshort-em-desc: new	257	short-long-desc: switch off regular	
long-noshort-sm: new	235	attribute if set	202
long-noshort-sm-desc: new	237	short-sc-footnote: switch off regular	
long-short-em: new	243	attribute if set	225
long-short-em-desc: new	244	short-sm-footnote: switch off regular	
long-short-sm: new	228	attribute if set	239
long-short-sm-desc: new	230	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	198
\Glsaccessstext: new	141	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	200
\@glxtr@doaccsupp: new	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	293	attribute if set	216
\Glsaccessdesc: new	144	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	145	set	203
\Glsaccessfirst: new	142	postfootnote: switch off regular	
\Glsaccessfirstplural: new	142	attribute if set	204
\Glsaccessname: new	140	0.5.2 (2015-12-08)	
\Glsaccessplural: new	141	\@GLSdesc@: added accessibility support	64
\Glsaccesssymbol: new	143	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	143	support	64
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	297	support	61
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	297	support	63
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	63
headuc attribute	298	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	62
headuc attribute	298	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	65
attribute	297	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	65
attribute	296	\@GLStext@: added accessibility support	60
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	64
attribute	294	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	64
attribute	293	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	61
attribute	294	\@GLSfirstplural@: added accessibility	
		support	62

\@Glsname@: add accessibility support ..	63	\GLSaccesssymbolplural: new ..	144, 149
\@Glsplural@: added accessibility support	62	\GLSaccessstext: new	141, 148
\@Glsymbol@: added accessibility support	65	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	65	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	60	\glsxtrabbrvfmt to here	54
\@glsdesc@: added accessibility support	63	\GlsXtrEnableInitialTagging: new	169
\@glsdescplural@: added accessibility support	64	\glsxtrfieldtitlecase: new	156
\@glsfirst@: added accessibility support	61	\GlsXtrFormatLocationList: new ...	51
\@glsfirstplural@: added accessibility support	62	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	63	new	180
\@Glsplural@: added accessibility support	62	\glsxtrtagfont: new	171
\@Glsymbol@: added accessibility support	64	\KV@printgloss@nonumberlist: added	53
\@Glsymbolplural@: added accessibility support	65	\mfu@checkword@do: added	170
\@Glstext@: added accessibility support	60	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	195
new	171	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	170	\@glsxtr@autoindex@at: new	166
\@glsxtr@tag: new	171	\@glsxtr@autoindex@encap: new ...	167
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	167
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	167
\@ifpackageloaded	140	\@glsxtr@autoindex@setname: new ..	165
removed \glsxtrabbrvfmt	192	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added	51	General: removed	
\Glossentrydesc: added	168	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	160	120
\Glossentrysymbol: added	169	\glsdescwidth: added	50
\glossentrysymbol: added	169	\glspagelistwidth: added	51
\GLSaccessdesc: new	144, 149	\glsxtrdoautoindexname: new	164
\GLSaccessdescplural: new ...	145, 149	\glsxtrpostnamehook: new	162
\GLSaccessfirst: new	142, 148	\if@glsxtr@format@override: new ..	164
\GLSaccessfirstplural: new ..	142, 148	\ProvidesGlossariesExtraLang: new	309
\GLSaccesslong: new	146, 150	\RequireGlossariesExtraLang: new	309
\GLSaccesslongpl: new	147, 150	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new	147	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new	147	new	96
\GLSaccessname: new	140, 147	\@GLSxtr@p@acrlong@: new	83
\GLSaccessplural: new	141, 148	\@GLSxtr@p@acrlongpl@: new	83
\GLSaccessshort: new	145, 149	\@GLSxtr@p@acrshort@: new	83
\GLSaccessshortpl: new	146, 150	\@GLSxtr@p@acrshortpl@: new	83
\GLSaccesssymbol: new	143, 148	\@GLSxtr@p@long@: new	82
		\@GLSxtr@p@longpl@: new	83
		\@GLSxtr@p@plural@: new	81
		\@GLSxtr@p@short@: new	82
		\@GLSxtr@p@shortpl@: new	82
		\@GLSxtr@p@text@: new	81
		\@GlsXtrEnableOnTheFly: new	47
		\@Glsxtr: new	47
		\@Glsxtr@p@acrlong@: new	83
		\@Glsxtr@p@acrlongpl@: new	83

\@Glsxtr@p@acrshort@: new	83	\glsenableentryunitcount: new	98
\@Glsxtr@p@acrshortpl@: new	83	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	82	entry's existence	152
\@Glsxtr@p@longpl@: new	82	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	81	entry's existence	152
\@Glsxtr@p@short@: new	81	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	82	check	172
\@Glsxtr@p@text@: new	81	\Glsxtr: new	47
\@Glsxtrpl: new	48	\glxtr: new	47
\@alt@glshyp@opt: new	77	\glxtrcat: new	47
\@glscalt@hyp@opt: new	77	\glxtrdowrglossaryhook: new	77
\@glscalt@hyp@opt@char: new	77	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	77	new	101
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	46
new	97	\Glsxtrpl: new	48
\@glsc@local@increment@currunitcount:		\glxtrpl: new	48
new	97	\glxtrpostlocalreset: new	89
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	89
new	74	\glxtrpostreset: new	89
\@glsxtr: new	47	\glxtrpostunset: new	89
\@glsxtr@addunitcounter: new	96	\glxtrprotectlinks: new	80
\@glsxtr@currunitcount: new	98	\GlsXtrSetAltModifier: new	77
\@glsxtr@ifunitcounter: new	96	\GlsXtrSetDefaultGlsOpts: new	76
\@glsxtr@p@acrlong@: new	83	\glxtrstarflywarn: new	47
\@glsxtr@p@acrlongpl@: new	83	\GlsXtrWarning: new	48
\@glsxtr@p@acrshort@: new	83	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	83	disables \setacronymstyle	103
\@glsxtr@p@long@: new	82	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	82	\@glsxtr@autoindexcrossrefs: new .	15
\@glsxtr@p@plural@: new	81	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	81	new	111
\@glsxtr@p@shortpl@: new	82	\@glsxtr@idx@entrynumberlist: new	113
\@glsxtr@p@text@: new	81	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	98	new	112
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	101	new	113
\@glsxtr@unitcountlist: new	96	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	48	new	112
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	104
empty see value if not set and added		\makeglossaries: new	105
'see' to field key map	38	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	46	\glxtrdiscardperiod: added check	
\cGlsformat: added	95	for first use	173
\cGlsformat: added	95	short-desc: fixed typo in	
\cGlsplformat: added	96	\glxtrinlinefullformat and	
\cGlsplformat: added	95	added missing second argument ...	208
\glstdisablehyper: added	79	1.02 (2016-04-25)	
\glstdohyperlink: added	78	\@glsxtr@current@style: new	49
\glstdonohyperlink: added	80	\Glsfmtfull: new	308

\glsfmtfull: new	307	\@GLSdescplural@: set abbreviation and regular format	64
\Glsfmtfullpl: new	308	\@GLSfirst@: set abbreviation format ..	61
\glsfmtfullpl: new	308	\@GLSfirstplural@: set abbreviation and regular format	63
\Glsfmtlong: new	306	\@GLSname@: set abbreviation and regular format	63
\glsfmtlong: new	306	\@GLSplural@: set abbreviation and regular format	62
\Glsfmtlongpl: new	307	\@GLSsymbol@: set regular format	65
\glsfmtlongpl: new	307	\@GLSsymbolplural@: set regular format	65
\Glsxtrheadfull: new	301	\@GLStext@: set abbreviation and regular format	60
\glsxtrheadfull: new	300	\@GLSuseri@: set regular format	66
\Glsxtrheadfullpl: new	302	\@GLSuserii@: set regular format	66
\glsxtrheadfullpl: new	301	\@GLSuseriii@: set regular format	66
\Glsxtrheadlong: new	300	\@GLSuseriv@: set regular format	67
\glsxtrheadlong: new	299	\@GLSuservi@: set regular format	67
\Glsxtrheadlongpl: new	300	\@GLSdesc@: set abbreviation and regular format	64
\glsxtrheadlongpl: new	299	\@GLSdescplural@: set abbreviation and regular format	64
\Glsxtrtitlefull: new	302	\@GLSfirst@: set abbreviation and regular format	61
\glsxtrtitlefull: new	301	\@GLSfirstplural@: set abbreviation and regular format	62
\Glsxtrtitlefullpl: new	302	\@GLSname@: set abbreviation and regular format	63
\glsxtrtitlefullpl: new	301	\@GLSplural@: set abbreviation and regular format	62
\Glsxtrtitlelong: new	300	\@GLSsymbol@: set regular format	65
\glsxtrtitlelong: new	299	\@GLSsymbolplural@: set regular format	65
\Glsxtrtitlelongpl: new	300	\@GLStext@: set abbreviation and regular format	60
\glsxtrtitlelongpl: new	299	\@GLSuseri@: set regular format	65
\ifglsxtrinsertinside: new	198	\@GLSuserii@: set regular format	66
postfootnote: added redef of		\@GLSuseriii@: set regular format	66
\glsxtrsetupfulldefs	205	\@GLSuseriv@: set regular format	66
stylemods: new	21	\@GLSuservi@: set regular format	67
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format	64
\@GLSfirstplural@: bug fix: misspelt cs name	63	\@GLSdescplural@: set abbreviation and regular format	62
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLSsymbol@: set regular format	65
\@GLSfirstplural@: bug fix: misspelt cs name	62	\@GLSsymbolplural@: set regular format	65
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	62	\@GLStext@: set abbreviation and regular format	60
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	62	\@GLSuseri@: set regular format	65
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	299	\@GLSuserii@: set regular format	66
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	294	\@GLSuseriii@: set regular format	66
1.04 (2015-04-30)		\@GLSuseriv@: set regular format	66
short-em-footnote: renamed from “footnote-em”	260	\@GLSuservi@: set regular format	67
1.04 (2016-05-02)		\@gls@preglossaryhook: added check for entry's existence	171
\@glsxtrpostloctag: new	53	\@GLSdesc@: set abbreviation and regular format	63
\@GLSdesc@: set abbreviation and regular format	64	\@GLSdescplural@: set abbreviation and regular format	64
		\@GLSfirst@: set abbreviation and regular format	61

\@glsfirstplural@: set abbreviation and regular format	62	\glsxtruserparen: new	264
\@glsname@: set abbreviation and regular format	63	\glsxtrusersuffix: new	265
\@glsplural@: set abbreviation and regular format	62	\GlsXtrWarnDeprecatedAbbrStyle: new	197
\@glsymbol@: set regular format	64	short-em-long-em: new	248
\@glsymbolplural@: set regular format	65	short-em-long-em-desc: new	249
\@glstext@: set abbreviation and regular format	60	short-em-nolong: new	251
\@glsxtr@deprecated@abbrstyle: new	197	short-em-nolong-desc: new	252
\@glsxtr@do@style: new	21	short-em-postfootnote: renamed from “postfootnote-em”	262
\@glsxtr@doloctag: new	53	short-footnote: new	204
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	113	short-long-user: new	271
\@glsxtr@pagetag: new	53	short-long-user-desc: new	272
\@glsxtr@pagetag: new	53	short-nolong: new	207
\@glsxtr@preloctag: new	53	short-nolong-desc: new	209
\@glsxtr@preloctag: new	52	short-postfootnote: new	206
\glossentrydesc: added glossdescfont attribute check	156	short-sc-footnote: renamed from “footnote-sc”	224
\Glossentryname: added glossnamefont attribute check	160	short-sc-nolong: new	219
\glossentryname: added glossnamefont attribute check	158	short-sc-nolong-desc: new	220
moved post name hook inside condition	160	short-sc-postfootnote: renamed from “postfootnote-sc”	226
\glsabbrvmfont: new	242	short-sm-footnote: renamed from “footnote-sm”	239
\glsabbrvuserfont: new	264	short-sm-nolong: new	233
\glsfirstabbrvmfont: new	242	short-sm-nolong-desc: new	235
\glsfirstabbrvuserfont: new	264	short-sm-postfootnote: renamed from “postfootnote-sm”	240
\glsfirstlongemfont: new	242	\letabbreviationstyle: new	197
\glsfirstlonguserfont: new	264	\newabbreviationstyle: bug fix: corrected test for existence	195
\glsifnotregularcategory: new ...	153	long-em-noshort-em: new	255
\glslongdefaultfont: new	182	long-em-noshort-em-desc: new	258
\glslongemfont: new	242	long-em-short-em: new	244
\glslongfont: new	182	long-em-short-em-desc: new	245
\glslonguserfont: new	264	long-noshort: new	213
\glsxtrassignfieldfont: new	60	long-noshort-desc: new	212
\GlsXtrEnablePreLocationTag: new .	52	long-noshort-em: renamed from “long-em”	253
\glsxtrfirstscfont: new	214	long-noshort-em-desc: renamed from “long-desc-em”	257
\glsxtrfirstsmfont: new	228	long-noshort-sc: renamed from “long-sc”	221
\glsxtrlongshortdescsort: new ...	199	long-noshort-sc-desc: renamed from “long-desc-sc”	223
\glsxtrpostnamehook: added category check	162	long-noshort-sm: renamed from “long-sm”	235
\glsxtrregularfont: new	54	long-noshort-sm-desc: renamed from \long-desc-sm	237
\glsxtruserfield: new	264		

long-short-user: new	265	docdef option changed to choice	14
long-short-user-desc: new	270	\glxtr@usesee: new	39
\renewabbreviationstyle: new	196	\glxtrusesee: new	39
style: new	21	\glxtruseseeformat: new	39
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	14
\eglssetwidest: new	327	1.07 (2016-08-15)	
\glsFindWidestAnyName: new	330	\@@glxtrp: new	84
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	335	nohyperfirst attribute	61
\glsFindWidestAnyNameSymbol: new	333	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	63
new	334	\@GLSxtrp: new	85
\glsFindWidestLevelTwo: new	331	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new ..	330	nohyperfirst attribute	61
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	335	nohyperfirst attribute	62
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	84
new	332	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	171
new	333	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new .	330	nohyperfirst attribute	61
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new	329	nohyperfirst attribute	62
\glsfirstlongfootnotefont: new ..	202	\@glxtrinmark: new	291
\glsgetwidestname: new	329	\@glxtrnotinmark: new	291
\glsgetwidestsubname: new	329	\@glxtrp: new	84
\glslongfootnotefont: new	202	\@glxtrp@opt: new	83
\glxtrAltTreeIndent: new	327	\glossxtrsetpopts: new	84
\glxtralttreeInit: new	327	\glsps: new	86
\glxtrAltTreePar: new	327	\glspt: new	86
\glxtrAltTreeSetHangIndent: new	336	\glxtr@entry@p: new	85
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new	202
new	337	\glxtrchecknohyperfirst: new	61
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new	156
new	327	\glxtrifinmark: new	291
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new	87
new	326	\Glsxtrp: new	86
\glxtrComputeTreeIndent: new ...	336	\glxtrp: new	85
\glxtrComputeTreeSubIndent: new	336	\glxtrsetpopts: new	84
\glxtrtreetopindent: new	327	short-long-desc: added text key	202
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form	246	plural key	202
\xglsetwidest: new	328	long-short-desc: added missing text	
1.06 (2016-06-18)		key	200
\@glsdoifexistsorwarn: new	14	fixed misspelling of \glsabbrvfont .	200
\@glxtr@docdefval: new	14	footnote: changed first forms to use	
\@glxtr@usesee: new	39	\glsfirstlongfootnotefont ...	202
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	25	from first keys	204

