

`glossaries-extra.sty v1.22: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2017-11-08

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	36
1.3.2	Document Definitions	43
1.3.3	Existing Glossary Style Modifications	48
1.3.4	Entry Formatting, Hyperlinks and Indexing	53
1.3.5	Entry Counting	88
1.3.6	Acronym Modifications	101
1.3.7	Indexing and Displaying Glossaries	103
1.3.8	Support for bib2gls	131
1.4	Integration with glossaries-accsupp	138
1.5	Categories	149
1.6	Abbreviations	174
1.6.1	Abbreviation Styles Setup	193
1.6.2	Predefined Styles (Default Font)	196
1.6.3	Predefined Styles (Small Capitals)	212
1.6.4	Predefined Styles (Fake Small Capitals)	226
1.6.5	Predefined Styles (Emphasized)	241
1.6.6	Predefined Styles (User Parentheses Hook)	262
1.6.7	Predefined Styles (Hyphen)	271
1.6.8	Predefined Styles (No Short on First Use)	285
1.7	Using Entries in Headings	288
1.8	Multi-Lingual Support	307
2	Style Adjustments (glossaries-extra-stylemods.sty)	309
2.1	Package Initialisation	309
2.2	List-Like Styles	310
2.3	Longtable Styles	313
2.4	Long Ragged Styles	315
2.5	Supertabular Styles	317
2.6	Super Ragged Styles	319
2.7	Inline Style	321
2.8	Tree Styles	321
2.9	Multicolumn Styles	337

3 bookindex style (glossary-bookindex.sty)	343
3.1 Package Initialisation and Options	343
Glossary	349
Change History	350
Index	365

1 Main Package Code (glossaries-extra.sty)

1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2017/11/08 v1.22 (NLCT)]
```

Requires xkeyval to define package options.

```
3 \RequirePackage{xkeyval}
```

Requires etoolbox package.

```
4 \RequirePackage{etoolbox}
```

Has glossaries already been loaded?

```
5 \@ifpackageloaded{glossaries}
```

```
6 {%
```

Already loaded so pass any options to \setupglossaries. This means that the options that can only be set when glossaries is loaded can't be used.

```
7 \newcommand{\glxstr@doption}[1]{\setupglossaries{#1}}%
```

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

```
9 }
```

```
10 {%
```

Not already loaded, so pass options to glossaries.

```
11 \newcommand{\glxstr@doption}[1]{%
```

```
12 \PassOptionsToPackage{#1}{glossaries}%
```

```
13 }%
```

Set the defaults.

```
14 \PassOptionsToPackage{toc}{glossaries}
```

```
15 \PassOptionsToPackage{nopostdot}{glossaries}
```

```
16 \PassOptionsToPackage{noredefwarn}{glossaries}
```

```
17 \@ifpackageloaded{polyglossia}%
```

```
18 {}%
```

```
19 {%
```

```
20 \@ifpackageloaded{babel}%
```

```
21 {\PassOptionsToPackage{translate=babel}{glossaries}}%
```

```
22 {}%
```

```
23 }%
```

```
24 \newcommand*{\@glxstr@declareoption}[2]{%
```

```
25 \DeclareOptionX{#1}{#2}%
```

```
26 \DeclareOption{#1}{#2}%
```

```
27 }
```

```
28 }
```

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

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

```

```

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

63 \define@choicekey{glossaries-extra.sty}{undefaction}[\val\nr]%
64 {warn,error}%
65 {%
66   \ifcase\nr\relax
67     \let\glstrundefaction\@glstr@warn@undefaction
68     \let\glstr@warnonexistssordo\@glstr@warn@onexistssordo
69     \let\@glstr@redef@forglsentries\@glstr@do@redef@forglsentries
70   \or
71     \let\glstrundefaction\@glstr@err@undefaction
72     \let\glstr@warnonexistssordo\@gobble
73     \let\@glstr@redef@forglsentries\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   \begingroup
84   \ifKV@glslink@noindex
85   \else
86     \edef\@gls@label{\glsdetoklabel{#1}}%
87     \let\glslabel\@gls@label
88     \glswriteentry{#1}%
89     {%
90       \ifdefempty{\@glxtr@thevalue}%
91       {%
92         \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
93         \else
94           \let\theHglentrycounter\@glxtr@theHvalue
95           \fi
96           \glxtr@saveentrycounter
97           \let\@do@wrglossary\@glxtr@dorecord
98       }%
99       {%
100         \let\theHglentrycounter\@glxtr@thevalue
101         \let\theHglentrycounter\@glxtr@theHvalue
102         \let\@do@wrglossary\@glxtr@dorecordnodefer
103       }%
104       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
105       \glxtr@do@wrglossary{#1}%
106       \else
107         \@glxtrwrglossmark
108         \@do@wrglossary
109       \fi
110     }%
111   \fi
112 \endgroup
113 }

```

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

```

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

```

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

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

```

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

```


122 \beginngroup

Save the label in case it's needed.

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

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

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

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

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

Check if thevalue has been set.

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

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

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

Save the entry counter.

```
142     \glsxtr@saveentrycounter
```

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

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

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

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

No need to escape special characters.

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

```

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

```

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

```

160 \newcommand*\@glsxtr@dorecord{%
161   \global\let\@glsrecordlocref\theglentrycounter
162   \let\@glsxtr@orgprefix\@glo@counterprefix
163   \ifx\theglentrycounter\theHglentrycounter
164     \def\@glo@counterprefix{}%
165   \else
166     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
167       {\theglentrycounter}{\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 \theglentrycounter 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\theglentrycounter\theHglentrycounter
179     \protected@write\@auxout{{\string\glsxtr@record
180       {\@gls@label}{\@gls@counter}{\@glsnumberformat}}%
181       {\theglentrycounter}}%
182   \else
183     \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
184       {\theglentrycounter}{\theHglentrycounter}}%
185     }%
186     \@do@gls@getcounterprefix
187     \protected@write\@auxout{{\string\glsxtr@record
188       {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}}%
189       {\theglentrycounter}}%
190   \fi
191   \@glsxtr@counterrecordhook
192 }

```

r@recordcounter

```

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

```

```

195 }

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

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

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

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

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

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

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

```

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

```

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

Add a key to store the group heading.

```

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

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

```

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

alsoindex

```

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

only

```

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

off

```

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

Now define the record package option.

```

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

Don't record.

```

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

```

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

```

Only record (don't index).

```

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

```

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

```

288     \def\glxtrsetaliasnoindex{}%

```

\@glxtr@setupsort@none was only introduced to glossaries v4.30, so it may not be available. If it's defined, use it to remove the unnecessary overhead of escaping and sanitizing the sort value.

```

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

```

Record and index.

```

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

```

Version 1.06 changes the docdef option to a choice rather than boolean setting. The available values are: false, true or restricted. The restricted option permits document definitions as long as they occur before the first glossary is displayed.

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

Need to provide conditional commands that are backward compatible:

if@glxtrdocdef
 307 \newcommand*\if@glxtrdocdef{\ifnum\@glxtr@docdefval>0 }
 lsxtrdocdeftrue
 308 \newcommand*\@glxtrdocdeftrue{\@glxtr@docdefval=1 }
 sxtrdocdeffalse
 309 \newcommand*\@glxtrdocdeffalse{\@glxtr@docdefval=0 }

By default don't allow entries to be defined in the document to encourage the user to define them in the preamble, but if the user is really determined to define them in the document allow them to request this.

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

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

oifexistsorwarn Need an error to notify user if an undefined entry is being referenced in the glossary for the docdef=restricted option. This is used by \glossentryname (but not by \glossentrydesc etc as one error per entry is sufficient).
 319 \newcommand*\@glsdofexistsorwarn{\glsdofexistsorwarn}

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

Switch off since this can increase the build time.

326 \@glxtrindexcrossrefsfalse

But allow see key to switch it on automatically.

oindexcrossrefs

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

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

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

iesExtraWarning Allow users to suppress warnings.

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

raWarningNoLine Allow users to suppress warnings.

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

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

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

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

postdot Shortcut for nopostdot=false

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

nopostdot Needs to redefine \@glxtr@defpostpunc

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

```

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

```

`glstrabbrvtype` Glossary type for abbreviations.

```

392 \newcommand*\glstrabbrvtype{\glstrdefaulttype}

```

`abbreviationsdef` Set by abbreviations option.

```

393 \newcommand*\@glstr@abbreviationsdef{}

```

`abbreviationsdef`

```

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

```



```

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

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

404 \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.

```

412 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
413   \newcommand*{\ab}{\cgl}{%
414     \newcommand*{\abp}{\cgl spl}{%
415       \newcommand*{\as}{\glstrshort}{%
416         \newcommand*{\asp}{\glstrshort pl}{%
417           \newcommand*{\al}{\glstrlong}{%
418             \newcommand*{\alp}{\glstrlong pl}{%
419               \newcommand*{\af}{\glstrfull}{%
420                 \newcommand*{\afp}{\glstrfull pl}{%
421                   \newcommand*{\Ab}{\cGls}{%
422                     \newcommand*{\Abp}{\cGls pl}{%
423                       \newcommand*{\As}{\glstrshort}{%
424                         \newcommand*{\Asp}{\glstrshort pl}{%
425                           \newcommand*{\Al}{\glstrlong}{%
426                             \newcommand*{\Alp}{\glstrlong pl}{%
427                               \newcommand*{\Af}{\glstrfull}{%
428                                 \newcommand*{\Afp}{\glstrfull pl}{%
429                                   \newcommand*{\AB}{\cGLS}{%
430                                     \newcommand*{\ABP}{\cGLS pl}{%
431                                       \newcommand*{\AS}{\GLSxtrshort}{%
432                                         \newcommand*{\ASP}{\GLSxtrshort pl}{%
433                                           \newcommand*{\AL}{\GLSxtrlong}{%
434                                             \newcommand*{\ALP}{\GLSxtrlong pl}{%
435                                               \newcommand*{\AF}{\GLSxtrfull}{%
436                                                 \newcommand*{\AFP}{\GLSxtrfull pl}{%
437                                                   \newcommand*{\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}{\cglS}%
442   \newcommand*{\acp}{\cglSpl}%
443   \newcommand*{\acs}{\glSxtrshort}%
444   \newcommand*{\acsp}{\glSxtrshortpl}%
445   \newcommand*{\acl}{\glSxtrlong}%
446   \newcommand*{\aclp}{\glSxtrlongpl}%
447   \newcommand*{\acf}{\glSxtrfull}%
448   \newcommand*{\acfp}{\glSxtrfullpl}%
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   \newcommand*{\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}{\glSxtrnewsymbol}%
473   }{}%
474   \ifdef\printnumbers
475   {%
476     \newcommand*{\newnum}{\glSxtrnewnumber}%
477   }{}%
478   \let\GlsXtrDefineOtherShortcuts\relax
479 }

```

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

@setupshortcuts Command used to set the shortcuts option.

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

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

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

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

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

```

518      \GlsXtrDefineAbbreviationShortcuts
519      \GlsXtrDefineOtherShortcuts
520    }%

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

    Leave none and false as last option.

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

```

`lsxtr@doaccsupp`

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

`lsxtr@redefstyles`

```

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

```

stylemods

```

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

```

glstr@do@style

```

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

```

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

```

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

```

Defer actual style change:

```

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

```

Set this as the default style:

```

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

```

Set this style:

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

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

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

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

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

sxtrwrglossmark Does nothing by default.

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

debug Provide extra debug options.

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

Pass all other options to glossaries.

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

Process options.

```
615 \ProcessOptionsX
```

Load glossaries if not already loaded.

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

Load the glossaries-accsupp package if required.

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

Redefine \glspostdescription if required.

```
618 \let\@glxtr@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   \glxtrtitleorpdforheading
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\@glxtr@org@doseeglossary\@do@seeglossary
```

r@doseeglossary

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

oindex@glossary

```
643 \newcommand*{\@glxtr@dosee@alsoindex@glossary}[2]{%
644   \@glxtr@recordsee{#1}{#2}%
645   \@glxtr@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\@glxtr@org@gloautosee\@glo@autosee
```

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

```
648 \if@glxtr@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\glxtr@org@@do@wrglossary\@@do@wrglossary
```

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

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

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

```
687 \let\glxtr@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\glxtr@indexonly@saveentrycounter
```

Set up record option if required.

```
689 \glxtr@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\@glxtrundeftag{\glxtrundeftag}%
693 }
```

1.2 Extra Utilities

`trifemptyglossary` `\glxtrifemptyglossary{<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{\glxtrifemptyglossary}[3]{%
695   \ifcsdef{glolist@#1}%
696   {%
697     \ifcsstring{glolist@#1}{,}{#2}{#3}%
698   }%
699   {%
700     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
701     #2%
702   }%
703 }
```

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

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

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

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

vide@storagekey Unstarred version.

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

```

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

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

```

Provide the no-link command if not already defined.

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

vide@storagekey Starred version.

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

```

```

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

```

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

`\GlsXtrFmtField`

```

745 \newcommand{\GlsXtrFmtField}{useri}

```

`\GlsXtrFmtDefaultOptions`

```

746 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

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

```

`\glxtrentryfmt` No link or indexing.

```

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

```

`@glxtrentryfmt`

```

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

```

```

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

```

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

```

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

```

trfieldlistgadd Similarly but uses `\listcsgadd`.

```

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

```

trfieldlisteadd Similarly but uses `\listcseadd`.

```

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

```

trfieldlistxadd Similarly but uses `\listcsxadd`.

```

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

```

Now provide commands to iterate over these lists.

fieldddolistloop

```

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

```

ieldforlistloop

```

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

```

List element tests:

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

```

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

```

`rfieldxifinlist` Expands item.

```

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

```

`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.

```

804 \newrobustcmd{\glxtrifhasfield}{%
805   \@ifstar{\s@glxtrifhasfield}{\@glxtrifhasfield}%
806 }

```

`lsxtrifhasfield` Unstarred version adds grouping.

```

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

```

`lsxtrifhasfield` Starred version omits grouping.

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

`etfieldifexists`

```

826 \newcommand*{\glxtrsetfieldifexists}[3]{\glsetdoifexists{#1}{#3}}

```

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

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

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

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

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

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

`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.

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

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

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

`xGlsXtrSetField`

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

`eGlsXtrSetField`

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

`XtrIfFieldEqStr`

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

```

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

```

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

```

862 \ifglentrycounter
863   \newcommand*{\glxtrpageref}[1]{\pageref{glentry-\glsdetoklabel{#1}}}
864 \else
865   \ifglssubentrycounter
866     \newcommand*{\glxtrpageref}[1]{\pageref{glentry-\glsdetoklabel{#1}}}
867   \else
868     \newcommand*{\glxtrpageref}[1]{\gls{#1}}
869   \fi
870 \fi

```

`lossarypreamble`

```

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

```

`lossarypreamble`

```

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

```

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`

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

`ewglossaryentry` Starred version.

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

`ewglossaryentry` Unstarred version.

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

The following is different from the base glossaries.sty:

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



```

929     \egroup
930 }%
931 }

```

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

```

932 \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.

```

933 \renewcommand{\newignoredglossary}{%
934   \@ifstar\glxtr@s@newignoredglossary\glxtr@org@newignoredglossary
935 }

```

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

```

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

```

`\ignoredglossary` Starred form.

```

964 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
965   \ifcsdef{glolist@#1}
966   {%

```

```

967 \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
968 }%
969 {%
970 \ifdefempty\@ignored@glossaries
971 {%
972 \edef\@ignored@glossaries{#1}%
973 }%
974 {%
975 \eappto\@ignored@glossaries{,#1}%
976 }%
977 \csgdef{glolist@#1}{,}%
978 \ifcsundef{gls@#1@entryfmt}%
979 {%
980 \defglsentryfmt[#1]{\glsentryfmt}%
981 }%
982 }%
983 }%
984 }

```

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

```

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

```

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

```

1006 \newcommand{\provideignoredglossary}{%
1007 \@ifstar\glxtr@s@provideignoredglossary\glxtr@provideignoredglossary
1008 }

```

`ignoredglossary` Unstarred version.

```

1009 \newcommand*{\glxtr@provideignoredglossary}[1]{%
1010   \ifcsdef{glolist@#1}
1011   {}%
1012   {%
1013     \ifdefempty\@ignored@glossaries
1014     {%
1015       \edef\@ignored@glossaries{#1}%
1016     }%
1017     {%
1018       \eappto\@ignored@glossaries{, #1}%
1019     }%
1020     \csgdef{glolist@#1}{,}%
1021     \ifcsundef{gls@#1@entryfmt}%
1022     {%
1023       \defglsentryfmt[#1]{\glsentryfmt}%
1024     }%
1025     {}%
1026     \ifdefempty\@gls@nohyperlist
1027     {%
1028       \renewcommand*{\@gls@nohyperlist}{#1}%
1029     }%
1030     {%
1031       \eappto\@gls@nohyperlist{, #1}%
1032     }%
1033   }%
1034 }

```

`ignoredglossary` Starred form.

```

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

```

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

is glossary label.

```
1054 \newcommand*{\glstrcopytoglossary}[2]{%
1055   \glsdoidexists{#1}%
1056   {%
1057     \ifcsdef{glolist@#2}
1058     {%
1059       \cseappto{glolist@#2}{#1,}%
1060     }%
1061     {%
1062       \glstrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1063     }%
1064   }%
1065 }
```

1.3.1 Existence Checks

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

```
1066 \renewcommand{\glsdoidexists}[2]{%
1067   \ifglstryexists{#1}{#2}%
1068   {%
```

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

```
1069   \edef\glslabel{\glsdetoklabel{#1}}%
1070   \glstrundefaction{Glossary entry ‘\glslabel’
1071     has not been defined}{You need to define a glossary entry before
1072     you can reference it.}%
1073   }%
1074 }
```

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

```
1075 \renewcommand{\glsdoidnoexists}[2]{%
1076   \ifglstryexists{#1}{%
1077     \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1078       has already been defined}{}}{#2}%
1079 }
```

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

```
1080 \ifdef\glsdoidexistsordo
1081 {%
1082   \renewcommand{\glsdoidexistsordo}[3]{%
1083     \ifglstryexists{#1}{#2}%
1084     {%
1085       \glstrundefaction{Glossary entry ‘\glsdetoklabel{#1}’
1086         has not been defined}{You need to define a glossary entry
1087         before you can use it.}%
1088       #3%
1089     }%
```

```

1090 }%
1091 }
1092 {%
1093   \glstr@warnonexistsordo\glsoifexistsordo
1094   \newcommand{\glsoifexistsordo}[3]{%
1095     \ifglstryexists{#1}{#2}%
1096     {%
1097       \glstrundefaction{Glossary entry ‘\glstoklabel{#1}’
1098       has not been defined}{You need to define a glossary entry
1099       before you can use it.}%
1100       #3%
1101     }%
1102   }%
1103 }

```

arynoexistsordo Similarly for \doifglossarynoexistsordo.

```

1104 \ifdef\doifglossarynoexistsordo
1105 {%
1106   \renewcommand{\doifglossarynoexistsordo}[3]{%
1107     \ifglossaryexists{#1}%
1108     {%
1109       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1110       #3%
1111     }%
1112     {#2}%
1113   }%
1114 }
1115 {%
1116   \glstr@warnonexistsordo\doifglossarynoexistsordo
1117   \newcommand{\doifglossarynoexistsordo}[3]{%
1118     \ifglossaryexists{#1}%
1119     {%
1120       \glstrundefaction{Glossary type ‘#1’ already exists}{}%
1121       #3%
1122     }%
1123     {#2}%
1124   }%
1125 }
1126

```

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.

```

1127 \appto\newglossaryentryposthook{%
1128   \ifdefvoid\@glo@see
1129   {\csxdef{glo@\@glo@label @see}{}}%
1130   {%

```

```

1131 \csxdef{glo@\@glo@label @see}{\@glo@see}%
1132 \if@glxtr@autoseeindex
1133 \@glxtr@autoindexcrossrefs
1134 \fi
1135 }%
1136 }
1137 \appto\@gls@keymap{,{see}{see}}

```

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

```

1138 \newcommand*{\glxtrusesee}[1]{%
1139 \glsdoifexists{#1}%
1140 {%
1141 \letcs{\@glo@see}{glo@\glsdetoklabel{#1}@see}%
1142 \ifdefempty\@glo@see
1143 {}%
1144 {%
1145 \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1146 }%
1147 }%
1148 }

```

`\glxtr@usesee`

```

1149 \newcommand*{\glxtr@usesee}[1][\seename]{%
1150 \@glxtr@usesee{#1}%
1151 }

```

`\@glxtr@usesee`

```

1152 \def\@glxtr@usesee[#1]#2\@end@glxtr@usesee{%
1153 \glxtruseseeformat{#1}{#2}%
1154 }

```

`truseseeformat` 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.

```

1155 \newcommand*{\glxtruseseeformat}[2]{%
1156 \glsseeformat{#1}{#2}{}%
1157 }

```

`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 `\glsaccessstext` for abbreviations.

```

1158 \renewcommand*{\glsseeitemformat}[1]{%
1159 \ifglshashshort{glslabel}{\glsaccessstext{#1}}{\glsaccessname{#1}}%
1160 }

```

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

```

1161 \newcommand*{\glxtruseealso}[1]{%
1162   \glstoifexists{#1}%
1163   {%
1164     \letcs{\@glo@see}{glo\glstoifexists{#1}@seealso}%
1165     \ifdefempty\@glo@see
1166     {}%
1167     {%
1168       \expandafter\glxtrusealsoformat\expandafter{\@glo@see}%
1169     }%
1170   }%
1171 }

```

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

```

1172 \newcommand*{\glxtrusealsoformat}[1]{%
1173   \glstoifexists{\seealsoformat{#1}}%
1174 }

```

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

```

1175 \newrobustcmd{\glxtrseelist}[1]{%
1176   \edef\@glo@tmp{\noexpand\glstoifexists{#1}}\@glo@tmp
1177 }

```

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

```

1178 \providecommand{\seealsoformat}{see also}

```

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

```

1179 \ifdef\@xdycrossrefhook
1180 {
1181   Add the cross-reference class definition to the hook.
1182   \appto\@xdycrossrefhook{%
1183     \write\glstoifexists{(define-crossref-class \string"seealso\string"
1184       :unverified )}%
1185     \write\glstoifexists{(markup-crossref-list
1186       :class \string"seealso\string"^^J\space\space\space
1187       :open \string"\string\glxtrusealsoformat\glstoifexists\string"
1188       :close \string"\glstoifexists\string")}%
1189   }

```

Append to class list.

```

1189   \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

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

```

1190 \newrobustcmd*{\glxtrindexseealso}[2]{%
1191   \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex

```

```

1192     \@glstr@recordsee{#1}{#2}%
1193 \fi
1194 \glsdoifexists{#1}%
1195 {%
1196     \@glstrwrglossmark
1197     \def\@gls@xref{#2}%
1198     \@onelevel@sanitize\@gls@xref
1199     \@gls@checkmkidxchars\@gls@xref
1200     \gls@glossary{\csname glo@#1@type\endcsname}{%
1201         (indexentry
1202             :tkey (\csname glo@#1@index\endcsname)
1203             :xref (\string"\@gls@xref\string")
1204             :attr \string"seealso\string"
1205         )
1206     }%
1207 }%
1208 }
1209 }
1210 {
    xindy not in use or glossaries version too old to support this.
1211 \newrobustcmd*{\glstrindexseealso}{\glssee[\seealsoname]}
1212 }

```

The alias key should be set to the label of the synonymous entry. The seealso key essentially behaves like `see=[\seealsoname]{\langle x r - l i s t \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`.

```

1213 \ifdef\gls@set@xr@key
1214 {

```

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

```

1215 \define@key{glossentry}{alias}{%
1216     \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1217 }
1218 \define@key{glossentry}{seealso}{%
1219     \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1220 }

```

Add to the key mappings.

```

1221 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1222 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1223 \appto\@newglossaryentryposthook{%
1224     \ifdefvoid\@glo@seealso

```



```

1225     {\csxdef{glo@\@glo@label @seealso}{}}}%
1226     {%
1227         \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1228         \if@glxtr@autoseeindex
1229             \@glxtr@autoindexcrossrefs
1230         \fi
1231     }%

```

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

```

1232     \ifdefvoid\@glo@alias
1233     {\csxdef{glo@\@glo@label @alias}{}}}%
1234     {%
1235         \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1236     }%
1237 }

```

Provide user-level commands to access the values.

`\glxtralias`

```

1238 \newcommand*{\glxtralias}[1]{\@gls@entry@field{#1}{alias}}

```

`trseealsolabels`

```

1239 \newcommand*{\glxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}

```

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

```

1240 \appto\@glo@autoseehook{%
1241     \ifdefvoid\@glo@alias
1242     {%
1243         \ifdefvoid\@glo@seealso
1244         {}%
1245         {%
1246             \edef\@do@glsssee{\noexpand\glxtrindexseealso
1247                 {\@glo@label}{\@glo@seealso}}}%
1248             \@do@glsssee
1249         }%
1250     }%
1251     {%

```

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

```

1252         \ifdefvoid\@glo@see
1253         {%
1254             \edef\@do@glsssee{\noexpand\glsssee{\@glo@label}{\@glo@alias}}}%
1255             \@do@glsssee
1256         }%
1257     }%
1258 }%
1259 }%
1260 }
1261 {

```

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

`\glsxtralias`

```
1262 \glsaddstoragekey*{alias}{\glsxtralias}
```

`trseealsolabels`

```
1263 \glsaddstoragekey*{seealso}{\glsxtrseealsolabels}
```

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.

```
1264 \appto\@newglossaryentryposthook{%
1265   \ifcvoid{glo@\@glo@label @alias}%
1266   {%
1267     \ifcvoid{glo@\@glo@label @seealso}%
1268     {}%
1269     {%
1270       \edef\@do@glsssee{\noexpand\glsxtrindexseealso
1271         {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1272       \@do@glsssee
1273     }%
1274   }%
1275   {%
```

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

```
1276   \ifdefvoid\@glo@see
1277   {%
1278     \edef\@do@glsssee{\noexpand\glsssee
1279       {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1280     \@do@glsssee
1281   }%
1282   {}%
1283   }%
1284 }

1285 }
```

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

```
1286 \AtEndDocument{\if@glxtrindexcrossrefs\glsxtraddallcrossrefs\fi}
```

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

```
1287 \newcommand*\glsxtraddallcrossrefs{%
1288   \forallglossaries{\@glo@type}%
1289   {%
1290     \forglsentries[\@glo@type]{\@glo@label}%
1291     {%
1292       \ifglssused{\@glo@label}%
```

```

1293      {\expandafter\@glxtr@addunusedxrefs\expandafter{\@glo@label}}{}%
1294    }%
1295  }%
1296 }

```

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

```

1297 \newcommand*{\@glxtr@addunusedxrefs}[1]{%
1298   \letcs{\@glo@see}{glo\@glsetoklabel{#1}@see}%
1299   \ifdefvoid\@glo@see
1300   {}%
1301   {%
1302     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1303   }%
1304   \letcs{\@glo@see}{glo\@glsetoklabel{#1}@seealso}%
1305   \ifdefvoid\@glo@see
1306   {}%
1307   {%
1308     \expandafter\glxtr@addunused\@glo@see\@end@glxtr@addunused
1309   }%
1310 }

```

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

```

1311 \newcommand*{\glxtr@addunused}[1][]{%
1312   \@glxtr@addunused
1313 }

```

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

```

1314 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1315   \@for\@glxtr@label:=#1\do
1316   {%
1317     \ifglxused{\@glxtr@label}{}%
1318     {%
1319       \glxadd[format=glxtrunusedformat]{\@glxtr@label}%
1320       \glxunset{\@glxtr@label}%
1321       \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1322     }%
1323   }%
1324 }

```

lxtrunusedformat

```

1325 \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.

```

1326 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries

```

```

1327 \renewcommand{\makenoidxglossaries}{%
1328   \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off
1329   {%
1330     \glxtr@orgmakenoidxglossaries
1331     Add marker to \@do@seeglossary
1332     \renewcommand{\@do@seeglossary}[2]{%
1333       \@glxtrwrglossmark
1334       \edef\@gls@label{\glsdetoklabel{##1}}%
1335       \protected@write\@auxout{}{%
1336         \string\@gls@reference
1337         {\csname glo@\@gls@label @type\endcsname}%
1338         {\@gls@label}%
1339         {%
1340           \string\glsseeformat##2}%
1341         }%
1342       }%
1343       Check for docdefs=restricted:
1344       \if@glxtrdocdefrestricted
1345       If restricted document definitions allowed, adjust \@gls@reference so that it doesn't test for
1346       existence.
1347       \renewcommand*\@gls@reference[3]{%
1348         \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1349         \ifinlistcs{##2}{\glsref@##1}%
1350         {}%
1351         {\listcsgadd{\glsref@##1}{##2}}%
1352         \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1353         {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1354         {}%
1355         \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1356       }%
1357       \else
1358       Disable document definitions.
1359       \@glxtrdocdeffalse
1360       \fi
1361       \disable@keys{glossaries-extra}{docdef}%
1362     }%
1363     {%
1364       \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1365         not permitted\MessageBreak
1366         with record=\@glxtr@record@setting\space package option}%
1367       {You may only use \string\makenoidxglossaries\space with the
1368         record=off option}%
1369     }%
1370   }%

```

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

```

1367 \renewcommand*{\gls@defdocnewglossaryentry}{%
1368   \ifcase\@glsxtr@docdefval
      docdef=false:
1369     \renewcommand*{\newglossaryentry}[2]{%
1370       \PackageError{glossaries-extra}{Glossary entries must
1371         be \MessageBreak defined in the preamble with \MessageBreak
1372         package option ‘docdef=false’\MessageBreak(consider using
1373         ‘docdef=restricted’)}{Move your glossary definitions to
1374         the preamble. You can also put them in a \MessageBreak separate file
1375         and load them with \string\loadglsentries.}%
1376     }%
1377   \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.
1378     \let\gls@checkseeallowed\relax
1379     \let\newglossaryentry\new@glossaryentry
1380   \or
      Restricted mode just needs to allow the see value.
1381     \let\gls@checkseeallowed\relax
1382   \fi
1383 }%

```

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`

```

1384 \newcommand*{\GlsXtrEnableOnTheFly}{%
1385   \@ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1386 }

```

`\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.)

```

1387 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1388   \renewcommand*{\glsdetoklabel}[1]{%
1389     \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1390     {%
1391       \expandafter\detokenize\expandafter{##1}%
1392     }%
1393     {\detokenize{##1}}%
1394   }%
1395   \@GlsXtrEnableOnTheFly
1396 }
1397 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end#3#4{%

```

```

1398 \expandafter\if\glsbackslash#1%
1399   #3%
1400 \else
1401   #4%
1402 \fi
1403 }

```

sxtrstarflywarn

```

1404 \newcommand*{\glsxtrstarflywarn}{%
1405   \GlossariesExtraWarning{Experimental starred version of
1406   \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1407   read the warnings in the glossaries-extra user manual)}}%
1408 }

```

rEnableOnTheFly

```

1409 \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:

\glsxtrcat

```

1410 \newcommand*{\glsxtrcat}{general}

```

\glsxtr

```

1411 \newcommand*{\glsxtr}[1] [] {%
1412   \def\glsxtr@keylist{##1}%
1413   \@glsxtr
1414 }

```

\@glsxtr

```

1415 \newcommand*{\@glsxtr}[2] [] {%
1416   \ifglsentryexists{##2}%
1417   {%
1418     \ifblank{##1}{ }\{\GlsXtrWarning{##1}{##2}}%
1419   }%
1420   {%
1421     \gls@defglossaryentry{##2}{name={##2},category=\glsxtrcat,
1422     description={\nopostdesc},##1}%
1423   }%
1424   \expandafter\gls\expandafter[\glsxtr@keylist]{##2}%
1425 }

```

\Glsxtr

```

1426 \newcommand*{\Glsxtr}[1] [] {%
1427   \def\glsxtr@keylist{##1}%
1428   \@Glsxtr
1429 }

```

```

\@Glsxtr
1430 \newcommand*{\@Glsxtr}[2] [] {%
1431 \ifglentryexists{##2}%
1432 {%
1433 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1434 }%
1435 {%
1436 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1437 description={\nopostdesc},##1}%
1438 }%
1439 \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
1440 }

\glstrpl
1441 \newcommand*{\glstrpl}[1] [] {%
1442 \def\glstr@keylist{##1}%
1443 \@glstrpl
1444 }

\@glstrpl
1445 \newcommand*{\@glstrpl}[2] [] {%
1446 \ifglentryexists{##2}%
1447 {%
1448 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1449 }%
1450 {%
1451 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1452 description={\nopostdesc},##1}%
1453 }%
1454 \expandafter\glsp\expandafter[\glstr@keylist]{##2}%
1455 }

\Glsxtrpl
1456 \newcommand*{\Glsxtrpl}[1] [] {%
1457 \def\glstr@keylist{##1}%
1458 \@Glsxtrpl
1459 }

\@Glsxtrpl
1460 \newcommand*{\@Glsxtrpl}[2] [] {%
1461 \ifglentryexists{##2}
1462 {%
1463 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1464 }%
1465 {%
1466 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1467 description={\nopostdesc},##1}%
1468 }%
1469 \expandafter\Glspl\expandafter[\glstr@keylist]{##2}%

```

```
1470 }
```

`\GlsXtrWarning`

```
1471 \newcommand*{\GlsXtrWarning}[2]{%
1472   \def\@glxtr@optlist{##1}%
1473   \@onelevel@sanitize\@glxtr@optlist
1474   \GlossariesExtraWarning{The options ‘\@glxtr@optlist’ have
1475     been ignored for entry ‘##2’ as it has already been defined}%
1476 }
```

Disable commands after the glossary:

```
1477 \renewcommand\@printglossary[2]{%
1478   \def\@glxtr@printglossopts{##1}%
1479   \@glxtr@orgprintglossary{##1}{##2}%
1480   \def\@glxtr{\@glxtr@disabledflycommand\glxtr}%
1481   \def\@glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1482   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1483   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1484 }
```

`@disabledflycommand`

```
1485 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1486   \PackageError{glossaries-extra}%
1487   {\string##1\space can’t be used after any of the \MessageBreak
1488     glossaries have been displayed}%
1489   {The on-the-fly commands enabled by
1490     \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1491     before the glossaries. If you want to use any entries \MessageBreak
1492     after any of the glossaries, you must use the standard \MessageBreak
1493     method of first defining the entry and then using the \MessageBreak
1494     entry with commands like \string\glx}%
1495   \@glxtr@disabledflycommand
1496 }%
1497 \newcommand*{\@glxtr@disabledflycommand}[2][\]{##2}
```

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

```
1498 \let\GlsXtrEnableOnTheFly\relax
1499 }
1500 \@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.

```
1501 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

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

etglossarystyle

```
1502 \renewcommand*{\setglossarystyle}[1]{%
1503   \ifcsundef{@glsstyle@#1}%
1504   {%
1505     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1506   }%
1507   {%
1508     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
1509   \protected@edef\@glsxtr@current@style{#1}%
1510 }%
1511 \ifx\@glossary@default@style\relax
1512   \protected@edef\@glossary@default@style{#1}%
1513 \fi
1514 }
```

In case we have an old version of glossaries:

```
1515 \ifdef\@glossary@default@style
1516 {}
1517 {%
1518   \let\@glossary@default@style\relax
1519 }
```

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

```
1520 \ifdef\glslistdottedwidth
1521 {%
1522   \ifdim\glslistdottedwidth=.5\hsize
1523     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1524     \AtBeginDocument{%
1525       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1526         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1527       \fi
1528     }%
1529   \fi
1530 }
1531 {}%
```

Similarly for \glsdescwidth:

\glsdescwidth

```
1532 \ifdef\glsdescwidth
1533 {%
1534   \ifdim\glsdescwidth=.6\hsize
1535     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1536     \AtBeginDocument{%
1537       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1538         \setlength{\glsdescwidth}{.6\columnwidth}%
1539       \fi
1540     }%
1541   \fi
1542 }
```

```

1539     \fi
1540   }%
1541 \fi
1542 }
1543 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

1544 \ifdef\glspagelistwidth
1545 {%
1546   \ifdim\glspagelistwidth=.1\hsize
1547     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1548     \AtBeginDocument{%
1549       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1550         \setlength{\glspagelistwidth}{.1\columnwidth}%
1551       \fi
1552     }%
1553 \fi
1554 }
1555 {}%

```

aryentrynumbers Has the nonnumberlist option been used?