switched from \glsfirstlongfont to	1.10 (2016-12-17)
\glsfirstlongfootnotefont ... 205	\@GLSp1@: fixed bug caused by typo in
\RestoreAcronyms: modified	command name 55
\@gls@link@checkfirsthyper to	1.11 (2017-01-19)
set \glsxtrifwasfirstuse 104	\@glsxtr@do@redef@for@gl@sentries:
1.08 (2016-12-13)	new 6
\@glsxtr@record: new 8	\@glsxtr@noidx@do: new 131
\@GLS@: added \@glsxtr@record 55	\@glsxtr@redef@for@gl@sentries: new . 6
\@GLSp1@: added \@glsxtr@record ... 55	\@glsxtr@shortcutsval: new 19
\@Gls@: added \@glsxtr@record 55	\@glsxtr@unsrt@getgroup@title: new 130
\@GLspl@: added \@glsxtr@record ... 55	\@print@noidx@glossary: added
\@gls@: added \@glsxtr@record 55	redefinition 115
\@gls@@link@: added	\glsxtr@addloc@listfield: added
\@glsxtr@record 56	group key 12
\@gls@field@link: added	added location key 12
\@glsxtr@record 54	\glsxtr@fields: new 123
\@gls@saveentrycounter: new 25	\glsxtr@linkprefix: new 123
\@glsdisp: added \@glsxtr@record .. 56	\glsxtr@org@new@ignored@glossary:
\@glspl@: added \@glsxtr@record ... 55	new 34
\@glsxtr@dorecord: new 10	\glsxtr@s@new@ignored@glossary: new 34
\@glsxtr@err@undefaction: new 6	\glsxtr@shortcutsval: new 123
\@glsxtr@record: new 7	\glsxtr@texencoding: new 123
\@glsxtr@warn@onexists@do: new ... 6	\glsxtr@writefields: new 123
\@glsxtr@warn@undefaction: new 6	\GlsXtrLoadResources: new 122
\@print@unsrt@glossary: new 127	\glsxtr@resourcefile: changed
General: added record package option ... 12	extension to .glstex 122
\glsadd: added \@glsxtr@record 59	\new@ignored@glossary: added starred
\glsdoifexists: now defines	version 34
\glslabel 37	1.12 (2017-02-03)
\glsxtr@do@wrglossary: new 25	\@glsxtr@recordcounter: new 10
\glsxtr@addloc@listfield: new 11	\@gls@preglossaryhook: check for
\glsxtr@indexonly@saveentrycounter:	definition 171
new 11	\@glsxtr@counter@recordhook: new . 125
\glsxtr@record: new 125	\@glsxtr@display@loc: new 115
\glsxtr@resource: new 123	\@glsxtr@docounter@record: new ... 125
\glsxtr@saveentrycounter: new 25	\@glsxtr@longnew@glossary@entry:
\glsxtr@setup@record: new 11	new 33
\glsxtr@assignfieldfont: added check	\@glsxtr@noop@recordcounter: new . 11
for existence 60	\@glsxtr@op@recordcounter: new ... 11
\glsxtr@resourcefile: new 122	\@glsxtr@provide@storagekey: new . 26
\printunsrt@glossaries: new 127	\@glsxtr@s@longnew@glossary@entry:
\printunsrt@glossary: new 127	new 33
1.09 (2016-12-16)	\@glsxtrentryfmt: new 28
\@glsxtr@gettype: new 111	\@glsxtr@index@aliased: new 75
\@glsxtr@mixed@assign@sortkey:	\@glsxtr@set@alias@noindex: new 75
new 111	\@new@glossary@entry@posthook: added
\@print@glossary: redefined to save	check for alias key 43
options 110	\@no@glsxtr@index@aliased: new 75
\glsxtr@make@glossaries: new 111	\@printunsrt@glossary: new 127

General: added target key to printgloss family	110	\GlsXtrLoadResources: removed restriction on only one per document	122
\apptoglossarypreamble: new	32	\glxtrlocrangefmt: new	116
\csGlsXtrLetField: new	31	\glxtrpostlongdescription: new ..	34
\eGlsXtrSetField: new	31	\glxtrprovidestoragekey: new	26
\gGlsXtrSetField: new	31	\GlsXtrRecordCounter: new	125
\glsdohyperlink: added check for alias field	78	\glxtrresourcecount: new	122
\glsnoidxdisplayloc: added redefinition	115	\glxtrresourcefile: added catcode change for @	122
\glissettoctitle: added patch	35	\glxtrsetaliasnoindex: new	75
\glxtr@counterrecord: new	125	\GlsXtrSetField: new	30
\glxtr@langtag: new	123	\glxtrsetfieldifexists: new	30
\glxtr@newabbreviation: new	178	\glxtrunsrtdo: new	130
\glxtr@org@newignoredglossary: Added check for existence	34	\Glsxtrusefield: new	30
\glxtr@pluralsuffixes: new	123	\glxtrusefield: new	30
\glxtr@provideignoredglossary: new	35	short-postlong-user: new	268
\glxtr@s@newignoredglossary: Added check for existence	34	short-postlong-user-desc: new ...	270
\glxtr@s@provideignoredglossary: new	36	\longnewglossaryentry: added starred version	33
\glxtrabbrvpluralsuffix: new ...	182	long-postshort-user: new	266
\glxtralias: new	42	long-postshort-user-desc: new ...	267
\glxtrcopytoglossary: new	36	postdot: new	15
\glxtrdeffield: new	30	\pretoglossarypreamble: new	32
\glxtrdisplayendloc: new	116	\print@noop@unsrtglossaryunit: new	130
\glxtrdisplayendlochook: new ...	116	\print@op@unsrtglossaryunit: new	129
\glxtrdisplaysingleloc: new	116	\printunsrtglossary: added starred form	127
\glxtrdisplaystartloc: new	116	\printunsrtglossaryhandler: new .	129
\glxtrdeffield: new	30	\printunsrtglossaryunit: new	11
\glxtrentryfmt: new	28	\printunsrtglossaryunitsetup: new	129
\glxtrfielddolistloop: new	29	\provideignoredglossary: new	35
\glxtrfieldforlistloop: new	29	\s@glxtr@provide@storagekey: new	26
\glxtrfieldifinlist: new	29	\s@printunsrtglossary: new	127
\glxtrfieldlistadd: new	28	\xGlsXtrSetField: new	31
\glxtrfieldlistead: new	29	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new	29	\@glsdisp: removed	
\glxtrfieldlistxadd: new	29	\@glxtr@org@glsdisp	56
\glxtrfieldxifinlist: new	29	\glxtrsetaliasnoindex: switched to \providecommand	75
\glxtrfmt: new	27	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new	27	\@gls@link: added redefinition	57
\GlsXtrFmtField: new	27	\@gls@noidx@getgrouptitle: new ..	113
\glxtrifkeydefined: new	26	\@gls@removespaces: new	116
\glxtrindexaliased: new	75	\@glxtr@do@automake@err: new ...	124
\GlsXtrLetField: new	31	\@glxtr@org@gloautosee: new	23
\GlsXtrLetFieldToField: new	31	\@glxtr@record: added third arg	7
		\@glxtr@recordsee: new	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue	59	\@glo@autosee: added redefinition	24
added \glsadd option thevalue	59	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	79	bug	113
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	91	check for seealso field	44
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@	99	instead of \csname	131
\glsnavigation: new	114	\@glsxtr@dorecordnodefer: new	10
\glsxtr@org@getgrouptitle: new ..	114	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new	7	misspelt command	128
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake	124	new	129
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	116	\@gls@setupsort@none	13
\glsxtrgetgrouptitle: new	114	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new	56	redefinition	24
\glsxtrlocationhyperlink: new ...	117	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new	114	\providecommand lines	123
\glsxtrsupphypernumber: new	117	\@glsxtrautoindex: new	165
\ifglsxtrwrglossbefore: new	56	\@glsxtrautoindexentry: new	165
1.15 (2017-05-10)		\@glsxtrautoindexsort: new	165
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new	40
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new	43
short-em-long-em: fixed spelling of		\@glsxtrseelist: new	40
\glsabbrvfont	248	\@glsxtruseseealso: new	39
short-long: fixed spelling of		\@glsxtruseseealsoformat: new	40
\glsabbrvfont	200	\@seealso: new	40
short-long-user: fixed spelling of		autoseeindex: new	15
\glsabbrvfont	271	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	177
\glsabbrvfont	268	\@glsxtr@markwordseps: new	177
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	270	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	112
\glsabbrvfont	245	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	266	\glsxtrundeftag	113
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	267	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	112
\glsabbrvfont	198	\@glsxtrifhyphenstart: new	273
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	265	in the abbreviation styles	198
footnote: fixed spelling of		\glsabbrvhyphenfont: new	274
\glsabbrvfont	203	\glsabbrvonlyfont: new	287
postfootnote: fixed spelling of		\glsabbrvscfont: new	214
\glsabbrvfont	204	\glsabbrvsmfont: new	228

\glsabbrvuserfont: initialised to default font	264	short-long-user-desc: corrected first forms	272
\glsfirstabbrvhyphenfont: new ...	274	short-nolong-desc-noreg: new ...	209
\glsfirstabbrvonlyfont: new	287	short-nolong-noreg: new	207
\glsfirstabbrvscfont: new	214	long-em-noshort-em-desc-noreg: new	260
\glsfirstabbrvsmfont: new	228	long-em-noshort-em-noreg: new ...	256
\glsfirstlonghyphenfont: new ...	274	long-hyphen-noshort-desc-noreg: new	276
\glsfirstlongonlyfont: new	287	long-hyphen-postshort-hyphen: new	279
\glslonghyphenfont: new	274	long-hyphen-postshort-hyphen-desc: new	281
\glslongonlyfont: new	287	long-hyphen-short-hyphen: new ...	274
\glslonguserfont: initialised to default font	264	long-hyphen-short-hyphen-desc: new	275
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong	178	long-noshort-desc-noreg: new ...	212
\GlsXtrDefineAcShortcuts: new ...	18	long-noshort-noreg: new	213
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside	192	long-only-short-only: new	287
\glsxtrhyphensuffix: new	274	long-only-short-only-desc: new ..	289
\glsxtrifhyphenstart: new	273	long-short-user-desc: corrected first forms	270
\glsxtrlonghyphen: new	278	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	275	stylemods: changed default value to “default”	21
\glsxtrlonghyphenshort: new	273	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	199	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new	289	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref	10
\glsxtronlydescsort: new	288	\@glsxtr@dorecordnodefer: Use \theglscopycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new	287	\@glsxtr@record@setting: new	12
\glsxtrparen: new	180	\@glsxtr@record@setting@alsoindex: new	12
\glsxtrposthyphenlong: new	284	\@glsxtrifhasfield: new	30
\glsxtrposthyphenshort: new	278	General: added \glslink option theHvalue	57
\glsxtrposthyphensubsequent: new	279	added \glslink option thevalue ...	57
\glsxtrshortdescname: new	208	\glsxtr@writefields: removed double-quotes around \jobname ..	124
\glsxtrshorthyphen: new	283	\glsxtrdoautoindexname: changed format test	165
\glsxtrshorthyphenlong: new	281	\glsxtrhyperlink: new	79
\glsxtrshortlongdescname: new ...	201	\glsxtrifhasfield: new	29
\glsxtrshortlongdescsort: new ...	201	\GlsXtrSetDefaultNumberFormat: new	7
\Glsxtrsubsequentfmt: new	194	\s@glsxtrifhasfield: new	30
\glsxtrsubsequentfmt: new	194		
\Glsxtrsubsequentplfmt: new	194		
\glsxtrsubsequentplfmt: new	194		
\glsxtrword: new	177		
\glsxtrwordsep: new	177		
short-hyphen-long-hyphen: new ...	282		
short-hyphen-long-hyphen-desc: new	283		
short-hyphen-postlong-hyphen: new	284		
short-hyphen-postlong-hyphen-desc: new	286		

1.20 (2017-09-11)		
\@glstrhypernameprefix: new	111
\glsdohypertarget: added redefinition		111
\printunstrtglossaryunitsetup:		
switched from redefining		
\glolinkprefix to		
\@glstrhypernameprefix	129
1.21 (2017-11-03)		
\@glstr@record: added check for		
default options	9
\@glstrwrglossmark: new	22
\glslink: changed \let to \def	80
\@glstr@checkgroup: new	130
\@glstr@defpostpunc: new	15
\@glstr@do@record@wrglossary:		
new	7
\@glstr@dosee@alsoindex@glossary:		
new	23
\@glstr@doseeglossary: new	23
\@glstr@noidx@do: removed code		
dealing with the group	131
\@glstr@record@setting@off: new	.	12
\@glstr@record@setting@only: new		12
\@glstr@rglstrigger@record: new		136
\@glstrglossentry: new	125
\@glstrnewgls: new	132
\@glstrsetaliasnoindex: changed to		
use \glstrifhasfield instead of		
\ifglshasfield	75
\@glstrwrglossmark: new	22
\@rGLS: new	138
\@rGLS@: new	138
\@rGLSpl: new	138
\@rGLSpl@: new	139
\@rGls: new	137
\@rGls@: new	137
\@rGlspl: new	138
\@rGlspl@: new	138
\@rgls: new	136
\@rgls@: new	137
\@rglspl: new	137
\@rglspl@: new	137
General: adjusted mcolalttree	342
ac	20
modified index to remove hard coded		
\space	322
modified list to remove hard coded		
\space	312
moved conditional outside of		
\glsgroupskip	315–319, 321
new	345
redefined altlistgroup to discourage		
breaks after group headings	313
redefined altlisthypergroup to		
discourage breaks after group		
headings	314
redefined alttreegroup to discourage		
breaks after group headings	338
redefined alttreehypergroup to		
discourage breaks after group		
headings	338
redefined indexgroup to discourage		
breaks after group headings	323
redefined indexhypergroup to		
discourage breaks after group		
headings	323
redefined listgroup to discourage		
breaks after group headings	313
redefined listhypergroup to		
discourage breaks after group		
headings	313
redefined mcolalttreegroup to		
discourage breaks after group		
headings	342
redefined mcolalttreehypergroup to		
discourage breaks after group		
headings	343
redefined mcolalttreespannav to		
discourage breaks after group		
headings	343
redefined mcolindexgroup to		
discourage breaks after group		
headings	339
redefined mcolindexhypergroup to		
discourage breaks after group		
headings	339
redefined mcolindexspannav to		
discourage breaks after group		
headings	339
redefined mcoltreegroup to		
discourage breaks after group		
headings	340
redefined mcoltreehypergroup to		
discourage breaks after group		
headings	340
redefined mcoltreenamegroup to		
discourage breaks after group		

headings	341	\glxtrbookindexbetween: new	346
redefined		\glxtrbookindexbookmark: new ...	346
mcoltreenonamehypergroup to		\glxtrbookindexcols: new	345
discourage breaks after group		\glxtrbookindexcolspread: new ..	346
headings	341	\glxtrbookindexfirstmark: new ..	350
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	350
discourage breaks after group		\glxtrbookindexformatheader: new	346
headings	342	\glxtrbookindexgroupskip: new ..	346
redefined mcoltreepannav to		\glxtrbookindexlastmark: new ...	350
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	350
headings	340	\glxtrbookindexmarkentry: new ..	349
redefined treegroup to discourage		\glxtrbookindexname: new	345
breaks after group headings	324	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new	345
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings	325	new	345
redefined treenonamegroup to		\glxtrbookindexprelocation: new	345
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings	326	new	346
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	346
discourage breaks after group		\glxtrbookindexsubname: new	345
headings	326	\glxtrbookindexsubprelocation:	
debug: new	22	new	345
\gglsetwidest: new	327	\glxtrbookindexsubsubatendgroup:	
\gl:disablehyper: added check for		new	346
existence	79	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	79	new	346
\gl:enablehyper: changed to use \def		\glxtrbookindexthepage: new	349
rather than \let	79	\glxtrdetoklocation: new	135
\Glsfmtname: new	304	\glxtrenablerecordcount: new ...	135
\gl:fmtname: new	303	\glxtrglossentry: new	125
\glshex: new	132	\glxtrgroupfield: new	130
\gl:listchildpostlocation: new ..	312	\GlsXtrheadname: new	295
\gl:listchildprelocation: new ...	312	\glxtrheadname: new	295
\gl:listprelocation: new	312	\GlsXtrIfFieldEqStr: new	31
\gl:navhyperlink: patched	77	\glxtriflabelinlist: new	129
\gl:seeitemformat: new	39	\glxtrifrecordtrigger: new	135
\gl:showtarget: new	23	\glxtrindexseealso: added check	
\gl:treechildprelocation: new ...	322	that the entry exists	40
\gl:treeprelocation: new	322	\glxtrinithyperoutside: new	57
\gl:triggerrecordformat: new	136	\GlsXtrLocationRecordCount: new ..	134
\gl:useabbrvfont: new	192	\glxtrnewgls: new	132, 133
\gl:uselongfont: new	192	\glxtrnewGLSlike: new	134
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new	133
new	8	\glxtrnewrgls: new	134
\glxtr@org@@do@wrglossary: new ..	25	\glxtrnewrGLSlike: new	134
\glxtr@org@dohyperlink: new	77	\glxtrnewrglslike: new	134
\glxtr@setbookindexmark: new ...	350	\glxtrprelocation: new	311, 345
\glxtrbookindexatendgroup: new ..	346	\GlsXtrRecordCount: new	134

\glxtrrecordtriggervalue: new ..	135	1.22 (2017-11-08)	
\glxtrresourcefile: now disables		\@glxtr@nopostpunc: new	110
record key	122	\@glxtr@orgprintglossary: changed	
\glxtrresourceinit: new	132	explicit \let for \nopostdesc to	
\GlsXtrSetRecordCountAttribute:		\glxtractivatenopost	109
new	135	\@glxtrglossentryother: new	126
\glxtrtitlename: new	295	\glossentrynameother: new	162
\glxtrtitleorpdforheading: new ..	291	\glseeitemformat: switched check	
\GlsXtrTotalRecordCount: new	134	from regular to short	39
\glxtrwrglossmark: new	22	\glxtr@setaccessdisplay: new ...	162
short-em: new	250	\glxtr@writefields: provide	
short-sc: corrected first letter		\glxtr@record in aux file	123
uppercasing	218	\glxtractivatenopost: new	109
short-sm: corrected first letter		\glxtrbookindexprelocation:	
uppercasing	233	removed check for no post dot	345
\ifglxtr@hyperoutside: new	57	\glxtrglossentryother: new	126
all: new	310	\glxtrnopostpunc: new	110
nolong-short: new	210	1.23 (2017-11-12)	
nolong-short-em: new	252	\@@glxtrfmt: added check for indexing	28
nolong-short-noreg: new	210	added grouping	27
nolong-short-sc: new	220	new	27
nolong-short-sm: new	235	\@glxtr@nopostpunc@postdesc: new	110
nopostdot: new	15	\@glxtr@restore@postpunc: new ..	110
\printunsrtglossaryentryprocesshook:		\@glxtrentryfmt: fixed missing label	
new	128, 129	argument	28
\printunsrtglossarypredoglossary:		\@glxtrfmt: new	27
new	129	\eglsupdatewidest: new	328
\rGLS: new	138	\gglsupdatewidest: new	328
\rGls: new	137	\glsupdatewidest: new	328
\rgls: new	136	\GlsXtrDefineAbbreviationShortcuts:	
\rGLSformat: new	139	changed \newabbr definition to use	
\rGlsformat: new	139	\providecommand	17
\rglsformat: new	139	\GlsXtrDefineAcShortcuts: changed	
\rGLSpl: new	138	\newabbr definition to use	
\rGlspl: new	138	\providecommand	18
\rglspl: new	137	\glxtrfmtdisplay: new	28
\rGLSplformat: new	139	\glxtrifcustomdiscardperiod: new	172
\rGlsplformat: new	139	\GlsXtrIfFieldUndef: new	30
\rglsplformat: new	139	\glxtrrestorepostpunc: new	110
\s@glxtrifhasfield: switched from		\s@glxtrfmt: new	27
\ifdef to \ifundef	30	\s@glxtrfmt: new	27
		\xglsupdatewidest: new	328

Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

Symbols	
\.	15, 16, 173, 322
\@	122
\@cGLS@	91, 100
\@cGLSpl@	91, 100
\@cGls@	91, 99
\@cGlspl@	91, 100
\@cgls@	91, 99
\@cglspl@	91, 99
\@do@wrglossary	8, 9, 106
\@do@wrglossary	11, 13, 25, 59, 75
\@glo@assign@sortkey	111
\@glo@list	6
\@glo@type	127
\@glossarysec	346
\@gls@expand@field	26
\@glslocalreset	89
\@glslocalunset	89
\@glsreset	89
\@glsunset	89
\@glsxtr@autoindex@escspch	167, 168
\@glsxtr@checkspch	165, 166, 168
\@glsxtr@disabledflycommand	49
\@glsxtr@org@postdescription	110
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 125
\@glsxtrfmt	27
\@glsxtrp	84, 85
\@glsxtrpostloctag	52
\@glsxtrpreloctag	52
\@glsxtrwrglossmark	8, 11, 23, 25, 41, 45, 106
\@newglossaryentry@defcounters	90
\@newglossaryentry@defunitcounters	98
\@par	327
\@ACRlong	81
\@ACRlongpl	81
\@ACRshort	80
\@ACRshortpl	81
\@Acrlong	81
\@Acrlongpl	81
\@Acrshort	80
\@Acrshortpl	81
\@GLS@	80, 94, 95, 138
\@GLSdesc@	64
\@GLSpl@	80, 94, 95, 139
\@GLSplural@	81
\@GLSsymbol@	65
\@GLStext@	81
\@GLSxtr@full	183
\@GLSxtr@fullpl	185
\@GLSxtr@p@acrlong@	81
\@GLSxtr@p@acrlongpl@	81
\@GLSxtr@p@acrshort@	80
\@GLSxtr@p@acrshortpl@	81
\@GLSxtr@p@long@	80
\@GLSxtr@p@longpl@	80
\@GLSxtr@p@plural@	80
\@GLSxtr@p@short@	80
\@GLSxtr@p@shortpl@	80
\@GLSxtr@p@text@	80
\@GLSxtrlong	80, 188
\@GLSxtrlongpl	80, 191
\@GLSxtrp	88
\@GLSxtrshort	80, 186
\@GLSxtrshortpl	80, 189
\@Gls@	80, 93, 94, 137
\@Gls@acentryname	102
\@Gls@entry@field	72, 87, 163
\@Gls@entryname	102
\@GlsXtrEnableOnTheFly	46
\@Glspl@	80, 93, 94, 138
\@Glsplural@	81
\@Glstext@	81
\@Glsxtr	47, 49

\@Glsxtr@full	183	\@firstofone	60, 128, 157, 164, 170
\@Glsxtr@fullpl	184	\@firstofthree	56,
\@Glsxtr@p@acrlong@	81	60, 68–71, 77, 182, 184, 185, 187, 189, 190	
\@Glsxtr@p@acrlongpl@	81	\@firstoftwo ..	61–65, 69, 71, 74, 77, 104,
\@Glsxtr@p@acrshort@	80	162, 174, 175, 183–185, 189–191, 291, 292	
\@Glsxtr@p@acrshortpl@	81	\@for	6, 21, 44, 90,
\@Glsxtr@p@long@	80	102, 105, 108, 114, 128, 135, 155, 162, 169	
\@Glsxtr@p@longpl@	80	\@glo@alias	41, 42
\@Glsxtr@p@plural@	80	\@glo@assign@sortkey	108
\@Glsxtr@p@short@	80	\@glo@autosee	23
\@Glsxtr@p@shortpl@	80	\@glo@autoseehook	42
\@Glsxtr@p@text@	80	\@glo@category	96
\@Glsxtrlong	80, 187	\@glo@check@sortallowed	108
\@Glsxtrlongpl	80, 191	\@glo@counterprefix	10, 117
\@Glsxtrp	87	\@glo@countunit	96
\@Glsxtrpl	48, 49	\@glo@default@sorttype	108
\@Glsxtrshort	80, 186	\@glo@desc	33
\@Glsxtrshortpl	80, 189	\@glo@descplural	33
\@acrlong	81	\@glo@group	12
\@acrlongpl	81	\@glo@label	
\@acrshort	80	11, 12, 26, 38, 41–43, 72, 79, 329–336
\@acrshortpl	81	\@glo@location	12
\@afterheading		\@glo@loclist	11
.....	313, 314, 323–326, 339–342, 349	\@glo@name	165
\@alt@glshyp@opt	77	\@glo@no@assign@sortkey	111
\@auxout	10, 11,	\@glo@parent	331, 332
45, 53, 92, 100, 105, 106, 118, 122–125, 349		\@glo@see	38–40, 42–44
\@bibgl@restoreat	122	\@glo@seealso	41, 42
\@cGLS	94	\@glo@sort	165
\@cGLS@	91, 94, 100	\@glo@sorttype	108, 115
\@cGLSpl	95	\@glo@text	56
\@cGLSpl@	91, 95, 100	\@glo@thislettergrp	131
\@cGls@	91, 99	\@glo@thisvalue	264
\@cGlspl@	91, 100	\@glo@tmp	26, 40, 72
\@cgl@	91, 99	\@glo@type	43, 78, 102, 105, 108, 109,
\@cglspl@	91, 99	111, 114, 115, 117, 118, 121, 122, 127, 128	
\@disable@onlypremakeg	106	\@glo@types	153, 154, 329–335
\@do@auxoutstuff	118	\@glossary@default@style ..	49, 50, 109, 344
\@do@gl@getcounterprefix	10	\@glossarystyle	109
\@do@gl@see	42, 43	\@gl@	80, 93, 94, 137
\@do@newglossaryentry ..	102, 103, 179, 180	\@gl@@link	56
\@do@seeglossary	13, 23, 45, 106	\@gl@ReturnAfterFi	117
\@do@wrglossary	58, 59, 136	\@gl@actualchar	166
\@empty	60, 68–71, 110, 165, 166, 182–191	\@gl@adjustmode	59
\@end@glxtr@addunused	44	\@gl@alt@hyp@opt	77
\@end@glxtr@gettype	108, 111	\@gl@alt@hyp@opt@char	77
\@end@glxtr@usesee	39	\@gl@alt@hyp@opt@keys	77
\@end@glxtrifhyphenstart	273	\@gl@automake	108
\@endfortrue	162, 195	\@gl@between	114

\@gls@checkedmkidx	165, 166, 168	\@gls@noidxloclist@sep	112
\@gls@checkmkidxchars	41, 165	\@gls@noref@warn	106, 115
\@gls@codepage	118	\@gls@org@glsnoidxdisplayloc ..	112, 113
\@gls@counter	9, 10, 57, 59, 75, 136	\@gls@org@glsseeformat	112, 113
\@gls@currentlettergroup ...	115, 128, 131	\@gls@preglossaryhook	109, 170
\@gls@declareoption	5	\@gls@prevlevel	337, 338, 342, 343
\@gls@default@longpl	178, 179	\@gls@quotechar	165
\@gls@doautomake	108, 124	\@gls@reference	45, 105, 106
\@gls@doautomake@err	124	\@gls@saveentrycounter .	13, 25, 58, 59, 136
\@gls@encapchar	166	\@gls@see@noindex	24, 122
\@gls@entry@count	91, 92	\@gls@setdefault@glslink@opts	9, 28, 58, 76
\@gls@entry@field	26, 30, 42, 72, 85–88, 91, 126, 127	\@gls@setsort	58
\@gls@entry@unitcount	100	\@gls@setupsort@none	13
\@gls@field@font	60–67	\@gls@short	178, 179
\@gls@field@link	60–67, 72, 73	\@gls@shortpl	176, 179
\@gls@getcounterprefix	10	\@gls@sort	130
\@gls@getgrouptitle	114, 128	\@gls@thisval	162
\@gls@grptitle	78, 114	\@gls@tmp	114
\@gls@hyp@opt	72, 73, 77, 94, 95, 133, 136–138, 182–191	\@gls@tmpb	168
\@gls@hyp@opt@cs	77	\@gls@type	106, 108, 195, 329–336
\@gls@ifinlist	129	\@gls@write@entrycounts	91
\@gls@increment@currcount	91	\@gls@write@entryunitcounts	100
\@gls@increment@currunitcount	99	\@gls@write@entryunitcounts@do	101
\@gls@keymap ..	11, 12, 26, 39, 41, 72, 123, 162	\@gls@xref	11, 41
\@gls@label ...	8–10, 45, 76, 77, 106, 125, 195	\@gls@abbrv@current@abbreviation	178, 192
\@gls@levelchar	166	\@gls@sacronymlists	102
\@gls@link	27, 54, 56, 68–71, 182–191	\@gls@doifexistsorwarn ...	14, 158–161, 163
\@gls@link@checkfirsthyper	56, 104	\@gls@entry	92, 100, 101
\@gls@link@label	57, 136	\@gls@link	58, 78–80
\@gls@link@nocheckfirsthyper	54, 67–71, 182–191	\@gls@nextpages	109
\@gls@link@opts	57	\@gls@nonextpages	109
\@gls@list	114	\@gls@numberformat	9, 10, 57, 59, 75, 136, 162, 165
\@gls@local@increment@currcount	91	\@gls@order	105
\@gls@local@increment@currunitcount	99	\@gls@pl@	80, 93, 94, 137
\@gls@location	131, 132	\@gls@plural@	81
\@gls@loclist	112, 113, 131, 132	\@gls@punc@token	174, 175
\@gls@long	178	\@gls@recordlocref	10
\@gls@longpl	176, 178, 179	\@gls@showtarget	79
\@gls@map	162	\@gls@style@altlist	312
\@gls@nohyperlist	34, 36	\@gls@style@altlistgroup	313
\@gls@noidx@do	115	\@gls@style@altlisthypergroup	314
\@gls@noidx@getgrouptitle	128	\@gls@style@alttree	326
\@gls@noidx@nosanitizesort	107	\@gls@style@alttreegroup	338
\@gls@noidx@sanitizesort	107	\@gls@style@alttreehypergroup	338
\@gls@noidxloclist@finalsep	112	\@gls@style@index	322
\@gls@noidxloclist@prev	112	\@gls@style@indexgroup	323
		\@gls@style@indexhypergroup	323