```

1556 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1557 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1558   \glsnonnumberlistfalse
1559   \renewcommand*{\glossaryentrynumbers}[1]{%
1560     \ifglsentryexists{\glscurrententrylabel}%
1561     {%
1562       \@glsxtrpreloctag
1563       \GlsXtrFormatLocationList{#1}%
1564       \@glsxtrpostloctag
1565       \gls@save@numberlist{#1}%
1566     }{}%
1567   }%
1568 \else
1569   \glsnonnumberlisttrue
1570   \renewcommand*{\glossaryentrynumbers}[1]{%
1571     \ifglsentryexists{\glscurrententrylabel}%
1572     {%
1573       \gls@save@numberlist{#1}%
1574     }{}%
1575   }%
1576 \fi

```

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

```

1577 \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`

```

1578 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1579   \let\@glsxtrpreloctag\@glsxtrpreloctag
1580   \let\@glsxtrpostloctag\@glsxtrpostloctag
1581   \renewcommand*{\@glsxtr@pagetag}{#1}%
1582   \renewcommand*{\@glsxtr@pagetag}{#2}%
1583   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
1584     \csgdef{\@glsxtr@preloctag##1}{##2}%
1585   }%
1586   \renewcommand*{\@glsxtr@doloctag}{%
1587     \ifcsundef{\@glsxtr@preloctag\@glscurrententrylabel}%
1588     {%
1589       \GlossariesWarning{Missing pre-location tag for ‘\@glscurrententrylabel’.
1590         Rerun required}%
1591     }%
1592     {%
1593       \csuse{\@glsxtr@preloctag\@glscurrententrylabel}%
1594     }%
1595   }%
1596 }
1597 \@onlypreamble\GlsXtrEnablePreLocationTag

```

`glsxtrpreloctag`

```

1598 \newcommand*{\@glsxtrpreloctag}{%
1599   \let\@glsxtr@org@delimN\delimN
1600   \let\@glsxtr@org@delimR\delimR
1601   \let\@glsxtr@org@glsgignore\glsgignore
1602   \gdef is required as the delimiters may occur inside a scope.
1603   \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1604   \renewcommand*{\delimN}{%
1605     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1606     \@glsxtr@org@delimN}%
1607   \renewcommand*{\delimR}{%
1608     \gdef\@glsxtr@thisloctag{\@glsxtr@pagetag}%
1609     \@glsxtr@org@delimR}%
1610   \renewcommand*{\glsgignore}[1]{%
1611     \gdef\@glsxtr@thisloctag{\relax}%
1612     \@glsxtr@org@glsgignore{##1}}%
1613   \@glsxtr@doloctag
1613 }

```

`glsxtrpreloctag`

```

1614 \newcommand*{\@glsxtrpreloctag}{}

```

```

@glxtr@pagetag
1615 \newcommand*{\@glxtr@pagetag}{}%

glxtr@pagetag
1616 \newcommand*{\@glxtr@pagetag}{}%

lsxtrpostloctag
1617 \newcommand*{\@glxtrpostloctag}{%
1618   \let\delimN\@glxtr@org@delimN
1619   \let\delimR\@glxtr@org@delimR
1620   \let\glsignore\@glxtr@org@glsignore
1621   \protected@write\@auxout{}{%
1622     {\string\@glxtr@savepreloctag{\glscurrententrylabel}\@glxtr@thisloctag}}%
1623 }

lsxtrpostloctag
1624 \newcommand*{\@glxtrpostloctag}{%

lsxtr@preloctag
1625 \newcommand*{\@glxtr@savepreloctag}[2]{%
1626 \protected@write\@auxout{}{%
1627   \string\providecommand\string\@glxtr@savepreloctag[2]{%

glxtr@doloctag
1628 \newcommand*{\@glxtr@doloctag}{%

ss@nonumberlist Modify the nonumberlist key to use \GlsXtrFormatLocationList (and also save the number
list):
1629 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1630   \XKV@plfalse
1631   \XKV@sttrue
1632   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1633   {%
1634     \csname glsnonumberlist\XKV@resa\endcsname
1635     \ifglsnonumberlist
1636       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1637     \else
1638       \def\glossaryentrynumbers##1{%
1639         \@glxtrpreloctag
1640         \GlsXtrFormatLocationList{##1}%
1641         \@glxtrpostloctag
1642         \gls@save@numberlist{##1}}%
1643       \fi
1644   }%
1645 }

```

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.

```
1646 \renewcommand*{\glsentryfmt}{%
1647   \ifglshasshort{\glslabel}{\glsssetabbrvfmt{\glscategory{\glslabel}}}{}%
1648   \glsifregular{\glslabel}%
1649   {\glstrregularfont{\glsgenentryfmt}}}%
1650   {%
1651     \ifglshasshort{\glslabel}%
1652     {\glstrgenabbrvfmt}%
1653     {\glstrregularfont{\glsgenentryfmt}}}%
1654   }%
1655 }
```

`\glstrregularfont` Font used for regular entries.

```
1656 \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.

```
1657 \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).

```
1658   \@glstr@record{#2}{#3}{glslink}%
1659   \glsoifexists{#3}%
1660   {%
```

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).

```
1661   \let\glstrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1662   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1663   \def\glscustomtext{#4}%
1664   \@glstr@field@linkdefs
1665   #1%
1666   \@gls@link[#2]{#3}{#4}%
1667   \let\ifKV@glslink@hyper\glstrorg@ifKV@glslink@hyper
1668   }%
```

```

1669 \glspostlinkhook
1670 }

```

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.

```

1671 \let\@glsxtr@org@gls@\@gls@
1672 \def\@gls@#1#2{%
1673   \@glsxtr@record{#1}{#2}{glslink}%
1674   \@glsxtr@org@gls@{#1}{#2}%
1675 }%

```

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

```

1676 \let\@glsxtr@org@glspl@\@glspl@
1677 \def\@glspl@#1#2{%
1678   \@glsxtr@record{#1}{#2}{glslink}%
1679   \@glsxtr@org@glspl@{#1}{#2}%
1680 }%

```

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

```

1681 \let\@glsxtr@org@Gls@\@Gls@
1682 \def\@Gls@#1#2{%
1683   \@glsxtr@record{#1}{#2}{glslink}%
1684   \@glsxtr@org@Gls@{#1}{#2}%
1685 }%

```

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

```

1686 \let\@glsxtr@org@Glspl@\@Glspl@
1687 \def\@Glspl@#1#2{%
1688   \@glsxtr@record{#1}{#2}{glslink}%
1689   \@glsxtr@org@Glspl@{#1}{#2}%
1690 }%

```

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

```

1691 \let\@glsxtr@org@GLS@\@GLS@
1692 \def\@GLS@#1#2{%
1693   \@glsxtr@record{#1}{#2}{glslink}%
1694   \@glsxtr@org@GLS@{#1}{#2}%
1695 }%

```

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

```

1696 \let\@glsxtr@org@GLSpl@\@GLSpl@
1697 \def\@GLSpl@#1#2{%
1698   \@glsxtr@record{#1}{#2}{glslink}%
1699   \@glsxtr@org@GLSpl@{#1}{#2}%
1700 }%

```

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

```

1701 \renewcommand*{\@glsdisp}[3][\]{%
1702   \@glsxtr@record{#1}{#2}{glslink}%
1703   \glsdoifexists{#2}{%
1704     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
1705     \let\glsifplural\@secondoftwo
1706     \let\glsapscase\@firstofthree
1707     \def\glscustomtext{#3}%
1708     \def\glsinsert{}%
1709     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1710     \@gls@link[#1]{#2}{\@glo@text}%
1711     \ifKV@glslink@local
1712       \glslocalunset{#2}%
1713     \else
1714       \glsunset{#2}%
1715     \fi
1716   }%
1717   \glspostlinkhook
1718 }
```

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

```

1719 \renewcommand*{\@gls@@link}[3][\]{%
1720   \@glsxtr@record{#1}{#2}{glslink}%
1721   \glsdoifexistsordo{#2}{%
1722     {%
1723       \let\do@gls@link@checkfirsthyper\relax
1724       \@gls@link[#1]{#2}{#3}%
1725     }%
1726     {%
1727       \glstextformat{#3}%
1728     }%
1729   \glspostlinkhook
1730 }
```

glsxtrinitwrgloss Set the default if the wrgloss is omitted.

```

1731 \newcommand*{\glsxtrinitwrgloss}{%
1732   \glsifattribute{\glslabel}{wrgloss}{after}%
1733   {%
1734     \glsxtrinitwrglossbeforefalse
1735   }%
1736   {%
1737     \glsxtrinitwrglossbeforetrue
1738   }%
1739 }
```

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

```

1740 \newif\ifglsxtrinitwrglossbefore
1741 \glsxtrinitwrglossbeforetrue
```

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

```
1742 \define@choicekey{glslink}{wrgloss}[\val\nr]{before,after}%
1743 {%
1744   \ifcase\nr\relax
1745     \glstrinitwrglossbeforetrue
1746   \or
1747     \glstrinitwrglossbeforefalse
1748   \fi
1749 }
```

```
1750 \define@key{glslink}{thevalue}{\def\@glstr@thevalue{#1}}
```

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

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

```
1752 \define@boolkey{glslink}[glstr@]{hyperoutside}[true]{%
1753 \glstr@hyperoutsidetrue
```

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

```
1754 \newcommand*{\glstrnithyperoutside}{%
1755   \glsifattribute{\glslabel}{hyperoutside}{false}%
1756   {%
1757     \glstr@hyperoutsidefalse
1758   }%
1759   {%
1760     \glstr@hyperoutsidetrue
1761   }%
1762 }
```

`\@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`.

```
1763 \def\@gls@link[#1]#2#3{%
1764   \leavevmode
1765   \edef\glslabel{\glsdetoklabel{#2}}%
1766   \def\@gls@link@opts{#1}%
1767   \let\@gls@link@label\glslabel
1768   \let\@glsnumberformat\@glstr@defaultnumberformat
1769   \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
1770   \edef\glstype{\csname glo@\glslabel @type\endcsname}%
1771   \let\org@ifKV@glslink@hyper@ifKV@glslink@hyper
```

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

```
1772 \def\@glstr@thevalue{%
1773 \def\@glstr@theHvalue{\@glstr@thevalue}%
```

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

```
1774 \glstrinitwrgloss
```


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

```
1775 \glxtrinithyperoutside
```

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

```
1776 \@gls@setdefault@glslink@opts
1777 \do@gl:disablehyperinlist
1778 \do@gl:link@checkfirsthyper
1779 \setkeys{glslink}{#1}%
1780 \glslinkpostsetkeys
```

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

```
1781 \ifdefempty{\@glxtr@thevalue}%
1782 {%
1783   \@gls@saveentrycounter
1784 }%
1785 {%
1786   \let\theglentrycounter\@glxtr@thevalue
1787   \def\theHglentrycounter{\@glxtr@theHvalue}%
1788 }%
1789 \@gls@setsort{\glslabel}%
```

Check textformat attribute (new to v1.21).

```
1790 \glshasattribute{\glslabel}{textformat}%
1791 {%
1792   \edef\@glxtr@attrval{\glsggetattribute{\glslabel}{textformat}}%
1793   \ifcsdef{\@glxtr@attrval}%
1794   {%
1795     \letcs{\@glxtr@textformat}{\@glxtr@attrval}%
1796   }%
1797   {%
1798     \GlossariesExtraWarning{Unknown control sequence name
1799       '\@glxtr@attrval' supplied in textformat attribute
1800       for entry '\glslabel'. Reverting to default \string\glstextformat}%
1801     \let\@glxtr@textformat\glstextformat
1802   }%
1803 }%
1804 {%
1805   \let\@glxtr@textformat\glstextformat
1806 }%
```

Do write if it should occur before the link text:

```
1807 \ifglxtrinitwrglossbefore
1808   \@do@wrglossary{#2}%
1809 \fi
```

Do the link text:

```
1810 \ifKV@glslink@hyper
1811   \ifglxtr@hyperoutside
1812     \@glslink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1813   \else
1814     \@glxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
```

```

1815 \fi
1816 \else
1817 \ifglxtr@hyperoutside
1818 \glsdonohyperlink{\glolinkprefix\glslabel}{\@glxtr@textformat{#3}}%
1819 \else
1820 \@glxtr@textformat{\glsdonohyperlink{\glolinkprefix\glslabel}{#3}}%
1821 \fi
1822 \fi

```

Do write if it should occur after the link text:

```

1823 \ifglxtrinitwrglossbefore
1824 \else
1825 \@do@wrglossary{#2}%
1826 \fi

```

As the original definition:

```

1827 \let@ifKV@glslink@hyper\org@ifKV@glslink@hyper
1828 }

```

```

1829 \define@key{glossadd}{thevalue}{\def\@glxtr@thevalue{#1}}

```

```

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

```

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

```

1831 \renewrobustcmd*{\glsadd}[2][\]{%
1832 \glxtrifinmark
1833 }%
1834 {%
1835 \@gls@adjustmode
1836 \@glxtr@record{#1}{#2}{glossadd}%
1837 \glsoifexists{#2}%
1838 {%
1839 \let\@glsnumberformat\@glxtr@defaultnumberformat
1840 \edef\@gls@counter{\csname glo@\glsdetoklabel{#2}@counter\endcsname}%
1841 \def\@glxtr@thevalue{}%
1842 \def\@glxtr@theHvalue{\@glxtr@thevalue}%
1843 \setkeys{glossadd}{#1}%
1844 \ifdefempty{\@glxtr@thevalue}%
1845 {%
1846 \@gls@saveentrycounter
1847 }%
1848 {%
1849 \let\theglentrycounter\@glxtr@thevalue
1850 \def\theHglentrycounter{\@glxtr@theHvalue}%
1851 }%
1852 \@do@wrglossary{#2}%
1853 }%
1854 }%
1855 }

```

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

```
1856 \newcommand*{\@glsxtr@field@linkdefs}{%
1857   \let\glsxtrifwasfirstuse\@secondoftwo
1858   \let\glsifplural\@secondoftwo
1859   \let\glscapscase\@firstofthree
1860   \let\glsinsert\@empty
1861 }
```

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

assignfieldfont

```
1862 \newcommand*{\glsxtrassignfieldfont}[1]{%
1863   \ifglstryexists{#1}%
1864   {%
1865     \ifglshasshort{#1}%
1866     {%
1867       \glsetabbrvfmt{\glscategory{#1}}%
1868       \glsifregular{#1}%
1869       {\let\@gls@field@font\glsxtrregularfont}%
1870       {\let\@gls@field@font\@firstofone}%
1871     }%
1872     {%
1873       \glsifnotregular{#1}%
1874       {\let\@gls@field@font\@firstofone}%
1875       {\let\@gls@field@font\glsxtrregularfont}%
1876     }%
1877   }%
1878   {%
1879     \let\@gls@field@font\@gobble
1880   }%
1881 }
```

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

```
1882 \def\@glstext@#1#2[#3]{%
1883   \glsxtrassignfieldfont{#2}%
1884   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesstext{#2}#3}}%
1885 }
```

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

```
1886 \def\@GLStext@#1#2[#3]{%
1887   \glsxtrassignfieldfont{#2}%
1888   \@gls@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1889   {\@gls@field@font{\@GLSaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1890 }
```

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

```
1891 \def\@Glstext@#1#2[#3]{%
1892   \glsxtrassignfieldfont{#2}%
1893 }
```

```

1893 \@gls@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%
1894 {\@gls@field@font{\Glsaccessstext{#2}#3}}}%
1895 }

```

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`

```

1896 \newcommand*{\glstrchecknohyperfirst}[1]{%
1897 \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1898 }

```

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

```

1899 \def\@glsfirst@#1#2[#3]{%
1900 \glstrassignfieldfont{#2}%
    Ensure that \glsfirst honours the nohyperfirst attribute.
1901 \@gls@field@link
1902 [\let\glstrifwasfirstuse\@firstoftwo
1903 \glstrchecknohyperfirst{#2}%
1904 ]{#1}{#2}%
1905 {\@gls@field@font{\glsaccessfirst{#2}#3}}}%
1906 }

```

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

```

1907 \def\@Glsfirst@#1#2[#3]{%
1908 \glstrassignfieldfont{#2}%
    Ensure that \Glsfirst honours the nohyperfirst attribute.
1909 \@gls@field@link
1910 [\let\glstrifwasfirstuse\@firstoftwo
1911 \let\glscapscase\@secondofthree
1912 \glstrchecknohyperfirst{#2}%
1913 ]%
1914 {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}}%
1915 }

```

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

```

1916 \def\@GLSfirst@#1#2[#3]{%
1917 \glstrassignfieldfont{#2}%
    Ensure that \GLSfirst honours the nohyperfirst attribute.
1918 \@gls@field@link
1919 [\let\glstrifwasfirstuse\@firstoftwo
1920 \let\glscapscase\@thirdofthree
1921 \glstrchecknohyperfirst{#2}%
1922 ]%
1923 {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1924 }

```

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

```
1925 \def\@glsplural@#1#2[#3]{%
1926   \glsxtrassignfieldfont{#2}%
1927   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1928   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1929 }
```

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

```
1930 \def\@Glsplural@#1#2[#3]{%
1931   \glsxtrassignfieldfont{#2}%
1932   \@gls@field@link
1933   [\let\glsifplural\@firstoftwo
1934   \let\glscapscase\@secondofthree
1935   ]%
1936   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1937 }
```

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

```
1938 \def\@GLSplural@#1#2[#3]{%
1939   \glsxtrassignfieldfont{#2}%
1940   \@gls@field@link
1941   [\let\glsifplural\@firstoftwo
1942   \let\glscapscase\@thirdofthree
1943   ]%
1944   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1945 }
```

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

```
1946 \def\glsfirstplural@#1#2[#3]{%
1947   \glsxtrassignfieldfont{#2}%
1948   \gls@field@link
1949   [\let\glsxtrifwasfirstuse\@firstoftwo
1950   \let\glsifplural\@firstoftwo
1951   \glsxtrchecknohyperfirst{#2}%
1952   ]%
1953   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1954 }
```

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

```
1955 \def\Glsfirstplural@#1#2[#3]{%
1956   \glsxtrassignfieldfont{#2}%
1957   \gls@field@link
1958   [\let\glsxtrifwasfirstuse\@firstoftwo
1959   \let\glsifplural\@firstoftwo
1960   \let\glscapscase\@secondofthree
```

```

1961 \glstrchecknohyperfirst{#2}%
1962 ]%
1963 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1964 }

```

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

```

1965 \def\@GLSfirstplural@#1#2[#3]{%
1966 \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
1967 \@gls@field@link
1968 [\let\glstrifwasfirstuse\@firstoftwo
1969 \let\glsifplural\@firstoftwo
1970 \let\glscapscase\@thirdofthree
1971 \glstrchecknohyperfirst{#2}%
1972 ]%
1973 {#1}{#2}%
1974 {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
1975 }

```

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

```

1976 \def\@glsname@#1#2[#3]{%
1977 \glstrassignfieldfont{#2}%
1978 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
1979 }

```

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

```

1980 \def\@GLSname@#1#2[#3]{%
1981 \glstrassignfieldfont{#2}%
1982 \@gls@field@link
1983 [\let\glscapscase\@secondoftwo]{#1}{#2}%
1984 {\@gls@field@font{\Glsaccessname{#2}#3}}%
1985 }

```

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

```

1986 \def\@GLSname@#1#2[#3]{%
1987 \glstrassignfieldfont{#2}%
1988 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
1989 {#1}{#2}%
1990 {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
1991 }

```

\@glsdesc@

```

1992 \def\@glsdesc@#1#2[#3]{%
1993 \glstrassignfieldfont{#2}%
1994 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
1995 }

```

```

\@Glsdesc@   First letter uppercase version.
1996 \def\@Glsdesc@#1#2[#3]{%
1997   \glstrassignfieldfont{#2}%
1998   \@gls@field@link
1999   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2000   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2001 }

\@GLSdesc@   All uppercase version.
2002 \def\@GLSdesc@#1#2[#3]{%
2003   \glstrassignfieldfont{#2}%
2004   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2005   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2006 }

@glsdscplural@   No case-changing version.
2007 \def\@glsdscplural@#1#2[#3]{%
2008   \glstrassignfieldfont{#2}%
2009   \@gls@field@link
2010   [\let\glscapscase\@secondoftwo
2011   \let\glsifplural\@firstoftwo
2012   ]{#1}{#2}{\@gls@field@font{\glaccessdscplural{#2}#3}}%
2013 }

@Glsdscplural@   First letter uppercase version.
2014 \def\@Glsdscplural@#1#2[#3]{%
2015   \glstrassignfieldfont{#2}%
2016   \@gls@field@link
2017   [\let\glscapscase\@secondoftwo
2018   \let\glsifplural\@firstoftwo
2019   ]{#1}{#2}{\@gls@field@font{\Glsaccessdscplural{#2}#3}}%
2020 }

@GLSdscplural@   All uppercase version.
2021 \def\@GLSdsc@#1#2[#3]{%
2022   \glstrassignfieldfont{#2}%
2023   \@gls@field@link
2024   [\let\glscapscase\@thirdoftwo
2025   \let\glsifplural\@firstoftwo
2026   ]%
2027   {#1}{#2}%
2028   {\@gls@field@font{\GLSaccessdscplural{#2}\mfirstucMakeUppercase{#3}}}%
2029 }

\@glssymbol@
2030 \def\@glssymbol@#1#2[#3]{%
2031   \glstrassignfieldfont{#2}%
2032   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2033 }

```

\@Glssymbol@ First letter uppercase version.

```
2034 \def\@Glssymbol@#1#2[#3]{%
2035   \glstrassignfieldfont{#2}%
2036   \@gls@field@link
2037   [\let\glscapscase\@secondoftwo]%
2038   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2039 }
```

\@GLSsymbol@ All uppercase version.

```
2040 \def\@GLSsymbol@#1#2[#3]{%
2041   \glstrassignfieldfont{#2}%
2042   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2043   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2044 }
```

lssymbolplural@ No case-changing version.

```
2045 \def\@lssymbolplural@#1#2[#3]{%
2046   \glstrassignfieldfont{#2}%
2047   \@gls@field@link
2048   [\let\glscapscase\@secondoftwo
2049   \let\glsifplural\@firstoftwo
2050   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2051 }
```

lssymbolplural@ First letter uppercase version.

```
2052 \def\@lssymbolplural@#1#2[#3]{%
2053   \glstrassignfieldfont{#2}%
2054   \@gls@field@link
2055   [\let\glscapscase\@secondoftwo
2056   \let\glsifplural\@firstoftwo
2057   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2058 }
```

LSsymbolplural@ All uppercase version.

```
2059 \def\@LSsymbol@#1#2[#3]{%
2060   \glstrassignfieldfont{#2}%
2061   \@gls@field@link
2062   [\let\glscapscase\@thirdoftwo
2063   \let\glsifplural\@firstoftwo
2064   ]%
2065   {#1}{#2}%
2066   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2067 }
```

\@Glsuseri@ First letter uppercase version.

```
2068 \def\@Glsuseri@#1#2[#3]{%
2069   \glstrassignfieldfont{#2}%
2070   \@gls@field@link
```



```

2071 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2072 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2073 }

```

\@GLSuseri@ All uppercase version.

```

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

```

\@Glsuserii@ First letter uppercase version.

```

2079 \def\@Glsuserii@#1#2[#3]{%
2080   \glstrassignfieldfont{#2}%
2081   \@gls@field@link
2082   [\let\glscapscase\@secondoftwo]%
2083   {#1}{#2}{\@gls@field@font{\Glsentryuserii{#2}#3}}%
2084 }

```

\@GLSuserii@ All uppercase version.

```

2085 \def\@GLSuserii@#1#2[#3]{%
2086   \glstrassignfieldfont{#2}%
2087   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2088   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuserii{#2}#3}}}%
2089 }

```

\@Glsuseriii@ First letter uppercase version.

```

2090 \def\@Glsuseriii@#1#2[#3]{%
2091   \glstrassignfieldfont{#2}%
2092   \@gls@field@link
2093   [\let\glscapscase\@secondoftwo]%
2094   {#1}{#2}{\@gls@field@font{\Glsentryuseriii{#2}#3}}%
2095 }

```

\@GLSuseriii@ All uppercase version.

```

2096 \def\@GLSuseriii@#1#2[#3]{%
2097   \glstrassignfieldfont{#2}%
2098   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2099   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriii{#2}#3}}}%
2100 }

```

\@Glsuseriv@ First letter uppercase version.

```

2101 \def\@Glsuseriv@#1#2[#3]{%
2102   \glstrassignfieldfont{#2}%
2103   \@gls@field@link
2104   [\let\glscapscase\@secondoftwo]%
2105   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2106 }

```

\@GLSuseriv@ All uppercase version.

```
2107 \def\@GLSuseriv@#1#2[#3]{%
2108   \glstrassignfieldfont{#2}%
2109   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2110   {#1}{#2}%
2111   {\@gls@field@font{\mfirstucMakeUppercase{\glentryuseriv{#2}#3}}}%
2112 }
```

\@Glsuserv@ First letter uppercase version.

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

\@GLSuserv@ All uppercase version.

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

\@Glsuservi@ First letter uppercase version.

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

\@GLSuservi@ All uppercase version.

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

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

\@acrshort No case change.

```
2135 \def\@acrshort#1#2[#3]{%
2136   \glsdoifexists{#2}%
2137   {%
2138     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2139     \let\glxtrifwasfirstuse\@secondoftwo
2140     \let\glsifplural\@secondoftwo
```

```

2141 \let\glscapscase\@firstofthree
2142 \let\glsinsert\@empty
2143 \def\glscustomtext{%
2144     \acronymfont{\glsaccesssshort{#2}}#3%
2145 }%
2146 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2147 }%
2148 \glspostlinkhook
2149 }

```

\@Acrshort First letter uppercase.

```

2150 \def\@Acrshort#1#2[#3]{%
2151     \glsdoifexists{#2}%
2152     {%
2153         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2154         \let\glstrifwasfirstuse\@secondoftwo
2155         \let\glsifplural\@secondoftwo
2156         \let\glscapscase\@secondofthree
2157         \let\glsinsert\@empty
2158         \def\glscustomtext{%
2159             \acronymfont{\Glsaccesssshort{#2}}#3%
2160         }%
2161         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2162     }%
2163     \glspostlinkhook
2164 }

```

\@ACRshort All uppercase.

```

2165 \def\@ACRshort#1#2[#3]{%
2166     \glsdoifexists{#2}%
2167     {%
2168         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2169         \let\glstrifwasfirstuse\@secondoftwo
2170         \let\glsifplural\@secondoftwo
2171         \let\glscapscase\@thirdofthree
2172         \let\glsinsert\@empty
2173         \def\glscustomtext{%
2174             \mfirstucMakeUppercase{\acronymfont{\glsaccesssshort{#2}}#3}%
2175         }%
2176         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2177     }%
2178     \glspostlinkhook
2179 }

```

\@acrshortpl No case change.

```

2180 \def\@acrshortpl#1#2[#3]{%
2181     \glsdoifexists{#2}%
2182     {%
2183         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

2184 \let\glxtrifwasfirstuse\@secondoftwo
2185 \let\glsifplural\@firstoftwo
2186 \let\glscapscase\@firstofthree
2187 \let\glsinsert\@empty
2188 \def\glscustomtext{%
2189     \acronymfont{\glsaccessshortpl{#2}}#3%
2190 }%
2191 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2192 }%
2193 \glspostlinkhook
2194 }

```

\@Acrshortpl First letter uppercase.

```

2195 \def\@Acrshortpl#1#2[#3]{%
2196     \glsdoifexists{#2}%
2197     {%
2198         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2199         \let\glxtrifwasfirstuse\@secondoftwo
2200         \let\glsifplural\@firstoftwo
2201         \let\glscapscase\@secondofthree
2202         \let\glsinsert\@empty
2203         \def\glscustomtext{%
2204             \acronymfont{\Glsaccessshortpl{#2}}#3%
2205         }%
2206         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2207     }%
2208     \glspostlinkhook
2209 }

```

\@ACRshortpl All uppercase.

```

2210 \def\@ACRshortpl#1#2[#3]{%
2211     \glsdoifexists{#2}%
2212     {%
2213         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2214         \let\glxtrifwasfirstuse\@secondoftwo
2215         \let\glsifplural\@firstoftwo
2216         \let\glscapscase\@thirdofthree
2217         \let\glsinsert\@empty
2218         \def\glscustomtext{%
2219             \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2220         }%
2221         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2222     }%
2223     \glspostlinkhook
2224 }

```

\@acrlong No case change.

```

2225 \def\@acrlong#1#2[#3]{%
2226     \glsdoifexists{#2}%

```

```

2227 {%
2228   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2229   \let\glxtrifwasfirstuse\@secondoftwo
2230   \let\gl@ifplural\@secondoftwo
2231   \let\gl@scapscase\@firstofthree
2232   \let\gl@insert\@empty
2233   \def\gl@customtext{%
2234     \acronymfont{\gl@saccesslong{#2}}#3%
2235   }%
2236   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2237 }%
2238 \glspostlinkhook
2239 }

```

\@Acrlong First letter uppercase.

```

2240 \def\@Acrlong#1#2[#3]{%
2241   \gl@doifexists{#2}%
2242   {%
2243     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2244     \let\glxtrifwasfirstuse\@secondoftwo
2245     \let\gl@ifplural\@secondoftwo
2246     \let\gl@scapscase\@secondofthree
2247     \let\gl@insert\@empty
2248     \def\gl@customtext{%
2249       \acronymfont{\Glsaccesslong{#2}}#3%
2250     }%
2251     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2252   }%
2253   \glspostlinkhook
2254 }

```

\@ACRlong All uppercase.

```

2255 \def\@ACRlong#1#2[#3]{%
2256   \gl@doifexists{#2}%
2257   {%
2258     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2259     \let\glxtrifwasfirstuse\@secondoftwo
2260     \let\gl@ifplural\@secondoftwo
2261     \let\gl@scapscase\@thirdofthree
2262     \let\gl@insert\@empty
2263     \def\gl@customtext{%
2264       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2265     }%
2266     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2267   }%
2268   \glspostlinkhook
2269 }

```

\@acrlongpl No case change.

```

2270 \def\@acrlongpl#1#2[#3]{%
2271   \glsdoifexists{#2}%
2272   {%
2273     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2274     \let\glxtrifwasfirstuse\@secondoftwo
2275     \let\gl@sifplural\@firstoftwo
2276     \let\glscapscase\@firstofthree
2277     \let\gl$insert\@empty
2278     \def\glscustomtext{%
2279       \acronymfont{\gl@saccesslongpl{#2}}#3%
2280     }%
2281     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2282   }%
2283   \glspostlinkhook
2284 }

```

\@Acrlongpl First letter uppercase.

```

2285 \def\@Acrlongpl#1#2[#3]{%
2286   \glsdoifexists{#2}%
2287   {%
2288     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2289     \let\glxtrifwasfirstuse\@secondoftwo
2290     \let\gl@sifplural\@firstoftwo
2291     \let\glscapscase\@secondofthree
2292     \let\gl$insert\@empty
2293     \def\glscustomtext{%
2294       \acronymfont{\Glsaccesslongpl{#2}}#3%
2295     }%
2296     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2297   }%
2298   \glspostlinkhook
2299 }

```

\@ACrlongpl All uppercase.

```

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

```

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

\@glsaddkey

```

2315 \renewcommand*{\@glsaddkey}[7]{%
2316   \key@ifundefined{glossentry}{#1}%
2317   {%
2318     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2319     \appto\@gls@keymap{,{#1}{#1}}%
2320     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2321     \appto\@newglossaryentryposthook{%
2322       \letcs{\@glo@tmp}{@glo@#1}%
2323       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2324     }%
2325     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2326     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2327   \ifcsdef{@gls@user@#1@}%
2328   {%
2329     \PackageError{glossaries}%
2330     {Can't define '\string#5' as helper command
2331     '\expandafter\string\csname @gls@user@#1@endcsname' already
2332     exists}%
2333     {}%
2334   }%
2335   {%
2336     \expandafter\newcommand\expandafter*\expandafter
2337     {\csname @gls@user@#1@endcsname}[2][ ]{%
2338       \new@ifnextchar[%
2339         {\csuse{@gls@user@#1@}{##1}{##2}}%
2340         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2341     \csdef{@gls@user@#1@}##1##2[##3]{%
2342       \@gls@field@link{##1}{##2}{#3{##2}##3}%
2343     }%
2344     \newrobustcmd*{#5}{%
2345       \expandafter\@gls@hyp@opt\csname @gls@user@#1@endcsname}%
2346     }%

```

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

```

2347   \ifcsdef{@Gls@user@#1@}%
2348   {%
2349     \PackageError{glossaries}%
2350     {Can't define '\string#6' as helper command
2351     '\expandafter\string\csname @Gls@user@#1@endcsname' already
2352     exists}%
2353     {}%
2354   }%
2355   {%
2356     \expandafter\newcommand\expandafter*\expandafter

```

```

2357     {\csname @Gls@user@#1\endcsname}[2] [] {%
2358         \new@ifnextchar [%
2359             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2360             {\csuse{@Gls@user@#1@}{##1}{##2} [] }}}%
2361     \csdef{@Gls@user@#1@}##1##2[##3]{%
2362         \@gls@field@link[\let\glscapscase\secondofthree]%
2363         {##1}{##2}{#4{##2}##3}%
2364     }%
2365     \newrobustcmd*{#6}{%
2366         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2367     }%

```

Finally the all caps version (modified):

```

2368     \ifcsdef{@GLS@user@#1@}%
2369     {%
2370         \PackageError{glossaries}%
2371         {Can't define '\string#7' 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\thirdofthree]%
2384         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2385     }%
2386     \newrobustcmd*{#7}{%
2387         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2388     }%
2389 }%
2390 {%
2391     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2392 }%
2393 }

```

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

```

2394 \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.