<code>\@glsstyle@inline</code>	322	<code>\@glsxtr@autoseeindextrue</code>	15
<code>\@glsstyle@list</code>	312	<code>\@glsxtr@bookindex@atendgroup</code> ..	347, 349
<code>\@glsstyle@listdotted</code>	311	<code>\@glsxtr@bookindex@atsubendgroup</code> ..	348
<code>\@glsstyle@listgroup</code>	313	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	348
<code>\@glsstyle@listhypergroup</code>	313	<code>\@glsxtr@bookindex@between</code>	347, 349
<code>\@glsstyle@mcolalttree</code>	342	<code>\@glsxtr@bookindex@sep</code>	347, 348
<code>\@glsstyle@mcolalttreegroup</code>	342	<code>\@glsxtr@bookindex@subatendgroup</code> ..	
<code>\@glsstyle@mcolalttreehypergroup</code> ..	343	347, 349
<code>\@glsstyle@mcolalttreespannav</code>	343	<code>\@glsxtr@bookindex@subbetween</code> ..	347, 348
<code>\@glsstyle@mcolindexgroup</code>	339	<code>\@glsxtr@bookindex@subsep</code>	347, 348
<code>\@glsstyle@mcolindexhypergroup</code>	339	<code>\@glsxtr@bookindex@subsubatendgroup</code>	
<code>\@glsstyle@mcolindexspannav</code>	339	347, 349
<code>\@glsstyle@mcoltreegroup</code>	340	<code>\@glsxtr@bookindex@subsubbetween</code> ..	
<code>\@glsstyle@mcoltreehypergroup</code>	340	347, 348
<code>\@glsstyle@mcoltreenamegroup</code>	341	<code>\@glsxtr@bookindex@groupskip</code> ...	347, 349
<code>\@glsstyle@mcoltreenamehypergroup</code>	341	<code>\@glsxtr@cat</code>	90, 102, 135, 169
<code>\@glsstyle@mcoltreenamespannav</code> ..	342	<code>\@glsxtr@checkgroup</code>	128
<code>\@glsstyle@mcoltreespannav</code>	340	<code>\@glsxtr@counterrecordhook</code>	10, 11
<code>\@glsstyle@tree</code>	324	<code>\@glsxtr@csname</code>	97, 99
<code>\@glsstyle@treegroup</code>	324	<code>\@glsxtr@current@style</code>	50, 344
<code>\@glsstyle@treehypergroup</code>	325	<code>\@glsxtr@currentunitcount</code>	97, 99
<code>\@glsstyle@treenoname</code>	325	<code>\@glsxtr@currunitcount</code>	98, 100
<code>\@glsstyle@treenonamegroup</code>	326	<code>\@glsxtr@declareoption</code>	5, 15, 17, 20
<code>\@glsstyle@treenonamehypergroup</code> ...	326	<code>\@glsxtr@defaultnoglossarywarning</code> ..	20
<code>\@glstarget</code>	79, 80, 111	<code>\@glsxtr@defaultnumberformat</code>	
<code>\@glstext@</code>	81	7, 9, 57, 59, 75, 162, 165
<code>\@glswidestname</code>	329, 336	<code>\@glsxtr@defpostpunc</code>	15, 16, 23
<code>\@glsxtr</code>	47, 49	<code>\@glsxtr@deprecated@abbrstyle</code>	
<code>\@glsxtr@@do@wrglossary</code>	106	223, 224, 226,
<code>\@glsxtr@abbreviationsdef</code>	17, 24	228, 237, 239, 240, 242, 255, 258, 262, 264
<code>\@glsxtr@accessdisplay</code>	162, 163	<code>\@glsxtr@disabledflycommand</code>	49
<code>\@glsxtr@activate@initialtagging</code> ..		<code>\@glsxtr@display@loc</code>	115
.....	170, 171	<code>\@glsxtr@do@wrintex</code>	76
<code>\@glsxtr@addunitcounter</code>	96	<code>\@glsxtr@do@glsglisablehyperinlist</code> ..	74
<code>\@glsxtr@addunused</code>	44	<code>\@glsxtr@do@record@wrglossary</code>	8, 13
<code>\@glsxtr@addunusedxrefs</code>	43, 44	<code>\@glsxtr@do@redef@forglsentries</code>	7
<code>\@glsxtr@attrval</code>	58, 156–161, 163, 165	<code>\@glsxtr@do@style</code>	21, 309
<code>\@glsxtr@autoindex@at</code>	165, 166	<code>\@glsxtr@do@titlecaps@warn</code>	
<code>\@glsxtr@autoindex@doextra@esc</code>	165	157–160, 163, 170
<code>\@glsxtr@autoindex@encap</code>	165–167	<code>\@glsxtr@doabbreviationsdef</code>	17
<code>\@glsxtr@autoindex@esc</code>	165, 167, 168	<code>\@glsxtr@doaccsupp</code>	20, 22
<code>\@glsxtr@autoindex@escat</code>	166, 167	<code>\@glsxtr@docdefval</code>	14, 45
<code>\@glsxtr@autoindex@escencap</code> ...	166, 167	<code>\@glsxtr@docounterrecord</code>	11
<code>\@glsxtr@autoindex@esclevel</code> ...	166, 167	<code>\@glsxtr@doglossary</code>	128
<code>\@glsxtr@autoindex@escquote</code> ...	165, 167	<code>\@glsxtr@doiflabelinlist</code>	129
<code>\@glsxtr@autoindex@level</code>	166, 167	<code>\@glsxtr@doloctag</code>	52
<code>\@glsxtr@autoindex@setname</code>	164	<code>\@glsxtr@dorecord</code>	8, 9
<code>\@glsxtr@autoindexcrossrefs</code> ..	13, 14, 38, 42	<code>\@glsxtr@dorecordnodefer</code>	8, 9
<code>\@glsxtr@autoseeindexfalse</code>	13	<code>\@glsxtr@dosee@alsoindex@glossary</code> ..	13

\@glxtr@doseeglossary	13, 23	\@glxtr@notfoundinlist	175
\@glxtr@dostylewarn	195	\@glxtr@op@recordcounter	13
\@glxtr@enabletagging	169	\@glxtr@optlist	49
\@glxtr@end@	46	\@glxtr@org@starttoc	290, 291
\@glxtr@endescspch	165–168	\@glxtr@org@GLS@	55
\@glxtr@entrycount@org@localreset ..	91	\@glxtr@org@GLSpl@	55
\@glxtr@entrycount@org@localunset ..	91	\@glxtr@org@Gls@	55
\@glxtr@entrycount@org@reset	91	\@glxtr@org@Glspl@	55
\@glxtr@entrycount@org@unset	91	\@glxtr@org@Glsxtrtitlefirst ..	291, 293
\@glxtr@entryunitcount@org@localreset	99	\@glxtr@org@Glsxtrtitlefirstplural	291, 293
\@glxtr@entryunitcount@org@localunset	99	\@glxtr@org@Glsxtrtitlefull ..	292, 293
\@glxtr@entryunitcount@org@reset ..	99	\@glxtr@org@Glsxtrtitlefullpl ..	292, 293
\@glxtr@entryunitcount@org@unset ..	99	\@glxtr@org@Glsxtrtitlelong ..	292, 293
\@glxtr@err@undefaction	7, 13	\@glxtr@org@Glsxtrtitlelongpl ..	292, 293
\@glxtr@field@linkdefs	54	\@glxtr@org@Glsxtrtitlename ..	291, 292
\@glxtr@format@overridefalse	164	\@glxtr@org@Glsxtrtitleplural ..	291, 293
\@glxtr@format@overridefalse	164	\@glxtr@org@Glsxtrtitleshort ..	291, 292
\@glxtr@foundinlist	175	\@glxtr@org@Glsxtrtitleshortpl ..	291, 292
\@glxtr@full	182	\@glxtr@org@Glsxtrtitletext ..	291, 293
\@glxtr@fullpl	184	\@glxtr@org@MakeUppercase	291, 292
\@glxtr@gettype	108	\@glxtr@org@checkfirsthyper ...	73, 104
\@glxtr@glossdescfont	156–158	\@glxtr@org@delimN	52, 53
\@glxtr@glossnamefont	158–161, 163	\@glxtr@org@delimR	52, 53
\@glxtr@gobbleto@endescspch	168	\@glxtr@org@doseeglossary	23, 106
\@glxtr@groupheading	128, 130, 131	\@glxtr@org@gloautosee	24
\@glxtr@idx@displaynumberlist	107	\@glxtr@org@gls@	55
\@glxtr@idx@entrynumberlist	107	\@glxtr@org@glsdohypertarget	111
\@glxtr@ifcsstart	46	\@glxtr@org@glsignore	52, 53
\@glxtr@ifpunctoken	175	\@glxtr@org@glspl@	55
\@glxtr@ifunitcounter	96	\@glxtr@org@glsxtrtitlefirst ..	291, 293
\@glxtr@insert@dots	176	\@glxtr@org@glsxtrtitlefirstplural	291, 293
\@glxtr@insert@dots@next	177	\@glxtr@org@glsxtrtitlefull ..	292, 293
\@glxtr@insert@dots	178	\@glxtr@org@glsxtrtitlefullpl ..	292, 293
\@glxtr@label	44, 155, 156	\@glxtr@org@glsxtrtitlelong ..	291, 293
\@glxtr@loadstyles	310, 311	\@glxtr@org@glsxtrtitlelongpl ..	291, 293
\@glxtr@longnewglossaryentry	33	\@glxtr@org@glsxtrtitlename ..	291, 292
\@glxtr@mark@wordseps	177	\@glxtr@org@glsxtrtitleorpdforheading	291, 292
\@glxtr@mark@wordseps@next	177	\@glxtr@org@glsxtrtitleplural ..	291, 293
\@glxtr@markwordseps	178, 179	\@glxtr@org@glsxtrtitleshort ..	291, 292
\@glxtr@mixed@assign@sortkey	108	\@glxtr@org@glsxtrtitleshortpl ..	291, 292
\@glxtr@noidx@displaynumberlist ..	107	\@glxtr@org@glsxtrtitletext ..	291, 293
\@glxtr@noidx@do	130	\@glxtr@org@makeglossaries	105
\@glxtr@noidx@entrynumberlist	107	\@glxtr@org@markboth	290, 291
\@glxtr@noidx@numberlistloop	107	\@glxtr@org@markright	290
\@glxtr@noop@recordcounter	10, 13	\@glxtr@org@newacronymstyle ..	103, 104
\@glxtr@nopostpunc	109	\@glxtr@org@postdescription ..	110, 171
\@glxtr@nopostpunc@postdesc	110		

\@glxstr@org@see@noindex	122	\@glxstr@thisloctag	52, 53
\@glxstr@org@setacronymstyle ..	103, 104	\@glxstr@titlelabel	113, 114, 130
\@glxstr@org@theHvalue	8, 9	\@glxstr@tmp	21, 116
\@glxstr@org@prefix	10	\@glxstr@type	156
\@glxstr@org@printglossary	49, 110	\@glxstr@unitcountlist	96
\@glxstr@org@warndep	176	\@glxstr@unsrt@getgrouptitle	128
\@glxstr@p@acrlong@	81	\@glxstr@usesee	39
\@glxstr@p@acrlongpl@	81	\@glxstr@warn@onexistsordo	7, 13
\@glxstr@p@acrshort@	80	\@glxstr@warn@undefaction	7, 13
\@glxstr@p@acrshortpl@	81	\@glxstr@docdeffalse	45
\@glxstr@p@long@	80	\@glxstr@entryfmt	28
\@glxstr@p@longpl@	80	\@glxstr@fmt	27
\@glxstr@p@plural@	80	\@glxstr@glossentry	125
\@glxstr@p@short@	80	\@glxstr@glossentryother	126
\@glxstr@p@shortpl@	80	\@glxstr@hypernameprefix	111, 130
\@glxstr@p@text@	80	\@glxstr@trifhasfield	29
\@glxstr@p@pagetag	52	\@glxstr@trifhyphenstart	273
\@glxstr@p@pagetag	52	\@glxstr@indexaliased	75
\@glxstr@p@prevunitcount	98	\@glxstr@indexcrossrefsfalse	14
\@glxstr@p@printglossopts	49, 108, 110	\@glxstr@indexcrossrefstrue	15
\@glxstr@p@printunsrtglossaryskipentry	128, 129	\@glxstr@inmark	290
\@glxstr@p@provide@addstoragekey	27	\@glxstr@long	80, 187
\@glxstr@p@provide@storagekey	26	\@glxstr@longpl	80, 190
\@glxstr@record	13, 54–56, 59	\@glxstr@newgls	133, 134
\@glxstr@record@setting	8, 9, 12, 40, 44, 45, 105	\@glxstr@newgls@inner	132, 133
\@glxstr@record@setting@alsoindex ..	8, 9, 40, 105	\@glxstr@newgls@innercsname	133
\@glxstr@record@setting@off	44	\@glxstr@notinmark	290
\@glxstr@record@setting@only	105	\@glxstr@p	86
\@glxstr@record@see	13, 23, 40	\@glxstr@p@opt	84
\@glxstr@redef@forglsentries	7, 24	\@glxstr@p	48, 49
\@glxstr@redef@styles	21, 309	\@glxstr@postloctag	51–53
\@glxstr@reg@glosslist	105–108, 111	\@glxstr@preloctag	51–53
\@glxstr@restore@postpunc	110	\@glxstr@setaliasnoindex	74–76
\@glxstr@rglstrigger@record ...	137–139	\@glxstr@short	80, 185
\@glxstr@s@longnewglossaryentry	33	\@glxstr@shortpl	80, 188
\@glxstr@s@savepreloctag	52, 53	\@glxstr@undeftag	6, 25
\@glxstr@setentrycountunsetattr	90	\@glxstr@wrglossmark	22
\@glxstr@setentryunitcountunsetattr ..	101	\@gobble ..	7, 13, 15, 60, 129, 130, 177, 347–349
\@glxstr@setupshortcuts	19, 20, 24	\@gobbletwo	176
\@glxstr@shortcutsval	19, 124	\@ifnextchar	77
\@glxstr@swaptwo	175	\@ifpackageloaded	5, 16, 124, 140, 156, 158, 160, 162, 164, 309
\@glxstr@tag	170	\@ifstar ...	26, 27, 29, 33–35, 46, 77, 127, 169
\@glxstr@taggingcs	170	\@ifundefined	309
\@glxstr@textformat	58, 59	\@ignored@glossaries	34–36
\@glxstr@theHvalue	8, 9, 57–59, 136	\@input	122
\@glxstr@thevalue	8, 9, 57–59, 136	\@input@	117
		\@istfilename	105
		\@makeglossary	105

\aclp	18		
\ACP	18		
\Acp	18		
\acp	18		
\ACRfullfmt	103		
\Acrfullfmt	103		
\acrfullfmt	103		
\ACRfullplfmt	103		
\Acrfullplfmt	103		
\acrfullplfmt	103		
\acronymentry	102		
\acronymfont	68–71, 83, 103, 104		
\acronymname	16		
\acronymsort	102		
\acronymstype	17, 102, 103		
\acrpluralsuffix	102, 124		
\ACS	18		
\Acs	18		
\acs	18		
\ACSP	18		
\Acsp	18		
\acsp	18		
\actualchar	168		
\addtolength	337		
\advance	92, 101, 123, 133		
\AF	17		
\Af	17		
\af	17		
\AFP	17		
\Afp	17		
\afp	17		
\AL	17		
\Al	17		
\al	17		
\ALP	17		
\Alp	17		
\alp	17		
\AnyTrackedLanguages	309		
\appto	11, 12, 21, 26, 38–43, 72, 76, 90, 98, 128, 130, 164, 174, 177, 310		
\arabic	349		
\AS	17		
\As	17		
\as	17		
\ASP	17		
\Asp	17		
\asp	17		
\AtBeginDocument	22, 25, 50, 51, 124		
\AtEndDocument	43, 91, 100, 118		
		B	
		babel package	164, 166, 174
		\begin	15, 16, 115, 119, 120, 128, 312, 314–321, 340–343, 347
		\beginngroup	8, 9, 27, 28, 75, 125–127
		\bgroup	33, 109
		bib2gls	3, 28, 130, 132, 136
		C	
		\catcode	122
		category attributes:	
		apospplural	179
		discardperiod	173
		entrycount	89, 90, 92, 101
		firstuc	160
		glossdesc	156
		glossdescfont	156
		glossname	158
		glossnamefont	158, 160
		headuc	293
		indexname	165
		indexonlyfirst	76
		insertdots	178
		markshortwords	178
		markwords	178, 179, 273, 274, 282
		nohyper	74
		nohyperfirst	61–63
		noshortplural	179
		regular	54, 95, 198, 200, 202–204, 207–210, 212, 213, 216, 217, 219, 220, 222, 224– 226, 230, 232–234, 237–239, 241, 244– 250, 252, 254, 256, 258–260, 262, 265, 271, 272, 274–276, 278, 282, 283, 287, 289
		textformat	58
		\cdot	22
		\centering	346
		\cGLS	17, 18, 90, 101
		\cGls	17, 18, 90, 101
		\cgls	17, 18, 90, 101
		\cGLSformat	94
		\cGlsformat	93
		\cglsformat	93, 95
		\cGLSpl	17, 18, 90, 101
		\cGlspl	17, 18, 90, 101
		\cglspl	17, 18, 90, 101
		\cGLSplformat	94
		\cGlsplformat	93
		\cglsplformat	93, 95
		\changes	16, 295

entry categories:		
abbreviation	192	
general	150, 152	
index	154	
\epreto	165	
\equal	121	
etoolbox package	5	
\expandafter	22, 26, 27, 39, 40, 43, 44, 46–48, 72, 73, 77, 84, 95, 96, 106–108, 111, 114, 116, 128, 130, 133, 139, 156, 159, 160, 162, 163, 165, 168, 175, 178, 179, 273	
\expandonce	102, 131, 165, 166, 199, 276	
F		
\fi	7–11, 13–15, 17, 19, 20, 22–24, 28, 32, 38, 40, 42, 43, 45–47, 50, 51, 53, 56–59, 75, 76, 92, 100, 101, 104, 107–111, 113, 116–124, 135, 136, 164–166, 168, 175, 177, 179, 185–191, 194, 199, 201, 203–212, 215, 217–229, 231–245, 247–263, 265–267, 269, 272, 273, 276, 279–282, 284, 286, 288, 312, 314–326, 328–338, 344, 346, 348	
first use	351	
flag	351	
text	351	
\firstacronymfont	103, 104	
fontspec package	124	
\footnote	202	
\forall glossaries	43, 127, 154, 156, 329, 330, 332–336	
\forall gloss entries	92, 101	
\ForEachTrackedDialect	309	
\forall gloss entries	6, 43, 154, 156, 329–336	
\forall listcsloop	29, 101, 115	
\forall listloop	112, 170	
\futurelet	174	
G		
\gdef	52, 166, 167	
\Genacrfullformat	103	
\genacrfullformat	103	
\GenericAcronymFields	103	
\Genplacrfullformat	103	
\genplacrfullformat	103	
\glo@grabfirst	130	
\glo@name	159, 160, 163	
\gloaliaslabel	79	
\global	10, 33, 109, 131	
\glolinkprefix	58, 59, 79, 124	
glossaries package	13, 23, 24, 38, 40–42, 108, 311	
glossaries-accsupp package	20, 22, 140	
glossaries-extra package	2	
glossaries-extra-stylemods package	20, 171, 309	
glossaries-stylemods package	345	
glossaries.sty package	33	
\GlossariesExtraWarning	6, 15, 32, 47, 49, 58, 104, 106, 116, 120, 122, 127, 156–159, 161, 163, 170, 197	
\GlossariesExtraWarningNoLine	15, 92, 101	
\GlossariesWarning	52, 107, 109, 112, 113, 195	
\GlossariesWarningNoLine	106, 118	
glossary styles:		
altlist	312	
altlistgroup	313	
altlisthypergroup	314	
almtree	326, 327, 337	
almtreegroup	338	
almtreehypergroup	338	
index	322	
indexgroup	323	
indexhypergroup	323	
inline	322	
list	312	
listdotted	311	
listdottedstyle	312	
listgroup	313	
listhypergroup	313	
mcolalmtree	342	
mcolalmtreegroup	342	
mcolalmtreehypergroup	343	
mcolalmtreespannav	343	
mcolindexgroup	339	
mcolindexhypergroup	339	
mcolindexspannav	339	
mcoltreegroup	340	
mcoltreehypergroup	340	
mcoltreenonamegroup	341	
mcoltreenonamehypergroup	341	
mcoltreenonamespannav	342	
mcoltreespannav	340	
sublistdotted	312	
tree	324	
treegroup	324	
treehypergroup	325	
treenoname	325	
treenonamegroup	326	
treenonamehypergroup	326	
glossary-bookindex package	311	

glossary-hypernav package	78	253, 255, 257, 258, 261, 263, 265, 266,
glossary-long package	316	269–272, 274, 276, 278–280, 282, 285, 287
glossary-longbooktabs package	316	\glsabbrvhyphenfont 274, 275, 279–286
\glossaryentrynumbers	53, 109, 131, 132	\glsabbrvonlyfont 287, 289
\glossaryheader	115,	\glsabbrvscfont 214–219, 221–223, 225–227
128, 312–321, 323–326, 337–341, 343, 347		\glsabbrvsmfont 228–237, 239–241
\glossaryname	108	\glsabbrvuserfont 264–271
\glossarypostamble	115, 128, 130	\GLSaccessdesc 64
\glossarypreamble	115, 127	\Glsaccessdesc 64, 157, 169
\glossarysection	115, 121, 127, 130	\glsaccessdesc 63, 157, 173
\glossarytitle	35, 108, 109, 115, 121, 127	\GLSaccessdescplural 64
\glossarytoctitle	35, 108, 115, 121, 127	\Glsaccessdescplural 64
\glossentry	109, 131,	\glsaccessdescplural 64
132, 311, 312, 314–321, 323–325, 337, 347		\GLSaccessfirst 61
\glossentrydesc	311–321, 323–327	\Glsaccessfirst 61
\glossentryname		\glsaccessfirst 61
126, 311–321, 323–325, 337, 338, 345		\GLSaccessfirstplural 63
\glossentrynameother	127	\Glsaccessfirstplural 63
\glossentrysymbol	315–319, 321, 323–325, 327	\glsaccessfirstplural 62
\glossxtrsetpopts	171	\Glsaccesslong 70,
\GLS	90, 101, 135	180, 188, 199, 207, 211, 212, 215, 221–
\Gls	48, 90, 101, 135	224, 229, 235, 236, 238, 243, 245, 253–
\gls	32, 47, 49, 90, 101, 107, 119, 135	257, 259, 265, 267, 275, 277, 280, 283, 288
\gls@assign@desc	33	\glsaccesslong
\gls@assign@field	11, 12, 26, 72	70, 180, 187, 188, 199, 201, 203–206,
\gls@checkseeallowed	46, 106	208–212, 215, 217, 218, 220–226, 228,
\gls@codepage	118	229, 231–240, 242, 243, 245, 247–263,
\gls@defdocnewglossaryentry	90, 98	265–267, 269, 272, 275–277, 280, 283, 288
\gls@defglossaryentry	33, 47, 48	\Glsaccesslongpl
\gls@dotocitle	109	71, 181, 191, 199, 207, 211, 212, 215,
\gls@glossary	41	221–224, 229, 235–238, 243, 245, 253–
\gls@grplabel	78	259, 266, 267, 275, 277, 280, 281, 283, 288
\gls@level	131	\glsaccesslongpl 71, 180,
\gls@noidxglossary	106	190, 191, 199, 201, 203, 204, 206, 207,
\gls@org@glossaryentryfield	109	209–212, 215, 217–226, 228, 229, 231,
\gls@org@glossarysubentryfield	109	233–240, 242, 243, 245, 247, 249–263,
\gls@save@numberlist	51, 53	265, 267, 269, 272, 275, 277, 280, 283, 288
\gls@set@xr@key	41	\GLSaccessname 63
\gls@tmplen	329–338	\Glsaccessname 63
\gls@type	106	\glsaccessname 39, 63
\glsabbrvdefaultfont	181, 198, 200,	\GLSaccessplural 62
203, 205, 206, 208, 211, 264, 274, 276, 287		\Glsaccessplural 62
\glsabbrvmfont		\glsaccessplural 62
242–251, 253, 255, 257, 258, 260–263		\Glsaccessshort 68, 186, 194, 201,
\glsabbrvfont	81, 82, 103, 181, 185,	203–206, 209, 210, 217, 218, 220, 225–
186, 189, 190, 192, 194, 198–206, 208,		228, 231, 233, 234, 240–242, 247, 249,
211, 213, 215, 216, 218, 219, 222, 223,		250, 252, 261, 263, 269, 272, 280, 285, 286
225, 227, 229, 231, 232, 234, 236, 237,		\glsaccessshort
239, 241, 243, 245, 247, 248, 250, 251,		68, 180, 185, 186, 194, 199, 201,

203, 205–210, 212, 215, 217–222, 224–227, 229, 231–236, 238–243, 245, 247, 248, 250–257, 259, 261, 263, 265–267, 269, 272, 275, 277, 280, 283, 285, 286, 288	
\Glsaccesssshortpl	69, 189, 194, 201, 203–206, 209, 210, 217, 218, 220, 226–228, 231, 233, 234, 240–242, 247, 249, 250, 252, 261–263, 269, 272, 280, 285, 286, 288
\glsaccesssshortpl	69, 180, 181, 189, 190, 194, 199, 201, 203–205, 207–210, 212, 215, 217–222, 224–229, 231–245, 247, 249–254, 256–259, 261, 263, 265–267, 269, 272, 275, 277, 280, 283, 285, 286, 288
\GLSaccessssymbol	65
\Glsaccessssymbol	65, 169
\glsaccessssymbol	64, 169, 173
\GLSaccessssymbolplural	65
\Glsaccessssymbolplural	65
\glsaccessssymbolplural	65
\GLSaccesstext	60
\Glsaccesstext	61
\glsaccesstext	39, 60
\glsacrshortcutsrue	19, 20
\glsacspacemax	104
\glsadd	28, 44, 119
\glsadd options	
theHvalue	9
thevalue	9
\glsaddstoragekey	42, 43, 150
\glsbackslash	46
\glsapscase	56, 60–71, 73, 182–193
\glsacategory	54, 60, 74, 81, 82, 151–153, 156–163, 169, 172, 182–186, 188–190
\glsacategorylabel	74, 176, 178, 179, 204, 226, 241, 262, 266, 268, 270, 279, 281, 284, 286
\glsclsebrace	40, 120, 121
\glscurrententrylabel	51–53, 109, 117, 125, 126, 128, 129, 171, 172
\glscurrentfieldvalue	27, 28, 30, 31, 264
\glsclustomtext	54, 56, 68–71, 182–192, 194
\glsclefaulttype	6, 16, 32, 108, 119, 127, 129
\glsclescriptionaccessdisplay	144, 157
\glsclescriptionpluralaccessdisplay	144, 145
\glsclescriptionwidth	314–321
\glscltoklabel	8, 9, 28–33, 37–39, 44–46, 57, 59, 75, 79, 91, 96–101, 106, 109, 112, 113, 125, 126, 130, 131, 134–136, 156, 159, 160, 163, 331, 332
\glsclisplaynumberlist	107, 111
\glsclohyperlink	77, 78, 80
\glsclohypertarget	80, 111
\glsclolfeexists	14, 23, 30, 36, 39, 40, 54, 56, 59, 67–71, 79, 106, 112, 113, 125, 126, 182–191
\glsclolfeexistsordo	27, 28, 56
\glsclolfeexistsorwarn	14, 156, 157, 168, 169
\glsclolfeexists	33
\glsclonohyperlink	59, 79, 80
\glsclosanitizesort	107
\glsclenableentrycount	90, 92, 100
\glsclenableentryunitcount	91, 101
\glsclentrycounter	117
\glsclentrycurrcount	91, 92, 98
\Glsclentrydesc	144, 149, 157
\glsclentrydesc	144, 149, 158
\Glsclentrydescplural	145, 149
\glsclentrydescplural	144, 145, 149
\Glsclentryfirst	95, 139, 142, 148
\glsclentryfirst	95, 139, 142, 148, 305
\Glsclentryfirstplural	96, 139, 142, 148
\glsclentryfirstplural	95, 139, 142, 143, 148, 306
\glsclentryfmt	34–36
\Glsclentryfull	103
\glsclentryfull	103
\Glsclentryfullpl	103
\glsclentryfullpl	103
\glsclentryitem	126, 127, 311, 312, 314–321, 323–325, 337, 348
\Glsclentrylong	82, 83, 95, 139, 146, 150
\glsclentrylong	82, 83, 95, 139, 146, 150, 204, 227, 241, 262, 268, 270, 284, 306, 307
\Glsclentrylongpl	82, 83, 96, 147, 150
\glsclentrylongpl	82, 83, 95, 147, 150, 307
\Glsclentrylongplural	139
\glsclentrylongplural	139
\Glsclentryname	140, 147, 158, 160, 161
\glsclentryname	125, 126, 140, 147, 165, 303, 304, 329–336, 350
\glsclentrynumberlist	107, 113, 334–336
\Glsclentryplural	141, 148
\glsclentryplural	141, 148, 305
\glsclentryprevcount	91, 92, 98
\glsclentryprevmaxcount	99
\glsclentryprevtotalcount	98

<code>\Glsentryshort</code>	81, 83, 145, 149	<code>\glsfirstlongemfont</code>	
<code>\glsentryshort</code>	81–	244–246, 248, 249, 255, 256, 258–260
	83, 104, 145, 149, 266, 268, 279, 302, 303	<code>\glsfirstlongfont</code> ...	180, 181, 198–201,
<code>\Glsentryshortpl</code>	82, 83, 146, 150	203, 205–213, 215, 216, 218, 219, 222,	
<code>\glsentryshortpl</code> ..	82, 83, 146, 149, 150, 303	223, 225, 227, 229, 231, 232, 234, 236,	
<code>\Glsentrysymbol</code>	143, 148	237, 239, 241, 243, 245, 247, 248, 250,	
<code>\glsentrysymbol</code>	143, 148, 333–335	251, 253, 255, 257, 258, 261, 263, 265,	
<code>\Glsentrysymbolplural</code>	144, 149	266, 269, 271, 274, 276, 280, 282, 285, 288	
<code>\glsentrysymbolplural</code>	143, 144, 149	<code>\glsfirstlongfootnotefont</code>	
<code>\Glsentrytext</code>	141, 147	202–206, 225–228, 239–242, 260–263
<code>\glsentrytext</code>	79, 140, 141, 147, 148, 304	<code>\glsfirstlonghyphenfont</code>	273–276, 278–285
<code>\glsentrytype</code>	125, 126	<code>\glsfirstlongonlyfont</code>	287–289
<code>\Glsentryuseri</code>	66	<code>\glsfirstlonguserfont</code>	265–272
<code>\glsentryuseri</code>	66	<code>\GLSfirstplural</code>	298
<code>\Glsentryuserii</code>	66	<code>\Glsfirstplural</code>	298, 299
<code>\glsentryuserii</code>	66	<code>\glsfirstplural</code>	298
<code>\Glsentryuseriii</code>	66	<code>\glsfirstpluralaccessdisplay</code> ..	142, 143
<code>\glsentryuseriii</code>	66	<code>\glsforeachincategory</code>	195
<code>\Glsentryuseriv</code>	66	<code>\glsgenentryfmt</code>	54
<code>\glsentryuseriv</code>	67	<code>\glsgetattribute</code>	58, 78,
<code>\Glsentryuserv</code>	67	92, 96–98, 117, 135, 156–159, 161, 163, 165	
<code>\glsentryuserv</code>	67	<code>\glsgetcategoryattribute</code>	151
<code>\Glsentryuservi</code>	67	<code>\glsgetgrouptitle</code>	313, 314, 323–326, 338–344
<code>\glsentryuservi</code>	67	<code>\glsgetwidestname</code>	327
<code>\glsfieldfetch</code>	79	<code>\glsgroupheading</code>	
<code>\glsfieldxdef</code>	155, 156	131, 312–321, 323–326, 337–343, 349
<code>\glsfindwidesttoplevelname</code>	329	<code>\glsgroupskip</code>	131, 312, 314–324, 326, 338, 349
<code>\GLSfirst</code>	297, 298	<code>\glshasattribute</code>	
<code>\Glsfirst</code>	298	58, 78, 92, 97, 99, 101, 117, 135,
<code>\glsfirst</code>	297	156–161, 163, 164, 198, 200, 202, 203,	
<code>\glsfirstabbrvdefaultfont</code>		205, 208–210, 213, 214, 216, 217, 225,	
181, 198, 201, 203, 205, 206, 208, 211, 276		227, 229, 230, 232, 239, 241, 243–249,	
<code>\glsfirstabbrvfont</code>	243–263	256, 260, 262, 265, 266, 268, 270–272,	
<code>\glsfirstabbrvfont</code>	103, 180, 181,	274–276, 278, 279, 281–283, 285–287, 289	
198–212, 215, 216, 218, 219, 222, 223,		<code>\glshascategoryattribute</code>	152
225, 227, 229, 231, 232, 234, 236, 237,		<code>\glshyperlink</code>	79
239, 241, 243, 245, 247, 248, 250, 251,		<code>\glshypernavsep</code>	114
253, 255, 257, 258, 261, 263, 265, 266,		<code>\glshypernumber</code>	117, 164
269, 271, 274, 276, 277, 280, 282, 285, 288		<code>\glsifattribute</code> ...	56, 57, 61, 74, 76, 85,
<code>\glsfirstabbrvhyphenfont</code>		154, 157–160, 163, 171, 173, 174, 293–302	
.....	273–275, 279, 280, 282–286	<code>\glsifcategory</code>	154
<code>\glsfirstabbrvonlyfont</code>	288	<code>\glsifcategoryattribute</code>	
<code>\glsfirstabbrvscfont</code>	214–228	74, 152, 153, 178, 179
<code>\glsfirstabbrvsmfont</code>	229–242	<code>\glsifnotregular</code>	60
<code>\glsfirstabbrvuserfont</code>	265–272	<code>\glsifnotregularcategory</code>	153
<code>\glsfirstaccessdisplay</code>	142	<code>\glsifplural</code>	
<code>\glsfirstlongdefaultfont</code>	56, 60, 62–65, 67–71, 173, 174, 182–193
198, 201, 206, 208, 211, 214–224, 229–		<code>\glsifregular</code>	54, 60, 95, 96, 139
238, 243, 244, 246, 247, 250–254, 257, 258		<code>\glsifregularcategory</code>	153