```

2395 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2396 \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.

```

2397 \ifglssused{\glslabel}%
2398 {\let\glstrifwasfirstuse\@secondoftwo}
2399 {\let\glstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.

2400 \edef\glscategorylabel{\glscategory{\glslabel}}%
2401 \ifglssused{\glslabel}%
2402 {%
2403   \glssifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2404   {\KV@glsslink@hyperfalse}{}}%
2405 }%
2406 {%
2407   \glssifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2408   {\KV@glsslink@hyperfalse}{}}%
2409 }%
2410 \glsslinkcheckfirsthyperhook
2411 }

```

`\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.

```

2412 \ifdefined\do@glssdisablehyperinlist
2413 {%
2414   \let\@glssxtr\do@glssdisablehyperinlist\do@glssdisablehyperinlist
2415   \renewcommand*{\do@glssdisablehyperinlist}{%
2416     \@glssxtr\do@glssdisablehyperinlist
2417     \glssifattribute{\glslabel}{nohyper}{true}{\KV@glsslink@hyperfalse}{}}%
2418   }
2419 }
2420 {}

```

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

```

2421 \define@boolkey{glsslink}{noindex}[true]{}
2422 \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`

```

2423 \ifdefined\@glss@setdefault@glsslink@opts
2424 {
2425   \renewcommand*{\@glss@setdefault@glsslink@opts}{%
2426     \KV@glsslink@noindexfalse
2427     \@glssxtrsetaliasnoindex
2428   }
2429 }
2430 {

```

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

```

2431 \newcommand*{\@glsglslSetDefaultGlslink@opts}{%
2432   \KV@glsglslLink@noindexfalse
2433   \@glsglslSetAliasNoindex
2434 }
2435 \pretol@do@glsglslDisableHyperinlist{\@glsglslSetDefaultGlslink@opts}
2436 }

```

`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.)

```

2437 \providecommand*{\glsglslSetAliasNoindex}{%
2438   \KV@glsglslLink@noindextrue
2439 }

```

`setaliasnoindex`

```

2440 \newcommand*{\@glsglslSetAliasNoindex}{%
2441   \glsglslIfHasField{alias}{\glsglslLabel}%
2442   {%
2443     \let\glsglslIndexAliased\glsglslIndexAliased
2444     \glsglslSetAliasNoindex
2445     \let\glsglslIndexAliased\@no@glsglslIndexAliased
2446   }%
2447   {%
2448 }

```

`xtrindexaliased`

```

2449 \newcommand{\@glsglslXtrIndexAliased}{%
2450   \ifKV@glsglslLink@noindex
2451   \else
2452     \begingroup
2453     \let\@glsglslNumberFormat\glsglslXtr@defaultNumberFormat
2454     \edef\@glsglslCounter{\csname glo@glsglslDetokLabel{\glsglslLabel}@counter\endcsname}%
2455     \glsglslXtr@saveEntryCounter
2456     \do@wrglossary{\glsglslAlias{\glsglslLabel}}%
2457   \endgroup
2458   \fi
2459 }

```

`xtrindexaliased`

```

2460 \newcommand{\@no@glsglslXtrIndexAliased}{%
2461   \PackageError{glossaries-extra}{\string\glsglslXtrIndexAliased\space
2462     not permitted outside definition of \string\glsglslSetAliasNoindex}%
2463   {%
2464 }

```

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

```

2465 \let\glsglslIndexAliased\@no@glsglslIndexAliased

```

tDefaultGlsOpts Set the default options for \glslink etc.

```
2466 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2467   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2468     \setkeys{glslink}{#1}%
2469     \@glsxtrsetaliasnoindex
2470   }%
2471 }
```

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

```
2472 \newcommand*{\glsxtrifindexing}[2]{%
2473   \ifKV@glslink@noindex #2\else #1\fi
2474 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2475 \renewcommand*{\glswriteentry}[2]{%
2476   \glsxtrifindexing
2477   {%
2478     \ifglsindexonlyfirst
2479       \ifglsused{#1}
2480       {\glsxtrdoautoindexname{#1}{dualindex}}}%
2481     {#2}%
2482   \else
2483     \glsifattribute{#1}{indexonlyfirst}{true}%
2484     {\ifglsused{#1}
2485      {\glsxtrdoautoindexname{#1}{dualindex}}}%
2486     {#2}}%
2487   {#2}%
2488   \fi
2489   }%
2490   {%
2491 }
```

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

```
2492 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
2493   \glsxtrdowrglossaryhook{\@gls@label}%
2494 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2495 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
2496   \glsxtrdowrglossaryhook{\@gls@label}%
2497 }
```

xtr@do@@wrindex

```
2498 \newcommand*{\@glsxtr@do@@wrindex}{%
```

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

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

alt@glsl@hyp@opt User version

<code>lt@hyp@opt@char</code>	Contains the character used as the command modifier.
------------------------------	--

1t@hyp@opt@keys	Contains the option list used as the command modifier.
-----------------	--

rSetAltModifier

org@dohyperlink

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

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

```
2526 \ifdef\glsnavhyperlink
2527 {
2528   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
2529     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
```

Scope:

```
2530   {%
2531     \let\glsdohyperlink\glsxtr@org@dohyperlink
2532     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2533   }%
2534 }%
2535 }
2536 {}
```

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

```
2537 \renewcommand*{\glsdohyperlink}[2]{%
2538   \glsasattribute{\glslabel}{targeturl}%
2539   {%
2540     \glsasattribute{\glslabel}{targetname}%
2541     {%
2542       \glsasattribute{\glslabel}{targetcategory}%
2543       {%
2544         \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2545           {\glsgetattribute{\glslabel}{targetcategory}}%
2546           {\glsgetattribute{\glslabel}{targetname}}%
2547           {\glsxtrprotectlinks#2}}%
2548       }%
2549     }%
2550     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2551       }%
2552       {\glsgetattribute{\glslabel}{targetname}}%
2553       {\glsxtrprotectlinks#2}}%
2554   }%
2555 }%
2556 {%
2557   \href{\glsgetattribute{\glslabel}{targeturl}}{%
2558     {\glsxtrprotectlinks#2}}%
2559 }%
2560 }%
2561 {}
```

Check for alias.

```

2562 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2563 \ifdefvoid\gloaliaslabel
2564 {%
2565     \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2566 }%
2567 {%

```

Redirect link to the alias target.

```

2568     \glsxtrhyperlink
2569     {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2570     {\glsxtrprotectlinks#2}}%
2571 }%
2572 }%
2573 }

```

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

```

2574 \ifdef\@glsshowtarget
2575 {
2576     \newcommand{\glsxtrhyperlink}[2]{%
2577         \@glsshowtarget{#1}%
2578         \hyperlink{#1}{#2}%
2579     }%
2580 }
2581 {
2582     \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2583 }

```

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.

```

2584 \renewrobustcmd*{\glsdohyperlink}[2][\glentrytext{\@glo@label}]{%
2585     \glsdoifexists{#2}%
2586     {%
2587         \def\@glo@label{#2}%
2588         {\edef\glslabel{#2}%
2589         \@glslink{\glolinkprefix\glslabel}{#1}}%
2590     }%
2591 }

```

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

```

2592 \renewcommand{\glsdisablehyper}{%
2593     \KV@glslink@hyperfalse
2594     \def\@glslink{\glsdohyperlink}%
2595     \let\@glstarget\@secondoftwo
2596 }

```

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

```
2597 \renewcommand{\glsenablehyper}{%
2598   \KV@glslink@hypertrue
2599   \def\@glslink{\glsdohyperlink}%
2600   \def\@glstarget{\glsdohypertarget}%
2601 }
```

`\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.

```
2602 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

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

```
2603 \ifcsundef{hyperlink}%
2604 {%
2605   \def\@glslink{\glsdonohyperlink}
2606 }%
2607 {%
2608   \def\@glslink{\glsdohyperlink}
2609 }
```

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

```
2610 \newcommand*{\glstrprotectlinks}{%
2611   \KV@glslink@hyperfalse
2612   \KV@glslink@noindextrue
2613   \let\@gls@\@glstr@p@text@
2614   \let\@Gls@\@Glsxtr@p@text@
2615   \let\@GLS@\@GLSxtr@p@text@
2616   \let\@glspl@\@glstr@p@plural@
2617   \let\@Glspl@\@Glsxtr@p@plural@
2618   \let\@GLSpl@\@GLSxtr@p@plural@
2619   \let\@glstrshort@\@glstr@p@short@
2620   \let\@Glsxtrshort@\@Glsxtr@p@short@
2621   \let\@GLSxtrshort@\@GLSxtr@p@short@
2622   \let\@glstrlong@\@glstr@p@long@
2623   \let\@Glsxtrlong@\@Glsxtr@p@long@
2624   \let\@GLSxtrlong@\@GLSxtr@p@long@
2625   \let\@glstrshortpl@\@glstr@p@shortpl@
2626   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
2627   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
2628   \let\@glstrlongpl@\@glstr@p@longpl@
2629   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
2630   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
2631   \let\@acrshort@\@glstr@p@acrshort@
2632   \let\@Acrshort@\@Glsxtr@p@acrshort@
2633   \let\@ACRshort@\@GLSxtr@p@acrshort@}
```

```

2634 \let\@acrshortpl\@glsxtr@p@acrshortpl@
2635 \let\@Acrshortpl\@Glsxtr@p@acrshortpl@
2636 \let\@ACRshortpl\@GLSxtr@p@acrshortpl@
2637 \let\@acrlong\@glsxtr@p@acrlong@
2638 \let\@Acrlong\@Glsxtr@p@acrlong@
2639 \let\@ACRlong\@GLSxtr@p@acrlong@
2640 \let\@acrlongpl\@glsxtr@p@acrlongpl@
2641 \let\@Acrlongpl\@Glsxtr@p@acrlongpl@
2642 \let\@ACRlongpl\@GLSxtr@p@acrlongpl@
2643 }

```

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

@glsxtr@p@text@

```

2644 \def\@glsxtr@p@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}

```

@Glsxtr@p@text@

```

2645 \def\@Glsxtr@p@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}

```

@GLSxtr@p@text@

```

2646 \def\@GLSxtr@p@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}

```

lsxtr@p@plural@

```

2647 \def\@lsxtr@p@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}

```

lSxtr@p@plural@

```

2648 \def\@lSxtr@p@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}

```

LSxtr@p@plural@

```

2649 \def\@LSxtr@p@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}

```

glsxtr@p@short@

```

2650 \def\@glsxtr@p@short@#1#2[#3]{%
2651 {%
2652   \glssetabbrvfmt{\glscategory{#2}}%
2653   \glsabbrvfont{\gl Sentryshort{#2}}#3%
2654 }%
2655 }

```

Glsxtr@p@short@

```

2656 \def\@Glsxtr@p@short@#1#2[#3]{%
2657 {%
2658   \glssetabbrvfmt{\glscategory{#2}}%
2659   \glsabbrvfont{\Glsentryshort{#2}}#3%
2660 }%
2661 }

```


GLSxtr@p@short@

```
2662 \def\@GLSxtr@p@short@#1#2[#3]{%
2663   {%
2664     \glsetabbrvfmt{\glscategory{#2}}%
2665     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshort{#2}}#3}%
2666   }%
2667 }
```

sxtr@p@shortpl@

```
2668 \def\@glsxtr@p@shortpl@#1#2[#3]{%
2669   {%
2670     \glsetabbrvfmt{\glscategory{#2}}%
2671     \glsabbrvfont{\glsentryshortpl{#2}}#3%
2672   }%
2673 }
```

Sxtr@p@shortpl@

```
2674 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2675   {%
2676     \glsetabbrvfmt{\glscategory{#2}}%
2677     \glsabbrvfont{\Glsentryshortpl{#2}}#3%
2678   }%
2679 }
```

Sxtr@p@shortpl@

```
2680 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2681   {%
2682     \glsetabbrvfmt{\glscategory{#2}}%
2683     \mfirstucMakeUppercase{\glsabbrvfont{\glsentryshortpl{#2}}#3}%
2684   }%
2685 }
```

@glsxtr@p@long@

```
2686 \def\@glsxtr@p@long@#1#2[#3]{\{\glsentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2687 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2688 \def\@GLSxtr@p@long@#1#2[#3]{%
2689   {\mfirstucMakeUppercase{\glsfont{\glsentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2690 \def\@glsxtr@p@longpl@#1#2[#3]{\{\glsentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2691 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glsfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2692 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2693   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2694 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

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

xtr@p@acrshort@
2696 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2697   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2698 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

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

r@p@acrshortpl@
2700 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2701   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2702 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
2703 \def\@Glsxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
2704 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2705   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
2706 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2707 \def\@Glsxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2708 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2709   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2710 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
2711 \newcommand*{\glsxtrsetpopts}[1]{%
2712   \renewcommand*{\@glsxtrp@opt}{#1}%
2713 }
```

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

```
2714 \newcommand*{\glossxtrsetpopts}{%
2715   \glsxtrsetpopts{noindex}%
2716 }
```

`\@@glsxtrp`

```
2717 \newrobustcmd*{\@@glsxtrp}[2]{%
    Add scope.
2718   {%
2719     \let\glspostlinkhook\relax
2720     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2721   }%
2722 }
```

`\@glsxtrp`

```
2723 \newrobustcmd*{\@glsxtrp}[2]{%
2724   \ifcsdef{gls#1}%
2725     {%
2726       \@glsxtrp{gls#1}{#2}%
2727     }%
2728     {%
2729       \ifcsdef{glsxtr#1}%
2730         {%
2731           \@glsxtrp{glsxtr#1}{#2}%
2732         }%
2733         {%
2734           \PackageError{glossaries-extra}{‘#1’ not recognised by
2735             \string\glsxtrp}{}%
2736         }%
2737       }%
2738 }
```

`\@Glsxtrp`

```
2739 \newrobustcmd*{\@Glsxtrp}[2]{%
2740   \ifcsdef{Gls#1}%
2741     {%
2742       \@glsxtrp{Gls#1}{#2}%
2743     }%
2744     {%
2745       \ifcsdef{Glsxtr#1}%
2746         {%
2747           \@glsxtrp{Glsxtr#1}{#2}%
2748         }%
2749     }
```

```

2749   {%
2750       \PackageError{glossaries-extra}{‘#1’ not recognised by
2751           \string\Glsxtrp}{}%
2752   }%
2753 }%
2754 }

```

\@GLSxtrp

```

2755 \newrobustcmd*{\@GLSxtrp}[2]{%
2756   \ifcsdef{GLS#1}%
2757   {%
2758       \@@glxtrp{GLS#1}{#2}%
2759   }%
2760   {%
2761       \ifcsdef{GLSxtr#1}%
2762       {%
2763           \@@glxtrp{GLSxtr#1}{#2}%
2764       }%
2765       {%
2766           \PackageError{glossaries-extra}{‘#1’ not recognised by
2767               \string\GLSxtrp}{}%
2768       }%
2769   }%
2770 }

```

\glxtr@entry@p

```

2771 \newrobustcmd*{\glxtr@headentry@p}[2]{%
2772   \gl@ifattribute{#1}{headuc}{true}%
2773   {%
2774       \mfirstucMakeUppercase{\@glx@entry@field{#1}{#2}}%
2775   }%
2776   {%
2777       \@glx@entry@field{#1}{#2}%
2778   }%
2779 }

```

\glxtrp Not robust as it needs to expand somewhat.

```

2780 \ifdef\texorpdfstring
2781 {
2782   \newcommand{\glxtrp}[2]{%
2783     \protect\NoCaseChange
2784     {%
2785         \protect\texorpdfstring
2786         {%
2787             \protect\glxtrifinmark
2788             {%
2789                 \ifcsdef{glxtrhead#1}%
2790                 {%
2791                     {\protect\csuse{glxtrhead#1}{#2}}%

```

```

2792         }%
2793         {%
2794         \glxtr@headentry@p{#2}{#1}%
2795         }%
2796     }%
2797     {%
2798     \@glxtrp{#1}{#2}%
2799     }%
2800 }%
2801 {%
2802 \protect\@gls@entry@field{#2}{#1}%
2803 }%
2804 }%
2805 }
2806 }
2807 {
2808 \newcommand{\glxtrp}[2]{%
2809 \protect\NoCaseChange
2810 {%
2811 \protect\glxtrifinmark
2812 {%
2813 \ifcsdef{glxtrhead#1}%
2814 {%
2815 {\protect\csuse{glxtrhead#1}}%
2816 }%
2817 {%
2818 \glxtr@headentry@p{#2}{#1}%
2819 }%
2820 }%
2821 {%
2822 \@glxtrp{#1}{#2}%
2823 }%
2824 }%
2825 }
2826 }

```

Provide short synonyms for the most common option.

`\glsp`

```
2827 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
2828 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```
2829 \ifdef\texorpdfstring
```

```
2830 {
```

```
2831 \newcommand{\Glsxtrp}[2]{%
```

```

2832 \protect\NoCaseChange
2833 {%
2834 \protect\texorpdfstring
2835 {%
2836 \protect\glxtrifinmark
2837 {%
2838 \ifcsdef{Glsxtrhead#1}%
2839 {%
2840 {\protect\csuse{Glsxtrhead#1}{#2}}%
2841 }%
2842 {%
2843 \protect\@Gls@entry@field{#2}{#1}%
2844 }%
2845 }%
2846 {%
2847 \@Glsxtrp{#1}{#2}%
2848 }%
2849 }%
2850 {%
2851 \protect\@gls@entry@field{#2}{#1}%
2852 }%
2853 }%
2854 }
2855 }
2856 {
2857 \newcommand{\Glsxtrp}[2]{%
2858 \protect\NoCaseChange
2859 {%
2860 \protect\glxtrifinmark
2861 {%
2862 \ifcsdef{Glsxtrhead#1}%
2863 {%
2864 {\protect\csuse{Glsxtrhead#1}}%
2865 }%
2866 {%
2867 \protect\@Gls@entry@field{#2}{#1}%
2868 }%
2869 }%
2870 {%
2871 \@Glsxtrp{#1}{#2}%
2872 }%
2873 }%
2874 }
2875 }

```

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

```

2876 \ifdef\texorpdfstring
2877 {
2878 \newcommand{\GLSxtrp}[2]{%

```

```

2879 \protect\NoCaseChange
2880 {%
2881 \protect\texorpdfstring
2882 {%
2883 \protect\glsxtrifinmark
2884 {%
2885 \ifcsdef{GLSxtr#1}%
2886 {%
2887 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2888 }%
2889 {%
2890 \protect\mfirstucMakeUppercase
2891 {%
2892 \protect\@gls@entry@field{#2}{#1}%
2893 }%
2894 }%
2895 }%
2896 {%
2897 \@GLSxtrp{#1}{#2}%
2898 }%
2899 }%
2900 {%
2901 \protect\@gls@entry@field{#2}{#1}%
2902 }%
2903 }%
2904 }
2905 }
2906 {
2907 \newcommand{\GLSxtrp}[2]{%
2908 \protect\NoCaseChange
2909 {%
2910 \protect\glsxtrifinmark
2911 {%
2912 \ifcsdef{GLSxtr#1}%
2913 {%
2914 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2915 }%
2916 {%
2917 \protect\mfirstucMakeUppercase
2918 {%
2919 \protect\@gls@entry@field{#2}{#1}%
2920 }%
2921 }%
2922 }%
2923 {%
2924 \@GLSxtrp{#1}{#2}%
2925 }%
2926 }%
2927 }

```

2928 }

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.

```
2929 \renewcommand*{\@glsunset}[1]{%
2930   \@glsunset{#1}%
2931   \glsxtrpostunset{#1}%
2932 }%
```

glsxtrpostunset

```
2933 \newcommand*{\glsxtrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2934 \renewcommand*{\@glslocalunset}[1]{%
2935   \@glslocalunset{#1}%
2936   \glsxtrpostlocalunset{#1}%
2937 }%
```

rpostlocalunset

```
2938 \newcommand*{\glsxtrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2939 \renewcommand*{\@glsreset}[1]{%
2940   \@glsreset{#1}%
2941   \glsxtrpostreset{#1}%
2942 }%
```

glsxtrpostreset

```
2943 \newcommand*{\glsxtrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2944 \renewcommand*{\@glslocalreset}[1]{%
2945   \@glslocalreset{#1}%
2946   \glsxtrpostlocalreset{#1}%
2947 }%
```

rpostlocalreset

```
2948 \newcommand*{\glsxtrpostlocalreset}[1]{}
```

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

```
2949 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```


Enable entry counting:

```
2950 \glsenableentrycount
```

Redefine \gls etc:

```
2951 \renewcommand*{\gls}{\cglsl}%
2952 \renewcommand*{\Gls}{\cGls}%
2953 \renewcommand*{\glspl}{\cglspl}%
2954 \renewcommand*{\Glspl}{\cGlspl}%
2955 \renewcommand*{\GLS}{\cGLS}%
2956 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2957 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2958 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2959 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2960 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2961 can't be used with \string\GlsXtrEnableEntryCounting}%
2962 {Use one or other but not both commands}}%
2963 }
```

ycountunsetattr

```
2964 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2965 \@for\@glsxtr@cat:=#1\do
2966 {%
2967 \ifdefempty{\@glsxtr@cat}{}%
2968 {%
2969 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
2970 }%
2971 }%
2972 }
```

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

nableentrycount

```
2973 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
2974 \appto\@newglossaryentry@defcounters{\@newglossaryentry@defcounters}%
```

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

```
2975 \renewcommand*{\gls@defdocnewglossaryentry}{%
2976 \renewcommand*{\newglossaryentry}[2]{%
2977 \PackageError{glossaries}{\string\newglossaryentry\space
2978 may only be used in the preamble when entry counting has
2979 been activated}{If you use \string\glsenableentrycount\space
2980 you must place all entry definitions in the preamble not in
2981 the document environment}%
2982 }%
2983 }%
```

New commands to access new fields:

```

2984 \newcommand*\glsentrycurrcount}[1]{%
2985   \ifcsundef{glo@glsdetoklabel{##1}@currcount}%
2986   {0}{\@gls@entry@field{##1}{currcount}}%
2987 }%
2988 \newcommand*\glsentryprevcount}[1]{%
2989   \ifcsundef{glo@glsdetoklabel{##1}@prevcount}%
2990   {0}{\@gls@entry@field{##1}{prevcount}}%
2991 }%

```

Adjust post unset and reset:

```

2992 \let\@glsxtr@entrycount@org@unset\glsxtrpostunset
2993 \renewcommand*\glsxtrpostunset}[1]{%
2994   \@glsxtr@entrycount@org@unset{##1}%
2995   \@gls@increment@currcount{##1}%
2996 }%
2997 \let\@glsxtr@entrycount@org@localunset\glsxtrpostlocalunset
2998 \renewcommand*\glsxtrpostlocalunset}[1]{%
2999   \@glsxtr@entrycount@org@localunset{##1}%
3000   \@gls@local@increment@currcount{##1}%
3001 }%
3002 \let\@glsxtr@entrycount@org@reset\glsxtrpostreset
3003 \renewcommand*\glsxtrpostreset}[1]{%
3004   \@glsxtr@entrycount@org@reset{##1}%
3005   \csgdef{glo@glsdetoklabel{##1}@currcount}{0}%
3006 }%
3007 \let\@glsxtr@entrycount@org@localreset\glsxtrpostlocalreset
3008 \renewcommand*\glsxtrpostlocalreset}[1]{%
3009   \@glsxtr@entrycount@org@localreset{##1}%
3010   \csdef{glo@glsdetoklabel{##1}@currcount}{0}%
3011 }%

```

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

```

3012 \let\@cgl@s\@cgl@s
3013 \let\@cgl spl\@cgl spl
3014 \let\@cGls\@cGls
3015 \let\@cGlspl\@cGlspl
3016 \let\@cGLS\@cGLS
3017 \let\@cGLSpl\@cGLSpl

```

The rest is as the original definition.

```

3018 \AtEndDocument{\@gls@write@entrycounts}%
3019 \renewcommand*\@gls@entry@count}[2]{%
3020   \csgdef{glo@glsdetoklabel{##1}@prevcount}{##2}%
3021 }%
3022 \let\glsenableentrycount\relax
3023 \renewcommand*\glsenableentryunitcount{%
3024   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space

```

```

3025     can't be used with \string\glsenableentrycount}%
3026     {Use one or other but not both commands}%
3027 }%
3028 }

```

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

```

3029 \newcommand*{\@gls@write@entrycounts}{%
3030   \immediate\write\@auxout
3031   {\string\providecommand*\string\@gls@entry@count}[2]{}%
3032   \count@=0\relax
3033   \forallglsentries{\@glsentry}{%
3034     \glshasattribute{\@glsentry}{entrycount}%
3035     {%
3036       \ifglsused{\@glsentry}%
3037       {%
3038         \immediate\write\@auxout
3039         {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3040       }%
3041     }%
3042     \advance\count@ by \@ne
3043   }%
3044 }%
3045 }%
3046 \ifnum\count@=0
3047   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3048   \MessageBreak with \string\glsenableentrycount\space but the
3049   \MessageBreak attribute 'entrycount' hasn't
3050   \MessageBreak been assigned to any of the defined
3051   \MessageBreak entries}%
3052 \fi
3053 }

```

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

```

3054 \newcommand*{\glstrifcounttrigger}[3]{%
3055   \glshasattribute{#1}{entrycount}%
3056   {%
3057     \ifnum\glsentryprevcount{#1}>\glsgetattribute{#1}{entrycount}\relax
3058     #3%
3059   \else
3060     #2%
3061   \fi
3062 }%
3063 {#3}%
3064 }

```

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

\@@cgl's@

```
3065 \def\@@cgl's@#1#2[#3]{%
3066   \gl'sxtrifcounttrigger{#2}%
3067   {%
3068     \cgl'sformat{#2}{#3}%
3069     \gl'sunset{#2}%
3070   }%
3071   {%
3072     \@gl's@{#1}{#2}[#3]%
3073   }%
3074 }%
```

\@@cgl'spl@

```
3075 \def\@@cgl'spl@#1#2[#3]{%
3076   \gl'sxtrifcounttrigger{#2}%
3077   {%
3078     \cgl'splformat{#2}{#3}%
3079     \gl'sunset{#2}%
3080   }%
3081   {%
3082     \@gl'spl@{#1}{#2}[#3]%
3083   }%
3084 }%
```

\@@cGl's@

```
3085 \def\@@cGl's@#1#2[#3]{%
3086   \gl'sxtrifcounttrigger{#2}%
3087   {%
3088     \cGl'sformat{#2}{#3}%
3089     \gl'sunset{#2}%
3090   }%
3091   {%
3092     \@Gl's@{#1}{#2}[#3]%
3093   }%
3094 }%
```

\@@cGl'spl@

```
3095 \def\@@cGl'spl@#1#2[#3]{%
3096   \gl'sxtrifcounttrigger{#2}%
3097   {%
3098     \cGl'splformat{#2}{#3}%
3099     \gl'sunset{#2}%
3100   }%
3101   {%
3102     \@Gl'spl@{#1}{#2}[#3]%
3103   }%
3104 }%
```

\@@cGLS@

```
3105 \def\@@cGLS@#1#2[#3]{%
3106   \glxtrifcounttrigger{#2}%
3107   {%
3108     \cGLSformat{#2}{#3}%
3109     \glset{#2}%
3110   }%
3111   {%
3112     \@GLS@{#1}{#2}[#3]%
3113   }%
3114 }%
```

\@@cGLSpl@

```
3115 \def\@@cGLSpl@#1#2[#3]{%
3116   \glxtrifcounttrigger{#2}%
3117   {%
3118     \cGLSplformat{#2}{#3}%
3119     \glset{#2}%
3120   }%
3121   {%
3122     \@GLSpl@{#1}{#2}[#3]%
3123   }%
3124 }%
```

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

\@cgl's@

```
3125 \def\@cgl's@#1#2[#3]{\@gl's@{#1}{#2}[#3]}
```

\@cGL's@

```
3126 \def\@cGL's@#1#2[#3]{\@GL's@{#1}{#2}[#3]}
```

\@cgl'spl@

```
3127 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}
```

\@cGL'spl@

```
3128 \def\@cGL'spl@#1#2[#3]{\@GL'spl@{#1}{#2}[#3]}
```

Add all upper case versions not provided by glossaries.

\cGLS

```
3129 \newrobustcmd*{\cGLS}{\@gl'shyp@opt\@cGLS}
```

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

```
3130 \newcommand*{\@cGLS}[2][ ]{%
3131   \new@ifnextchar[{\@cGLS@{#1}{#2}}{\@cGLS@{#1}{#2}[ ]}%
3132 }
```

```

\@cGLS@
3133 \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.
3134 \newcommand*\cGLSformat[2]{%
3135   \expandafter\mfirstucMakeUppercase\expandafter{\cglformat{#1}{#2}}%
3136 }

\cGLSpl
3137 \newrobustcmd*\cGLSpl{\@gls@hyp@opt\@cGLSpl}

\@cGLSpl  Defined the un-starred form. Need to determine if there is a final optional argument
3138 \newcommand*\@cGLSpl[2][{}]{%
3139   \new@ifnextchar[\@cGLSpl@{#1}{#2}}{\@cGLSpl@{#1}{#2}[{}]}%
3140 }

\@cGLSpl@
3141 \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.
3142 \newcommand*\cGLSplformat[2]{%
3143   \expandafter\mfirstucMakeUppercase\expandafter{\cglsplformat{#1}{#2}}%
3144 }

      Modify the trigger formats to check for the regular attribute.

\cglformat
3145 \renewcommand*\cglformat[2]{%
3146   \glsifregular{#1}
3147   {\glsentryfirst{#1}}%
3148   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
3149 }

\cGlsformat
3150 \renewcommand*\cGlsformat[2]{%
3151   \glsifregular{#1}
3152   {\Glsentryfirst{#1}}%
3153   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%
3154 }

\cglsplformat
3155 \renewcommand*\cglsplformat[2]{%
3156   \glsifregular{#1}
3157   {\glsentryfirstplural{#1}}%
3158   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}%#2%
3159 }

```

\cGlsplformat

```
3160 \renewcommand*{\cGlsplformat}[2]{%
3161   \glsifregular{#1}
3162   {\Glsentryfirstplural{#1}}%
3163   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}#2}%
3164 }
```

New code similar to above for unit counting.

defunitcounters

```
3165 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3166   \edef\@glo@countunit{\csuse{\glstr@categoryattr@@\@glo@category @unitcount}}%
3167   \ifdefvoid\@glo@countunit
3168     {}%
3169     {%
3170       \@glstr@ifunitcounter{\@glo@countunit}%
3171       {}%
3172       {\expandafter\@glstr@addunitcounter\expandafter{\@glo@countunit}}%
3173     }%
3174 }
```

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

```
3175 \newcommand*{\@glstr@unitcountlist}{}%
```

@addunitcounter

```
3176 \newcommand*{\@glstr@addunitcounter}[1]{%
3177   \listadd{\@glstr@unitcountlist}{#1}%
3178   \ifcsundef{glstr@theunit@#1}
3179     {%
3180       \ifcsdef{theH#1}%
3181       {\csdef{glstr@theunit@#1}{\csuse{theH#1}}}%
3182       {\csdef{glstr@theunit@#1}{\csuse{the#1}}}%
3183     }%
3184   {}%
3185 }
```

r@ifunitcounter

```
3186 \newcommand*{\@glstr@ifunitcounter}[3]{%
3187   \xifinlist{#1}{\@glstr@unitcountlist}{#2}{#3}%
3188 }
```

urrentunitcount

```
3189 \newcommand*{\@glstr@currentunitcount}[1]{%
3190   glo@\glstetoklabel{#1}@currunit@\glstetattribute{#1}{unitcount}.%
3191   \csuse{glstr@theunit@\glstetattribute{#1}{unitcount}}%
3192 }
```

previousunitcount

```
3193 \newcommand*\@glsxtr@previousunitcount[1]{%
3194   glo@glstdetoklabel{#1}@prevunit@glsggetattribute{#1}{unitcount}.%
3195   \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}%
3196 }
```

t@currunitcount

```
3197 \newcommand*\@gls@increment@currunitcount[1]{%
3198   \glshasattribute{#1}{unitcount}%
3199   {%
3200     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3201     \ifcsundef{\@glsxtr@csname}%
3202     {%
3203       \csgdef{\@glsxtr@csname}{1}%
3204       \listcsxadd
3205         {glo@glstdetoklabel{#1}@unitlist}%
3206         {\glsggetattribute{#1}{unitcount}.%
3207          \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}}%
3208     }%
3209   }%
3210   {%
3211     \csxdef{\@glsxtr@csname}%
3212     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3213   }%
3214 }%
3215 {}%
3216 }
```

t@currunitcount

```
3217 \newcommand*\@gls@local@increment@currunitcount[1]{%
3218   \glshasattribute{#1}{unitcount}%
3219   {%
3220     \edef\@glsxtr@csname{\@glsxtr@currentunitcount{#1}}%
3221     \ifcsundef{\@glsxtr@csname}%
3222     {%
3223       \csdef{\@glsxtr@csname}{1}%
3224       \listcseadd
3225         {glo@glstdetoklabel{#1}@unitlist}%
3226         {\glsggetattribute{#1}{unitcount}.%
3227          \csuse{glsxtr@theunit@glsggetattribute{#1}{unitcount}}}%
3228     }%
3229   }%
3230   {%
3231     \csedef{\@glsxtr@csname}%
3232     {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3233   }%
3234 }%
3235 {}%
3236 }
```


r@currunitcount

```
3237 \newcommand*{\@glxstr@currunitcount}[2]{%
3238   \ifcsundef
3239     {glo@\glstdetoklabel{#1}@currunit@#2}%
3240     {0}%
3241     {\csuse{glo@\glstdetoklabel{#1}@currunit@#2}}%
3242 }%
```

r@prevunitcount

```
3243 \newcommand*{\@glxstr@prevunitcount}[2]{%
3244   \ifcsundef
3245     {glo@\glstdetoklabel{#1}@prevunit@#2}%
3246     {0}%
3247     {\csuse{glo@\glstdetoklabel{#1}@prevunit@#2}}%
3248 }%
```

entryunitcount

```
3249 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3250   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3251   \renewcommand*{\gls@defdocnewglossaryentry}{%
3252     \renewcommand*\newglossaryentry[2]{%
3253       \PackageError{glossaries}{\string\newglossaryentry\space
3254         may only be used in the preamble when entry counting has
3255         been activated}{If you use \string\glsenableentryunitcount\space
3256         you must place all entry definitions in the preamble not in
3257         the document environment}%
3258     }%
3259   }%
```

New commands to access new fields:

```
3260 \newcommand*{\glsentrycurrcount}[1]{%
3261   \@glxstr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3262   \csuse{glxstr@theunit@\glsgetattribute{##1}{unitcount}}}%
3263 }%
3264 \newcommand*{\glsentryprevcount}[1]{%
3265   \@glxstr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}}.%
3266   \csuse{glxstr@theunit@\glsgetattribute{##1}{unitcount}}}%
3267 }%
```

Access total count:

```
3268 \newcommand*{\glsentryprevtotalcount}[1]{%
3269   \ifcsundef{glo@\glstdetoklabel{##1}@prevunittotal}%
3270     {0}%
3271     {%
3272       \number\csuse{glo@\glstdetoklabel{##1}@prevunittotal}
3273     }%
3274 }%
```

Access max value:

```

3275 \newcommand*{\glstentryprevmaxcount}[1]{%
3276   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
3277     {0}%
3278     {%
3279       \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
3280     }%
3281   }%

```

Adjust post unset and reset:

```

3282 \let\@glxstr@entryunitcount@org@unset\glxstrpostunset
3283 \renewcommand*{\glxstrpostunset}[1]{%
3284   \@glxstr@entryunitcount@org@unset{##1}%
3285   \@glst@increment@currunitcount{##1}%
3286 }%
3287 \let\@glxstr@entryunitcount@org@localunset\glxstrpostlocalunset
3288 \renewcommand*{\glxstrpostlocalunset}[1]{%
3289   \@glxstr@entryunitcount@org@localunset{##1}%
3290   \@glst@local@increment@currunitcount{##1}%
3291 }%
3292 \let\@glxstr@entryunitcount@org@reset\glxstrpostreset
3293 \renewcommand*{\glxstrpostreset}[1]{%
3294   \glshasattribute{##1}{unitcount}%
3295   {%
3296     \edef\@glxstr@csname{\@glxstr@currentunitcount{##1}}%
3297     \ifcsundef{\@glxstr@csname}%
3298       {}%
3299       {\csgdef{\@glxstr@csname}{0}}%
3300   }%
3301   {}%
3302 }%
3303 \let\@glxstr@entryunitcount@org@localreset\glxstrpostlocalreset
3304 \renewcommand*{\glxstrpostlocalreset}[1]{%
3305   \@glxstr@entryunitcount@org@localreset{##1}%
3306   \glshasattribute{##1}{unitcount}%
3307   {%
3308     \edef\@glxstr@csname{\@glxstr@currentunitcount{##1}}%
3309     \ifcsundef{\@glxstr@csname}%
3310       {}%
3311       {\csdef{\@glxstr@csname}{0}}%
3312   }%
3313   {}%
3314 }%

```

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

```

3315 \let\@cglst@\@cglst@
3316 \let\@cglstpl@\@cglstpl@
3317 \let\@cGlst@\@cGlst@

```

```

3318 \let\@cGlspl@\@cGlspl@
3319 \let\@cGLS@\@cGLS@
3320 \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3321 \AtEndDocument{\@gls@write@entryunitcounts}%
3322 \renewcommand*{\@gls@entry@unitcount}[3]{%
3323   \csgdef{glo@\glsdetoklabel{##1}@prevunit@##3}{##2}%
3324   \ifcsundef{glo@\glsdetoklabel{##1}@prevunittotal}%
3325   {\csgdef{glo@\glsdetoklabel{##1}@prevunittotal}{##2}}%
3326   {%
3327     \csxdef{glo@\glsdetoklabel{##1}@prevunittotal}{
3328       \number\numexpr\csuse{glo@\glsdetoklabel{##1}@prevunittotal}+##2}%
3329     }%
3330     \ifcsundef{glo@\glsdetoklabel{##1}@prevunitmax}%
3331     {\csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}}%
3332     {%
3333       \ifnum\csuse{glo@\glsdetoklabel{##1}@prevunitmax}<##2
3334       \csgdef{glo@\glsdetoklabel{##1}@prevunitmax}{##2}%
3335     }%
3336   }%
3337 }%
3338 \let\glsenableentryunitcount\relax
3339 \renewcommand*{\glsenableentrycount}{%
3340   \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3341     can't be used with \string\glsenableentryunitcount}%
3342   {Use one or other but not both commands}%
3343 }%
3344 }
3345 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3346 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

3347 \newcommand*{\@gls@write@entryunitcounts@do}[1]{%
3348   \immediate\write\@auxout
3349   {\string\@gls@entry@unitcount
3350     {\@gls@entry}%
3351     {\@glsxtr@currunitcount{\@gls@entry}{#1}%
3352     }%
3353     {#1}}%
3354 }

```

entryunitcounts

```

3355 \newcommand*{\@gls@write@entryunitcounts}{%
3356   \immediate\write\@auxout
3357   {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3358   \count@=0\relax

```

```

3359 \forallglentries{\@glentry}{%
3360   \glshasattribute{\@glentry}{unitcount}%
3361   {%
3362     \ifglused{\@glentry}%
3363     {%
3364       \forlistcsloop
3365         {\@glswrite@entryunitcounts@do}%
3366         {glo@\glsetoklabel{\@glentry}@unitlist}%
3367     }%
3368   }%
3369   \advance\count@ by \@ne
3370 }%
3371 {}%
3372 }%
3373 \ifnum\count@=0
3374   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3375     \MessageBreak with \string\glsenableentryunitcount\space but the
3376     \MessageBreak attribute ‘unitcount’ hasn’t
3377     \MessageBreak been assigned to any of the defined
3378     \MessageBreak entries}%
3379 \fi
3380 }

```

`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.