<code>\glsifusetranslator</code>	35	253, 255, 257, 258, 261, 263, 265, 266,
<code>\glsignore</code>	52, 53	269, 271, 274, 276, 280, 282, 285, 288, 289
<code>\glsinlinedescformat</code>	322	
<code>\glsinlinesubdescformat</code>	322	<code>\glslongfootnotefont</code>
<code>\glsinsert</code>	56,	202, 203, 205, 225, 227, 239, 241, 261, 263
	60, 68–71, 182–194, 273, 279, 281, 284–286	<code>\glslonghyphenfont</code>
<code>\glskeylisttok</code>	102, 103, 178, 180 274, 276, 278–280, 282, 284, 285
<code>\glslabel</code>	8,	<code>\glslongonlyfont</code>
	9, 27, 37, 39, 54, 56–59, 74, 75, 78, 79,	287, 288
	104, 136, 172, 173, 192–194, 204, 227,	<code>\glslongpltok</code>
	241, 262, 266, 268, 270, 279, 281, 284–286	. 179, 180, 198–202, 211, 213–217, 221,
<code>\glslabeltok</code>		225, 229–231, 236, 239, 243, 244, 246–
	102, 178, 179, 198–203, 205, 206, 208–	249, 253, 255, 258, 260, 265–268, 270–
	211, 213–219, 221, 225, 227, 229–232,	272, 274–276, 278, 279, 281–283, 287, 289
	234, 236, 239, 241, 243–251, 253, 255,	<code>\glslongpluralaccessdisplay</code>
	256, 258, 260, 262, 265–268, 270–272,	147
	274–276, 278, 279, 281–283, 285–287, 289	<code>\glslongtok</code> .. 102, 178, 180, 198–202, 204,
<code>\glsletentryfield</code>	165	206, 208, 211, 213–219, 221, 225, 226,
<code>\glslink</code>	103	229–232, 234–236, 239, 240, 243–251,
<code>\glslink options</code>		253, 255, 258, 260, 262, 265–268, 270–
counter	9	272, 274–276, 278, 279, 281–284, 287, 289
format	164	<code>\glslonguserfont</code>
hyper	289	264–271
hyperoutside	57	<code>\glsmcols</code>
noindex	8, 74, 289	340–343
theHvalue	58	<code>\GLSname</code>
thevalue	58, 132	295
wrgloss	8, 56, 57	<code>\Glsname</code>
<code>\glslinkcheckfirsthyperhook</code>	74	295
<code>\glslinkpostsetkeys</code>	58, 136	<code>\glsnameaccessdisplay</code> .. 140, 158, 159, 161
<code>\glslinkvar</code>	77	<code>\glsnamefont</code>
<code>\glslistchildpostlocation</code>	312	158, 160, 161, 163
<code>\glslistchildprelocation</code>	312, 313	<code>\glsnavhyperlink</code>
<code>\glslistdottedwidth</code>	311	114
<code>\glslistgroupheaderfmt</code>	313, 314	<code>\glsnavhyperlinkname</code>
<code>\glslistnavigationitem</code>	313, 314	78
<code>\glslistprelocation</code>	312, 313	<code>\glsnavhypertarget</code>
<code>\glslocalunset</code>	56, 136 313, 314, 324–326, 339–344
<code>\glslongaccessdisplay</code>	146	<code>\glsnavigation</code> .. 313, 314, 324–326, 338–343
<code>\glslongdefaultfont</code> .. 182, 198, 201, 202,		<code>\glsnextpages</code>
	206, 208, 211, 215, 216, 218, 219, 221–	109
	223, 229, 231, 232, 234, 236–238, 243,	<code>\glsnoidxdisplayloc</code>
	247, 250, 251, 253, 254, 257, 264, 274, 287	112, 113
<code>\glslongemfont</code> .. 242, 245, 248, 255, 258, 259		<code>\glsnoidxdisplaylocclsthandler</code> 112
<code>\glslongfont</code> .. 82, 83, 182, 187, 188, 190–		<code>\glsnoidxloclist</code>
	192, 198, 199, 201, 203, 205, 206, 208,	113, 131, 132
	211, 213, 215, 216, 218, 219, 222, 223,	<code>\glsnoidxnumberlistloophandler</code> 112
	225, 227, 229, 231, 232, 234, 236, 237,	<code>\glsnonextpages</code>
	239, 241, 243, 245, 247, 248, 250, 251,	109
		<code>\glsnonumberlistfalse</code>
		51
		<code>\glsnonumberlisttrue</code>
		51
		<code>\glsnopostdotfalse</code>
		110
		<code>\glsnopostdottrue</code>
		110
		<code>\glsnumberlistloop</code>
		107
		<code>\glsnumlistlastsep</code>
		112
		<code>\glsnumlistsep</code>
		112
		<code>\glsopenbrace</code>
		40, 120, 121
		<code>\glsorder</code>
		105
		<code>\glspagelistwidth</code>
		315, 317, 319, 321
		<code>\glspar</code>
		130
		<code>\GLSpl</code>
		90, 101, 135
		<code>\Glspl</code>
		48, 90, 101, 135

<code>\glsp</code>	48, 90, 101, 135	<code>\glstextformat</code>	56, 58
<code>\GLSplural</code>	296, 297	<code>\glstextup</code>	214
<code>\Glsplural</code>	297	<code>\glstreechildpredesc</code>	323, 324
<code>\glsplural</code>	296, 297	<code>\glstreechildprelocation</code>	323, 324, 326
<code>\glspluralaccessdisplay</code>	141	<code>\glstreegroupheaderfmt</code>	323–326, 338–344, 346
<code>\glspluralsuffix</code>	124, 178, 182	<code>\glstreeindent</code>	324, 325, 336–338
<code>\glspostdescription</code>	15, 16, 110, 171, 311–321, 323–327	<code>\glstreeitem</code>	322, 340, 347, 348
<code>\glspostinline</code>	322	<code>\glstreenamebox</code>	337, 338
<code>\glspostlinkhook</code>	55, 56, 68–71, 84, 182–191	<code>\glstreenamefmt</code>	323–325, 327, 329–338
<code>\glsprestandardsort</code>	107	<code>\glstreenavigationfmt</code>	324–326, 338–343
<code>\glsresetentrylist</code>	115, 128	<code>\glstreepredesc</code>	323–325
<code>\glsee</code>	41–43	<code>\glstreeprelocation</code>	322–325, 327
<code>\glseeformat</code>	39, 40, 45, 106, 112, 113	<code>\glstreesubitem</code>	322, 348
<code>\glseelist</code>	40	<code>\glstreesubsubitem</code>	323, 348
<code>\glsssetabbrvfmt</code>	54, 60, 81, 82, 156–161, 163, 169, 182–186, 188–190, 192	<code>\GlsrLetField</code>	31
<code>\glsssetAttribute</code>	198, 200, 202, 203, 205, 206, 208, 209, 211, 213, 214, 216–219, 221, 225, 227, 229–232, 234, 236, 239, 241, 243–246, 248–251, 253, 255, 256, 258, 260, 262, 265, 266, 268, 270–272, 274–276, 278, 279, 281–283, 285–287, 289	<code>\glstype</code>	56, 57, 68–71, 136, 182–191
<code>\glsssetcategoryattribute</code>	90, 102, 104, 135, 151, 152, 154, 155, 169	<code>\glset</code>	44, 56, 93, 94, 136
<code>\glsssetnoexpandfield</code>	11, 12	<code>\glswrite</code>	40, 105
<code>\glsssettoctitle</code>	109	<code>\glswriteentry</code>	8, 9
<code>\glssshortaccessdisplay</code>	145	<code>\Glsxtr</code>	49
<code>\glssshortpltok</code>	179, 180, 198, 200–204, 206, 208, 214–219, 225, 226, 229–233, 239, 240, 243–251, 260, 262, 265, 266, 268, 270–272, 274, 275, 279, 281–284, 286, 287, 289	<code>\glxtr</code>	49
<code>\glssshortpluralaccessdisplay</code>	146	<code>\glxtr@@do@wrglossary</code>	8, 9, 11, 13
<code>\glssshorttok</code>	102, 178–180, 198–202, 204, 206, 208, 213–219, 221, 225, 226, 229–233, 235, 239, 240, 243, 244, 246–251, 253, 255, 260, 262, 265–268, 270–272, 274, 275, 278, 279, 281–284, 286, 287, 289	<code>\glxtr@addloclistfield</code>	13
<code>\glsssubentryitem</code>	126, 127, 311–321, 323–325, 337, 348	<code>\glxtr@addunused</code>	44
<code>\glssymbolaccessdisplay</code>	143	<code>\glxtr@applyabbrvfmt</code>	192
<code>\glssymbolpluralaccessdisplay</code>	143, 144	<code>\glxtr@applyabbrvstyle</code>	176, 178, 195
<code>\glstarget</code>	126, 127, 311–321, 323–325, 337, 338, 348	<code>\glxtr@counterrecord</code>	125
<code>\GLStext</code>	296	<code>\glxtr@do@alsoindex@wrglossary</code>	13
<code>\GlsText</code>	296	<code>\glxtr@doption</code>	5, 15, 16, 22, 24
<code>\glstext</code>	296	<code>\glxtr@fields</code>	123
<code>\glstextaccessdisplay</code>	140, 141	<code>\glxtr@headentry@p</code>	85, 86
		<code>\glxtr@hyperoutsidefalse</code>	57
		<code>\glxtr@hyperoutsidettrue</code>	57
		<code>\glxtr@ifnextpunc</code>	174
		<code>\glxtr@ifpunctoken</code>	175
		<code>\glxtr@indexonly@saveentrycounter</code>	13, 25
		<code>\glxtr@keylist</code>	47, 48
		<code>\glxtr@label</code>	350
		<code>\glxtr@langtag</code>	124
		<code>\glxtr@linkprefix</code>	123, 124
		<code>\glxtr@makeglossaries</code>	105
		<code>\glxtr@newabbreviation</code>	103, 177
		<code>\glxtr@next</code>	175
		<code>\glxtr@org@@do@wrglossary</code>	25
		<code>\glxtr@org@dohyperlink</code>	78
		<code>\glxtr@org@getgrouptitle</code>	114
		<code>\glxtr@org@newignoredglossary</code>	34

<code>\glxtr@orgmakenoidxglossaries</code>	44	<code>\glxtrbookindexgroupskip</code>	349
<code>\glxtr@pluralsuffixes</code>	123, 124	<code>\glxtrbookindexlastmarkfmt</code>	350
<code>\glxtr@process</code>	128, 129	<code>\glxtrbookindexname</code>	345, 348
<code>\glxtr@provideignoredglossary</code>	35	<code>\glxtrbookindexparentchildsep</code>	345, 347
<code>\glxtr@punctlist</code>	174, 175	<code>\glxtrbookindexparentschildsep</code>	347, 348
<code>\glxtr@record</code>	10, 123	<code>\glxtrbookindexprelocation</code> ...	345, 348
<code>\glxtr@recordsee</code>	11	<code>\glxtrbookindexsubbetween</code>	348
<code>\glxtr@resource</code>	122, 123	<code>\glxtrbookindexsubname</code>	348
<code>\glxtr@s@newignoredglossary</code>	34	<code>\glxtrbookindexsubprelocation</code> ...	348
<code>\glxtr@s@provideignoredglossary</code> ...	35	<code>\glxtrbookindexsubsubbetween</code>	348
<code>\glxtr@saveentrycounter</code>	8, 9, 11, 75	<code>\glxtrbookindexthepage</code>	349, 350
<code>\glxtr@setaccessdisplay</code>	163	<code>\glxtrcat</code>	47, 48
<code>\glxtr@setbookindexmark</code>	349	<code>\glxtrchecknohyperfirst</code>	61–63
<code>\glxtr@setup@record</code>	12, 13, 24, 25	<code>\glxtrComputeTreeIndent</code>	337
<code>\glxtr@shortcutsval</code>	123, 124	<code>\glxtrComputeTreeSubIndent</code>	337
<code>\glxtr@texencoding</code>	124	<code>\GlsXtrdefaultsubsequentfmt</code> ...	194, 196
<code>\glxtr@usesee</code>	39	<code>\glxtrdefaultsubsequentfmt</code> ...	194, 196
<code>\glxtr@warnonexistsordo</code>	7, 13, 37, 38	<code>\GlsXtrdefaultsubsequentplfmt</code> .	194, 196
<code>\glxtr@writefields</code>	122	<code>\glxtrdefaultsubsequentplfmt</code> .	194, 196
<code>\glxtrabbrvfootnote</code>	202–204, 225–227, 239–241, 260–262	<code>\GlsXtrDefineAbbreviationShortcuts</code>	19, 20
<code>\glxtrabbrvpluralsuffix</code>	124, 182, 198, 200, 203, 205, 206, 208, 211, 214, 228, 242, 265, 274, 276, 280, 285, 287	<code>\GlsXtrDefineAcShortcuts</code>	19, 20
<code>\glxtrabbrvtype</code>	16, 17, 180	<code>\GlsXtrDefineOtherShortcuts</code>	19, 20
<code>\glxtractivatenopost</code>	109	<code>\glxtrdetoklocation</code>	134, 135
<code>\glxtraddallcrossrefs</code>	43	<code>\glxtrdiscardperiod</code>	172
<code>\glxtralias</code>	75	<code>\glxtrdisplayendloc</code>	115
<code>\glxtrAltTreeIndent</code>	327	<code>\glxtrdisplayendlochook</code>	116
<code>\glxtralttreeInit</code>	337, 342, 343	<code>\glxtrdisplayingleloc</code>	116
<code>\glxtrAltTreePar</code>	327	<code>\glxtrdisplaystartloc</code>	115
<code>\glxtrAltTreeSetHangIndent</code> ...	327, 337	<code>\glxtrdoautoindexname</code>	76, 77, 162
<code>\glxtrAltTreeSetSubHangIndent</code> ...	338	<code>\glxtrdopostpunc</code>	204, 227, 241, 262
<code>\glxtralttreeSubSymbolDescLocation</code> ..	338	<code>\glxtrdowrglossaryhook</code>	76
<code>\glxtralttreeSymbolDescLocation</code> ..	327, 337	<code>\glxtremsuffix</code>	243, 245, 246, 248, 250, 251, 253, 255, 257, 258, 261, 263
<code>\glxtrassignfieldfont</code>	60–67	<code>\GlsXtrEnableEntryCounting</code>	101
<code>\glxtrautoindex</code>	165	<code>\GlsXtrEnableEntryUnitCounting</code>	90
<code>\glxtrautoindexassignsort</code>	165	<code>\GlsXtrEnableOnTheFly</code>	47, 49
<code>\glxtrautoindexentry</code>	165	<code>\glxtrfieldlistgadd</code>	125
<code>\glxtrbookindexatendgroup</code>	347	<code>\glxtrfieldtitlecase</code>	157–160, 163
<code>\glxtrbookindexatsubendgroup</code>	348	<code>\glxtrfieldtitlecasecs</code>	156
<code>\glxtrbookindexatsubsubendgroup</code> ..	348	<code>\glxtrfieldxifinlist</code>	129
<code>\glxtrbookindexbetween</code>	347	<code>\glxtrfirstscfont</code>	214
<code>\glxtrbookindexbookmark</code>	349	<code>\glxtrfirstsmfont</code>	228
<code>\glxtrbookindexcols</code>	347	<code>\GlsXtrFmtDefaultOptions</code>	27, 28
<code>\glxtrbookindexcolspread</code>	347	<code>\glxtrfmtdisplay</code>	27, 28
<code>\glxtrbookindexfirstmarkfmt</code>	350	<code>\GlsXtrFmtField</code>	27, 28
<code>\glxtrbookindexformatheader</code>	349	<code>\GlsXtrFormatLocationList</code>	51, 53, 334–336
		<code>\GLSxtrfull</code>	17, 18, 301

<code>\Glsxtrfull</code>	17, 18, 301, 302	<code>\glsxtrheadlongpl</code>	292
<code>\glsxtrfull</code>	17, 18, 301	<code>\Glsxtrheadname</code>	292
<code>\Glsxtrfullformat</code>	181, 194, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 249, 251, 252, 254, 256, 258, 260, 261, 263, 265, 267, 269, 272, 275, 277, 280, 283, 285, 288	<code>\glsxtrheadname</code>	125, 126, 292
<code>\glsxtrfullformat</code> 181, 193, 194, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 222, 224, 225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 248, 250, 252, 254, 256, 258, 259, 261, 263, 265, 266, 269, 272, 275, 277, 280, 283, 285, 288	<code>\Glsxtrheadplural</code>	292
<code>\Glsxtrfullpl</code>	17, 18, 301, 302	<code>\glsxtrheadplural</code>	292
<code>\Glsxtrfullpl</code>	17, 18, 302	<code>\Glsxtrheadshort</code>	292
<code>\glsxtrfullpl</code>	17, 18, 301	<code>\glsxtrheadshort</code>	292
<code>\Glsxtrfullplformat</code> 181, 193, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237, 238, 240, 241, 243, 245, 247, 249, 251, 252, 254, 256, 258, 260, 261, 263, 266, 267, 269, 272, 275, 277, 280, 283, 285, 288	<code>\Glsxtrheadshortpl</code>	292
<code>\glsxtrfullplformat</code>	193, 196, 199, 201, 203, 205, 207, 209, 212, 215, 217, 219, 220, 223–225, 227, 229, 231, 233, 234, 237–239, 241, 243, 245, 247, 248, 251, 252, 254, 256, 258, 259, 261, 263, 265, 267, 269, 272, 275, 277, 280, 283, 285, 288	<code>\glsxtrheadshortpl</code>	292
<code>\glsxtrfullsep</code> 180, 181, 198–201, 204–210, 212, 214–222, 224, 226–238, 240, 242–259, 261–264, 273–275, 277, 279, 282–284, 288	<code>\Glsxtrheadtext</code>	292
<code>\glsxtrgenabbrvfmt</code>	54	<code>\glsxtrheadtext</code>	292
<code>\glsxtrgetgroupitle</code>	114, 349	<code>\glsxtrhyperlink</code>	79, 117
<code>\glsxtrgroupfield</code>	131	<code>\glsxtrhyphensuffix</code>	274, 282
<code>\Glsxtrheadfirst</code>	292	<code>\glsxtrifcounttrigger</code>	93, 94
<code>\glsxtrheadfirst</code>	292	<code>\glsxtrifcustomdiscardperiod</code>	172
<code>\Glsxtrheadfirstplural</code>	292	<code>\glsxtrifemptyglossary</code>	115, 121, 127
<code>\glsxtrheadfirstplural</code>	292	<code>\glsxtrifhasfield</code>	31, 75, 345
<code>\Glsxtrheadfull</code>	292	<code>\glsxtrifhyphenstart</code> 273, 276, 278, 279, 282, 284
<code>\glsxtrheadfull</code>	292	<code>\glsxtrifindexing</code>	76
<code>\Glsxtrheadfullpl</code>	292	<code>\glsxtrifinmark</code>	59, 85–88, 291, 292
<code>\glsxtrheadfullpl</code>	292	<code>\glsxtrifnextpunc</code>	174, 175
<code>\Glsxtrheadlong</code>	292	<code>\glsxtrifperiod</code>	172–174
<code>\glsxtrheadlong</code>	292	<code>\glsxtrifrecordtrigger</code>	137–139
<code>\Glsxtrheadlongpl</code>	292	<code>\glsxtrifwasfirstuse</code> 60–63, 67–71, 74, 104, 173, 183, 185–191, 204, 205, 226, 227, 241, 262, 266, 268, 270, 279, 281, 284, 286
		<code>\glsxtrindexaliased</code>	75
		<code>\glsxtrindexseealso</code>	42, 43
		<code>\glsxtrinithyperoutside</code>	58
		<code>\glsxtrinitwrgloss</code>	57, 136
		<code>\glsxtrinitwrglossbeforefalse</code>	56, 57
		<code>\glsxtrinitwrglossbeforetrue</code>	56, 57
		<code>\Glsxtrinlinefullformat</code>	181, 183, 196, 204, 206, 207, 209, 210, 212, 218, 220– 222, 224, 226, 228, 233–236, 238, 240, 242, 250, 251, 253, 254, 256, 257, 259, 261, 263, 267, 269, 277, 280, 286, 288, 308
		<code>\glsxtrinlinefullformat</code>	181–183, 196, 203, 205, 206, 208, 210, 212, 218, 220– 222, 224, 226, 227, 232, 234–236, 238, 240, 242, 250–252, 254, 255, 257, 259, 261, 263, 267, 269, 277, 280, 285, 288, 307
		<code>\Glsxtrinlinefullplformat</code>	181, 184, 196, 204, 206, 207, 209, 210, 212, 218, 220–222, 224, 226, 228, 233–236, 238, 240, 242, 250, 252–254, 256, 258, 259, 262, 263, 267, 269, 277, 281, 286, 288, 308

\glxstrinlinefullplformat	\glxstrparen
..... 181, 184, 185, 196,	180, 181, 198–201, 204–210, 212, 214–
204, 205, 207, 208, 210, 212, 218, 220–	222, 224, 226, 228–238, 240, 242–259,
222, 224, 226, 228, 232, 234–236, 238,	261–264, 273–275, 277, 279, 282–284, 288
240, 242, 250–252, 254, 255, 257, 259,	\Glsxtrpl
261, 263, 267, 269, 277, 280, 286, 288, 308	49
\glxstrinsertinsidefalse	\glxstrpl
198	49
\glxstrlocationhyperlink	\glxstrpostdescription .
117	110, 154, 171, 322
\glxstrlocrangefmt	\glxstrposthyphenlong
116	284, 286
\GLSxtrlong	\glxstrposthyphenshort
17, 18, 299, 300	279, 281
\Glsxtrlong	\glxstrposthyphensubsequent
17, 18, 300
\glxstrlong	279, 281, 285, 286
17, 18, 299	\glxstrpostlink
\glxstrlonghyphen	172
280	\glxstrpostlinkendsentence
\glxstrlonghyphennoshort	172
276, 277	\glxstrpostlinkhook
\glxstrlonghyphenshort	172
275	\glxstrpostlocalreset
\GLSxtrlongpl	89, 91, 99
17, 18, 299, 300	\glxstrpostlocalunset
\Glsxtrlongpl	89, 91, 99
17, 18, 300	\glxstrpostlongdescription
\glxstrlongpl	33
17, 18, 299	\glxstrpostnamehook
\glxstrlongshortdescname	159–161, 164
..... 199, 215, 230, 244, 245, 270, 275, 281	\GlsXtrPostNewAbbreviation
\glxstrlongshortdescsort
..... 199, 215, 230, 244, 245, 270, 275, 281	180, 196, 198, 200,
\glxstrmarkhook	202–204, 206, 208–211, 213, 214, 216–
290	219, 221, 225, 226, 229, 230, 232, 234,
\glxstrnewabbrevpresetkeyhook	236, 239, 241, 243–251, 253, 255, 256,
179	258, 260, 262, 265, 266, 268, 270–272,
\glxstrnewnumber	274–276, 278, 279, 281–284, 286, 287, 289
18	\glxstrpostreset
\glxstrnewsymbol	89, 91, 99
18	\glxstrpostunset
\glxstrNoGlossaryWarning	89, 91, 99
20, 117	\glxstrprelocation
\GlsXtrNoGlsWarningAutoMake
121	312, 314, 316, 318, 320, 322, 345
\GlsXtrNoGlsWarningBuildInfo	\glxstrprotectlinks
121	78–80
\GlsXtrNoGlsWarningCheckFile	\GlsXtrRecordCounter
121	11
\GlsXtrNoGlsWarningEmptyMain	\glxstrrecordtriggervalue
121	135
\GlsXtrNoGlsWarningEmptyNotMain ...	\glxstrregularfont
121	54, 60
\GlsXtrNoGlsWarningEmptyStart	\glxstrresourcecount
121	122, 123
\GlsXtrNoGlsWarningHead	\glxstrresourcefile
121	122
\GlsXtrNoGlsWarningMisMatch	\glxstrresourceinit
121	122
\GlsXtrNoGlsWarningNoOut	\glxstrrestoremarkhook
122	290
\GlsXtrNoGlsWarningTail	\glxstrrestorepostpunc
122	110
\glxstrnopostpunc	\glxstrscfont
109	214
\glxstronlydescname	\glxstrscsuffix
289
\glxstronlydescsort	215, 216, 218, 219, 222, 223, 225, 227
289	\GlsXtrSetActualChar
\glxstronlysuffix	168
287	\glxstrsetaliasnoindex
\glxstrorg@ifKV@glslink@hyper	13, 75
54	\GlsXtrSetEncapChar
\glxstrorglong	168
178, 199, 276	\GlsXtrSetEscChar
\glxstrorgshort	168
178, 199	\glxstrsetfieldifexists
\GLSxtrp	31
85	\GlsXtrSetLevelChar
\Glsxtrp	168
85	\glxstrsetpopts
\glxstrp	84
84, 86	

<code>\glxtrsetupfulldefs</code>	<code>\glxtrtitleplural</code>
..... 182–185, 205, 227, 241, 262	291–293, 305
<code>\GLSxtrshort</code>	<code>\Glsxtrtitleshort</code>
17, 18, 88, 293, 294	291, 292, 303
<code>\Glsxtrshort</code>	<code>\glxtrtitleshort</code>
17, 18, 294	291, 292, 302
<code>\glxtrshort</code>	<code>\Glsxtrtitleshortpl</code>
17, 18, 293	291, 292, 303
<code>\glxtrshortdescname</code> ...	<code>\glxtrtitleshortpl</code>
208, 219, 233, 251	291, 292, 303
<code>\glxtrshorthyphen</code>	<code>\Glsxtrtitletext</code>
285	291–293, 304
<code>\glxtrshorthyphenlong</code>	<code>\glxtrtitletext</code>
283	291–293, 304
<code>\glxtrshortlongdescname</code>	<code>\GlsXtrTotalRecordCount</code>
..... 201, 217, 231, 247, 249, 272, 283, 286	135
<code>\glxtrshortlongdescsort</code>	<code>\glxtrtreetopindent</code>
..... 201, 217, 231, 247, 249, 272, 283, 286	327, 336
<code>\GLSxtrshortpl</code>	<code>\glxtrundefaction</code>
17, 18, 293, 294	7, 13, 25, 34, 37, 38
<code>\Glsxtrshortpl</code>	<code>\glxtrundeftag</code>
17, 18, 294, 295	25, 112, 113
<code>\glxtrshortpl</code>	<code>\glxtrunsrtdo</code>
17, 18, 294	129
<code>\glxtrsmfont</code>	<code>\GlsXtrUseAbbrStyleFmts</code>
228 200, 202, 208, 210, 211, 213,
<code>\glxtrsmsuffix</code>	214, 216, 218, 221, 230, 232, 235, 244,
..... 229, 231, 232, 234, 236, 237, 239, 241	246, 248, 249, 252, 257, 260, 268, 270,
<code>\Glsxtrsubsequentfmt</code>	271, 273, 275, 276, 278, 281, 283, 287, 289
..... 193, 196, 211, 222, 223,	<code>\GlsXtrUseAbbrStyleSetup</code> 207, 209, 210,
236, 238, 253, 255, 257, 259, 277, 280, 285	213, 220, 223, 235, 237, 252, 256, 257, 260
<code>\glxtrsubsequentfmt</code>	<code>\glxtruserfield</code>
..... 193, 196, 211, 222, 223,	264
236, 237, 253, 255, 257, 259, 276, 280, 285	<code>\glxtruserparen</code>
<code>\Glsxtrsubsequentplfmt</code>	265–272
..... 192, 196, 211, 222, 223,	<code>\glxtrusersuffix</code>
236, 238, 254, 255, 257, 259, 277, 280, 285	265, 266, 269, 271
<code>\glxtrsubsequentplfmt</code>	<code>\glxtruseseealsoformat</code>
..... 192, 196, 211, 222, 223,	40
236, 237, 253, 255, 257, 259, 277, 280, 285	<code>\glxtruseseeeformat</code>
<code>\glxtrsuppllocationurl</code>	39
117	<code>\GlsXtrWarnDeprecatedAbbrStyle</code> 176, 197
<code>\glxtrtagfont</code>	<code>\GlsXtrWarning</code>
171	47, 48
<code>\Glsxtrtitlefirst</code>	<code>\glxtrword</code>
291–293, 305, 306	177
<code>\glxtrtitlefirst</code>	<code>\glxtrwordsep</code> 177, 273, 276, 278, 279, 282, 284
291–293, 305	<code>\glxtrwrglossmark</code>
<code>\Glsxtrtitlefirstplural</code>	22
291–293, 306	
<code>\glxtrtitlefirstplural</code>	
291–293, 306	
<code>\Glsxtrtitlefull</code>	
292, 293, 308	
<code>\glxtrtitlefull</code>	
292, 293, 307, 308	
<code>\Glsxtrtitlefullpl</code>	
292, 293, 308	
<code>\glxtrtitlefullpl</code>	
292, 293, 308	
<code>\Glsxtrtitlelong</code>	
292, 293, 307	
<code>\glxtrtitlelong</code>	
291–293, 306	
<code>\Glsxtrtitlelongpl</code>	
292, 293, 307	
<code>\glxtrtitlelongpl</code>	
291–293, 307	
<code>\Glsxtrtitlename</code>	
291, 292, 304	
<code>\glxtrtitlename</code>	
291, 292, 303, 304	
<code>\glxtrtitleorpdforheading</code>	
..... 23, 125, 126, 291, 292	
<code>\Glsxtrtitleplural</code>	
291–293, 305	

H

<code>\hangindent</code> .	324, 325, 327, 336–338, 342, 343
<code>\hbox</code>	311
<code>\hfill</code>	311
<code>\href</code>	78
<code>\hsize</code>	50, 51
<code>\hss</code>	311
<code>\hyperlink</code>	79
<code>\hyperpage</code>	164
<code>\hyperref</code>	78, 117
<code>hyperref package</code>	80, 164, 289, 302