```

3381 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3382   \glsenableentryunitcount

```

Redefine `\gls` etc:

```

3383   \renewcommand*{\gls}{\cgl}%
3384   \renewcommand*{\Gls}{\cGls}%
3385   \renewcommand*{\glspl}{\cglspl}%
3386   \renewcommand*{\Glspl}{\cGlspl}%
3387   \renewcommand*{\GLS}{\cGLS}%
3388   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3389   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3390   \let\GlsXtrEnableEntryUnitCounting\@glsxtr@setentryunitcountunsetattr
3391   \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3392     \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3393       can’t be used with \string\GlsXtrEnableEntryUnitCounting}%
3394     {Use one or other but not both commands}}%
3395 }

```

`tcountunsetattr`

```

3396 \newcommand*{\@glsxtr@setentryunitcountunsetattr}[3]{%
3397   \@for\@glsxtr@cat:=#1\do
3398   {%
3399     \ifdefempty{\@glsxtr@cat}{}%
3400     {%
3401       \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3402       \glssetcategoryattribute{\@glsxtr@cat}{unitcount}{#3}%
3403     }%
3404   }%
3405 }

```

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`

```

3406 \renewcommand*{\SetGenericNewAcronym}{%
3407   \let\@Gls@entryname\@Gls@acrentryname
3408   \renewcommand{\newacronym}[4][]{%
3409     \ifdefempty{\@glsacronymlists}%
3410     {%
3411       \def\@glo@type{\acronymtype}%
3412       \setkeys{glossentry}{##1}%
3413       \DeclareAcronymList{\@glo@type}%
3414     }%
3415   }%
3416   \glskeylisttok{##1}%
3417   \glslabeltok{##2}%
3418   \glsshorttok{##3}%
3419   \glslongtok{##4}%
3420   \newacronymhook
3421   \protected@edef\@do@newglossaryentry{%
3422     \noexpand\newglossaryentry{\the\glslabeltok}%
3423     {%
3424       type=\acronymtype,%
3425       name={\expandonce{\acronymentry{##2}}},%
3426       sort={\acronymssort{\the\glsshorttok}{\the\glslongtok}},%
3427       text={\the\glsshorttok},%
3428       short={\the\glsshorttok},%
3429       shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3430       long={\the\glslongtok},%
3431       longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3432       category=acronym,

```

```

3433     \GenericAcronymFields,%
3434     \the\glskeylisttok
3435   }%
3436 }%
3437 \do@newglossaryentry
3438 }%
3439 \renewcommand*{\acrfullfmt}[3]{%
3440   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3441 \renewcommand*{\Acrfullfmt}[3]{%
3442   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3443 \renewcommand*{\ACRfullfmt}[3]{%
3444   \glslink[##1]{##2}{%
3445     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3446 \renewcommand*{\acrfullplfmt}[3]{%
3447   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3448 \renewcommand*{\Acrfullplfmt}[3]{%
3449   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3450 \renewcommand*{\ACRfullplfmt}[3]{%
3451   \glslink[##1]{##2}{%
3452     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3453 \renewcommand*{\glstryfull}[1]{\genacrfullformat{##1}{}}}%
3454 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3455 \renewcommand*{\glstryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3456 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3457 }

```

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:

```

3458 \let\@glstr@org@setacronymstyle\setacronymstyle
3459 \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`.

```

3460 \newcommand*{\MakeAcronymsAbbreviations}{%
3461   \renewcommand*{\newacronym}[4][]{%
3462     \glstr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3463   }%
3464   \renewcommand*{\firstacronymfont}[1]{\glstr@firstabbrvfont{##1}}%
3465   \renewcommand*{\acronymfont}[1]{\glstr@abbrvfont{##1}}%
3466   \renewcommand*{\setacronymstyle}[1]{%
3467     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}}
3468     unavailable.
3469     Use \string\setabbreviationstyle\space instead.
3470     The original acronym interface can be restored with
3471     \string\RestoreAcronyms}{}%
3472   }%
3473   \renewcommand*{\newacronymstyle}[1]{%

```

```

3474 \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3475 available unless you restore the original acronym interface with
3476 \string\RestoreAcronyms}%
3477 \@glstr@org@newacronymstyle{##1}%
3478 }%
3479 }

```

Switch acronyms to abbreviations:

```
3480 \MakeAcronymsAbbreviations
```

RestoreAcronyms Restore acronyms to glossaries interface.

```

3481 \newcommand*{\RestoreAcronyms}{%
3482 \SetGenericNewAcronym
3483 \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3484 \renewcommand{\acronymfont}[1]{##1}%
3485 \let\setacronymstyle\@glstr@org@setacronymstyle
3486 \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.

```

3487 \renewcommand*\@gls@link@checkfirsthyper{%
3488 \ifglused{\glslabel}%
3489 {\let\glstrifwasfirstuse\@secondoftwo}
3490 {\let\glstrifwasfirstuse\@firstoftwo}%
3491 \@glstr@org@checkfirsthyper
3492 }
3493 \glssetcategoryattribute{acronym}{regular}{false}%
3494 \setacronymstyle{long-short}%
3495 }

```

\glsacspace Allow the user to customise the maximum value.

```

3496 \renewcommand*{\glsacspace}[1]{%
3497 \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3498 \ifdim\dimen@<\glsacspacemax~\else\space\fi
3499 }

```

\glsacspacemax Value used in the above.

```
3500 \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

```
3501 \newcommand*{\@glstr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```
3502 \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
```

```
3503 \renewcommand*{\makeglossaries}[1] [] {%
3504   \ifx\@glxtr@record@setting\@glxtr@record@setting@only
3505     \PackageError{glossaries-extra}{\string\makeglossaries\space
3506       not permitted\MessageBreak with record=only package option}%
3507     {You may only use \string\makeglossaries\space with
3508       record=off or record=alsoindex options}%
3509   \else
3510     \ifblank{#1}%
3511     {\@glxtr@org@makeglossaries}%
3512     {%
3513       \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
3514         \PackageError{glossaries-extra}{\string\makeglossaries[#1]\space
3515           not permitted\MessageBreak with record=alsoindex package option}%
3516         {You may only use the hybrid \string\makeglossaries[...]\space with
3517           record=off option}%
3518       \else
3519         \edef\@glxtr@reg@glosslist{#1}%
3520         \ifundef{\glswrite}{\newwrite\glswrite}{}%
3521         \protected@write\@auxout{}{\string\providecommand
3522           \string\@glxtr@order[1]{}%
3523           \protected@write\@auxout{}{\string\providecommand
3524             \string\@istfilename[1]{}%
3525             \protected@write\@auxout{}{\string\@istfilename{\istfilename}}%
3526             \protected@write\@auxout{}{\string\@glxtr@order{\glxtr@order}}%
3527             \protected@write\@auxout{}{\string\glxtr@makeglossaries{#1}}%
3528             \write\@auxout{\string\providecommand\string\@glxtr@reference[3]{}%

```

Iterate through each supplied glossary type and activate it.

```
3529   \@for\@glo@type:=#1\do{%
3530     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3531   }%
```

New glossaries must be created before `\makeglossaries`:

```
3532   \renewcommand*\newglossary[4] [] {%
3533     \PackageError{glossaries}{New glossaries
3534       must be created before \string\makeglossaries}{You need
3535       to move \string\makeglossaries\space after all your
3536       \string\newglossary\space commands}}%
```

Any subsequent instances of this command should have no effect

```
3537   \let\@makeglossary\relax
3538   \let\makeglossary\relax
```



```

3539      \renewcommand\makeglossaries[1][]{}%
      Disable all commands that have no effect after \makeglossaries
3540      \@disable@onlypremakeg
      Allow see key:
3541      \let\gls@checkseeallowed\relax
      Adjust \@do@seeglossary. This needs to check for the entries existence.
3542      \renewcommand*{\@do@seeglossary}[2]{%
3543      \glsdoifexists{##1}%
3544      {%
3545      \edef\gls@label{\glsdetoklabel{##1}}%
3546      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3547      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3548      {\@glsxtr@org@doseeglossary{##1}{##2}}%
3549      {%
3550      @@glsxtrwrglossmark
3551      \protected@write\@auxout{}{%
3552      \string\gls@reference
3553      {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
3554      }%
3555      }%
3556      }%
3557      }%
      Adjust \@do@@wrglossary
3558      \let\glsxtr@@do@@wrglossary\@do@@wrglossary
3559      \def\@do@@wrglossary{%
3560      \edef\gls@type{\csname glo@\gls@label @type\endcsname}%
3561      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3562      {\@glsxtr@@do@@wrglossary}%
3563      {\gls@noidxglossary}%
3564      }%
      Suppress warning about no \makeglossaries
3565      \let\warn@nomakeglossaries\relax
3566      \def\warn@noprntglossary{%
3567      \GlossariesWarningNoLine{No \string\printglossary\space
3568      or \string\printglossaries\space
3569      found.^^J(Remove \string\makeglossaries\space if you don't want
3570      any glossaries.)^^JThis document will not have a glossary}%
3571      }%
      Only warn for glossaries not listed.
3572      \renewcommand{\@gls@noref@warn}[1]{%
3573      \edef\gls@type{##1}%
3574      \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3575      {%
3576      \GlossariesExtraWarning{Can't use
3577      \string\printnoidxglossary[type={\@gls@type}]
3578      when '\@gls@type' is listed in the optional argument of

```

```

3579         \string\makeglossaries}%
3580     }%
3581     {%
3582         \GlossariesWarning{Empty glossary for
3583         \string\printnoidxglossary[type={##1}].
3584         Rerun may be required (or you may have forgotten to use
3585         commands like \string\gls)}%
3586     }%
3587 }%

```

Adjust display number list to check for type:

```

3588     \renewcommand*{\glsdisplaynumberlist}[1]{%
3589     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3590     {\@glxtr@idx@displaynumberlist{##1}}%
3591     {\@glxtr@noidx@displaynumberlist{##1}}%
3592 }%

```

Adjust entry list:

```

3593     \renewcommand*{\glsentrynumberlist}[1]{%
3594     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3595     {\@glxtr@idx@entrynumberlist{##1}}%
3596     {\@glxtr@noidx@entrynumberlist{##1}}%
3597 }%

```

Adjust number list loop

```

3598     \renewcommand*{\glsnumberlistloop}[2]{%
3599     \expandafter\DTLifinlist\expandafter{##1}{\@glxtr@reg@glosslist}%
3600     {%
3601         \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3602         not available for glossary ‘##1’}{}%
3603     }%
3604     {\@glxtr@noidx@numberlistloop{##1}{##2}}%
3605 }%

```

Only sanitize sort for normal indexing glossaries.

```

3606     \renewcommand*{\glsprestandardsort}[3]{%
3607     \expandafter\DTLifinlist\expandafter{##2}{\@glxtr@reg@glosslist}%
3608     {%
3609         \glsdosanitizesort
3610     }%
3611     {%
3612         \ifglssanitizesort
3613         \@gls@noidx@sanitizesort
3614         \else
3615         \@gls@noidx@nosanitizesort
3616         \fi
3617     }%
3618 }%

```

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

```

3619 \renewcommand*\new@glossaryentry[2]{%
3620 \PackageError{glossaries-extra}{Glossary entries must be defined
3621 in the preamble\MessageBreak when you use the optional argument
3622 of \string\makeglossaries}{Either move your definitions to the
3623 preamble or don't use the optional argument of
3624 \string\makeglossaries}%
3625 }%

```

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).

```

3626 \let\@glo@assign@sortkey\@glstr@mixed@assign@sortkey
3627 \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3628 \expandafter\@glstr@gettype\expandafter,\@glstr@printglossopts,%
3629 type=\glsdefaulttype,\@end@glstr@gettype
3630 \def\@glo@sorttype{\@glo@default@sorttype}%
3631 }%

```

Check automake setting:

```

3632 \ifglsautomake
3633 \renewcommand*\@gls@doautomake{%
3634 \@for\@gls@type:=\@glstr@reg@glosslist\do{%
3635 \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3636 }%
3637 }%
3638 \fi

```

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

```

3639 \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3640 \fi
3641 }%
3642 \fi
3643 }

```

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

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

```

3644 \newcommand{\@glstr@orgprintglossary}[2]{%
3645 \def\@glo@type{\glsdefaulttype}%

```

Add check here.

```

3646 \def\glossarytitle{%
3647 \ifcsdef{\glo@type@\@glo@type @title}%
3648 {\csuse{\glo@type@\@glo@type @title}}%
3649 {\glossaryname}}%
3650 \def\glossarytoctitle{\glossarytitle}%

```

```

3651 \let\org@glossarytitle\glossarytitle
3652 \def\@glossarystyle{%
3653   \ifx\@glossary@default@style\relax
3654     \GlossariesWarning{No default glossary style provided \MessageBreak
3655       for the glossary '\@glo@type'. \MessageBreak
3656       Using deprecated fallback. \MessageBreak
3657       To fix this set the style with \MessageBreak
3658       \string\setglossarystyle\space or use the \MessageBreak
3659       style key=value option}%
3660   \fi
3661 }%
3662 \def\gls@dotoc@title{\gls@settoc@title{\@glo@type}}%
3663 \let\org@glossaryentrynumbers\glossaryentrynumbers
3664 \bgroup
3665   \@printgloss@setsort
3666   \setkeys{printgloss}{#1}%
3667   \ifx\glossarytitle\org@glossarytitle
3668   \else
3669     \cslet{\@glo@type@\@glo@type @title}{\glossarytitle}%
3670   \fi
3671   \let\currentglossary\@glo@type
3672   \let\org@glossaryentrynumbers\glossaryentrynumbers
3673   \let\glsnonextpages\@glsnonextpages
3674   \let\glsnextpages\@glsnextpages

3675   \glsxtractivatenopost
3676   \gls@dotoc@title
3677   \@glossarystyle
3678   \let\gls@org@glossaryentryfield\glossentry
3679   \let\gls@org@glossarysubentryfield\subglossentry
3680   \renewcommand{\glossentry}[1]{%
3681     \xdef\glscurrententrylabel{\glsdetoklabel{##1}}%
3682     \gls@org@glossaryentryfield{##1}%
3683   }%
3684   \renewcommand{\subglossentry}[2]{%
3685     \xdef\glscurrententrylabel{\glsdetoklabel{##2}}%
3686     \gls@org@glossarysubentryfield{##1}{##2}%
3687   }%
3688   \@gls@preglossaryhook
3689   #2%
3690 \egroup
3691 \global\let\glossaryentrynumbers\org@glossaryentrynumbers
3692 \global\let\warn@noprntglossary\relax
3693 }

```

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

```

3694 \newcommand*{\glsxtractivatenopost}{%
3695   \let\nopostdesc\@nopostdesc
3696   \let\glsxtrnopostpunc\@glsxtr@nopostpunc
3697 }

```

lsxtrnopostpunc

```
3698 \newrobustcmd*{\glxtrnopostpunc}{}
```

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

```
3699 \newcommand{\@glxtr@nopostpunc}{%
3700 \let\@glxtr@org@postdescription\glspostdescription
3701 \ifglsnopostdot
3702 \renewcommand{\glspostdescription}{%
3703 \glsnopostdottrue
3704 \let\glspostdescription\@glxtr@org@postdescription
3705 \glxtrpostdescription}%
3706 \else
3707 \renewcommand{\glspostdescription}{%
3708 \let\glspostdescription\@glxtr@org@postdescription
3709 \glxtrpostdescription}%
3710 \fi
3711 \glsnopostdotfalse
3712 }
```

\@printglossary Redefine.

```
3713 \renewcommand{\@printglossary}[2]{%
3714 \def\@glxtr@printglossopts{#1}%
3715 \@glxtr@orgprintglossary{#1}{#2}%
3716 }
```

Add a key that switches off the entry targets:

```
3717 \define@choicekey{printgloss}{target}[\val\nr]{true,false}[true]{%
3718 \ifcase\nr
3719 \let\@glstarget\glsdohypertarget
3720 \else
3721 \let\@glstarget\@secondoftwo
3722 \fi
3723 }
```

hypernameprefix

```
3724 \newcommand{\@glxtrhypernameprefix}{}
```

New to v1.20:

```
3725 \define@key{printgloss}{targetnameprefix}{%
3726 \renewcommand{\@glxtrhypernameprefix}{#1}%
3727 }
```

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

```
3728 \let\@glxtr@org@glsdohypertarget\glsdohypertarget
3729 \renewcommand{\glsdohypertarget}[2]{%
3730 \@glxtr@org@glsdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3731 }
```

@makeglossaries For the benefit of makeglossaries

```
3732 \newcommand*{\glxtr@makeglossaries}[1]{}
```

@glxtr@gettype Get just the type.

```
3733 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
```

```
3734 \def\@glo@type{#2}%
```

```
3735 }
```

@assign@sortkey Assign the sort key.

```
3736 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
```

```
3737 \edef\@glo@type{\@glo@type}%
```

```
3738 \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
```

```
3739 {%
```

```
3740 \@glo@no@assign@sortkey{#1}%
```

```
3741 }%
```

```
3742 {%
```

```
3743 \@glo@assign@sortkey{#1}%
```

```
3744 }%
```

```
3745 }%
```

Display number list for the regular version:

splaynumberlist

```
3746 \let\@glxtr@idx@displaynumberlist\glsdisplaynumberlist
```

Display number list for the “noidx” version:

splaynumberlist

```
3747 \newcommand*{\@glxtr@noidx@displaynumberlist}[1]{%
```

```
3748 \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
```

```
3749 \ifdef\@gls@loclist
```

```
3750 {%
```

```
3751 \def\@gls@noidxloclist@sep{%
```

```
3752 \def\@gls@noidxloclist@sep{%
```

```
3753 \def\@gls@noidxloclist@sep{%
```

```
3754 \glsnumlistsep
```

```
3755 }%
```

```
3756 \def\@gls@noidxloclist@finalsep{\glsnumlistlastsep}%
```

```
3757 }%
```

```
3758 }%
```

```
3759 \def\@gls@noidxloclist@finalsep{}%
```

```
3760 \def\@gls@noidxloclist@prev{}%
```

```
3761 \forlistloop{\glsnoidxdisplayloclisthandler}{\@gls@loclist}%
```

```
3762 \@gls@noidxloclist@finalsep
```

```
3763 \@gls@noidxloclist@prev
```

```
3764 }%
```

```
3765 {%
```

```
3766 \glxtrundeftag
```

```
3767 \glsdoifexists{#1}%
```

```

3768   {%
3769       \GlossariesWarning{Missing location list for ‘#1’. Either
3770           a rerun is required or you haven’t referenced the entry.}%
3771   }%
3772 }%
3773 }%
3774

```

And for the number list loop:

@numberlistloop

```

3775 \newcommand*{\@glsxtr@noidx@numberlistloop}[3]{%
3776   \letcs{\@gls@loclist}{glo@glstoklabel{#1}@loclist}%
3777   \let\@gls@org@glsnoidxdisplayloc\glsnoidxdisplayloc
3778   \let\@gls@org@glssseeformat\glssseeformat
3779   \let\glsnoidxdisplayloc#2\relax
3780   \let\glssseeformat#3\relax
3781   \ifdef\@gls@loclist
3782   {%
3783       \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3784   }%
3785   {%
3786       \glxtrundeftag
3787       \glsoifexists{#1}%
3788       {%
3789           \GlossariesWarning{Missing location list for ‘##1’. Either
3790               a rerun is required or you haven’t referenced the entry.}%
3791       }%
3792   }%
3793   \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3794   \let\glssseeformat\@gls@org@glssseeformat
3795 }%

```

Same for entry number list.

entrynumberlist

```

3796 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3797   \letcs{\@gls@loclist}{glo@glstoklabel{#1}@loclist}%
3798   \ifdef\@gls@loclist
3799   {%
3800       \glsnoidxloclist{\@gls@loclist}%
3801   }%
3802   {%
3803       \glxtrundeftag
3804       \glsoifexists{#1}%
3805       {%
3806           \GlossariesWarning{Missing location list for ‘#1’. Either
3807               a rerun is required or you haven’t referenced the entry.}%
3808       }%

```

```

3809 }%
3810 }%

```

entrynumberlist

```

3811 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3812 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3813   \protected@edef\@glsxtr@titlelabel{#1}%
3814   \ifdefvoid\@glsxtr@titlelabel
3815   {%
3816     {%
3817       \protected@edef\@glsxtr@titlelabel{\csuse{glsxtr@grouptitle@#1}}%
3818     }%
3819     \ifdefvoid{\@glsxtr@titlelabel}%
3820     {%
3821       \DTLifint{#1}%
3822       {%
3823         \ifnum#1<256\relax
3824           \edef#2{\char#1\relax}%
3825         \else
3826           \edef#2{#1}%
3827         \fi
3828       }%
3829       {%
3830         \ifcsundef{#1groupname}%
3831         {\def#2{#1}}%
3832         {\letcs#2{#1groupname}}%
3833       }%
3834     }%
3835     {%
3836       \let#2\@glsxtr@titlelabel
3837     }%
3838 }

```

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

```

3839 \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.

```

3840 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3841   \protected@edef\@glsxtr@titlelabel{glsxtr@grouptitle@#1}%
3842   \@onelevel@sanitize\@glsxtr@titlelabel
3843   \ifcsdef{\@glsxtr@titlelabel}
3844   {\letcs{#2}{\@glsxtr@titlelabel}}%
3845   {\glsxtr@org@getgrouptitle{#1}{#2}}%
3846 }
3847 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```


trsetgrouptitle Sets the title for the given group label.

```
3848 \newcommand{\glstrsetgrouptitle}[2]{%
3849   \protected@edef\glstr@titlelabel{\glstr@grouptitle@#1}%
3850   \@onelevel@sanitize\glstr@titlelabel
3851   \csxdef{\glstr@titlelabel}{#2}%
3852 }
```

\glsnavigation Redefine to use new user-level command.

```
3853 \renewcommand*{\glsnavigation}{%
3854   \def\@gls@between{}%
3855   \ifcsundef{\gls@hypergroup@list@\gls@type}%
3856   {%
3857     \def\@gls@list{}%
3858   }%
3859   {%
3860     \expandafter\let\expandafter\@gls@list
3861       \csname @gls@hypergroup@list@\gls@type\endcsname
3862   }%
3863   \@for\@gls@tmp:=\@gls@list\do{%
3864     \@gls@between
3865     \glstrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3866     \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3867     \let\@gls@between\glshypernavsep
3868   }%
3869 }
```

@noidx@glossary

```
3870 \renewcommand*{\@print@noidx@glossary}{%
3871   \ifcsdef{\glsref@\gls@type}%
3872   {%
3873     \ifcsdef{\gls@sortmacro@\gls@sorttype}%
3874     {%
3875       \csuse{\gls@sortmacro@\gls@sorttype}{\@gls@type}%
3876     }%
3877     {%
3878       \PackageError{glossaries}{Unknown sort handler ‘\@gls@sorttype’}{}%
3879     }%
3880     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3881     \glossarypreamble
```

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

```
3882   \def\@gls@currentlettergroup{}%
3883   \begin{theglossary}%
3884     \glossaryheader
3885     \glsresetentrylist
3886     \forlistcsloop{\@gls@noidx@do}{\glsref@\gls@type}%
3887     \end{theglossary}%
3888   \glossarypostamble
```

```

3889 }%
3890 {%

```

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

```

3891 \glstrifemptyglossary{\@glo@type}%
3892 {}%
3893 {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
3894 \@gls@noref@warn{\@glo@type}%
3895 }%
3896 }

```

noidxdisplayloc Patch to check for range formations.

```

3897 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3898 \setentrycounter[#1]{#2}%
3899 \@glstr@display@loc#3\empty\end@glstr@display@loc{#4}%
3900 }

```

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

```

3901 \def\@glstr@display@loc#1#2\end@glstr@display@loc#3{%
3902 \ifx#1(\relax
3903 \glstrdisplaystartloc{#2}{#3}%
3904 \else
3905 \ifx#1)\relax
3906 \glstrdisplayendloc{#2}{#3}%
3907 \else
3908 \glstrdisplaysingleloc{#1#2}{#3}%
3909 \fi
3910 \fi
3911 }

```

isplaysingleloc Single location.

```

3912 \newcommand*{\glstrdisplaysingleloc}[2]{%
3913 \csuse{#1}{#2}%
3914 }

```

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

displaystartloc Start of a location range.

```

3915 \newcommand*{\glstrdisplaystartloc}[2]{%
3916 \edef\glstrlocrangefmt{#1}%
3917 \ifx\glstrlocrangefmt\empty
3918 \def\glstrlocrangefmt{glsnumberformat}%
3919 \fi
3920 \expandafter\glstrdisplaysingleloc
3921 \expandafter{\glstrlocrangefmt}{#2}%
3922 }

```

trdisplayendloc End of a location range.

```
3923 \newcommand*{\glxtrdisplayendloc}[2]{%
3924   \edef\@glxtr@tmp{#1}%
3925   \ifdefempty{\@glxtr@tmp}{\def\@glxtr@tmp{glsnumberformat}}{}%
3926   \ifx\glxtrlocrangefmt\@glxtr@tmp
3927   \else
3928     \GlossariesExtraWarning{Mismatched end location range
3929       (start=\glxtrlocrangefmt, end=\@glxtr@tmp)}%
3930   \fi
3931   \expandafter\glxtrdisplayendloohook\expandafter{\@glxtr@tmp}{#2}%
3932   \expandafter\glxtrdisplaysingleloc
3933   \expandafter{\glxtrlocrangefmt}{#2}%
3934   \def\glxtrlocrangefmt{}%
3935 }
```

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

```
3936 \newcommand*{\glxtrdisplayendloohook}[2]{}
```

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

```
3937 \newcommand*{\glxtrlocrangefmt}{}%
```

ls@removespaces Redefine to allow adjustments to location hyperlink.

```
3938 \def\@gls@removespaces#1 #2\@nil{%
3939   \toks@=\expandafter{\the\toks@#1}%
3940   \ifx\@#2\@nil
3941     \edef\x{\the\toks@}%
3942     \ifx\x\empty
3943     \else
3944       \glxtrlocationhyperlink{\glstrycounter}{\@glo@counterprefix}{\the\toks@}%
3945     \fi
3946   \else
3947     \@gls@ReturnAfterFi{%
3948       \@gls@removespaces#2\@nil
3949     }%
3950   \fi
3951 }
```

cationhyperlink

```
3952 \newcommand*{\glxtrlocationhyperlink}[3]{%
3953   \ifdefvoid\glxtrsupplocationurl
3954   {%
3955     \glxtrhyperlink{#1#2#3}{#3}%
3956   }%
3957   {%
3958     \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
3959   }%
3960 }
```

supphypernumber

```
3961 \newcommand*{\glxtrsupphypernumber}[1]{%
3962 {%
3963   \glshasattribute{\glscurrententrylabel}{externallocation}%
3964   {%
3965     \def\glxtrsupplocationurl{%
3966       \glsggetattribute{\glscurrententrylabel}{externallocation}}%
3967   }%
3968   {%
3969     \def\glxtrsupplocationurl{}%
3970   }%
3971   \glshypernumber{#1}%
3972 }%
3973 }
```

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

@print@glossary

```
3974 \renewcommand{\@print@glossary}{%
3975   \makeatletter
3976   \@input@{\jobname.\csname @glo@type @in\endcsname}%
3977   \IfFileExists{\jobname.\csname @glo@type @in\endcsname}%
3978   {%
3979     {\glxtrNoGlossaryWarning{\@glo@type}}%
3980     \ifglxindy
3981       \ifcsundef{@xdy@\@glo@type @language}%
3982       {%
3983         \edef\@do@auxoutstuff{%
3984           \noexpand\AtEndDocument{%
3985             \noexpand\immediate\noexpand\write\@auxout{%
3986               \string\providecommand\string\@xdylanguage[2]{}}%
3987             \noexpand\immediate\noexpand\write\@auxout{%
3988               \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
3989           }%
3990         }%
3991       }%
3992     {%
3993       \edef\@do@auxoutstuff{%
3994         \noexpand\AtEndDocument{%
3995           \noexpand\immediate\noexpand\write\@auxout{%
3996             \string\providecommand\string\@xdylanguage[2]{}}%
3997           \noexpand\immediate\noexpand\write\@auxout{%
3998             \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
3999               @language\endcsname}}%
4000         }%
4001       }%
4002     }%
4003     \@do@auxoutstuff
}
```

```

4004 \edef\@do@auxoutstuff{%
4005 \noexpand\AtEndDocument{%
4006 \noexpand\immediate\noexpand\write\@auxout{%
4007 \string\providecommand\string\@gls@codepage[2]{}}%
4008 \noexpand\immediate\noexpand\write\@auxout{%
4009 \string\@gls@codepage{\@glo@type}{\gls@codepage}}%
4010 }%
4011 }%
4012 \@do@auxoutstuff
4013 \fi
4014 \renewcommand*{\@warn@nomakeglossaries}{%
4015 \GlossariesWarningNoLine{\string\makeglossaries\space
4016 hasn't been used,^^Jthe glossaries will not be updated}%
4017 }%
4018 }

```

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

`\GlsWarningHead` Header message.

```

4019 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4020 This document is incomplete. The external file associated with
4021 the glossary '#1' (which should be called \texttt{#2})
4022 hasn't been created.%
4023 }

```

`\GlsWarningEmptyStart` No entries have been added to the glossary.

```

4024 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4025 This has probably happened because there are no entries defined
4026 in this glossary.%
4027 }

```

`\GlsWarningEmptyMain` The default “main” glossary is empty.

```

4028 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4029 If you don't want this glossary,
4030 add \texttt{nomain} to your package option list when you load
4031 \texttt{glossaries-extra.sty}. For example:%
4032 }

```

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

```

4033 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4034 Did you forget to use \texttt{type=#1} when you defined your
4035 entries? If you tried to load entries into this glossary with
4036 \texttt{\string\loadglsentries} did you remember to use
4037 \texttt{[#1]} as the optional argument? If you did, check that
4038 the definitions in the file you loaded all had the type set
4039 to \texttt{\string\glsdefaulttype}.%
4040 }

```

`\GlsWarningCheckFile` Advisory message to check the file contents.

```

4041 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4042   Check the contents of the file \texttt{#1}. If
4043   it's empty, that means you haven't indexed any of your entries in this
4044   glossary (using commands like \texttt{\string\gls} or
4045   \texttt{\string\glsadd}) so this list can't be generated.
4046   If the file isn't empty, the document build process hasn't been
4047   completed.%
4048 }

```

WarningAutoMake Message when automake option has been used.

```

4049 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4050   You may need to rerun \LaTeX. If you already have, it may be that
4051   \TeX's shell escape doesn't allow you to run
4052   \ifglxindy xindy\else makeindex\fi. Check the
4053   transcript file \texttt{\jobname.log}. If the shell escape is
4054   disabled, try one of the following:
4055
4056   \begin{itemize}
4057     \item Run the external (Lua) application:
4058
4059         \texttt{makeglossaries-lite.lua \string\jobname\string}
4060
4061     \item Run the external (Perl) application:
4062
4063         \texttt{makeglossaries \string\jobname\string}
4064   \end{itemize}
4065
4066   Then rerun \LaTeX\ on this document.
4067   \GlossariesExtraWarning{Rerun required to build the
4068   glossary ‘#1’ or check TeX's shell escape allows
4069   you to run \ifglxindy xindy\else makeindex\fi}%
4070 }

```

WarningMismatch Mismatching \makenoidxglossaries.

```

4071 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
4072   You need to either replace \texttt{\string\makenoidxglossaries}
4073   with \texttt{\string\makeglossaries} or replace
4074   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4075   \texttt{\string\printnoidxglossary}
4076   (or \texttt{\string\printnoidxglossaries}) and then rebuild
4077   this document.%
4078 }

```

WarningBuildInfo Build advice.

```

4079 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4080   Try one of the following:
4081   \begin{itemize}
4082     \item Add \texttt{automake} to your package option list when you load

```

```

4083         \texttt{glossaries-extra.sty}. For example:
4084
4085         \texttt{\string\usepackage[automake]%
4086             \glspopenbrace glossaries-extra\glsclosebrace}
4087
4088     \item Run the external (Lua) application:
4089
4090         \texttt{makeglossaries-lite.lua \string"\jobname\string"}
4091
4092     \item Run the external (Perl) application:
4093
4094         \texttt{makeglossaries \string"\jobname\string"}
4095 \end{itemize}
4096
4097 Then rerun \LaTeX\ on this document.%
4098 }

```

`oGlsWarningTail` Final paragraph.

```

4099 \newcommand{\GlsXtrNoGlsWarningTail}{%
4100 This message will be removed once the problem has been fixed.%
4101 }

```

`GlsWarningNoOut` No out file created. Build advice.

```

4102 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4103 The file \texttt{#1} doesn't exist. This most likely means you haven't used
4104 \texttt{\string\makeglossaries} or you have used
4105 \texttt{\string\nofiles}. If this is just a draft version of the
4106 document, you can suppress this message using the
4107 \texttt{\nomissingglstext} package option.%
4108 }

```

`glossarywarning`

```

4109 \newcommand*{@@glstxr@defaultnoglossarywarning}[1]{%
4110 \glossarysection[\glossarytoctitle]{\glossarytitle}
4111 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glotype@{@glo@type @in\endcsname}
4112 \par
4113 \glstxrifemptyglossary{#1}%
4114 {%
4115 \GlsXtrNoGlsWarningEmptyStart\space
4116 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4117 \medskip
4118 \noindent\texttt{\string\usepackage[nomain\ifglstacronym ,acronym\fi]%
4119 \glspopenbrace glossaries-extra\glsclosebrace}
4120 \medskip
4121 }%
4122 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}}%
4123 }%
4124 {%
4125 \IfFileExists{\jobname.\csname @glotype@{@glo@type @out\endcsname}

```

```

4126 {%
4127   \GlsXtrNoGlsWarningCheckFile
4128     {\jobname.\csname @glotype@\@glo@type @out\endcsname}
4129
4130   \ifglautomake
4131
4132     \GlsXtrNoGlsWarningAutoMake{#1}
4133
4134   \else
4135
4136     \ifthenelse{\equal{#1}{main}}{%
4137       {%
4138         \GlsXtrNoGlsWarningEmptyMain\par
4139         \medskip
4140         \noindent\texttt{\string\usepackage[nomain]%
4141           \glsopenbrace glossaries-extra\glsclosebrace}
4142         \medskip
4143       }%
4144     }%
4145
4146     \ifdefequal\makeglossaries\@no@makeglossaries
4147     {%
4148       \GlsXtrNoGlsWarningMisMatch
4149     }%
4150     {%
4151       \GlsXtrNoGlsWarningBuildInfo
4152     }%
4153   \fi
4154 }%
4155 {%
4156   \GlsXtrNoGlsWarningNoOut
4157   {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4158 }%
4159 }%
4160 \par
4161 \GlsXtrNoGlsWarningTail
4162 }

```

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

glxtrresourcefile 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.

```

4163 \newcommand*\glxtrresourcefile[2] [] {%

```

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

```

4164   \disable@keys{glossaries-extra.sty}{record}%
4165   \glxtr@writefields
4166   \protected@write\auxout{\glxtrresourceinit}{\string\glxtr@resource{#1}{#2}}%
4167   \let\@glxtr@org@see@noindex\@gls@see@noindex
4168   \let\@gls@see@noindex\relax

```



```

4169 \IfFileExists{#2.glstex}%
4170 {%

```

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

```

4171 \edef\@bibgls@restoreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4172 \makeatletter
4173 \@input{#2.glstex}%
4174 \@bibgls@restoreat
4175 }%
4176 {%
4177 \GlossariesExtraWarning{No file '#2.glstex'}%
4178 }%
4179 \let\@gls@see@noindex\@glsxtr@org@see@noindex
4180 }
4181 \@onlypreamble\glsxtrresourcefile

```

trresourcecount

```

4182 \newcount\glsxtrresourcecount

```

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

```

4183 \newcommand*{\GlsXtrLoadResources}[1][{}]{%
4184 \ifnum\glsxtrresourcecount=0\relax
4185 \glsxtrresourcefile[#1]{\jobname}%
4186 \else
4187 \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
4188 \fi
4189 \advance\glsxtrresourcecount by 1\relax
4190 }

```

glsxtr@resource

```

4191 \newcommand*{\glsxtr@resource}[2]{%

```

\glsxtr@fields

```

4192 \newcommand*{\glsxtr@fields}[1]{%

```

xtr@texencoding

```

4193 \newcommand*{\glsxtr@texencoding}[1]{%

```

\glsxtr@langtag

```

4194 \newcommand*{\glsxtr@langtag}[1]{%

```

@pluralsuffixes

```

4195 \newcommand*{\glsxtr@pluralsuffixes}[4]{%

```

tr@shortcutsval

```

4196 \newcommand*{\glsxtr@shortcutsval}[1]{%

```

gsxtr@linkprefix

```
4197 \newcommand*{\gsxtr@linkprefix}[1]{}
```

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

```
4198 \newcommand*{\gsxtr@writefields}{%
```

```
4199   \protected@write\@auxout{}%
```

```
4200   {\string\providecommand*{\string\gsxtr@fields}[1]{}}%
```

```
4201   \protected@write\@auxout{}%
```

```
4202   {\string\providecommand*{\string\gsxtr@resource}[2]{}}%
```

```
4203   \protected@write\@auxout{}%
```

```
4204   {\string\providecommand*{\string\gsxtr@pluralsuffixes}[4]{}}%
```

```
4205   \protected@write\@auxout{}%
```

```
4206   {\string\providecommand*{\string\gsxtr@shortcutsval}[1]{}}%
```

```
4207   \protected@write\@auxout{}%
```

```
4208   {\string\providecommand*{\string\gsxtr@linkprefix}[1]{}}%
```

```
4209   \protected@write\@auxout{}{\string\gsxtr@fields{\@gls@keymap}}%
```

```
4210   \protected@write\@auxout{}%
```

```
4211   {\string\providecommand*{\string\gsxtr@record}[5]{}}%
```

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

```
4212 \ifdef\CurrentTrackedLanguageTag
```

```
4213 {%
```

```
4214   \protected@write\@auxout{}%
```

```
4215   {\string\gsxtr@langtag{\CurrentTrackedLanguageTag}}%
```

```
4216 }%
```

```
4217 {}%
```

```
4218 \protected@write\@auxout{}{\string\gsxtr@pluralsuffixes
```

```
4219   {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
```

```
4220   {\gsxtrabbrvpluralsuffix}}%
```

```
4221 \ifdef\inputencodingname
```

```
4222 {%
```

```
4223   \protected@write\@auxout{}{\string\gsxtr@texencoding{\inputencodingname}}%
```

```
4224 }%
```

```
4225 {%
```

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.)

```
4226   \@ifpackageloaded{fontspec}%
```

```
4227   {\protected@write\@auxout{}{\string\gsxtr@texencoding{utf8}}}%
```

```
4228   {}%
```

```
4229 }%
```

```
4230 \protected@write\@auxout{}{\string\gsxtr@shortcutsval{\@gsxtr@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.

```
4231 \AtBeginDocument
```

```

4232   {\protected@write\@auxout{}\string\glstr@linkprefix{\glolinkprefix}}}%
4233   \let\glstr@writefields\relax

```

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

```

4234   \ifglstr@automake
4235     \IfFileExists{\jobname.aux}%
4236     {\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.

```

4237     \ifx\@glstr@doautomake\@glstr@doautomake@err
4238       \let\@glstr@doautomake\relax
4239     \fi
4240   \fi
4241 }

```

do@automake@err

```

4242 \newcommand*{\@glstr@doautomake@err}{%
4243   \PackageError{glossaries}{You must use
4244     \string\makeglossaries\space with automake=true}
4245   {%
4246     Either remove the automake=true setting or
4247     add \string\makeglossaries\space to your document preamble.%
4248   }%
4249 }

```

Allow locations specific to a particular counter to be recorded.

\glstr@record

```

4250 \newcommand*{\glstr@record}[5]{}

```

r@counterrecord Aux file command.

```

4251 \newcommand*{\glstr@counterrecord}[3]{%
4252   \glstrfieldlistgadd{#1}{record.#2}{#3}%
4253 }

```

unterrecordhook Hook used by \@glstr@dorecord.

```

4254 \newcommand*{\@glstr@counterrecordhook}{}

```

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

```

4255 \newcommand*{\GlsXtrRecordCounter}[1]{%
4256   \@glstr@recordcounter{#1}%
4257 }
4258 \@onlypreamble\GlsXtrRecordCounter

```

docounterrecord

```

4259 \newcommand*{\@glstr@docounterrecord}[1]{%

```

```

4260 \protected@write\@auxout{}\string\glxtr@counterrecord
4261 {\@glslabel}{#1}{\csuse{the#1}}}%
4262 }

```

`\glxtrglossentry` 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.

```

4263 \newcommand*\glxtrglossentry}[1]{%
4264   \glxtrtitleorpdforheading
4265   {\@glxtrglossentry{#1}}%
4266   {\glsentryname{#1}}%
4267   {\glxtrheadname{#1}}%
4268 }

```

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

```

4269 \newrobustcmd*\@glxtrglossentry}[1]{%
4270   \glxtrtitleorpdforheading
4271   {%
4272     \glsdofexists{#1}%
4273     {%
4274       \begingroup
4275         \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4276         \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4277         \ifglshasparent{#1}%
4278         {\glssubentryitem{#1}}%
4279         {\glsentryitem{#1}}%
4280         \glstarget{#1}{\glossentryname{#1}}%
4281       \endgroup
4282     }%
4283   }%
4284   {\glsentryname{#1}}%
4285   {\glxtrheadname{#1}}%
4286 }

```

`\glossentryother` As `\glxtrglossentry` but uses a different field. First argument is command to use in the header. The second argument is the entry's label. The third argument is the internal field label. This needs to be expandable in case it occurs in a sectioning command so it can't have an optional argument.

```

4287 \newcommand*\glxtrglossentryother}[3]{%
4288   \ifstrepty{#1}%
4289   {%
4290     \ifcsdef{glxtrhead#3}%
4291     {%
4292       \glxtrtitleorpdforheading

```

```

4293     {\@glxtrglossentryother{#2}{#3}{#1}}%
4294     {\@gls@entry@field{#2}{#3}}%
4295     {\csuse{glxtrhead#3}{#2}}%
4296 }%
4297 {%
4298     \glxtrtitleorpdforheading
4299     {\@glxtrglossentryother{#2}{#3}{#1}}%
4300     {\@gls@entry@field{#2}{#3}}%
4301     {\@gls@entry@field{\NoCaseChange{#2}}{#3}}%
4302 }%
4303 }%
4304 {%
4305     \glxtrtitleorpdforheading
4306     {\@glxtrglossentryother{#2}{#3}{#1}}%
4307     {\@gls@entry@field{#2}{#3}}%
4308     {#1}}%
4309 }%
4310 }

```

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

```

4311 \newrobustcmd*{\@glxtrglossentryother}[3]{%
4312     \glxtrtitleorpdforheading
4313     {%
4314         \glsdoifexists{#1}%
4315         {%
4316             \begingroup
4317                 \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4318                 \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4319                 \ifglshasparent{#1}%
4320                 {\glssubentryitem{#1}}%
4321                 {\glsentryitem{#1}}%
4322                 \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4323             \endgroup
4324         }%
4325     }%
4326     {\@gls@entry@field{#1}{#2}}%
4327     {#3}%
4328 }

```

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

```

4329 \newcommand*{\printunsrtglossary}{%
4330     \@ifstar\s@printunsrtglossary\@printunsrtglossary
4331 }

```

`ntunsrtglossary` Unstarred version.

```

4332 \newcommand*{\@printunsrtglossary}[1][\]{%
4333     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4334 }

```

ntunsrtglossary Starred version.

```
4335 \newcommand*{\s@printunsrtglossary}[2][\{%
4336   \begin{group}
4337     #2%
4338     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4339   \end{group}
4340 }
```

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

```
4341 \newcommand*{\printunsrtglossaries}{%
4342   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4343 }
```

@unsrt@glossary

```
4344 \newcommand*{\@print@unsrt@glossary}{%
4345   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4346   \glossarypreamble
4347   check for empty list
4348   \glsxtrifemptyglossary{\@glo@type}%
4349   {%
4350     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4351   }%
4352   \key@ifundefined{glossentry}{group}%
4353   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4354   {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4355   \def\@gls@currentlettergroup{}}%
```

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

```
4356 \def\@glsxtr@doglossary{%
4357   \begin{theglossary}%
4358   \glossaryheader
4359   \glsresetentrylist
4360 }%
4361 \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4362   :\expandafter=\csname glolist@\@glo@type\endcsname\do{%
4363   \ifdefempty{\glscurrententrylabel}
4364   }%
4365   {%
```

Provide a hook (for example to measure width).

```
4366   \let\glsxtr@process\@firstofone
4367   \let\printunsrtglossaryskipentry
4368   \@glsxtr@printunsrtglossaryskipentry
4369   \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%
```

Don't check group for child entries.

```

4370 \glxtr@process
4371 {%
4372 \ifglshasparent{\glscurrententrylabel}{}%
4373 {%
4374 \@glxtr@checkgroup\glscurrententrylabel
4375 \expandafter\appto\expandafter\@glxtr@doglossary\expandafter
4376 {\@glxtr@groupheading}%
4377 }%
4378 \eappto\@glxtr@doglossary{%
4379 \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4380 }%
4381 }%
4382 }%
4383 \appto\@glxtr@doglossary{\end{theglossary}}%
4384 \printunsrtglossarypredoglossary
4385 \@glxtr@doglossary
4386 }%
4387 \glossarypostamble
4388 }

```

entryprocesshook

```

4389 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}

```

entryprocesshook

```

4390 \newcommand*{\printunsrtglossaryskipentry}{%
4391 \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4392 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4393 }

```

entryprocesshook

```

4394 \newcommand*{\@glxtr@printunsrtglossaryskipentry}{%
4395 \let\glxtr@process\@gobble
4396 }

```

rypredoglossary

```

4397 \newcommand*{\printunsrtglossarypredoglossary}{}

```

lossary@handler

```

4398 \newcommand{\@printunsrt@glossary@handler}[1]{%
4399 \xdef\glscurrententrylabel{#1}%
4400 \printunsrtglossaryhandler\glscurrententrylabel
4401 }

```

glossaryhandler

```

4402 \newcommand{\printunsrtglossaryhandler}[1]{%
4403 \glxtrunsrtdo{#1}%
4404 }

```

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.

```
4405 \newrobustcmd*{\glstriflabelinlist}[4]{%
4406   \protected@edef\@glstr@doiflabelinlist{\noexpand\@gls@ifinlist{#1}{#2}}%
4407   \@glstr@doiflabelinlist{#3}{#4}%
4408 }
```

srtglossaryunit

```
4409 \newcommand{\print@op@unsrtglossaryunit}[2][{}]{%
4410   \s@printunsrtglossary[type=\glsdefaulttype,#1]{%
4411     \printunsrtglossaryunitsetup{#2}%
4412   }%
4413 }
```

ossaryunitsetup

```
4414 \newcommand*{\printunsrtglossaryunitsetup}[1]{%
4415   \renewcommand{\printunsrtglossaryhandler}[1]{%
4416     \glstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4417     {\glstrunsrtdo{##1}}%
4418   }%
4419 }
```

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

```
4420 \ifcsundef{theH#1}%
4421 {%
4422   \renewcommand*{\@glstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4423 }%
4424 {%
4425   \renewcommand*{\@glstrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4426 }%
4427 \renewcommand*{\glossarysection}[2][{}]{%
4428   \appto\glossarypostamble{\glspare\medskip\glspare}%
4429 }
```

srtglossaryunit

```
4430 \newcommand{\print@noop@unsrtglossaryunit}[2][{}]{%
4431   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4432     requires the record=only or record=alsoindex package option}{}%
4433 }
```

t@getgrouptitle

```
4434 \newrobustcmd*{\@glstr@unsrt@getgrouptitle}[2]{%
4435   \protected@edef\@glstr@titlelabel{\glstr@grouptitle@#1}%
4436   \@onelevel@sanitize\@glstr@titlelabel
4437   \ifcsdef{\@glstr@titlelabel}
4438   {\letcs{#2}{\@glstr@titlelabel}}%
```



```

4439 {\def#2{#1}}%
4440 }

```

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

```

4441 \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).

```

4442 \newcommand*{\glxtrgroupfield}{group}

```

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

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

```

4443 \newcommand*{\@glxtr@checkgroup}[1]{%
4444   \def\@glxtr@groupheading{}%
4445   \key@ifundefined{glossentry}{group}%
4446   {%
4447     \letcs{\@gls@sort}{glo@\glsdetoklabel{#1}@sort}%
4448     \expandafter\glo@grabfirst\@gls@sort{}{}\@nil
4449   }%
4450   {%
4451     \protected@edef\@glo@thislettergrp{%
4452       \csuse{glo@\glsdetoklabel{#1}@\glxtrgroupfield}}%
4453   }%
4454   \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4455   {}%
4456   {%
4457     \ifdefempty{\@gls@currentlettergroup}{}%
4458     {\def\@glxtr@groupheading{\glsgroupskip}}%
4459     \eappto\@glxtr@groupheading{%
4460       \noexpand\glsgroupheading{\expandonce\@glo@thislettergrp}%
4461     }%
4462   }%
4463   \let\@gls@currentlettergroup\@glo@thislettergrp
4464 }

```

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

```

4465 \newcommand{\@glxtr@noidx@do}[1]{%
4466   \ifgl Sentryexists{#1}%
4467   {%

```

```

4468 \global\letcs{\@gls@loclist}{glo@glstetoklabel{#1}@loclist}%
4469 \global\letcs{\@gls@location}{glo@glstetoklabel{#1}@location}%
4470 \ifglshasparent{#1}%
4471 {%
4472   \gls@level=\csuse{glo@glstetoklabel{#1}@level}\relax
4473   \ifdefvoid{\@gls@location}%
4474   {%
4475     \ifdefvoid{\@gls@loclist}%
4476     {%
4477       \subglossentry{\gls@level}{#1}{}%
4478     }%
4479     {%
4480       \subglossentry{\gls@level}{#1}%
4481       {%
4482         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4483       }%
4484     }%
4485   }%
4486   {%
4487     \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4488   }%
4489 }%
4490 {%

4491   \ifdefvoid{\@gls@location}%
4492   {%
4493     \ifdefvoid{\@gls@loclist}
4494     {%
4495       \glossentry{#1}{}%
4496     }%
4497     {%
4498       \glossentry{#1}%
4499       {%
4500         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4501       }%
4502     }%
4503   }%
4504   {%
4505     \glossentry{#1}%
4506     {%
4507       \glossaryentrynumbers{\@gls@location}%
4508     }%
4509   }%
4510 }%
4511 }%
4512 {}%
4513 }

```

1.3.8 Support for bib2gls

Some useful commands for bib2gls users.

`\glshex`

```
4514 \newcommand*{\glshex}{\string\u}
```

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

```
4515 \newcommand*{\glstrresourceinit}{}
```

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

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

`\glstrnewgls`

```
4516 \newcount\@glstrnewgls@inner
```

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

`\@glstrnewgls`

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

```
4517 \newcommand*{\@glstrnewgls}[4]{%
4518   \ifdef{#3}%
4519   {%
4520     \PackageError{glossaries-extra}{Command \string#3\space already
4521 defined}{}%
4522   }%
4523   {%
4524     \ifcsdef{@#4like@#2}%
4525     {%
4526       \advance\@glstrnewgls@inner by \@ne
4527       \def\@glstrnewgls@innercsname{@#4like\number\@glstrnewgls@inner @#2}%
4528     }%
4529     {\def\@glstrnewgls@innercsname{@#4like@#2}}%
4530     \expandafter\newrobustcmd\expandafter*\expandafter
4531     #3\expandafter{\expandafter\@gls@hyp@opt\csname\@glstrnewgls@innercsname\endcsname}%
4532     \ifstrempy{#1}%
4533     {%
4534       \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2][]{%
4535         \new@ifnextchar[%
4536         {\csname @#4@\endcsname{##1}{#2##2}}%
4537         {\csname @#4@\endcsname{##1}{#2##2}[]}%
4538       }%
4539     }%
```

```

4540      {%
4541      \expandafter\newcommand\expandafter*\csname\@glstrnewgls@innercsname\endcsname[2] [] {%
4542      \new@ifnextchar [%
4543      {\csname @#4@\endcsname{#1,##1}{#2##2}}}%
4544      {\csname @#4@\endcsname{#1,##1}{#2##2} [] }%
4545      }%
4546      }%
4547      }%
4548      }

```

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

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

```

4549 \newrobustcmd*{\glstrnewgls}[3] [] {%
4550   \@glstrnewgls{#1}{#2}{#3}{gls}%
4551 }

```

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

```

4552 \newrobustcmd*{\glstrnewglslike}[6] [] {%
4553   \@glstrnewgls{#1}{#2}{#3}{gls}%
4554   \@glstrnewgls{#1}{#2}{#4}{glspl}%
4555   \@glstrnewgls{#1}{#2}{#5}{Gls}%
4556   \@glstrnewgls{#1}{#2}{#6}{Glspl}%
4557 }

```

`\glstrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSpl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

```

4558 \newrobustcmd*{\glstrnewGLSlike}[4] [] {%
4559   \@glstrnewgls{#1}{#2}{#3}{GLS}%
4560   \@glstrnewgls{#1}{#2}{#4}{GLSpl}%
4561 }

```

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

```

4562 \newrobustcmd*{\glstrnewrgls}[3] [] {%
4563   \@glstrnewgls{#1}{#2}{#3}{rgls}%
4564 }

```

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

```

4565 \newrobustcmd*{\glstrnewrglslike}[6] [] {%
4566   \@glstrnewgls{#1}{#2}{#3}{rgls}%
4567   \@glstrnewgls{#1}{#2}{#4}{rglspl}%
4568   \@glstrnewgls{#1}{#2}{#5}{rGls}%
4569   \@glstrnewgls{#1}{#2}{#6}{rGlspl}%
4570 }

```

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

```
4571 \newrobustcmd*{\glxtrnewrGLSlike}[4][\rGLS]{%
4572   \@glxtrnewgls{#1}{#2}{#3}{rGLS}%
4573   \@glxtrnewgls{#1}{#2}{#4}{rGLSpl}%
4574 }
```

Provide easy access to record count fields.

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

```
4575 \newcommand*{\GlsXtrTotalRecordCount}[1]{%
4576   \ifcsdef{glo@\glxtrdetoklabel{#1}@recordcount}%
4577   {\csname glo@\glxtrdetoklabel{#1}@recordcount\endcsname}%
4578   {0}%
4579 }
```

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

```
4580 \newcommand*{\GlsXtrRecordCount}[2]{%
4581   \ifcsdef{glo@\glxtrdetoklabel{#1}@recordcount.#2}%
4582   {\csname glo@\glxtrdetoklabel{#1}@recordcount.#2\endcsname}%
4583   {0}%
4584 }
```

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

```
4585 \newcommand*{\GlsXtrLocationRecordCount}[3]{%
4586   \ifcsdef{glo@\glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}}%
4587   {\csname glo@\glxtrdetoklabel{#1}@recordcount.#2.\glxtrdetoklocation{#3}\endcsname}%
4588   {0}%
4589 }
```

`\trdetoklocation`

```
4590 \newcommand*{\glxtrdetoklocation}[1]{#1}
```

`\ablerecordcount`

```
4591 \newcommand*{\glxtrablerecordcount}{%
4592   \renewcommand*{\gls}{\rgls}%
4593   \renewcommand*{\Gls}{\rGls}%
4594   \renewcommand*{\glspl}{\rglspl}%
4595   \renewcommand*{\Glspl}{\rGlspl}%
4596   \renewcommand*{\GLS}{\rGLS}%
4597   \renewcommand*{\GLSpl}{\rGLSpl}%
4598 }
```

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

```
4599 \newcommand*{\glxtrrecordtriggervalue}[1]{%
```

```

4600 \GlsXtrTotalRecordCount{#1}%
4601 }

```

dCountAttribute

```

4602 \newcommand*{\GlsXtrSetRecordCountAttribute}[2]{%
4603 \@for\@glxstr@cat:=#1\do
4604 {%
4605 \ifdefempty{\@glxstr@cat}{}%
4606 {%
4607 \glsssetcategoryattribute{\@glxstr@cat}{recordcount}{#2}%
4608 }%
4609 }%
4610 }

```

rifrecordtrigger

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

```

4611 \newcommand*{\glxstrifrecordtrigger}[3]{%
4612 \glshasattribute{#1}{recordcount}%
4613 {%
4614 \ifnum\glxstrrecordtriggervalue{#1}>\glssgetattribute{#1}{recordcount}\relax
4615 #3%
4616 \else
4617 #2%
4618 \fi
4619 }%
4620 {#3}%
4621 }

```

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

```

4622 \newcommand*{\@glxstr@rglstrigger@record}[3]{%
4623 \edef\glslabel{\glssdetoklabel{#2}}%
4624 \let\@gls@link@label\glslabel
4625 \def\@glxstr@thevalue{%
4626 \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4627 \def\@glsnumberformat{glstriggerrecordformat}%
4628 \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4629 \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4630 \def\@glxstr@thevalue{%
4631 \def\@glxstr@theHvalue{\@glxstr@thevalue}%
4632 \glxstrinitwrgloss
4633 \setkeys{glslink}{#1}%
4634 \glslinkpostsetkeys
4635 \ifdefempty{\@glxstr@thevalue}%
4636 {%
4637 \@gls@saveentrycounter
4638 }%

```

```

4639 {%
4640   \let\theglsentrycounter\@glstr@thevalue
4641   \def\theHglentrycounter{\@glstr@theHvalue}%
4642 }%
4643 \ifglstrinitwrglossbefore
4644   \@do@wrglossary{#2}%
4645 \fi
4646 #3%
4647 \ifglstrinitwrglossbefore
4648 \else
4649   \@do@wrglossary{#2}%
4650 \fi
4651 \ifKV@glslink@local
4652   \glsllocalunset{#2}%
4653 \else
4654   \glunset{#2}%
4655 \fi
4656 }

```

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

```

4657 \newcommand*{\glstriggerrecordformat}[1]{%

```

\rgls

```

4658 \newrobustcmd*{\rgls}{\@glshyp@opt\@rgls}

```

\@rgls

```

4659 \newcommand*{\@rgls}[2][{}]{%
4660   \new@ifnextchar[{\@rgls@{#1}{#2}}{\@rgls@{#1}{#2}}[]}%
4661 }

```

\@rgls@

```

4662 \def\@rgls@#1#2[#3]{%
4663   \glstrifrecordtrigger{#2}%
4664   {%
4665     \@glstr@rglstrigger@record{#1}{#2}{\rglsformat{#2}{#3}}%
4666   }%
4667   {%
4668     \@gl@{#1}{#2}[#3]%
4669   }%
4670 }%

```

\rglsp1

```

4671 \newrobustcmd*{\rglsp1}{\@glshyp@opt\@rglsp1}

```

\@rglsp1

```

4672 \newcommand*{\@rglsp1}[2][{}]{%
4673   \new@ifnextchar[{\@rglsp1@{#1}{#2}}{\@rglsp1@{#1}{#2}}[]}%
4674 }

```

```

\@rglsp1@
4675 \def\@rglsp1@#1#2[#3]{%
4676   \glxtrifrecordtrigger{#2}%
4677   {%
4678     \@glxtr@rglstrigger@record{#1}{#2}{\rglsp1format{#2}{#3}}%
4679   }%
4680   {%
4681     \@glsp1@{#1}{#2}[#3]%
4682   }%
4683 }%

\@rGls
4684 \newrobustcmd*{\@rGls}{\@gls@hyp@opt\@rGls}

\@rGls
4685 \newcommand*{\@rGls}[2][{}]{%
4686   \new@ifnextchar[{\@rGls@{#1}{#2}}{\@rGls@{#1}{#2}[]}%
4687 }

\@rGls@
4688 \def\@rGls@#1#2[#3]{%
4689   \glxtrifrecordtrigger{#2}%
4690   {%
4691     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsformat{#2}{#3}}%
4692   }%
4693   {%
4694     \@Gls@{#1}{#2}[#3]%
4695   }%
4696 }%

\@rGlspl
4697 \newrobustcmd*{\@rGlspl}{\@gls@hyp@opt\@rGlspl}

\@rGlspl
4698 \newcommand*{\@rGlspl}[2][{}]{%
4699   \new@ifnextchar[{\@rGlspl@{#1}{#2}}{\@rGlspl@{#1}{#2}[]}%
4700 }

\@rGlspl@
4701 \def\@rGlspl@#1#2[#3]{%
4702   \glxtrifrecordtrigger{#2}%
4703   {%
4704     \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4705   }%
4706   {%
4707     \@Glspl@{#1}{#2}[#3]%
4708   }%
4709 }%

```


\rGLS

4710 \newrobustcmd*{\rGLS}{\@gls@hyp@opt\rGLS}

\@rGLS

4711 \newcommand*{\@rGLS}[2][\%
4712 \new@ifnextchar[\@rGLS@{#1}{#2}]{\@rGLS@{#1}{#2}[]}%
4713 }

\@rGLS@

4714 \def\rGLS@#1#2[#3]{%
4715 \glsxtrifrecordtrigger{#2}%
4716 {%
4717 \glsxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%
4718 }%
4719 {%
4720 \@GLS@{#1}{#2}[#3]%
4721 }%
4722 }%

\rGLSpl

4723 \newrobustcmd*{\rGLSpl}{\@gls@hyp@opt\rGLSpl}

\@rGLSpl

4724 \newcommand*{\@rGLSpl}[2][\%
4725 \new@ifnextchar[\@rGLSpl@{#1}{#2}]{\@rGLSpl@{#1}{#2}[]}%
4726 }

\@rGLSpl@

4727 \def\rGLSpl@#1#2[#3]{%
4728 \glsxtrifrecordtrigger{#2}%
4729 {%
4730 \glsxtr@rglstrigger@record{#1}{#2}{\rGLSplformat{#2}{#3}}%
4731 }%
4732 {%
4733 \@GLSpl@{#1}{#2}[#3]%
4734 }%
4735 }%

\rglsformat

4736 \newcommand*{\rglsformat}[2]{%
4737 \glsifregular{#1}
4738 {\glsentryfirst{#1}}%
4739 {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%
4740 }

\rglsplformat

4741 \newcommand*{\rglsplformat}[2]{%
4742 \glsifregular{#1}

```

4743 {\glsentryfirstplural{#1}}%
4744 {\ifglshaslong{#1}{\glsentrylongplural{#1}}{\glsentryfirstplural{#1}}#2%
4745 }

```

`\rGlsformat`

```

4746 \newcommand*{\rGlsformat}[2]{%
4747   \glsifregular{#1}
4748   {\Glsentryfirst{#1}}%
4749   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}#2%
4750 }

```

`\rGlsplformat`

```

4751 \newcommand*{\rGlsplformat}[2]{%
4752   \glsifregular{#1}
4753   {\Glsentryfirstplural{#1}}%
4754   {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}#2%
4755 }

```

`\rGLSformat`

```

4756 \newcommand*{\rGLSformat}[2]{%
4757   \expandafter\mfirstucMakeUppercase\expandafter{\rGlsformat{#1}{#2}}%
4758 }

```

`\rGLSplformat`

```

4759 \newcommand*{\rGLSplformat}[2]{%
4760   \expandafter\mfirstucMakeUppercase\expandafter{\rGlsplformat{#1}{#2}}%
4761 }

```

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.

```

4762 \@ifpackageloaded{glossaries-accsupp}
4763 {

```

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

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

```

4764   \newcommand*{\glsaccessname}[1]{%
4765     \glsnameaccessdisplay
4766     {%
4767       \glsentryname{#1}%
4768     }%
4769     {#1}%
4770   }

```

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

```
4771 \newcommand*{\Glsaccessname}[1]{%
4772   \glsnameaccessdisplay
4773   {%
4774     \Glsentryname{#1}%
4775   }%
4776   {#1}%
4777 }
```

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

```
4778 \newcommand*{\GLSaccessname}[1]{%
4779   \glsnameaccessdisplay
4780   {%
4781     \mfirstucMakeUppercase{\glsentryname{#1}}%
4782   }%
4783   {#1}%
4784 }
```

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

```
4785 \newcommand*{\glsaccesstext}[1]{%
4786   \glstextaccessdisplay
4787   {%
4788     \glsentrytext{#1}%
4789   }%
4790   {#1}%
4791 }
```

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

```
4792 \newcommand*{\GLSaccesstext}[1]{%
4793   \glstextaccessdisplay
4794   {%
4795     \Glsentrytext{#1}%
4796   }%
4797   {#1}%
4798 }
```

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

```
4799 \newcommand*{\GLSaccesstext}[1]{%
4800   \glstextaccessdisplay
4801   {%
4802     \mfirstucMakeUppercase{\glsentrytext{#1}}%
4803   }%
4804   {#1}%
4805 }
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

4806 \newcommand*{\glsaccessplural}[1]{%
4807   \glspluralaccessdisplay
4808   {%
4809     \glsentryplural{#1}%
4810   }%
4811   {#1}%
4812 }

```

Glsaccessplural Display the plural value (no link and no check for existence) with the first letter converted to upper case.

```

4813 \newcommand*{\Glsaccessplural}[1]{%
4814   \glspluralaccessdisplay
4815   {%
4816     \Glsentryplural{#1}%
4817   }%
4818   {#1}%
4819 }

```

GLSaccessplural Display the plural value (no link and no check for existence) converted to upper case.