I

<code>\if</code>	46
<code>\if@glxtr@autoseeindex</code>	23, 24, 38, 41
<code>\if@glxtr@format@override</code>	165
<code>\if@glxtrdocdefrestricted</code>	45
<code>\if@glxtrindexcrossrefs</code>	14, 43
<code>\ifblank</code>	26, 47, 48, 105
<code>\ifcase</code> ..	7, 12, 19, 20, 22, 45, 57, 110, 323, 348

<code>\ifcsdef</code>	25, 32, 34–37, 58, 72, 73, 84–88, 96, 108, 114, 115, 126, 130, 133, 134, 156–159, 161–163, 172, 176, 192, 196, 314–321
<code>\ifcsstring</code>	25, 152, 195
<code>\ifcsundef</code>	30, 32, 34–36, 45, 50, 52, 80, 91, 96–100, 113, 114, 118, 129, 152, 195–197, 311, 328, 329, 336, 350
<code>\ifcsvoid</code>	43, 151
<code>\ifdef</code>	13, 18, 24, 28, 37, 38, 40, 41, 50, 51, 74, 78, 79, 85–87, 108, 112, 113, 123, 124, 132, 154, 155, 168, 171, 264, 291, 302–308, 311–314, 322–326, 338–343, 346, 349, 350
<code>\ifdefempty</code>	7–9, 30, 34, 36, 39, 40, 58, 59, 90, 102, 105, 108, 116, 128, 131, 135, 136, 169, 177, 192, 347
<code>\ifdefequal</code>	44, 121, 131, 162
<code>\ifdefstring</code>	6, 31, 165, 170, 346
<code>\ifdefvoid</code>	38, 41–44, 79, 96, 113, 117, 131
<code>\ifdim</code>	50, 51, 104, 328–336
<code>\IfFileExists</code>	21, 117, 121, 122, 124, 310
<code>\ifglossaryexists</code>	38
<code>\ifglssacronym</code>	17, 121
<code>\ifglssacrshortcuts</code>	19
<code>\ifglssautomake</code>	108, 121, 124
<code>\ifglssentrycounter</code>	32
<code>\ifglssentryexists</code>	8, 37, 38, 47, 48, 51, 60, 131, 152, 171, 172
<code>\ifglssfieldeq</code>	150
<code>\ifglsshasfield</code>	27, 28, 264
<code>\ifglsshaslong</code>	95, 96, 139
<code>\ifglsshasparent</code>	126–128, 131, 329, 331, 332
<code>\ifglsshasshort</code>	39, 54, 60
<code>\ifglsshasymbol</code>	173, 323–325, 327
<code>\ifglssindexonlyfirst</code>	76
<code>\ifglssnogroupskip</code>	312, 314–324, 326, 338, 346
<code>\ifglssnonumberlist</code>	53
<code>\ifglssnopostdot</code>	15, 110
<code>\ifglsssanitizesort</code>	107
<code>\ifglsssubentrycounter</code>	32
<code>\ifglssused</code>	43, 44, 74, 76, 92, 101, 104, 192, 329–331, 333–335
<code>\ifglssxindy</code>	117, 119, 120
<code>\ifglssxtr@hyperoutside</code>	58, 59
<code>\ifglssxtrinitwrglossbefore</code>	56, 58, 59, 136
<code>\ifglssxtrininsertinside</code>	185–191, 194, 199, 201, 203–212, 215, 217–229, 231–245, 247–263, 265–267, 269, 272, 273, 276, 279–282, 284, 286, 288
<code>\ifHy@hyperindex</code>	164
<code>\ifinlistcs</code>	29, 45
<code>\ifinner</code>	23
<code>\ifKV@glsslink@hyper</code>	54, 57–59
<code>\ifKV@glsslink@local</code>	56, 136
<code>\ifKV@glsslink@noindex</code>	8, 9, 11, 28, 75, 76
<code>\ifmmode</code>	23
<code>\ifnum</code>	14, 92, 100, 101, 113, 122, 135, 324, 325, 337
<code>\ifstrempy</code>	126, 133
<code>\ifstrequal</code>	16, 21
<code>\ifthenelse</code>	121
<code>\IfTrackedLanguageFileExists</code>	309
<code>\ifundef</code>	30, 105, 170, 171
<code>\ifx</code>	8–10, 40, 50, 51, 105, 109, 115–117, 124, 165, 166, 168, 175, 177, 179, 273, 344
<code>\immediate</code>	92, 100, 118, 124
<code>\index</code>	165
<code>\indexspace</code>	312, 323–326, 338–344, 346, 349
<code>\input</code>	309
<code>\inputencodingname</code>	124
<code>\istfilename</code>	105
<code>\item</code>	119, 120, 311–314, 322–324, 339, 340
J	
<code>\jobname</code>	117, 119–122, 124
K	
<code>\key@ifundefined</code>	11, 12, 26, 72, 128, 130
<code>\KV@glsslink@hyperfalse</code>	61, 74, 79, 80
<code>\KV@glsslink@hypertrue</code>	80
<code>\KV@glsslink@noindexfalse</code>	74, 75
<code>\KV@glsslink@noindextrue</code>	75, 80
L	
<code>\LaTeX</code>	119, 120
<code>\leaders</code>	311
<code>\leavevmode</code>	34, 57
<code>\let</code>	5, 7–13, 15, 17–19, 23–25, 27, 33, 44, 46, 49, 50, 52–71, 73–75, 77–81, 84, 90, 91, 99–106, 108–114, 122, 124, 128, 129, 131, 136, 157, 158, 160–166, 170, 171, 175–179, 182–191, 194, 196, 205, 227, 241, 262, 290–293, 322, 323, 327, 329, 340, 347–349
<code>\letabbreviationstyle</code>	204, 206, 207, 209, 212, 213, 219, 220, 233, 235, 251, 252
<code>\letcs</code>	26, 30, 39, 44, 58, 72, 112–114, 130, 131, 156–163, 350
<code>\levelchar</code>	168

<code>\listadd</code>	96	255–258, 260, 262, 265–268, 270–272,
<code>\listbreak</code>	170	274–276, 278, 279, 281–284, 286, 287, 289
<code>\listcsadd</code>	28	<code>\newacronym</code>
<code>\listcseadd</code>	29, 97	102, 103
<code>\listcsgadd</code>	29, 45	<code>\newacronymhook</code>
<code>\listcsxadd</code>	29, 97	102
<code>\loadglsentries</code>	46, 119	<code>\newacronymstyle</code>
<code>\long</code>	33	103, 104
M		
<code>\MakeAcronymsAbbreviations</code>	104	<code>\newcommand</code>
<code>\makeatletter</code>	117, 122, 167	5–8, 10–12, 14–30, 32–
<code>\makeatother</code>	167	36, 38–40, 42–44, 46–49, 51–54, 56, 57,
<code>\makebox</code>	311, 337, 338	60, 61, 72, 73, 75–77, 79, 80, 83–92, 94–
<code>\makefirstuc</code>	170	104, 108–114, 116–156, 162, 164–178,
<code>makeglossaries</code>	111	180–192, 194–197, 199, 201, 202, 208,
<code>\makeglossaries</code>	105, 118, 120, 121, 124	214, 228, 242, 264, 265, 273, 274, 276,
<code>\makeglossary</code>	105	278, 279, 282, 284, 287, 289–310, 312,
<code>makeindex</code>	351	322, 326–329, 336, 337, 345, 346, 349, 350
<code>makeindex</code>	104	<code>\newcount</code>
<code>\makenoidxglossaries</code>	120	14, 122, 132
<code>\MakeTextUppercase</code>	292	<code>\newentry</code>
<code>\MakeUppercase</code>	291, 292	18
<code>\marginpar</code>	23	<code>\newglossary</code>
<code>\markboth</code>	291	16, 105
<code>\markright</code>	290	<code>\newglossaryentry</code>
<code>\maxdimen</code>	50, 51 18, 46, 90, 98, 102, 154, 155, 179
<code>\mbox</code>	313, 314, 337	<code>\newglossaryentry options</code>
<code>\medskip</code>	121, 130	alias
<code>\MessageBreak</code>		15, 38, 41–44
.....	45, 46, 49, 92, 101, 105, 108, 109, 195	desc
<code>mfistuc package</code>	170	144, 149
<code>\mfistucMakeUppercase</code>	60–	descplural
71, 73, 82, 83, 85, 88, 95, 103, 139–150,		144, 145, 149
159, 160, 163, 183, 185, 186, 188, 190–194		first 78, 141, 142, 148, 198, 297–299, 305, 351
<code>\mfu@checkword@arg</code>	170	firstplural
<code>\mfu@checkword@do</code>	170	142, 148, 198, 298, 306, 351
N		
<code>\NeedsTeXFormat</code>	5, 310, 345	group
<code>\new@glossaryentry</code>	46, 108	130, 131
<code>\new@ifnextchar</code>	27,	loclist
72, 73, 94, 95, 133, 136–138, 174, 182–191		28
<code>\newabbr</code>	17, 18	long
<code>\newabbreviation</code>	17, 18	146, 150, 306
<code>\newabbreviationhook</code>	179	longplural
<code>\newabbreviationstyle</code> ..	198–202, 204,	147, 150, 307
206–221, 223, 224, 226, 228, 230–233,		name
235, 237, 239, 240, 243–249, 251–253,		39, 140, 147, 164, 295, 303
		plural
		141, 148, 198, 296, 297, 304
		see
		15, 24, 38, 39, 41, 44, 46, 106
		seealso
		15, 38, 39, 41, 43, 44, 362
		short
		145, 149, 176
		shortplural
		146, 150, 176
		symbol
		143, 148
		symbolplural
		143, 144, 149
		text
		78,
		140, 141, 147, 148, 198, 200, 295, 296, 304
		<code>\newglossarystyle</code>
		346
		<code>\newif</code>
		56, 164, 198
		<code>\newlength</code>
		327
		<code>\newnum</code>
		18
		<code>\newrobustcmd</code>
		27–31, 40, 41,
		72, 73, 84, 85, 94, 95, 110, 114, 125, 126,
		129, 130, 133, 134, 136–138, 163, 170,
		171, 182–192, 273, 291, 293–302, 329–335
		<code>\newsym</code>
		18
		<code>\newterm</code>
		154
		<code>\newtoks</code>
		176

<code>\newwrite</code>	105	<code>docdef</code>	14, 44, 45, 90, 98
<code>\nobreak</code>	313, 314, 323–326, 339–342	false	45
<code>\NoCaseChange</code>	85–88, 126, 293–302	restricted	14
<code>\noexpand</code> 10, 11, 21, 40, 42, 43, 102, 118, 122,	128, 129, 131, 165, 166, 179, 310, 347, 348	true	46
<code>\nofiles</code>	121	<code>docdefs</code>	
<code>\noindent</code>	121, 325, 326, 340–343	restricted	45
<code>\nopagebreak</code>	323–326, 338–345, 349	<code>nonumberlist</code>	51
<code>\nopostdesc</code>	34, 47, 48, 109, 154	<code>nopostdot</code>	15
<code>\nr</code>	7, 12, 14, 19, 20, 22, 57, 110	false	15
<code>\ns@GLSxtrfull</code>	183	<code>numbers</code>	18
<code>\ns@GLSxtrfull</code>	183	<code>postdot</code>	15
<code>\ns@glxtrfull</code>	182	<code>record</code>	7, 12, 44, 54, 105, 122, 360
<code>\ns@GLSxtrfullpl</code>	185	alsoindex	8, 10
<code>\ns@GLSxtrfullpl</code>	184	only	7, 8
<code>\ns@glxtrfullpl</code>	184	<code>shortcuts</code>	19
<code>\ns@GLSxtrlong</code>	188	ac	19
<code>\ns@GLSxtrlong</code>	187	all	19
<code>\ns@glxtrlong</code>	187	false	19
<code>\ns@GLSxtrlongpl</code>	191	none	19
<code>\ns@GLSxtrlongpl</code>	190, 191	true	19
<code>\ns@glxtrlongpl</code>	190	<code>style</code>	21
<code>\ns@GLSxtrshort</code>	186	<code>stylemods</code>	21
<code>\ns@GLSxtrshort</code>	186	<code>symbols</code>	18, 155
<code>\ns@glxtrshort</code>	185	<code>undefaction</code>	37
<code>\ns@GLSxtrshortpl</code>	189	error	6
<code>\ns@GLSxtrshortpl</code>	189	warn	6
<code>\ns@glxtrshortpl</code>	188	xindy	40, 41
<code>\null</code>	20	<code>\PackageError</code>	6, 11,
<code>\number</code>	97–100, 122, 133	21, 24, 45, 46, 49, 50, 72, 73, 75, 84, 85,	
<code>\numexpr</code>	97, 100	90, 91, 98, 100, 101, 103, 105, 107, 108,	
O			
<code>\or</code>	7, 13, 19, 20, 22, 46, 57, 323, 348	115, 124, 128, 130, 132, 195–197, 310, 311	
<code>\org@glossaryentrynumbers</code>	51, 109	<code>\PackageWarning</code>	15
<code>\org@glossarytitle</code>	109	<code>\PackageWarningNoLine</code>	15
<code>\org@ifKV@glslink@hyper</code>	57, 59	<code>\pageref</code>	32
P			
<code>\p@gl@s@hyp@opt</code>	77	<code>\par</code> . 121, 122, 313, 314, 323–327, 337–343, 346	
<code>package options:</code>		<code>\parindent</code>	
abbreviations	16, 17	322, 324, 325, 327, 337, 338, 340–344, 347	
accsupp	20, 140	<code>\parskip</code>	322, 324, 325, 340–342, 347
acronym	17	<code>\PassOptionsToPackage</code>	5, 21
automake	108, 119, 124	<code>\pdfbookmark</code>	346
true	124	<code>\preglossarypreamble</code>	32
autoseeindex	24	<code>\preto</code>	75
false	23	<code>\print@noop@unsrtglossaryunit</code> ...	11, 13
debug		<code>\print@op@unsrtglossaryunit</code>	13
showtargets	79	<code>\printabbreviations</code>	17
		<code>\printglossaries</code>	106, 120
		<code>\printglossary</code>	17, 106, 120
		<code>\printglossary options</code>	
		nonumberlist	53

90–92, 98, 100, 101, 103–109, 118–125, 128–130, 132, 158, 160, 161, 163, 165, 349	tracklang package 123
\strut 311–321, 325	U
\subglossentry	\u 132
. . . . 109, 131, 311–321, 323–325, 337, 348	\undef 13, 169
\subitem 322, 323	\underline 171
\subsubitem 323	\unskip 34, 44, 311
T	\usepackage 120, 121
\tablehead 318–321	V
\tabletail 318–321	\val 7, 12, 14, 19, 20, 22, 57, 110
\tabularnewline 314–322	W
\TeX 119	\warn@nomakeglossaries 106
\texorpdfstring 28, 85–88, 291, 302–308	\warn@noprintglossary 106, 109
textcase package 290	\write 40, 92, 100, 105, 118, 124
\textsc 214	X
\textsmaller 228	\x 117
\texttt 23, 118–121	\xcapitalisewords 156
\the 102, 103, 116, 117, 122, 168, 179, 180, 198–206, 208–211, 213–219, 221, 225–227, 229–236, 239–241, 243–251, 253, 255, 256, 258, 260, 262, 265–268, 270–272, 274–276, 278, 279, 281–287, 289	\xdef 109, 129
\theglentrycounter 8–10, 58, 59, 136	\xifinlist 96
\theHglentrycounter 8–10, 58, 59, 136	\xifinlistcs 29
\theindex 164	xindy 351
\this@dialect 309	xindy 104
\thisgrptitle 349	xkeyval package 5
\toks@ 116, 117, 168	\XKV@checkchoice 53
	\XKV@plfalse 53
	\XKV@resa 53
	\XKV@sttrue 53