```

4820 \newcommand*{\GLSaccessplural}[1]{%
4821   \glspluralaccessdisplay
4822   {%
4823     \mfirstucMakeUppercase{\glsentryplural{#1}}%
4824   }%
4825   {#1}%
4826 }

```

\glsaccessfirst Display the first value (no link and no check for existence).

```

4827 \newcommand*{\glsaccessfirst}[1]{%
4828   \glsfirstaccessdisplay
4829   {%
4830     \glsentryfirst{#1}%
4831   }%
4832   {#1}%
4833 }

```

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

```

4834 \newcommand*{\Glsaccessfirst}[1]{%
4835   \glsfirstaccessdisplay
4836   {%
4837     \Glsentryfirst{#1}%
4838   }%
4839   {#1}%
4840 }

```

\GLSaccessfirst Display the first value (no link and no check for existence) converted to upper case.

```

4841 \newcommand*{\GLSaccessfirst}[1]{%

```

```

4842 \glsfirstaccessdisplay
4843 {%
4844     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4845 }%
4846 {#1}%
4847 }

```

`\glsfirstplural` Display the firstplural value (no link and no check for existence).

```

4848 \newcommand*{\glsaccessfirstplural}[1]{%
4849     \glsfirstpluralaccessdisplay
4850     {%
4851         \glsentryfirstplural{#1}%
4852     }%
4853     {#1}%
4854 }

```

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

```

4855 \newcommand*{\Glsaccessfirstplural}[1]{%
4856     \glsfirstpluralaccessdisplay
4857     {%
4858         \Glsentryfirstplural{#1}%
4859     }%
4860     {#1}%
4861 }

```

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

```

4862 \newcommand*{\GLSaccessfirstplural}[1]{%
4863     \glsfirstpluralaccessdisplay
4864     {%
4865         \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4866     }%
4867     {#1}%
4868 }

```

`\glsaccesssymbol` Display the symbol value (no link and no check for existence).

```

4869 \newcommand*{\glsaccesssymbol}[1]{%
4870     \glssymbolaccessdisplay
4871     {%
4872         \glsentrysymbol{#1}%
4873     }%
4874     {#1}%
4875 }

```

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

```

4876 \newcommand*{\Glsaccesssymbol}[1]{%
4877     \glssymbolaccessdisplay

```

```

4878     {%
4879     \Glsentrysymbol{#1}%
4880     }%
4881     {#1}%
4882 }

```

GLSaccesssymbol Display the symbol value (no link and no check for existence) converted to upper case.

```

4883 \newcommand*{\GLSaccesssymbol}[1]{%
4884 \glssymbolaccessdisplay
4885 {%
4886 \mfirstucMakeUppercase{\glentrysymbol{#1}}%
4887 }%
4888 {#1}%
4889 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence).

```

4890 \newcommand*{\glaccesssymbolplural}[1]{%
4891 \glssymbolpluralaccessdisplay
4892 {%
4893 \glentrysymbolplural{#1}%
4894 }%
4895 {#1}%
4896 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4897 \newcommand*{\GLSaccesssymbolplural}[1]{%
4898 \glssymbolpluralaccessdisplay
4899 {%
4900 \Glsentrysymbolplural{#1}%
4901 }%
4902 {#1}%
4903 }

```

esssymbolplural Display the symbolplural value (no link and no check for existence) converted to upper case.

```

4904 \newcommand*{\GLSaccesssymbolplural}[1]{%
4905 \glssymbolpluralaccessdisplay
4906 {%
4907 \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
4908 }%
4909 {#1}%
4910 }

```

\glsaccessdesc Display the desc value (no link and no check for existence).

```

4911 \newcommand*{\glsaccessdesc}[1]{%
4912 \glsdescriptionaccessdisplay
4913 {%
4914 \glentrydesc{#1}%

```

```

4915     }%
4916     {#1}%
4917 }

```

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

```

4918 \newcommand*{\Glsaccessdesc}[1]{%
4919 \glsdescriptionaccessdisplay
4920 {%
4921 \Glsentrydesc{#1}%
4922 }%
4923 {#1}%
4924 }

```

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

```

4925 \newcommand*{\GLSaccessdesc}[1]{%
4926 \glsdescriptionaccessdisplay
4927 {%
4928 \mfirstucMakeUppercase{\glsentrydesc{#1}}%
4929 }%
4930 {#1}%
4931 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence).

```

4932 \newcommand*{\glsaccessdescplural}[1]{%
4933 \glsdescriptionpluralaccessdisplay
4934 {%
4935 \glsentrydescplural{#1}%
4936 }%
4937 {#1}%
4938 }

```

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

```

4939 \newcommand*{\Glsaccessdescplural}[1]{%
4940 \glsdescriptionpluralaccessdisplay
4941 {%
4942 \Glsentrydescplural{#1}%
4943 }%
4944 {#1}%
4945 }

```

`ccessdescplural` Display the descplural value (no link and no check for existence) converted to upper case.

```

4946 \newcommand*{\GLSaccessdescplural}[1]{%
4947 \glsdescriptionpluralaccessdisplay
4948 {%
4949 \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
4950 }%

```

```

4951     {#1}%
4952 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

4953 \newcommand*{\glsaccessshort}[1]{%
4954   \glsshortaccessdisplay
4955   {%
4956     \glentryshort{#1}%
4957   }%
4958   {#1}%
4959 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

4960 \newcommand*{\Glsaccessshort}[1]{%
4961   \glsshortaccessdisplay
4962   {%
4963     \Glsentryshort{#1}%
4964   }%
4965   {#1}%
4966 }

```

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

```

4967 \newcommand*{\GLSaccessshort}[1]{%
4968   \glsshortaccessdisplay
4969   {%
4970     \mfirstucMakeUppercase{\glentryshort{#1}}%
4971   }%
4972   {#1}%
4973 }

```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

4974 \newcommand*{\glsaccessshortpl}[1]{%
4975   \glsshortpluralaccessdisplay
4976   {%
4977     \glentryshortpl{#1}%
4978   }%
4979   {#1}%
4980 }

```

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

4981 \newcommand*{\Glsaccessshortpl}[1]{%
4982   \glsshortpluralaccessdisplay
4983   {%
4984     \Glsentryshortpl{#1}%
4985   }%
4986   {#1}%
4987 }

```


LSaccessshortpl Display the shortplural value (no link and no check for existence) converted to upper case.

```
4988 \newcommand*{\LSaccessshortpl}[1]{%
4989 \glsshortpluralaccessdisplay
4990 {%
4991 \mfirstucMakeUppercase{\glsentryshortpl{#1}}%
4992 }%
4993 {#1}%
4994 }
```

\glsaccesslong Display the long form (no link and no check for existence).

```
4995 \newcommand*{\glsaccesslong}[1]{%
4996 \glslongaccessdisplay{\glsentrylong{#1}}{#1}%
4997 }
```

\Glsaccesslong Display the long form (no link and no check for existence).

```
4998
4999 \newcommand*{\Glsaccesslong}[1]{%
5000 \glslongaccessdisplay{\Glsentrylong{#1}}{#1}%
5001 }
```

\GLSaccesslong Display the long value (no link and no check for existence) converted to upper case.

```
5002 \newcommand*{\GLSaccesslong}[1]{%
5003 \glslongaccessdisplay
5004 {%
5005 \mfirstucMakeUppercase{\glsentrylong{#1}}%
5006 }%
5007 {#1}%
5008 }
```

glsaccesslongpl Display the long plural form (no link and no check for existence).

```
5009 \newcommand*{\glsaccesslongpl}[1]{%
5010 \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5011 }
```

Glsaccesslongpl Display the long plural form (no link and no check for existence).

```
5012
5013 \newcommand*{\Glsaccesslongpl}[1]{%
5014 \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5015 }
```

GLSaccesslongpl Display the longplural value (no link and no check for existence) converted to upper case.

```
5016 \newcommand*{\GLSaccesslongpl}[1]{%
5017 \glslongpluralaccessdisplay
5018 {%
5019 \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5020 }%
5021 {#1}%
5022 }
```

End of if part

5023 }

5024 {

No accessibility support. Just define these commands to do \glentry<xxx>

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

5025 \newcommand*{\glsaccessname}[1]{\glentryname{#1}}

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

5026 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

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

5027 \newcommand*{\GLSaccessname}[1]{%

5028 \protect\mfirstucMakeUppercase{\glentryname{#1}}}

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

5029 \newcommand*{\glsaccesstext}[1]{\glentrytext{#1}}

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

5030 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

\GLSaccesstext Display the text value (no link and no check for existence). converted to upper case.

5031 \newcommand*{\GLSaccesstext}[1]{%

5032 \protect\mfirstucMakeUppercase{\glentrytext{#1}}}

glsaccessplural Display the plural value (no link and no check for existence).

5033 \newcommand*{\glsaccessplural}[1]{\glentryplural{#1}}

Glsaccessplural Display the plural value (no link and no check for existence) with the first letter converted to upper case.

5034 \newcommand*{\Glsaccessplural}[1]{\Glsentryplural{#1}}

GLSaccessplural Display the plural value (no link and no check for existence). converted to upper case.

5035 \newcommand*{\GLSaccessplural}[1]{%

5036 \protect\mfirstucMakeUppercase{\glentryplural{#1}}}

\glsaccessfirst Display the first value (no link and no check for existence).

5037 \newcommand*{\glsaccessfirst}[1]{\glentryfirst{#1}}

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

5038 \newcommand*{\Glsaccessfirst}[1]{\Glsentryfirst{#1}}

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

```

5039 \newcommand*{\GLSaccessfirst}[1]{%
5040 \protect\mfirstucMakeUppercase{\glentryfirst{#1}}}
```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

5041 \newcommand*{\glsaccessfirstplural}[1]{\glentryfirstplural{#1}}
```

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

```

5042 \newcommand*{\Glsaccessfirstplural}[1]{\Glsentryfirstplural{#1}}
```

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.

```

5043 \newcommand*{\GLSaccessfirstplural}[1]{%
5044 \protect\mfirstucMakeUppercase{\glentryfirstplural{#1}}}
```

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

```

5045 \newcommand*{\glsaccesssymbol}[1]{\glentrysymbol{#1}}
```

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

```

5046 \newcommand*{\Glsaccesssymbol}[1]{\Glsentrysymbol{#1}}
```

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

```

5047 \newcommand*{\GLSaccesssymbol}[1]{%
5048 \protect\mfirstucMakeUppercase{\glentrysymbol{#1}}}
```

`esssymbolplural` Display the symbolplural value (no link and no check for existence).

```

5049 \newcommand*{\glsaccesssymbolplural}[1]{\glentrysymbolplural{#1}}
```

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

```

5050 \newcommand*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}
```

`esssymbolplural` Display the symbolplural value (no link and no check for existence). converted to upper case.

```

5051 \newcommand*{\GLSaccesssymbolplural}[1]{%
5052 \protect\mfirstucMakeUppercase{\glentrysymbolplural{#1}}}
```

`\glsaccessdesc` Display the desc value (no link and no check for existence).

```

5053 \newcommand*{\glsaccessdesc}[1]{\glentrydesc{#1}}
```

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

```

5054 \newcommand*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}
```

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

```

5055 \newcommand*{\GLSaccessdesc}[1]{%
5056 \protect\mfirstucMakeUppercase{\glentrydesc{#1}}}
```

`ccessdescplural` Display the descplural value (no link and no check for existence).

```

5057 \newcommand*{\glsaccessdescplural}[1]{\glentrydescplural{#1}}
```

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

```

5058 \newcommand*{\Glsaccessdescplural}[1]{\Glentrydescplural{#1}}
```

`ccessdescplural` Display the descplural value (no link and no check for existence). converted to upper case.

```

5059 \newcommand*{\GLSaccessdescplural}[1]{%
5060 \protect\mfirstucMakeUppercase{\glentrydescplural{#1}}}
```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5061 \newcommand*{\glsaccessshort}[1]{\glentryshort{#1}}
```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5062 \newcommand*{\Glsaccessshort}[1]{\Glentryshort{#1}}
```

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

```

5063 \newcommand*{\GLSaccessshort}[1]{%
5064 \protect\mfirstucMakeUppercase{\glentryshort{#1}}}
```

`lsaccessshortpl` Display the short plural form (no link and no check for existence).

```

5065 \newcommand*{\glsaccessshortpl}[1]{\glentryshortpl{#1}}
```

`lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5066 \newcommand*{\Glsaccessshortpl}[1]{\Glentryshortpl{#1}}
```

`LSaccessshortpl` Display the shortplural value (no link and no check for existence). converted to upper case.

```

5067 \newcommand*{\GLSaccessshortpl}[1]{%
5068 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```

5069 \newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```

5070 \newcommand*{\Glsaccesslong}[1]{\Glentrylong{#1}}
```

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

```

5071 \newcommand*{\GLSaccesslong}[1]{%
5072 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}
```

`glsaccesslongpl` Display the long plural form (no link and no check for existence).
5073 `\newcommand*{\glsaccesslongpl}[1]{\glsentrylongpl{#1}}`

`Glsaccesslongpl` Display the long plural form (no link and no check for existence).
5074 `\newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}`

`GLSaccesslongpl` Display the longplural value (no link and no check for existence). converted to upper case.
5075 `\newcommand*{\GLSaccesslongpl}[1]{%`
5076 `\protect\mfirstucMakeUppercase{\glsentrylongpl{#1}}}`

End of else part
5077 }

1.5 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.
5078 `\glsaddstoragekey{category}{general}{\glscategory}`

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.
5079 `\newcommand{\glsifcategory}[4]{%`
5080 `\ifglsfieldeq{#1}{category}{#2}{#3}{#4}%`
5081 }

Categories can have attributes.

`categoryattribute` `\glssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

5082 `\newcommand*{\glssetcategoryattribute}[3]{%`
5083 `\csdef{@glsxtr@categoryattr@@#1@#2}{#3}%`
5084 }

`categoryattribute` `\glsgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

5085 `\newcommand*{\glsgetcategoryattribute}[2]{%`
5086 `\csuse{@glsxtr@categoryattr@@#1@#2}%`
5087 }

categoryattribute `\glshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5088 \newcommand*{\glshascategoryattribute}[4]{%
5089   \ifcsvoid{@glstr@categoryattr@#1@#2}{#4}{#3}%
5090 }
```

\glissetattribute `\glissetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5091 \newcommand*{\glissetattribute}[3]{%
5092   \glsetcategoryattribute{\glscategory{#1}}{#2}{#3}%
5093 }
```

\glsggetattribute `\glsggetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```
5094 \newcommand*{\glsggetattribute}[2]{%
5095   \glsgcategoryattribute{\glscategory{#1}}{#2}%
5096 }
```

\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.

```
5097 \newcommand*{\glshasattribute}[4]{%
5098   \ifglstryexists{#1}%
5099   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5100   {#4}%
5101 }
```

categoryattribute `\glisifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```
5102 \newcommand{\glisifcategoryattribute}[5]{%
```

```

5103 \ifcsundef{@glxtr@categoryattr@@#1@#2}%
5104 {#5}%
5105 {\ifcsstring{@glxtr@categoryattr@@#1@#2}{#3}{#4}{#5}}%
5106 }

```

`\glsifattribute` `\glsifattribute{<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.

```

5107 \newcommand{\glsifattribute}[5]{%
5108   \ifglentryexists{#1}%
5109   {\glsifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5110   {#5}%
5111 }

```

Set attributes for the default general category:

```
5112 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5113 \glsetcategoryattribute{acronym}{regular}{true}
```

`regularcategory` Convenient shortcut to create add the regular attribute.

```

5114 \newcommand*{\glsetregularcategory}[1]{%
5115   \glsetcategoryattribute{#1}{regular}{true}%
5116 }

```

`ifregularcategory` `\glsifregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```

5117 \newcommand{\glsifregularcategory}[3]{%
5118   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
5119 }

```

`ifnotregularcategory` `\glsifnotregularcategory{<category>}{<true part>}{<false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```

5120 \newcommand{\glsifnotregularcategory}[3]{%
5121   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
5122 }

```

`\glsifregular` `\glsifregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5123 \newcommand{\glsifregular}[3]{%
5124   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
5125 }
```

`\glsifnotregular` `\glsifnotregular{<entry label>}{<true part>}{<false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5126 \newcommand{\glsifnotregular}[3]{%
5127   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
5128 }
```

`\glsforeachincategory` `\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.

```
5129 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
5130   \forallglossaries[#1]{#3}%
5131   {%
5132     \forlentries[#3]{#4}%
5133     {%
5134       \glsifcategory{#4}{#2}{#5}{}%
5135     }%
5136   }%
5137 }
```

`\glsforeachwithattribute` `\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.


```

5138 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5139   \forallglossaries[#1]{#4}%
5140   {%
5141     \forglsentries[#4]{#5}%
5142     {%
5143       \glsifattribute{#5}{#2}{#3}{#6}{}%
5144     }%
5145   }%
5146 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to `index` and add `\glstrpostdescription`.

```

5147 \ifdef\newterm
5148 {%

```

`\newterm`

```

5149   \renewcommand*{\newterm}[2][ ]{%
5150     \newglossaryentry{#2}%
5151     {type={index},category=index,name={#2},%
5152      description={\glstrpostdescription\nopostdesc},#1}%
5153   }

```

Indexed terms are regular by default.

```

5154   \glsssetcategoryattribute{index}{regular}{true}

```

`trpostdescindex`

```

5155   \newcommand*{\glstrpostdescindex}{}

5156 }
5157 {}

```

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`.

```

5158 \ifdef\printsymbols
5159 {%

```

`glstrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```

5160   \newcommand*{\glstrnewsymbol}[3][ ]{%
5161     \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5162   }

```

Symbols are regular by default.

```

5163   \glsssetcategoryattribute{symbol}{regular}{true}

```

`trpostdescsymbol`

```

5164   \newcommand*{\glstrpostdescsymbol}{}

```

```
5165 }
5166 {}
```

Similar for the numbers option.

```
5167 \ifdef\printnumbers
5168 {%
```

glsxtrnewnumber

```
5169 \ifdef\printnumbers
5170   \newcommand*\glsxtrnewnumber[3][]{%
5171     \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5172   }
```

Numbers are regular by default.

```
5173   \glssetcategoryattribute{number}{regular}{true}
```

rpostdescnumber

```
5174   \newcommand*\glsxtrpostdescnumber{}

5175 }
5176 {}
```

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.

```
5177 \newcommand*\glsxtrsetcategory[2]{%
5178   \@for\@glsxtr@label:=#1\do
5179   {%
5180     \glsfieldxdef{\@glsxtr@label}{category}{#2}%
5181   }%
5182 }
```

tcategoryforall 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.

```
5183 \newcommand*\glsxtrsetcategoryforall[2]{%
5184   \forallglossaries[#1]{\@glsxtr@type}{%
5185     \forglsentries[\@glsxtr@type]{\@glsxtr@label}%
5186     {%
5187       \glsfieldxdef{\@glsxtr@label}{category}{#2}%
5188     }%
5189   }%
5190 }
```

trfieldtitlecase

```
\glsxtrfieldtitlecase{<label>}{<field>}
```

Apply title casing to the contents of the given field.

```
5191 \newcommand*\glsxtrfieldtitlecase[2]{%
```

```

5192 \expandafter\glxtrfieldtitlecasecs\expandafter
5193   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
5194 }

```

fieldtitlecasecs The command used by \glxtrfieldtitlecase. May be redefined to use a different command, for example, \xcapitalisefmtwords.

```

5195 \newcommand*{\glxtrfieldtitlecasecs}[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.

```

5196 \ifpackageloaded{glossaries-accsupp}
5197 {
5198   \renewcommand*{\glossentrydesc}[1]{%
5199     \glsdoifexistsorwarn{#1}%
5200     {%
5201       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossdescfont attribute to determine the font applied.

```

5202   \glshasattribute{#1}{glossdescfont}%
5203   {%
5204     \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5205     \ifcsdef{\@glxtr@attrval}%
5206     {%
5207       \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5208     }%
5209     {%
5210       \GlossariesExtraWarning{Unknown control sequence name
5211         ‘\@glxtr@attrval’ supplied in glossdescfont attribute
5212         for entry ‘#1’. Ignoring}%
5213       \let\@glxtr@glossdescfont\@firstofone
5214     }%
5215   }%
5216   {\let\@glxtr@glossdescfont\@firstofone}%
5217   \glsifattribute{#1}{glossdesc}{firstuc}%
5218   {%
5219     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5220   }%
5221   {%
5222     \glsifattribute{#1}{glossdesc}{title}%
5223     {%
5224       \@glxtr@do@titlecaps@warn
5225       \glsdescriptionaccessdisplay
5226       {%
5227         \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5228       }%
5229       {#1}%

```

```

5230     }%
5231     {%
5232         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
5233     }%
5234 }%
5235 }%
5236 }
5237 }
5238 {
5239 \renewcommand*{\glossentrydesc}[1]{%
5240     \glsdoifexistsorwarn{#1}%
5241     {%
5242         \glssetabbrvfmt{\glscategory{#1}}%
5243         \glsattribute{#1}{glossdescfont}%
5244         {%
5245             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5246             \ifcsdef{\@glxtr@attrval}%
5247             {%
5248                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5249             }%
5250             {%
5251                 \GlossariesExtraWarning{Unknown control sequence name
5252                     '\@glxtr@attrval' supplied in glossdescfont attribute
5253                     for entry '#1'. Ignoring}%
5254                 \let\@glxtr@glossdescfont\@firstofone
5255             }%
5256         }%
5257         {\let\@glxtr@glossdescfont\@firstofone}%
5258         \glsifattribute{#1}{glossdesc}{firstuc}%
5259         {%
5260             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%
5261         }%
5262         {%
5263             \glsifattribute{#1}{glossdesc}{title}%
5264             {%
5265                 \@glxtr@do@titlecaps@warn
5266                 \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5267             }%
5268             {%
5269                 \@glxtr@glossdescfont{\glsentrydesc{#1}}%
5270             }%
5271         }%
5272     }%
5273 }
5274 }

```

`\glossentryname` If the glossname attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

5275 `\ifpackageloaded{glossaries-accsupp}`

```

5276 {
5277   \renewcommand*{\glossentryname}[1]{%
5278     \@glsdoifexistsorwarn{#1}%
5279     {%
5280       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5281     \glsattribute{#1}{glossnamefont}%
5282     {%
5283       \edef\@glsxtr@attrval{\glsattribute{#1}{glossnamefont}}%
5284       \ifcsdef{\@glsxtr@attrval}%
5285       {%
5286         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5287       }%
5288       {%
5289         \GlossariesExtraWarning{Unknown control sequence name
5290           '\@glsxtr@attrval' supplied in glossnamefont attribute
5291           for entry '#1'. Reverting to default \string\glsnamefont}%
5292         \let\@glsxtr@glossnamefont\glsnamefont
5293       }%
5294     }%
5295     {\let\@glsxtr@glossnamefont\glsnamefont}%
5296     \glsifattribute{#1}{glossname}{firstuc}%
5297     {%
5298       \glsnameaccessdisplay
5299       {%
5300         \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5301       }%
5302       {#1}%
5303     }%
5304     {%
5305       \glsifattribute{#1}{glossname}{title}%
5306       {%
5307         \@glsxtr@do@titlecaps@warn
5308         \glsnameaccessdisplay
5309         {%
5310           \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
5311         }%
5312         {#1}%
5313       }%
5314       {%
5315         \glsifattribute{#1}{glossname}{uc}%
5316         {%
5317           \glsnameaccessdisplay
5318           {%

```

Hide the label from the upper-casing command.

```

5319         \letcs{\glo@name}{\glo@glstoklabel{#1}@name}%
5320         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5321       }%

```

```

5322         {#1}%
5323     }%
5324     {%
5325         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@name}%
5326         \glsnameaccessdisplay
5327     {%
5328         \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5329     }%
5330     {#1}%
5331 }%
5332 }%
5333 }%

```

Do post-name hook:

```

5334     \glxtrpostnamehook{#1}%
5335 }%
5336 }
5337 }
5338 {
5339     \renewcommand*{\glossentryname}[1]{%
5340         \@glsdoifexistsorwarn{#1}%
5341     {%
5342         \glsetabbrvfmt{\glscategory{#1}}%
5343         \glshasattribute{#1}{glossnamefont}%
5344     {%
5345         \edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
5346         \ifcsdef{\@glxtr@attrval}%
5347         {%
5348             \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5349         }%
5350     {%
5351         \GlossariesExtraWarning{Unknown control sequence name
5352             '\@glxtr@attrval' supplied in glossnamefont attribute
5353             for entry '#1'. Reverting to default \string\glsnamefont}%
5354         \let\@glxtr@glossnamefont\glsnamefont
5355     }%
5356 }%
5357 {\let\@glxtr@glossnamefont\glsnamefont}%
5358 \gl@ifattribute{#1}{glossname}{firstuc}%
5359 {%
5360     \@glxtr@glossnamefont{\Glsentryname{#1}}%
5361 }%
5362 {%
5363     \gl@ifattribute{#1}{glossname}{title}%
5364     {%
5365         \@glxtr@do@titlecaps@warn
5366         \@glxtr@glossnamefont{\glxtrfieldtitlecase{#1}{name}}%
5367     }%
5368     {%
5369         \gl@ifattribute{#1}{glossname}{uc}%

```

```
5370      {%
```

Hide the label from the upper-casing command.

```
5371      \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
5372      \@glxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5373      }%
5374      {%
```

This little trick is used by glossaries to allow the user to redefine `\glsnamefont` to use `\makefirstuc`. Support it even though they can now use the `firstuc` attribute.

```
5375      \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
5376      \expandafter\@glxtr@glossnamefont\expandafter{\glo@name}%
5377      }%
5378      }%
5379      }%
```

Do post-name hook.

```
5380      \glxtrpostnamehook{#1}%
5381      }%
5382  }
5383 }
```

`\Glossentryname` Redefine to set the abbreviation format and accessibility support.

```
5384 \ifpackageloaded{glossaries-accsupp}
5385 {
5386   \renewcommand*{\Glossentryname}[1]{%
5387     \@glstdoifexistsorwarn{#1}%
5388     {%
5389       \glsetabbrvfmt{\glscategory{#1}}%
```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```
5390     \glshasattribute{#1}{glossnamefont}%
5391     {%
5392       \edef\@glxtr@attrval{\glsetattribute{#1}{glossnamefont}}%
5393       \ifcsdef{\@glxtr@attrval}%
5394       {%
5395         \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5396         }%
5397       {%
5398         \GlossariesExtraWarning{Unknown control sequence name
5399           ‘\@glxtr@attrval’ supplied in glossnamefont attribute
5400           for entry ‘#1’. Reverting to default \string\glsnamefont}%
5401         \let\@glxtr@glossnamefont\glsnamefont
5402         }%
5403       }%
5404       {\let\@glxtr@glossnamefont\glsnamefont}%
5405       \glsnameaccessdisplay
5406       {%
5407         \@glxtr@glossnamefont{\Glsentryname{#1}}%
5408         }%
5409       {#1}%
```

Do post-name hook:

```

5410     \glstrpostnamehook{#1}%
5411   }%
5412 }
5413 }
5414 {
5415   \renewcommand*{\Glossentryname}[1]{%
5416     \@glsoifexistsorwarn{#1}%
5417     {%
5418       \glsetabbrvfmt{\glscategory{#1}}%
5419       \glshasattribute{#1}{glossnamefont}%
5420       {%
5421         \edef\@glstr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5422         \ifcsdef{\@glstr@attrval}%
5423         {%
5424           \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5425         }%
5426         {%
5427           \GlossariesExtraWarning{Unknown control sequence name
5428             '\@glstr@attrval' supplied in glossnamefont attribute
5429             for entry '#1'. Reverting to default \string\glsgnamefont}%
5430           \let\@glstr@glossnamefont\glsgnamefont
5431         }%
5432       }%
5433       {\let\@glstr@glossnamefont\glsgnamefont}%
5434       \@glstr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5435     \glstrpostnamehook{#1}%
5436   }%
5437 }
5438 }

```

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.

```

5439 \newcommand*{\glstrpostnamehook}[1]{%
5440   \let\@glsgnumberformat\@glstr@defaultnumberformat
5441   \glstrdoautoindexname{#1}{indexname}%

```

Allow categories to hook in here.

```

5442   \csuse{glstrpostname\glscategory{#1}}%
5443 }

```

`etaccessdisplay`

```

5444 \@ifpackageloaded{glossaries-accsupp}
5445 {

```



```

5446 \newcommand*{\glxtr@setaccessdisplay}[1]{%
5447   \ifcsdef{gls#1accessdisplay}%
5448   {\letcs\glxtr@accessdisplay{gls#1accessdisplay}}%
5449   {%

```

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.

```

5450   \edef\@gls@thisval{#1}%
5451   \@for\@gls@map:=\@gls@keymap\do{%
5452     \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5453     \ifdefequal{\@this@key}{\@gls@thisval}%
5454     {%
5455       \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5456       \@endfortrue
5457     }%
5458   }%
5459 }%
5460 \ifcsdef{gls\@gls@thisval accessdisplay}%
5461 {\letcs\glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5462 {\let\glxtr@accessdisplay\@firstoftwo}%
5463 }%
5464 }
5465 }
5466 {%
5467 \newcommand*{\glxtr@setaccessdisplay}[1]{%
5468   \let\glxtr@accessdisplay\@firstoftwo
5469 }

```

`sentrynameother` Provide a command that works like `\glossentryname` but accesses a different field (which must be supplied using its internal field label).

```

5470 \newrobustcmd*{\glossentrynameother}[2]{%
5471   \@glsdoifexistsorwarn{#1}%
5472   {%

```

Accessibility support:

```

5473   \glxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5474   \glsetabbrvfmt{\glscategory{#1}}%
5475   \glshasattribute{#1}{glossnamefont}%
5476   {%
5477     \edef\@glxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5478     \ifcsdef{\@glxtr@attrval}%
5479     {%
5480       \letcs{\@glxtr@glossnamefont}{\@glxtr@attrval}%
5481     }%
5482   }%
5483   \GlossariesExtraWarning{Unknown control sequence name
5484     '\@glxtr@attrval' supplied in glossnamefont attribute

```

```

5485         for entry '#1'. Reverting to default \string\glsnamefont}%
5486         \let\@glsxtr@glossnamefont\glsnamefont
5487     }%
5488 }%
5489 {\let\@glsxtr@glossnamefont\glsnamefont}%
5490 \glsifattribute{#1}{glossname}{firstuc}%
5491 {%
5492     \@glsxtr@accessdisplay
5493     {\@glsxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5494     {#1}%
5495 }%
5496 {%
5497     \glsifattribute{#1}{glossname}{title}%
5498     {%
5499         \@glsxtr@do@titlecaps@warn
5500         \@glsxtr@accessdisplay
5501         {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
5502         {#1}%
5503     }%
5504     {%
5505         \glsifattribute{#1}{glossname}{uc}%
5506         {%
5507             \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5508             \@glsxtr@accessdisplay
5509             {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5510             {#1}%
5511         }%
5512         {%
5513             \letcs{\glo@name}{glo@\glsdetoklabel{#1}@#2}%
5514             \@glsxtr@accessdisplay
5515             {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}}%
5516             {#1}%
5517         }%
5518     }%
5519 }%

```

Do post-name hook.

```

5520     \glsxtrpostnamehook{#1}%
5521 }%
5522 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

5523 \newif\if@glsxtr@format@override
5524 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glsnumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5525 \@ifpackageloaded{hyperref}

```

5526 {
 If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so don't add it.

```
5527 \ifHy@hyperindex
5528   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5529     \@glsxtr@format@overridetrue
5530     \appto\theindex{\let\glshypernumber\@firstofone}%
5531   }
5532 \else
5533   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5534     \@glsxtr@format@overridetrue
5535     \appto\theindex{\let\glshypernumber\hyperpage}%
5536   }
5537 \fi
5538 }
5539 {
5540   \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5541     \@glsxtr@format@overridetrue
5542   }
5543 }
5544 \@onlypreamble\GlsXtrEnableIndexFormatOverride
```

doautoindexname

```
5545 \newcommand*{\glxtrdoautoindexname}[2]{%
5546   \glshasattribute{#1}{#2}%
5547   {%
```

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.

```
5548   \@glsxtr@autoindex@setname{#1}%
```

If the attribute value is simply "true" don't add an encap, otherwise use the value as the encap.

```
5549   \protected@edef\@glsxtr@attrval{\glsggetattribute{#1}{#2}}%
5550   \if@glsxtr@format@override
5551     \ifx\@glsnumberformat\@glsxtr@defaultnumberformat
5552     \else
5553       \let\@glsxtr@attrval\@glsnumberformat
5554     \fi
5555   \fi
5556   \ifdefstring{\@glsxtr@attrval}{true}%
5557   {}%
5558   {\eappto\@glo@name{\@glsxtr@autoindex@encap\@glsxtr@attrval}}%
5559   \expandafter\glxtrautoindex\expandafter{\@glo@name}%
5560   }%
5561   {%
5562 }
```

glxtrautoindex

```
5563 \newcommand*{\glxtrautoindex}{\index}
```

toindex@setname Assign \@glo@name for use with indexname attribute.

```
5564 \newcommand*{\@glsxtr@autoindex@setname}[1]{%
5565   \protected@edef\@glo@name{\glsxtr@autoindexentry{#1}}%
5566   \glsxtr@autoindexassignsort{\@glo@sort}{#1}%
5567   \@gls@checkmkidxchars\@glo@sort
5568   \@glsxtr@autoindex@doextra@esc\@glo@sort
5569   \epreto\@glo@name{\@glo@sort\@glsxtr@autoindex@at}%
5570 }
```

rautoindexentry Command used for the actual part when auto-indexing.

```
5571 \newcommand*{\glsxtr@autoindexentry}[1]{\string\glstryname{#1}}
```

trautoindexsort Used to assign the sort value when auto-indexing.

```
5572 \newcommand*{\glsxtr@autoindexassignsort}[2]{%
5573   \glsletentryfield{#1}{#2}{sort}%
5574 }
```

dex@doextra@esc

```
5575 \newcommand*{\@glsxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5576   \ifx\@glsxtr@autoindex@esc\@gls@quotechar
5577   \else
5578     \def\@gls@checkedmkidx{}%
5579     \edef\@glsxtr@checkspch{%
5580       \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
5581       \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
5582       \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5583     @@glsxtr@checkspch
5584     \let#1\@gls@checkedmkidx\relax
5585   \fi
```

Escape actual character unless it has already been escaped.

```
5586   \ifx\@glsxtr@autoindex@at\@gls@actualchar
5587   \else
5588     \def\@gls@checkedmkidx{}%
5589     \edef\@glsxtr@checkspch{%
5590       \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
5591       \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
5592       \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5593     @@glsxtr@checkspch
5594     \let#1\@gls@checkedmkidx\relax
5595   \fi
```

Escape level character unless it has already been escaped.

```
5596   \ifx\@glsxtr@autoindex@level\@gls@levelchar
5597   \else
5598     \def\@gls@checkedmkidx{}%
5599     \edef\@glsxtr@checkspch{%
5600       \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
```

```

5601      \noexpand\@empty\@glxtr@autoindex@level\noexpand\@nnil
5602      \@glxtr@autoindex@level\noexpand\@empty\noexpand\@glxtr@endescspch}%
5603      \@glxtr@checkspch
5604      \let#1\@glx@checkedmkidx\relax
5605 \fi

```

Escape encap character unless it has already been escaped.

```

5606 \ifx\@glxtr@autoindex@encap\@glx@encapchar
5607 \else
5608   \def\@glx@checkedmkidx{%
5609     \edef\@glxtr@checkspch{%
5610       \noexpand\@glxtr@autoindex@escencap\expandonce{#1}%
5611       \noexpand\@empty\@glxtr@autoindex@encap\noexpand\@nnil
5612       \@glxtr@autoindex@encap\noexpand\@empty\noexpand\@glxtr@endescspch}%
5613       \@glxtr@checkspch
5614       \let#1\@glx@checkedmkidx\relax
5615     \fi
5616 }

```

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.

```

5617 \newcommand*{\@glxtr@autoindex@at}{}

```

trSetActualChar Set the actual character.

```

5618 \newcommand*{\GlsXtrSetActualChar}[1]{%
5619   \gdef\@glxtr@autoindex@at{#1}%
5620   \def\@glxtr@autoindex@escat##1##2##3\@glxtr@endescspch{%
5621     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escat}{##1}{##2}{##3}%
5622   }%
5623 }
5624 \@onlypreamble\GlsXtrSetActualChar
5625 \makeatother
5626 \GlsXtrSetActualChar{@}
5627 \makeatletter

```

autoindex@encap Encap character for use with \index.

```

5628 \newcommand*{\@glxtr@autoindex@encap}{}

```

XtrSetEncapChar Set the encap character.

```

5629 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5630   \gdef\@glxtr@autoindex@encap{#1}%
5631   \def\@glxtr@autoindex@escencap##1##2##3\@glxtr@endescspch{%
5632     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escencap}{##1}{##2}{##3}%
5633   }%
5634 }
5635 \GlsXtrSetEncapChar{|}
5636 \@onlypreamble\GlsXtrSetEncapChar

```

autoindex@level Level character for use with \index.

```
5637 \newcommand*{\@glxtr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
5638 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5639   \gdef\@glxtr@autoindex@level{#1}%
5640   \def\@glxtr@autoindex@esclevel##1#1##2#1##3\@glxtr@endescspch{%
5641     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@esclevel}{##1}{##2}{##3}%
5642   }%
5643 }
5644 \GlsXtrSetLevelChar{!}
5645 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
5646 \newcommand*{\@glxtr@autoindex@esc}{}
```

lXtrSetEscChar Set the escape character.

```
5647 \newcommand*{\GlsXtrSetEscChar}[1]{%
5648   \gdef\@glxtr@autoindex@esc{#1}%
5649   \def\@glxtr@autoindex@escquote##1#1##2#1##3\@glxtr@endescspch{%
5650     \@glxtr@autoindex@escspch{#1}{\@glxtr@autoindex@escquote}{##1}{##2}{##3}%
5651   }%
5652 }
5653 \GlsXtrSetEscChar{"}
5654 \@onlypreamble\GlsXtrSetEscChar
```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```
5655 \ifdef\actualchar
5656 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5657 {}
```

Quote character \quotechar:

```
5658 \ifdef\quotechar
5659 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5660 {}
```

Level character \levelchar:

```
5661 \ifdef\levelchar
5662 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5663 {}
```

Encap character \encapchar:

```
5664 \ifdef\encapchar
5665 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5666 {}
```

leto@endescspch

```
5667 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}
```

toindex@esc@spch

```
@@@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}
```

```
5668 \newcommand*{@@glxtr@autoindex@escspch}[5]{%
5669   \@gls@tmpb=\expandafter{\@gls@checkedmkidx}%
5670   \toks@={#3}%
5671   \ifx\@nnil#3\relax
5672     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
5673   \else
5674     \ifx\@nnil#4\relax
5675       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@}%
5676       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5677         #4#5\@glxtr@endescspch}%
5678     \else
5679       \edef\@gls@checkedmkidx{\the\@gls@tmpb\the\toks@
5680         \@glxtr@autoindex@esc#1}%
5681       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
5682     \fi
5683   \fi
5684   \@glxtr@checkspch
5685 }
```

\Glossentrydesc Redefine to set the abbreviation format and accessibility support.

```
5686 \renewcommand*{\Glossentrydesc}[1]{%
5687   \glsdoifexistsorwarn{#1}%
5688   {%
5689     \glssetabbrvfmt{\glscategory{#1}}%
5690     \Glsaccessdesc{#1}%
5691   }%
5692 }
```

\lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
5693 \renewcommand*{\lossentrysymbol}[1]{%
5694   \glsdoifexistsorwarn{#1}%
5695   {%
5696     \glssetabbrvfmt{\glscategory{#1}}%
5697     \Glsaccesssymbol{#1}%
5698   }%
5699 }
```

\lossentrysymbol Redefine to set the abbreviation format and accessibility support.

```
5700 \renewcommand*{\Glossentrysymbol}[1]{%
5701   \glsdoifexistsorwarn{#1}%
5702   {%
5703     \glssetabbrvfmt{\glscategory{#1}}%
5704     \Glsaccesssymbol{#1}%
5705   }%
5706 }
```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`eInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
5707 \newcommand*{\GlsXtrEnableInitialTagging}{%
5708   \@ifstar\s@glsextr@enabletagging\@glsextr@enabletagging
5709 }
5710 \@onlypreamble\GlsXtrEnableInitialTagging
```

`r@enabletagging` Starred version undefines command.

```
5711 \newcommand*{\s@glsextr@enabletagging}[2]{%
5712   \undef#2%
5713   \@glsextr@enabletagging{#1}{#2}%
5714 }
```

`r@enabletagging` Internal command.

```
5715 \newcommand*{\@glsextr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
5716   \@for\@glsextr@cat:=#1\do
5717   {%
5718     \ifdefempty\@glsextr@cat
5719     {}%
5720     {\glsssetcategoryattribute{\@glsextr@cat}{tagging}{true}}%
5721   }%
5722   \newrobustcmd*#2[1]{##1}%
5723   \def\@glsextr@taggingcs{#2}%
5724   \renewcommand*\@glsextr@activate@initialtagging{%
5725     \let#2\@glsextr@tag
5726   }%
5727   \ifundef\@gls@preglossaryhook
5728   {\GlossariesExtraWarning{Initial tagging requires at least
5729     glossaries.sty v4.19 to work correctly}}%
5730   {}%
5731 }
```

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`

```
5732 \ifundef\mfu@checkword@do
5733 {
5734   \newcommand*{\mfu@checkword@do}[1]{%
5735     \ifdefstring{\mfu@checkword@arg}{#1}%
5736     {%
5737       \let\@mfu@domakefirstuc\@firstofone
5738       \listbreak
5739     }%
5740 }
```



```

5740 {}%
5741 }

```

`\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.

```

5742 \ifundef\mfu@checkword
5743 {
5744   \newcommand{\@glstr@do@titlecaps@warn}{%
5745     \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5746       support not available}%

```

One warning should suffice.

```

5747   \let\@glstr@do@titlecaps@warn\relax
5748 }
5749 }
5750 {
5751   \renewcommand*\mfu@checkword[1]{%
5752     \def\mfu@checkword@arg{#1}%
5753     \let\@mfu@domakefirstuc\makefirstuc
5754     \forlistloop\mfu@checkword@do\@mfu@nocaplist
5755   }
5756 }
5757 }
5758 {}% no patch required

```

`@titlecaps@warn` Do warning if title case not supported.

```

5759 \newcommand*\@glstr@do@titlecaps@warn{}

```

`@initialtagging` Used in `\printglossary` but at least v4.19 of `glossaries` required.

```

5760 \newcommand*\@glstr@activate@initialtagging{}

```

`\@glstr@tag` Definition of tagging command when used in glossary.

```

5761 \newrobustcmd*\@glstr@tag[1]{%
5762   \glstr@attribute{\glstr@currententrylabel}{tagging}{true}%
5763   {\glstr@tagfont{#1}}{#1}%
5764 }

```

`\glstr@tagfont` Used in the glossary.

```

5765 \newcommand*\glstr@tagfont[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.

```

5766 \ifdef\@glstr@preglossaryhook
5767 {
5768   \renewcommand*\@glstr@preglossaryhook{%
5769     \@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.

```

5770 \ifundef\@glstr@org@postdescription
5771 {%
5772 \let\@glstr@org@postdescription\glspostdescription
5773 \renewcommand*{\glspostdescription}{%
5774 \ifglentryexists{\glscurrententrylabel}%
5775 {%
5776 \glstrpostdescription
5777 \@glstr@org@postdescription
5778 }%
5779 }%
5780 }%
5781 }%
5782 {}%
```

Enable the options used by `\@glstrp`:

```

5783 \glossxtrsetpopts
5784 }%
5785 }
5786 {}
```

`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.

```

5787 \newcommand*{\glstrpostdescription}{%
5788 \csuse{glstrpostdesc\glscategory{\glscurrententrylabel}}%
5789 }
```

`postdescgeneral`

```

5790 \newcommand*{\glstrpostdescgeneral}{}%
```

`xtrpostdescterm`

```

5791 \newcommand*{\glstrpostdescterm}{}%
```

`postdescacronym`

```

5792 \newcommand*{\glstrpostdescacronym}{}%
```

`descabbreviation`

```

5793 \newcommand*{\glstrpostdescabbreviation}{}%
```

`glspostlinkhook` Redefine the post link hook used by commands like `\gls` to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```

5794 \renewcommand*{\glspostlinkhook}{%
5795 \ifglentryexists{\glslabel}{\glstrpostlinkhook}{}%
5796 }
```

`trpostlinkhook` The entry label should already be stored in `\glslabel` by `\@gls@link`.

```

5797 \newcommand*{\glsxtrpostlinkhook}{%
5798   \glsxtrdiscardperiod{\glslabel}%
5799   {\glsxtrpostlinkendsentence}%
5800   {\glsxtrpostlink}%
5801 }

```

`\glsxtrpostlink`

```

5802 \newcommand*{\glsxtrpostlink}{%
5803   \csuse{glsxtrpostlink\glscategory{\glslabel}}%
5804 }

```

`linkendsentence` Done by `\glsxtrpostlinkhook` if a full stop is discarded.

```

5805 \newcommand*{\glsxtrpostlinkendsentence}{%
5806   \ifcsdef{glsxtrpostlink\glscategory{\glslabel}}
5807   {%
5808     \csuse{glsxtrpostlink\glscategory{\glslabel}}%
5809     .\spacefactor\sfcode'\. \relax
5810   }%
5811   {%
5812     \spacefactor\sfcode'\. \relax
5813   }%
5814 }

```

Put the full stop back.

Assume the full stop was discarded because the entry ends with a period, so adjust the space-
factor.

`dDescOnFirstUse` Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```

5815 \newcommand*{\glsxtrpostlinkAddDescOnFirstUse}{%
5816   \glsxtrifwasfirstuse{\space(\glsaccessdesc{\glslabel})}{}%
5817 }

```

`ymbolOnFirstUse` Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```

5818 \newcommand*{\glsxtrpostlinkAddSymbolOnFirstUse}{%
5819   \glsxtrifwasfirstuse
5820   {%
5821     \ifglshassymbol{\glslabel}{\space(\glsaccesssymbol{\glslabel})}{}%
5822   }%
5823   {}%
5824 }

```

`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).

```

5825 \newcommand*{\glxtrdiscardperiod}[3]{%
5826   \glxtrifwasfirstuse
5827   {%
5828     \glusifattribute{#1}{retainfirstuseperiod}{true}%
5829     {#3}%
5830     {%
5831       \glusifattribute{#1}{discardperiod}{true}%
5832       {%
5833         \glusifplural
5834         {%
5835           \glusifattribute{#1}{pluraldiscardperiod}{true}%
5836           {\glxtrifperiod{#2}{#3}}%
5837           {#3}%
5838         }%
5839         {%
5840           \glxtrifperiod{#2}{#3}%
5841         }%
5842       }%
5843     }%
5844   }%
5845 }%
5846 {%
5847   \glusifattribute{#1}{discardperiod}{true}%
5848   {%
5849     \glusifplural
5850     {%
5851       \glusifattribute{#1}{pluraldiscardperiod}{true}%
5852       {\glxtrifperiod{#2}{#3}}%
5853       {#3}%
5854     }%
5855     {%
5856       \glxtrifperiod{#2}{#3}%
5857     }%
5858   }%
5859   {#3}%
5860 }%
5861 }

```

`\glxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```

5862 \newcommand*{\glxtrifperiod}[1]{\new@ifnextchar.\{\@firstoftwo{#1}}

```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```

5863 \newcommand*{\glxtr@punclist}{.,;?!}

```

`\punctuationmark` Add character to punctuation list.

```

5864 \newcommand*{\glxtraddpunctuationmark}[1]{\appto\glxtr@punclist{#1}}

```

unctuationmarks Reset the punctuation list.

```
5865 \newcommand*{\glxtrsetpunctuationmarks}[1]{\def\glxtr@punclist{#1}}
```

```
\glxtrifpunc    \glxtrifnextpunc{<true part>}{<false part>}
```

Test if this is followed by a punctuation mark. (Adapted from \new@ifnextchar.)

```
5866 \newcommand*{\glxtrifnextpunc}[2]{%
5867   \def\reserved@a{#1}%
5868   \def\reserved@b{#2}%
5869   \futurelet\@glspunc@token\glxtr@ifnextpunc
5870 }
```

glxtr@ifnextpunc

```
5871 \newcommand*{\glxtr@ifnextpunc}{%
5872   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5873   \reserved@b
5874 }
```

glxtr@ifpunctoken Test if the token given in the first argument is in the punctuation list.

```
5875 \newcommand*{\glxtr@ifpunctoken}[1]{%
5876   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5877 }
```

glxtr@ifpunctoken

```
5878 \def\@glxtr@ifpunctoken#1#2{%
5879   \let\reserved@d=#2%
5880   \ifx\reserved@d\@nnil
5881     \let\glxtr@next\@glxtr@notfoundinlist
5882   \else
5883     \ifx#1\reserved@d
5884       \let\glxtr@next\@glxtr@foundinlist
5885     \else
5886       \let\glxtr@next\@glxtr@ifpunctoken
5887     \fi
5888   \fi
5889   \glxtr@next#1%
5890 }
```

glxtr@foundinlist

```
5891 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

@notfoundinlist

```
5892 \def\@glxtr@notfoundinlist#1{\@secondoftwo}
```

glsxtrdopostpunc

```
\glsxtrdopostpunc{<code>}
```

If this is followed by a punctuation character, do <code> after the character otherwise do <code> before whatever comes next.

```
5893 \newcommand{\glsxtrdopostpunc}[1]{%
5894   \glsxtrifnextpunc{\@glsxtr@swaptwo{#1}}{#1}%
5895 }
```

@glsxtr@swaptwo

```
5896 \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.

```
5897 \define@key{glsxtrabbrv}{category}{%
5898   \edef\glscategorylabel{#1}%
5899   \ifcsdef{@glsabbrv@current@#1}%
5900   {%
```

Warning should already have been issued.

```
5901   \let\@glsxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
5902   \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
5903   \glsxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
5904   \let\GlsXtrWarnDeprecatedAbbrStyle\@glsxtr@orgwarndep
5905 }%
5906 {}%
5907 }
```

Save the short plural form. This may be needed before the entry is defined.

```
5908 \define@key{glsxtrabbrv}{shortplural}{%
5909   \def\@gls@shortpl{#1}%
5910 }
```

Similarly for the long plural form.

```
5911 \define@key{glsxtrabbrv}{longplural}{%
5912   \def\@gls@longpl{#1}%
5913 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
5914 \newtoks\glsshortpltok
```

```

\glslongpltok
5915 \newtoks\glslongpltok

\glsxtr@insertdots Provided in case user wants to automatically insert dots between each letter of the abbrevi-
ation. This should be applied before defining the abbreviation to optimise the document
build. (Otherwise, it would have to be done each time the short form is required, which is an
unnecessary waste of time.) For this to work the short form must be expanded when passed
to \newabbreviation. Note that explicitly using the short or shortplural keys will override
this.
5916 \newcommand*{\@glsxtr@insertdots}[2]{%
5917   \def#1{}%
5918   \@glsxtr@insert@dots#1#2\@nnil
5919 }

\glsxtr@insert@dots
5920 \newcommand*{\@glsxtr@insert@dots}[2]{%
5921   \ifx\@nnil#2\relax
5922   \let\@glsxtr@insert@dots@next\@gobble
5923   \else
5924     \ifx\relax#2\relax
5925     \else
5926       \appto#1{#2.}%
5927     \fi
5928     \let\@glsxtr@insert@dots@next\@glsxtr@insert@dots
5929     \fi
5930     \@glsxtr@insert@dots@next#1%
5931 }

Similarly provide a way of replacing spaces with \glsxtrwordsep, which first needs to be
defined:

\glsxtrwordsep
5932 \newcommand*{\glsxtrwordsep}{\space}

Each word is marked with

\glsxtrword
5933 \newcommand*{\glsxtrword}[1]{#1}

\glsxtr@markwordseps
5934 \newcommand*{\@glsxtr@markwordseps}[2]{%
5935   \def#1{}%
5936   \@glsxtr@mark@wordseps#1#2 \@nnil
5937 }

\glsxtr@mark@wordseps
5938 \def\@glsxtr@mark@wordseps#1#2 #3{%
5939   \ifdefempty{#1}%

```

```

5940 {\def#1{\protect\glstrword{#2}}}%
5941 {\appto#1{\protect\glstrwordsep\protect\glstrword{#2}}}%
5942 \ifx\@nnil#3\relax
5943 \let\@glstr@mark@wordseps@next\relax
5944 \else
5945 \def\@glstr@mark@wordseps@next{%
5946 \@glstr@mark@wordseps#1#3}%
5947 \fi
5948 \@glstr@mark@wordseps@next
5949 }

```

newabbreviation Define a new generic abbreviation.

```

5950 \newcommand*{\newabbreviation}[4][]{%
5951 \glstr@newabbreviation{#1}{#2}{#3}{#4}%
5952 }

```

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.)

```

5953 \newcommand*{\glstr@newabbreviation}[4]{%
5954 \glscopylisttok{#1}%
5955 \glscopylabeltok{#2}%
5956 \glscopyshorttok{#3}%
5957 \glscopylongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

5958 \def\glstr@orgshort{#3}%
5959 \def\glstr@orglong{#4}%

```

Get the category.

```

5960 \def\glscategorylabel{abbreviation}%
5961 \glstr@applyabbrvstyle{\@glsabrv@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```

5962 \setkeys*{\glstr@abbrv}{shortplural,longplural}{#1}%

```

Set the default long plural

```

5963 \def\@glslongpl{#4\glspluralsuffix}%
5964 \let\@glslongpl\@glslongpl

```

Has the markwords attribute been set?

```

5965 \glscopycategoryattribute{\glscategorylabel}{markwords}{true}%
5966 {%
5967 \@glstr@markwordseps\@glslong{#4}%
5968 \expandafter\def\expandafter\@glslongpl\expandafter
5969 {\@glslong\glspluralsuffix}%
5970 \let\@glslongpl\@glslongpl

```

Update \glslongtok.

```

5971 \expandafter\glslongtok\expandafter{\@glslong}%
5972 }%
5973 {}%

```


Has the markshortwords attribute been set? (Not compatible with insertdots.)

```
5974 \glsifcategoryattribute{\glscategorylabel}{markshortwords}{true}%  
5975 {%  
5976   \@glstr@markwordseps\@gls@short{#3}%  
5977 }%  
5978 {%
```

Has the insertdots attribute been set?

```
5979 \glsifcategoryattribute{\glscategorylabel}{insertdots}{true}%  
5980 {%  
5981   \@glstr@insertdots\@gls@short{#3}%  
5982   \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%  
5983 }%  
5984 {\def\@gls@short{#3}}%  
5985 }%
```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```
5986 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%  
5987 {%  
5988   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5989     '\abbrvpluralsuffix}%  
5990 }%  
5991 {%
```

Has the noshortplural attribute been set?

```
5992 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%  
5993 {%  
5994   \let\@gls@shortpl\@gls@short  
5995 }%  
5996 {%  
5997   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short  
5998     \abbrvpluralsuffix}%  
5999 }%  
6000 }%
```

Update \glsshorttok:

```
6001 \expandafter\glsshorttok\expandafter{\@gls@short}%
```

Hook for further customisation if required:

```
6002 \glstrnewabbrevpresetkeyhook{#1}{#2}{#3}%
```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary. Ignore the category key (already obtained).

```
6003 \setkeys*{glstrabbrv}[category]{#1}%
```

Has the plural been explicitly set?

```
6004 \ifx\@gls@default@longpl\@gls@longpl  
6005 \else
```

Has the markwords attribute been set?

```
6006 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%  
6007 {%
```

```

6008      \expandafter\@glxtr@markwordseps\expandafter\@gls@longpl\expandafter
6009      {\@gls@longpl}%
6010    }%
6011    {}%
6012  \fi

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

6013  \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6014  \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

6015  \newabbreviationhook

```

Define this entry:

```

6016  \protected@edef\@do@newglossaryentry{%
6017    \noexpand\newglossaryentry{\the\glslabeltok}%
6018    {%
6019      type=\glxtrabbrvtype,%
6020      category=abbreviation,%
6021      short={\the\glsshorttok},%
6022      shortplural={\the\glsshortpltok},%
6023      long={\the\glslongtok},%
6024      longplural={\the\glslongpltok},%
6025      name={\the\glsshorttok},%
6026      \CustomAbbreviationFields,%
6027      \the\glskeylisttok
6028    }%
6029  }%
6030  \@do@newglossaryentry
6031  \GlsXtrPostNewAbbreviation
6032 }

```

`evpresetkeyhook` Hook for extra stuff in `\newabbreviation`

```

6033 \newcommand*{\glxtrnewabbrevpresetkeyhook}[3]{}

```

`NewAbbreviation` Hook used by abbreviation styles.

```

6034 \newcommand*{\GlsXtrPostNewAbbreviation}{}

```

`bbreviationhook` Hook for use with `\newabbreviation`.

```

6035 \newcommand*{\newabbreviationhook}{}

```

`reviationFields`

```

6036 \newcommand*{\CustomAbbreviationFields}{}

```

`\glxtrparen` For the parenthetical styles.

```

6037 \newcommand*{\glxtrparen}[1]{(#1)}

```

`lsxtrfullformat` Full format without case change.

```

6038 \newcommand*{\glxtrfullformat}[2]{%

```

```

6039 \glsfirstlongfont{\glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6040 \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6041 }

```

`\glsxtrfullformat` Full format with case change.

```

6042 \newcommand*{\Glsxtrfullformat}[2]{%
6043 \glsfirstlongfont{\Glsaccesslong{#1}}#2\glsxtrfullsep{#1}%
6044 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6045 }

```

`\glsxtrfullplformat` Plural full format without case change.

```

6046 \newcommand*{\Glsxtrfullplformat}[2]{%
6047 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6048 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6049 }

```

`\glsxtrfullplformat` Plural full format with case change.

```

6050 \newcommand*{\Glsxtrfullplformat}[2]{%
6051 \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glsxtrfullsep{#1}%
6052 \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6053 }

```

`\glsxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```

6054 \newcommand*{\glsxtrfullsep}[1]{\space}

```

In-line formats in case first use isn't compatible with `\glsentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glsxtrinfullformat` Full format without case change.

```

6055 \newcommand*{\Glsxtrinfullformat}{\glsxtrfullformat}

```

`\glsxtrinfullformat` Full format with case change.

```

6056 \newcommand*{\Glsxtrinfullformat}{\Glsxtrfullformat}

```

`\glsxtrfullplformat` Plural full format without case change.

```

6057 \newcommand*{\Glsxtrinfullplformat}{\glsxtrfullplformat}

```

`\glsxtrinfullplformat` Plural full format with case change.

```

6058 \newcommand*{\Glsxtrinfullplformat}{\Glsxtrfullplformat}

```

Redefine `\glsentryfull` etc to use the inline format. Since these commands as supposed to be expandable, they can only use the currently applied style. If there are mixed styles, you'll need to use the `\glsxtrfull` set of commands instead.

`\glsentryfull`

```

6059 \renewcommand*{\glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}

```

`\Glsentryfull`

```

6060 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinfullformat{#1}{}}

```

`\glsentryfullpl`

```
6061 \renewcommand*{\glsentryfullpl}[1]{\glsxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
6062 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\glsfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
6063 \newcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.

```
6064 \newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}
```

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.

```
6065 \newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}
```

`\glsabbrvdefaultfont`

```
6066 \newcommand*{\glsabbrvdefaultfont}[1]{#1}
```

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.

```
6067 \newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}
```

`\glslongdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.

```
6068 \newcommand*{\glslongdefaultfont}[1]{#1}
```

`\glsfirstlongfont` Font changing command used for the long form on first use or in the full format.

```
6069 \newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}
```

`\glsfirstlongdefaultfont`

```
6070 \newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}
```

`\glsbrvppluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.

```
6071 \newcommand*{\glsxtrabbrvppluralsuffix}{\glspluralsuffix}
```

`\glsabbrvppluralsuffix` Default plural suffix.

```
6072 \newcommand*{\abbrvppluralsuffix}{\glsxtrabbrvppluralsuffix}
```

`\glsxtrfull` Full form (no case-change).

```
6073 \newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}
6074 \newcommand*{\ns@glsxtrfull}[2][{}]{%
6075   \new@ifnextchar[{\@glsxtr@full{#1}{#2}}{%
6076     {\@glsxtr@full{#1}{#2}[]}%
6077 }
```

`\@glxtr@full` Low-level macro:

```
6078 \def\@glxtr@full#1#2[#3]{%
6079   \glstoifexists{#2}%
6080   {%
6081     \glsetabbrvfmt{\glscategory{#2}}%
6082     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6083     \let\gl@ifplural\@secondoftwo
6084     \let\glscapscase\@firstofthree
6085     \let\glinsert\@empty
6086     \def\glscustomtext{\glxtrinlinefullformat{#2}{#3}}%
```

What should `\glxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
6087   \glxtrsetupfulldefs
6088   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6089   }%
6090   \glspostlinkhook
6091 }
```

`trsetupfulldefs`

```
6092 \newcommand*{\glxtrsetupfulldefs}{%
6093   \let\glxtrifwasfirstuse\@firstoftwo
6094 }
```

`\Glsxtrfull` Full form (first letter uppercase).

```
6095 \newrobustcmd*{\Glsxtrfull}{\@gl@hyp@opt\@ns@Glsxtrfull}
6096 \newcommand*\ns@Glsxtrfull[2][{}]{%
6097   \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
6098   {\@Glsxtr@full{#1}{#2}[]}%
6099 }
```

`\@Glsxtr@full` Low-level macro:

```
6100 \def\@Glsxtr@full#1#2[#3]{%
6101   \glstoifexists{#2}%
6102   {%
6103     \glsetabbrvfmt{\glscategory{#2}}%
6104     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6105     \let\gl@ifplural\@secondoftwo
6106     \let\glscapscase\@secondofthree
6107     \let\glinsert\@empty
6108     \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6109     \glxtrsetupfulldefs
6110     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6111     }%
6112     \glspostlinkhook
6113 }
```

`\GLSxtrfull` Full form (all uppercase).

```
6114 \newrobustcmd*{\GLSxtrfull}{\@gls@hyp@opt\@ns@GLSxtrfull}
6115 \newcommand*\ns@GLSxtrfull[2][\@gls@hyp@opt\@ns@GLSxtrfull]{%
6116   \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
6117   {\@GLSxtr@full{#1}{#2}[]}%
6118 }
```

`\@GLSxtr@full` Low-level macro:

```
6119 \def\@GLSxtr@full#1#2[#3]{%
6120   \glsdoifexists{#2}%
6121   {%
6122     \glssetabbrvfmt{\gls@category{#2}}%
6123     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6124     \let\glsifplural\@secondoftwo
6125     \let\gls@scapscase\@thirdofthree
6126     \let\glsinsert\@empty
6127     \def\gls@customtext{\mfirstucMakeUppercase{\glsxtrinlinefullformat{#2}{#3}}}%
6128     \glsxtrsetupfulldefs
6129     \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
6130   }%
6131   \gls@postlinkhook
6132 }
```

`\glsxtrfullpl` Plural full form (no case-change).

```
6133 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\@ns@glsxtrfullpl}
6134 \newcommand*\ns@glsxtrfullpl[2][\@gls@hyp@opt\@ns@glsxtrfullpl]{%
6135   \new@ifnextchar[\@glsxtr@fullpl{#1}{#2}}%
6136   {\@glsxtr@fullpl{#1}{#2}[]}%
6137 }
```

`\@glsxtr@fullpl` Low-level macro:

```
6138 \def\@glsxtr@fullpl#1#2[#3]{%
6139   \glsdoifexists{#2}%
6140   {%
6141     \glssetabbrvfmt{\gls@category{#2}}%
6142     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6143     \let\glsifplural\@firstoftwo
6144     \let\gls@scapscase\@firstofthree
6145     \let\glsinsert\@empty
6146     \def\gls@customtext{\glsxtrinlinefullplformat{#2}{#3}}%
6147     \glsxtrsetupfullpldefs
6148     \@gls@link[#1]{#2}{\csname gls@\gls@type @entryfmt\endcsname}%
6149   }%
6150   \gls@postlinkhook
6151 }
```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```
6152 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6153 \newcommand*\ns@Glsxtrfullpl[2][\@gls@hyp@opt\@ns@Glsxtrfullpl]{%
6154   \new@ifnextchar[\@Glsxtr@fullpl{#1}{#2}}%
6155   {\@Glsxtr@fullpl{#1}{#2}[]}%
6156 }
```

```

6154 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6155         {\@Glsxtr@fullpl{#1}{#2}[]}%
6156 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6157 \def\@Glsxtr@fullpl#1#2[#3]{%
6158   \glsdoifexists{#2}%
6159   {%
6160     \glssetabbrvfmt{\glscategory{#2}}%
6161     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6162     \let\glsifplural\@firstoftwo
6163     \let\glscapscase\@secondofthree
6164     \let\glsinsert\@empty
6165     \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6166     \glsxtrsetupfulldefs
6167     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6168   }%
6169   \glspostlinkhook
6170 }

```

`\Glsxtrfullpl` Plural full form (all upper case).

```

6171 \newrobustcmd*{\Glsxtrfullpl}{\@gl@hyp@opt\@ns@Glsxtrfullpl}
6172 \newcommand*\ns@Glsxtrfullpl[2][]{%
6173   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6174   {\@Glsxtr@fullpl{#1}{#2}[]}%
6175 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6176 \def\@Glsxtr@fullpl#1#2[#3]{%
6177   \glsdoifexists{#2}%
6178   {%
6179     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6180     \let\glsifplural\@firstoftwo
6181     \let\glscapscase\@thirdofthree
6182     \let\glsinsert\@empty
6183     \def\glscustomtext{%
6184       \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6185     \glsxtrsetupfulldefs
6186     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6187   }%
6188   \glspostlinkhook
6189 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

6190 \newrobustcmd*{\glsxtrshort}{\@gl@hyp@opt\@ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6191 \newcommand*{\ns@glstrshort}[2] [] {%
6192   \new@ifnextchar[{\@glstrshort{#1}{#2}}{\@glstrshort{#1}{#2} []}%
6193 }

```

Read in the final optional argument:

```

6194 \def\@glstrshort#1#2[#3] {%
6195   \glstoifexists{#2}%
6196   {%

```

Need to make sure \glabbrvfont is set correctly.

```

6197     \glsetabbrvfmt{\glscategory{#2}}%
6198     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6199     \let\glstrifwasfirstuse\@secondoftwo
6200     \let\gl@ifplural\@secondoftwo
6201     \let\glscapscase\@firstofthree
6202     \let\glinsert\@empty
6203     \def\glscustomtext{%
6204       \glabbrvfont{\glaccessshort{#2}\ifglstrinsertinside#3\fi}%
6205       \ifglstrinsertinside\else#3\fi
6206     }%
6207     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6208   }%
6209   \glspostlinkhook
6210 }

```

\Glsxtrshort

```

6211 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\ns@Glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6212 \newcommand*{\ns@Glsxtrshort}[2] [] {%
6213   \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2} []}%
6214 }

```

Read in the final optional argument:

```

6215 \def\@Glsxtrshort#1#2[#3] {%
6216   \glstoifexists{#2}%
6217   {%
6218     \glsetabbrvfmt{\glscategory{#2}}%
6219     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6220     \let\glstrifwasfirstuse\@secondoftwo
6221     \let\gl@ifplural\@secondoftwo
6222     \let\glscapscase\@secondofthree
6223     \let\glinsert\@empty
6224     \def\glscustomtext{%
6225       \glabbrvfont{\Glsaccessshort{#2}\ifglstrinsertinside#3\fi}%
6226       \ifglstrinsertinside\else#3\fi
6227     }%
6228     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6229   }%
6230   \glspostlinkhook
6231 }

```


\GLSxtrshort

```
6232 \newrobustcmd*{\GLSxtrshort}{\@gls@hyp@opt\ns@GLSxtrshort}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6233 \newcommand*{\ns@GLSxtrshort}[2] [] {%
6234   \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2} []}]%
6235 }
```

Read in the final optional argument:

```
6236 \def\@GLSxtrshort#1#2[#3] {%
6237   \glsdoifexists{#2}%
6238   {%
6239     \glssetabbrvfmt{\glscategory{#2}}%
6240     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6241     \let\glsxtrifwasfirstuse\@secondoftwo
6242     \let\glsifplural\@secondoftwo
6243     \let\glsapscase\@thirdofthree
6244     \let\glsinsert\@empty
6245     \def\glscustomtext{%
6246       \mfirstucMakeUppercase
6247       {\glsabbrvfont{\glsaccessshort{#2}\ifglsxtrininsertinside#3\fi}%
6248       \ifglsxtrininsertinside\else#3\fi
6249     }%
6250   }%
6251   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6252   }%
6253   \glspostlinkhook
6254 }
```

\glsxtrlong

```
6255 \newrobustcmd*{\glsxtrlong}{\@gls@hyp@opt\ns@glsxtrlong}
```

Define the un-starred form. Need to determine if there is a final optional argument

```
6256 \newcommand*{\ns@glsxtrlong}[2] [] {%
6257   \new@ifnextchar[{\@glsxtrlong{#1}{#2}}{\@glsxtrlong{#1}{#2} []}]%
6258 }
```

Read in the final optional argument:

```
6259 \def\@glsxtrlong#1#2[#3] {%
6260   \glsdoifexists{#2}%
6261   {%
6262     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6263     \let\glsxtrifwasfirstuse\@secondoftwo
6264     \let\glsifplural\@secondoftwo
6265     \let\glsapscase\@firstofthree
6266     \let\glsinsert\@empty
6267     \def\glscustomtext{%
6268       \glsfont{\glsaccesslong{#2}\ifglsxtrininsertinside#3\fi}%
6269       \ifglsxtrininsertinside\else#3\fi
6270     }%
6271     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%

```

```

6272 }%
6273 \glspostlinkhook
6274 }

```

\Glsxtrlong

```

6275 \newrobustcmd*{\Glsxtrlong}{\@gls@hyp@opt\@ns@Glsxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
6276 \newcommand*{\ns@Glsxtrlong}[2] [] {%
6277   \new@ifnextchar[{\@Glsxtrlong{#1}{#2}}{\@Glsxtrlong{#1}{#2} []}%
6278 }

  Read in the final optional argument:
6279 \def\@Glsxtrlong#1#2[#3]{%
6280   \glsdoifexists{#2}%
6281   {%
6282     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6283     \let\glstrifwasfirstuse\@secondoftwo
6284     \let\glsifplural\@secondoftwo
6285     \let\glscapscase\@thirdofthree
6286     \let\glsinsert\@empty
6287     \def\glscustomtext{%
6288       \glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6289       \ifglstrinsertinside\else#3\fi
6290     }%
6291     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6292   }%
6293   \glspostlinkhook
6294 }

```

\GLSxtrlong

```

6295 \newrobustcmd*{\GLSxtrlong}{\@gls@hyp@opt\@ns@GLSxtrlong}

  Define the un-starred form. Need to determine if there is a final optional argument
6296 \newcommand*{\ns@GLSxtrlong}[2] [] {%
6297   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6298 }

  Read in the final optional argument:
6299 \def\@GLSxtrlong#1#2[#3]{%
6300   \glsdoifexists{#2}%
6301   {%
6302     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6303     \let\glstrifwasfirstuse\@secondoftwo
6304     \let\glsifplural\@secondoftwo
6305     \let\glscapscase\@thirdofthree
6306     \let\glsinsert\@empty
6307     \def\glscustomtext{%
6308       \mfirstucMakeUppercase
6309       {\glslongfont{\Glsaccesslong{#2}\ifglstrinsertinside#3\fi}%
6310       \ifglstrinsertinside\else#3\fi

```

```

6311     }%
6312     }%
6313     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6314     }%
6315     \glspostlinkhook
6316 }

```

Plural short forms:

\glsxtrshortpl

```

6317 \newrobustcmd*{\glsxtrshortpl}{\@gls@hyp@opt\@ns@glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6318 \newcommand*{\ns@glsxtrshortpl}[2][ ]{%
6319   \new@ifnextchar[{\@glsxtrshortpl{#1}{#2}}{\@glsxtrshortpl{#1}{#2}[]}%
6320 }

```

Read in the final optional argument:

```

6321 \def\@glsxtrshortpl#1#2[#3]{%
6322   \glsdoifexists{#2}%
6323   {%
6324     \glssetabbrvfmt{\glscategory{#2}}%
6325     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6326     \let\glsxtrifwasfirstuse\@secondoftwo
6327     \let\glsifplural\@firstoftwo
6328     \let\glscapscase\@firstofthree
6329     \let\glsinsert\@empty
6330     \def\glscustomtext{%
6331       \glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6332       \ifglsxtrininsertinside\else#3\fi
6333     }%
6334     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6335     }%
6336     \glspostlinkhook
6337 }

```

\Glsxtrshortpl

```

6338 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\@ns@Glsxtrshortpl}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6339 \newcommand*{\ns@Glsxtrshortpl}[2][ ]{%
6340   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2}[]}%
6341 }

```

Read in the final optional argument:

```

6342 \def\@Glsxtrshortpl#1#2[#3]{%
6343   \glsdoifexists{#2}%
6344   {%
6345     \glssetabbrvfmt{\glscategory{#2}}%
6346     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6347     \let\glsxtrifwasfirstuse\@secondoftwo

```

```

6348 \let\glsifplural\@firstoftwo
6349 \let\glscapscase\@secondofthree
6350 \let\glsinsert\@empty
6351 \def\glscustomtext{%
6352     \glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
6353     \ifglsxtrinsertinside\else#3\fi
6354 }%
6355 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6356 }%
6357 \glspostlinkhook
6358 }

```

\GLSxtrshortpl

```

6359 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\ns@GLSxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6360 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6361     \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6362 }

    Read in the final optional argument:
6363 \def\@GLSxtrshortpl#1#2[#3]{%
6364     \glsdoifexists{#2}%
6365     {%
6366         \glssetabbrvfmt{\glscategory{#2}}%
6367         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6368         \let\glxtrifwasfirstuse\@secondoftwo
6369         \let\glsifplural\@firstoftwo
6370         \let\glscapscase\@thirdofthree
6371         \let\glsinsert\@empty
6372         \def\glscustomtext{%
6373             \mfirstucMakeUppercase
6374             {\glsabbrvfont{\Glsaccessshortpl{#2}\ifglsxtrinsertinside#3\fi}%
6375             \ifglsxtrinsertinside\else#3\fi
6376         }%
6377     }%
6378     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6379 }%
6380 \glspostlinkhook
6381 }

```

Plural long forms:

\glxtrlongpl

```

6382 \newrobustcmd*{\glxtrlongpl}{\@gls@hyp@opt\ns@glxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6383 \newcommand*{\ns@glxtrlongpl}[2] [] {%
6384     \new@ifnextchar[{\@glxtrlongpl{#1}{#2}}{\@glxtrlongpl{#1}{#2} []}%
6385 }

```

Read in the final optional argument:

```

6386 \def\@glxstrlongpl#1#2[#3]{%
6387   \glstoifexists{#2}%
6388   {%
6389     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6390     \let\glxstrifwasfirstuse\@secondoftwo
6391     \let\gl@ifplural\@firstoftwo
6392     \let\glscapscase\@firstofthree
6393     \let\glinsert\@empty
6394     \def\glscustomtext{%
6395       \glslongfont{\glaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6396       \ifglxtrininsertinside\else#3\fi
6397     }%
6398     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6399   }%
6400   \glspostlinkhook
6401 }

```

\Glsxtrlongpl

```

6402 \newrobustcmd*{\Glsxtrlongpl}{\@gl@hyp@opt\ns@Glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6403 \newcommand*{\ns@Glsxtrlongpl}[2][]{%
6404   \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2}[]}%
6405 }

```

Read in the final optional argument:

```

6406 \def\@Glsxtrlongpl#1#2[#3]{%
6407   \glstoifexists{#2}%
6408   {%
6409     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6410     \let\glxstrifwasfirstuse\@secondoftwo
6411     \let\gl@ifplural\@firstoftwo
6412     \let\glscapscase\@secondofthree
6413     \let\glinsert\@empty
6414     \def\glscustomtext{%
6415       \glslongfont{\Glsaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6416       \ifglxtrininsertinside\else#3\fi
6417     }%
6418     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6419   }%
6420   \glspostlinkhook
6421 }

```

\GLSxtrlongpl

```

6422 \newrobustcmd*{\GLSxtrlongpl}{\@gl@hyp@opt\ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6423 \newcommand*{\ns@GLSxtrlongpl}[2][]{%
6424   \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2}[]}%
6425 }

```

Read in the final optional argument:

```

6426 \def\@GLSxtrlongpl#1#2[#3]{%
6427   \glsdoifexists{#2}%
6428   {%
6429     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6430     \let\glxtrifwasfirstuse\@secondoftwo
6431     \let\glsifplural\@firstoftwo
6432     \let\glscapscase\@thirdofthree
6433     \let\glinsert\@empty
6434     \def\glscustomtext{%
6435       \mfirstucMakeUppercase
6436       {\glslongfont{\glssaccesslongpl{#2}\ifglxtrininsertinside#3\fi}%
6437       \ifglxtrininsertinside\else#3\fi
6438     }%
6439   }%
6440   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6441 }%
6442 \glspostlinkhook
6443 }

```

`\glsssetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6444 \newcommand*{\glsssetabbrvfmt}[1]{%
6445   \ifcsdef{@glsabbrv@current@#1}%
6446   {\glxtr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6447   {\glxtr@applyabbrvfmt{@glsabbrv@current@abbreviation}}%
6448 }

```

`glsuseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6449 \newrobustcmd*{\glssuseabbrvfont}[2]{\glsssetabbrvfmt{#2}\glssabbrvfont{#1}}

```

`\glssuselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6450 \newrobustcmd*{\glssuselongfont}[2]{\glsssetabbrvfmt{#2}\glsslongfont{#1}}

```

`sxtrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6451 \newcommand*{\glssxtrgenabbrvfmt}{%
6452   \ifdefempty\glscustomtext
6453   {%
6454     \ifglssused\glslabel
6455     {%

```

Subsequent use:

```

6456     \glsifplural
6457     {%

```

Subsequent plural form:

```

6458     \glscapscase
6459     {%

```

Subsequent plural form, don't adjust case:

```
6460      \glxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6461      }%  
6462      {%
```

Subsequent plural form, make first letter upper case:

```
6463      \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%  
6464      }%  
6465      {%
```

Subsequent plural form, all caps:

```
6466      \mfirstucMakeUppercase  
6467      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%  
6468      }%  
6469      }%  
6470      {%
```

Subsequent singular form

```
6471      \glscapscase  
6472      {%
```

Subsequent singular form, don't adjust case:

```
6473      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6474      }%  
6475      {%
```

Subsequent singular form, make first letter upper case:

```
6476      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%  
6477      }%  
6478      {%
```

Subsequent singular form, all caps:

```
6479      \mfirstucMakeUppercase  
6480      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%  
6481      }%  
6482      }%  
6483      }%  
6484      {%
```

First use:

```
6485      \glsifplural  
6486      {%
```

First use plural form:

```
6487      \glscapscase  
6488      {%
```

First use plural form, don't adjust case:

```
6489      \glxtrfullplformat{\glslabel}{\glsinsert}%  
6490      }%  
6491      {%
```

First use plural form, make first letter upper case:

```
6492      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
6493      }%
6494      {%
```

First use plural form, all caps:

```
6495      \mfirstucMakeUppercase
6496      {\glsxtrfullplformat{\glslabel}{\glsinsert}}%
6497      }%
6498      }%
6499      {%
```

First use singular form

```
6500      \glscapscase
6501      {%
```

First use singular form, don't adjust case:

```
6502      \glsxtrfullformat{\glslabel}{\glsinsert}%
6503      }%
6504      {%
```

First use singular form, make first letter upper case:

```
6505      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6506      }%
6507      {%
```

First use singular form, all caps:

```
6508      \mfirstucMakeUppercase
6509      {\glsxtrfullformat{\glslabel}{\glsinsert}}%
6510      }%
6511      }%
6512      }%
6513      }%
6514      {%
```

User supplied text.

```
6515      \glscustomtext
6516      }%
6517 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6518 \newcommand*{\glsxtrsubsequentfmt}[2]{%
6519   \glsabbrvfont{\glsaccessshort{#1}\ifglsxtrininsertinside #2\fi}%
6520   \ifglsxtrininsertinside \else#2\fi
6521 }
6522 \let\glsxtrdefaultsubsequentfmt\glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6523 \newcommand*{\glsxtrsubsequentplfmt}[2]{%
6524   \glsabbrvfont{\glsaccessshortpl{#1}\ifglsxtrininsertinside #2\fi}%
6525   \ifglsxtrininsertinside \else#2\fi
```



```

6526 }
6527 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt

trsubsequentfmt Subsequent use format (singular, first letter uppercase).
6528 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6529   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
6530   \ifglxtrininsertinside \else#2\fi
6531 }
6532 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt

subsequentplfmt Subsequent use format (plural, first letter uppercase).
6533 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6534   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
6535   \ifglxtrininsertinside \else#2\fi
6536 }
6537 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

1.6.1 Abbreviation Styles Setup

```

breviationstyle
6538 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6539   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6540   {%
6541     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6542   }%
6543   {%
6544     \ifcsstring{@glsabbrv@current@#1}{#2}%
6545     {%
6546       Have abbreviations already been defined for this category?
6547       \ifcsstring{@glsabbrv@current@#1}{#2}%
6548       {%
6549         Style already set.
6550         }%
6551         {%
6552           \def\@glxtr@dostylewarn{%
6553             \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6554             {%
6555               \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
6556                 style has been switched \MessageBreak
6557                 for category ‘#1’, \MessageBreak
6558                 but there have already been entries \MessageBreak
6559                 defined for this category. Unwanted \MessageBreak
6560                 side-effects may result}}%
6561               \@endfortrue
6562             }%
6563             \@glxtr@dostylewarn
6564             Set up the style for the given category.
6565             \csdef{@glsabbrv@current@#1}{#2}%
6566             \glxtr@applyabbrvstyle{#2}%

```

```

6562     }%
6563 }%
6564 }

```

`\applyabbrvstyle` Apply the abbreviation style without existence check.

```

6565 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
6566   \csuse{@glsabbrv@dispstyle@setup@#1}%
6567   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6568 }

```

`\r@applyabbrvfmt` Only apply the style formats.

```

6569 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
6570   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6571 }

```

`\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.

```

6572 \newcommand*{\newabbreviationstyle}[3]{%
6573   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6574   {%
6575     \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6576       defined}{}%
6577   }%
6578   {%
6579     \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```

6580     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6581     #2}%
6582     \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

6583     \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6584     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6585     \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6586     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset `\glxtrsubsequentfmt` etc in case a style changes this.

```

6587     \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
6588     \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
6589     \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6590     \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6591     #3}%
6592   }%
6593 }

```

`\abbreviationstyle`

```

6594 \newcommand*{\renewabbreviationstyle}[3]{%
6595   \ifcsundef{@glsabbrv@dispstyle@setup@#1}

```

```

6596 {%
6597   \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6598 }%
6599 {%
6600   \csdef{@glsabbrv@dispstyle@setup@#1}{%
      Initialise hook to do nothing. The style may change this.
6601     \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6602     #2}%
6603   \csdef{@glsabbrv@dispstyle@fmts@#1}{%
      Assume in-line form is the same as first use. The style may change this.
6604     \renewcommand*{\glsxtrinlinefullformat}{\glsxtrfullformat}%
6605     \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6606     \renewcommand*{\glsxtrinlinefullplformat}{\glsxtrfullplformat}%
6607     \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6608     #3}%
6609 }%
6610 }

```

abbreviationstyle Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

6611 \newcommand*{\letabbreviationstyle}[2]{%
6612   \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6613   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6614 }

```

deprecated@abbrstyle `\@glsxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

6615 \newcommand*{\@glsxtr@deprecated@abbrstyle}[2]{%
6616   \csdef{@glsabbrv@dispstyle@setup@#1}{%
6617     \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6618     \csuse{@glsabbrv@dispstyle@setup@#2}%
6619   }%
6620   \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6621 }

```

deprecatedAbbrStyle Generate warning for deprecated style use.

```

6622 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6623   \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6624     use ‘#2’ instead}%
6625 }

```

eAbbrStyleSetup

```

6626 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6627   \ifcsundef{@glsabbrv@dispstyle@setup@#1}%

```

```

6628 {%
6629     \PackageError{glossaries-extra}%
6630     {Unknown abbreviation style definitions ‘#1’}{}%
6631 }%
6632 {%
6633     \csname @glsabbrv@dispstyle@setup@#1\endcsname
6634 }%
6635 }

```

seAbbrStyleFmts

```

6636 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
6637     \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6638     {%
6639         \PackageError{glossaries-extra}%
6640         {Unknown abbreviation style formats ‘#1’}{}%
6641     }%
6642     {%
6643         \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6644     }%
6645 }

```

1.6.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn’t set to “true”. If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

`xtrinsertinside` Switch to determine if the insert text should be inside or outside the font changing command. The default is outside.

```

6646 \newif\ifglsxtrinsertinside
6647 \glsxtrinsertinsidefalse

```

long-short

```

6648 \newabbreviationstyle{long-short}%
6649 {%
6650     \renewcommand*{\CustomAbbreviationFields}{%
6651         name={\protect\glsabbrvfont{\the\glsshorttok}},
6652         sort={\the\glsshorttok},
6653         first={\protect\glsfirstlongfont{\the\glslongtok}}%
6654         \protect\glsxtrfullsep{\the\glslabeltok}%
6655         \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6656         firstplural={\protect\glsfirstlongfont{\the\glslongtok}}%
6657         \protect\glsxtrfullsep{\the\glslabeltok}%
6658         \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

```

```

6659 plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6660 description={\the\glslongtok}}%

```

Unset the regular attribute if it has been set.

```

6661 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6662   \glshasattribute{\the\glslabeltok}{regular}%
6663   {%
6664     \glissetattribute{\the\glslabeltok}{regular}{false}%
6665   }%
6666   {}}%
6667 }%
6668 }%
6669 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6670 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6671 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6672 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6673 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6674 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6675 \renewcommand*{\glsxtrfullformat}[2]{%
6676   \glsfirstlongfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6677   \ifglsxtrininsertinside\else##2\fi
6678   \glsxtrfullsep{##1}%
6679   \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6680 }%
6681 \renewcommand*{\glsxtrfullplformat}[2]{%
6682   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6683   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6684   \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6685 }%
6686 \renewcommand*{\Glsxtrfullformat}[2]{%
6687   \glsfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6688   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6689   \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
6690 }%
6691 \renewcommand*{\Glsxtrfullplformat}[2]{%
6692   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6693   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6694   \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
6695 }%
6696 }

```

Set this as the default style for general abbreviations:

```

6697 \setabbreviationstyle{long-short}

```

ngshortdescsort

```

6698 \newcommand*{\glsxtrlongshortdescsort}{%

```

```

6699 \expandonce\glxtrorglong\space (\expandonce\glxtrorgshort)%
6700 }

```

ngshortdescname

```

6701 \newcommand*{\glxtrlongshortdescname}{%
6702   \protect\glslongfont{\the\glslongtok}
6703   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6704 }

```

long-short-desc User supplies description. The long form is included in the name.

```

6705 \newabbreviationstyle{long-short-desc}%
6706 {%
6707   \renewcommand*{\CustomAbbreviationFields}{%
6708     name={\glxtrlongshortdescname},
6709     sort={\glxtrlongshortdescsort},%
6710     first={\protect\glsfirstlongfont{\the\glslongtok}%
6711       \protect\glxtrfullsep{\the\glslabeltok}%
6712       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6713     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6714       \protect\glxtrfullsep{\the\glslabeltok}%
6715       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

```

The text key should only have the short form.

```

6716   text={\protect\glsabbrvfont{\the\glsshorttok}},%
6717   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6718 }%

```

Unset the regular attribute if it has been set.

```

6719 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6720   \glshasattribute{\the\glslabeltok}{regular}%
6721   {%
6722     \glissetattribute{\the\glslabeltok}{regular}{false}%
6723   }%
6724   {}%
6725 }%
6726 }%
6727 {%
6728   \GlsXtrUseAbbrStyleFmts{long-short}%
6729 }

```

short-long Short form followed by long form in parenthesis on first use.

```

6730 \newabbreviationstyle{short-long}%
6731 {%
6732   \renewcommand*{\CustomAbbreviationFields}{%
6733     name={\protect\glsabbrvfont{\the\glsshorttok}},
6734     sort={\the\glsshorttok},
6735     description={\the\glslongtok},%
6736     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6737       \protect\glxtrfullsep{\the\glslabeltok}%

```

```

6738 \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6739 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}}%
6740 \protect\glxtrfullsep{\the\glslabeltok}%
6741 \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6742 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6743 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6744 \glshasattribute{\the\glslabeltok}{regular}%
6745 {%
6746 \glssetattribute{\the\glslabeltok}{regular}{false}%
6747 }%
6748 {}%
6749 }%
6750 }%
6751 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6752 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
6753 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvdefaultfont{##1}}%
6754 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6755 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6756 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6757 \renewcommand*{\glxtrfullformat}[2]{%
6758 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6759 \ifglxtrinsertinside\else##2\fi
6760 \glxtrfullsep{##1}%
6761 \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6762 }%
6763 \renewcommand*{\glxtrfullplformat}[2]{%
6764 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6765 \ifglxtrinsertinside\else##2\fi
6766 \glxtrfullsep{##1}%
6767 \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6768 }%
6769 \renewcommand*{\Glsxtrfullformat}[2]{%
6770 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
6771 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6772 \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6773 }%
6774 \renewcommand*{\Glsxtrfullplformat}[2]{%
6775 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
6776 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
6777 \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6778 }%
6779 }

```

ortlongdescsort

```
6780 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}
```

ortlongdescname

```
6781 \newcommand*{\glxtrshortlongdescname}{%
6782   \protect\glsabbrvfont{\the\glsshorttok}
6783   \glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
6784 }
```

short-long-desc User supplies description. The long form is included in the name.

```
6785 \newabbreviationstyle{short-long-desc}%
6786 {%
6787   \renewcommand*{\CustomAbbreviationFields}{%
6788     name={\glxtrshortlongdescname},
6789     sort={\glxtrshortlongdescsort},
6790     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6791       \protect\glxtrfullsep{\the\glslabeltok}}%
6792     \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6793     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6794       \protect\glxtrfullsep{\the\glslabeltok}}%
6795     \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6796     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6797     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6798   }%
```

Unset the regular attribute if it has been set.

```
6799   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6800     \glshasattribute{\the\glslabeltok}{regular}%
6801     {%
6802       \glissetattribute{\the\glslabeltok}{regular}{false}%
6803     }%
6804   }%
6805 }%
6806 }%
6807 {%
6808   \GlsXtrUseAbbrStyleFmts{short-long}%
6809 }
```

ongfootnotefont Only used by the “footnote” styles.

```
6810 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6811 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote

```
\glxtrabbrvfootnote{<label>}{<long>}
```


Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
6812 \newcommand*{\glstrabbrvfootnote}[2]{\footnote{#2}}
```

footnote Short form followed by long form in footnote on first use.

```
6813 \newabbreviationstyle{footnote}%
6814 {%
6815   \renewcommand*{\CustomAbbreviationFields}{%
6816     name={\protect\glsabbrvfont{\the\glsshorttok}},
6817     sort={\the\glsshorttok},
6818     description={\the\glslongtok},%

6819     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6820       \protect\glstrabbrvfootnote{\the\glslabeltok}%
6821       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6822     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6823       \protect\glstrabbrvfootnote{\the\glslabeltok}%
6824       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

6825     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.

```
6826 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6827   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6828   \glsattribute{\the\glslabeltok}{regular}%
6829   {%
6830     \glssetattribute{\the\glslabeltok}{regular}{false}%
6831   }%
6832   {}%
6833 }%
6834 }%
6835 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
6836 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6837 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6838 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6839 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6840 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```
6841 \renewcommand*{\glstrfullformat}[2]{%
6842   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglstrinsertinside##2\fi}%
6843   \ifglstrinsertinside\else##2\fi
6844   \protect\glstrabbrvfootnote{##1}%
6845   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6846 }%

```

```

6847 \renewcommand*{\glxtrfullplformat}[2]{%
6848   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6849   \ifglxtrininsertinside\else##2\fi
6850   \protect\glxtrabbrvfootnote{##1}%
6851   {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6852 }%
6853 \renewcommand*{\Glsxtrfullformat}[2]{%
6854   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6855   \ifglxtrininsertinside\else##2\fi
6856   \protect\glxtrabbrvfootnote{##1}%
6857   {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6858 }%
6859 \renewcommand*{\Glsxtrfullplformat}[2]{%
6860   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6861   \ifglxtrininsertinside\else##2\fi
6862   \protect\glxtrabbrvfootnote{##1}%
6863   {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6864 }%

```

The first use full form and the inline full form use the short (long) style.

```

6865 \renewcommand*{\glxtrininlinefullformat}[2]{%
6866   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6867   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6868   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6869 }%
6870 \renewcommand*{\glxtrininlinefullplformat}[2]{%
6871   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6872   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6873   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6874 }%
6875 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
6876   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6877   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6878   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
6879 }%
6880 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
6881   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6882   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6883   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
6884 }%
6885 }

```

short-footnote

```
6886 \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.

```
6887 \newabbreviationstyle{postfootnote}%
```

```

6888 {%
6889   \renewcommand*{\CustomAbbreviationFields}{%
6890     name={\protect\glsabbrvfont{\the\glsshorttok}},
6891     sort={\the\glsshorttok},
6892     description={\the\glslongtok},%
6893     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
6894     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
6895     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.

```

6896   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6897     \csdef{glsxtrpostlink\glscategorylabel}{%
6898       \glsxtrifwasfirstuse
6899       {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

6900         \glsxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
6901         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}}%
6902     }%
6903   }%
6904 }%
6905 \glsattribute{\the\glslabeltok}{regular}%
6906 {%
6907   \glssetattribute{\the\glslabeltok}{regular}{false}%
6908 }%
6909 {}%
6910 }%

```

The footnote needs to be suppressed in the inline form, so `\glsxtrfull` must set the first use switch off.

```

6911   \renewcommand*{\glsxtrsetupfulldefs}{%
6912     \let\glsxtrifwasfirstuse\@secondoftwo
6913   }%
6914 }%
6915 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6916   \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6917   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6918   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6919   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6920   \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

6921   \renewcommand*{\glsxtrfullformat}[2]{%
6922     \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
6923     \ifglsxtrininsertinside\else##2\fi
6924   }%

```

```

6925 \renewcommand*{\glxtrfullplformat}[2]{%
6926   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6927   \ifglxtrininsertinside\else##2\fi
6928 }%
6929 \renewcommand*{\Glsxtrfullformat}[2]{%
6930   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6931   \ifglxtrininsertinside\else##2\fi
6932 }%
6933 \renewcommand*{\Glsxtrfullplformat}[2]{%
6934   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6935   \ifglxtrininsertinside\else##2\fi
6936 }%

```

The first use full form and the inline full form use the short (long) style.

```

6937 \renewcommand*{\glxtrinlinefullformat}[2]{%
6938   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6939   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6940   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6941 }%
6942 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6943   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6944   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6945   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6946 }%
6947 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6948   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6949   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6950   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6951 }%
6952 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6953   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6954   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6955   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6956 }%
6957 }

```

rt-postfootnote

```

6958 \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.

```

6959 \newabbreviationstyle{short}%
6960 {%
6961   \renewcommand*{\CustomAbbreviationFields}{%
6962     name={\protect\glsabbrvfont{\the\glsshorttok}},
6963     sort={\the\glsshorttok},
6964     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},

```

```

6965     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
6966     text={\protect\glsabbrvfont{\the\glsshorttok}},
6967     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
6968     description={\the\glslongtok}}%
6969 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6970   \glssetattribute{\the\glslabeltok}{regular}{true}}%
6971 }%
6972 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6973 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6974 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6975 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6976 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6977 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

6978 \renewcommand*{\glsxtrinlinefullformat}[2]{%
6979   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6980   \ifglsxtrininsertinside##2\fi}%
6981   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6982   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6983 }%
6984 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
6985   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6986   \ifglsxtrininsertinside##2\fi}%
6987   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6988   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6989 }%
6990 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6991   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
6992   \ifglsxtrininsertinside##2\fi}%
6993   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6994   \glsxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
6995 }%
6996 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6997   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
6998   \ifglsxtrininsertinside##2\fi}%
6999   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7000   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7001 }%

```

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.

```

7002 \renewcommand*{\glsxtrfullformat}[2]{%
7003   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7004   \ifglsxtrininsertinside\else##2\fi
7005 }%
7006 \renewcommand*{\glsxtrfullplformat}[2]{%
7007   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7008   \ifglsxtrininsertinside\else##2\fi

```

```

7009 }%
7010 \renewcommand*{\Glsxtrfullformat}[2]{%
7011   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7012   \ifglxtrinsertinside\else##2\fi
7013 }%
7014 \renewcommand*{\Glsxtrfullplformat}[2]{%
7015   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7016   \ifglxtrinsertinside\else##2\fi
7017 }%
7018 }

```

Set this as the default style for acronyms:

```

7019 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7020 \letabbreviationstyle{short-nolong}{short}

```

short-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7021 \newabbreviationstyle{short-nolong-noreg}%
7022 {%
7023   \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7024   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7025     \glshasattribute{\the\glslabeltok}{regular}%
7026     {%
7027       \glsssetattribute{\the\glslabeltok}{regular}{false}%
7028     }%
7029   }%
7030 }%
7031 }%
7032 {%
7033   \GlsXtrUseAbbrStyleFmts{short-nolong}%
7034 }

```

trshortdescname

```

7035 \newcommand*{\glxtrshortdescname}{%
7036   \protect\glsabbrvfont{\the\glssshorttok}%
7037 }

```

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.

```

7038 \newabbreviationstyle{short-desc}%
7039 {%
7040   \renewcommand*{\CustomAbbreviationFields}{%
7041     name={\glxtrshortdescname},
7042     sort={\the\glssshorttok},
7043     first={\protect\glsfirstabbrvfont{\the\glssshorttok}},
7044     firstplural={\protect\glsfirstabbrvfont{\the\glssshortpltok}},

```

```

7045     text={\protect\glsabbrvfont{\the\glsshorttok}},
7046     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7047     description={\the\glslongtok}}%
7048 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7049   \glssetattribute{\the\glslabeltok}{regular}{true}}%
7050 }%
7051 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7052 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
7053 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7054 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7055 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7056 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7057 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7058   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7059   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7060   \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7061 }%
7062 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7063   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7064   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7065   \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7066 }%
7067 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7068   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7069   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7070   \glsxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7071 }%
7072 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7073   \glsfirstabbrvfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7074   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7075   \glsxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7076 }%

```

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.

```

7077 \renewcommand*{\glsxtrfullformat}[2]{%
7078   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7079   \ifglsxtrininsertinside\else##2\fi
7080 }%
7081 \renewcommand*{\glsxtrfullplformat}[2]{%
7082   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
7083   \ifglsxtrininsertinside\else##2\fi
7084 }%
7085 \renewcommand*{\Glsxtrfullformat}[2]{%
7086   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
7087   \ifglsxtrininsertinside\else##2\fi
7088 }%

```

```

7089 \renewcommand*{\Glsxtrfullplformat}[2]{%
7090   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
7091   \ifglsxtrinsertinside\else##2\fi
7092 }%
7093 }

```

ort-nolong-desc

```
7094 \letabbreviationstyle{short-nolong-desc}{short-desc}
```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7095 \newabbreviationstyle{short-nolong-desc-noreg}%
7096 {%
7097   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
       Unset the regular attribute if it has been set.
7098   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7099     \glshasattribute{\the\glslabeltok}{regular}%
7100     {%
7101       \glsssetattribute{\the\glslabeltok}{regular}{false}%
7102     }%
7103     {}%
7104   }%
7105 }%
7106 {%
7107   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7108 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7109 \newabbreviationstyle{nolong-short}%
7110 {%
7111   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7112 }%
7113 {%
7114   \GlsXtrUseAbbrStyleFmts{short-nolong}%
       The inline full form displays the long form followed by the short form in parentheses.
7115   \renewcommand*{\glsxtrinlinefullformat}[2]{%
7116     \protect\glsfirstlongfont{\glsaccesslong{##1}%
7117       \ifglsxtrinsertinside##2\fi}%
7118     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7119     \glsxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7120   }%
7121   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7122     \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
7123       \ifglsxtrinsertinside##2\fi}%
7124     \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7125     \glsxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7126   }%
7127   \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```



```

7128 \protect\glsfirstlongfont{\glsaccesslong{##1}%
7129 \ifglxtrinsertinside##2\fi}%
7130 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7131 \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7132 }%
7133 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7134 \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
7135 \ifglxtrinsertinside##2\fi}%
7136 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7137 \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7138 }%
7139 }

```

ong-short-noreg Like `nolong-short` but doesn't set the regular attribute.

```

7140 \newabbreviationstyle{nolong-short-noreg}%
7141 {%
7142 \GlsXtrUseAbbrStyleSetup{nolong-short}%

```

Unset the regular attribute if it has been set.

```

7143 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7144 \glshasattribute{\the\glslabeltok}{regular}%
7145 {%
7146 \glissetattribute{\the\glslabeltok}{regular}{false}%
7147 }%
7148 {}%
7149 }%
7150 }%
7151 {%
7152 \GlsXtrUseAbbrStyleFmts{nolong-short}%
7153 }

```

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.

```

7154 \newabbreviationstyle{long-desc}%
7155 {%
7156 \renewcommand*{\CustomAbbreviationFields}{%
7157 name={\protect\protect\glslongfont{\the\glslongtok}},
7158 sort={\the\glslongtok},
7159 first={\protect\glsfirstlongfont{\the\glslongtok}},
7160 firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7161 text={\glslongfont{\the\glslongtok}},
7162 plural={\glslongfont{\the\glslongpltok}}}%
7163 }%
7164 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7165 \glissetattribute{\the\glslabeltok}{regular}{true}}%
7166 }%
7167 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7168 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
7169 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
7170 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
7171 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
7172 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7173 \renewcommand*\glxtrsubsequentfmt[2]{%
7174   \glslongfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7175   \ifglxtrininsertinside \else##2\fi
7176 }%
7177 \renewcommand*\glxtrsubsequentplfmt[2]{%
7178   \glslongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7179   \ifglxtrininsertinside \else##2\fi
7180 }%
7181 \renewcommand*\Glsxtrsubsequentfmt[2]{%
7182   \glslongfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7183   \ifglxtrininsertinside \else##2\fi
7184 }%
7185 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
7186   \glslongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7187   \ifglxtrininsertinside \else##2\fi
7188 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7189 \renewcommand*\glxtrininlinefullformat[2]{%
7190   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7191   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7192   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7193 }%
7194 \renewcommand*\glxtrininlinefullplformat[2]{%
7195   \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7196   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7197   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7198 }%
7199 \renewcommand*\Glsxtrininlinefullformat[2]{%
7200   \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7201   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7202   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7203 }%
7204 \renewcommand*\Glsxtrininlinefullplformat[2]{%
7205   \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
7206   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7207   \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7208 }%

```

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.

```

7209 \renewcommand*\glxtrfullformat[2]{%
7210   \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7211   \ifglxtrininsertinside\else##2\fi

```

```

7212 }%
7213 \renewcommand*{\glstrfullplformat}[2]{%
7214   \glstrfirstlongfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
7215   \ifglstrinsertinside\else##2\fi
7216 }%
7217 \renewcommand*{\GlsXtrfullformat}[2]{%
7218   \glstrfirstlongfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
7219   \ifglstrinsertinside\else##2\fi
7220 }%
7221 \renewcommand*{\GlsXtrfullplformat}[2]{%
7222   \glstrfirstlongfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
7223   \ifglstrinsertinside\else##2\fi
7224 }%
7225 }

```

ng-noshort-desc Provide a synonym that matches similar styles.

```
7226 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

short-desc-noreg Like long-noshort-desc but doesn't set the regular attribute.

```

7227 \newabbreviationstyle{long-noshort-desc-noreg}%
7228 {%
7229   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7230   \UnsettheRegularAttributeIfItHasBeenSet
7231   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7232     \glshasattribute{\the\glslabeltok}{regular}%
7233     {%
7234       \glsssetattribute{\the\glslabeltok}{regular}{false}%
7235     }%
7236   }%
7237 }%
7238 {%
7239   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7240 }

```

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.

```

7241 \newabbreviationstyle{long}%
7242 {%
7243   \renewcommand*{\CustomAbbreviationFields}{%
7244     name={\protect\glssabbrvfont{\the\glssshorttok}},
7245     sort={\the\glssshorttok},
7246     first={\protect\glstrfirstlongfont{\the\glslongtok}},
7247     firstplural={\protect\glstrfirstlongfont{\the\glslongpltok}},
7248     text={\glslongfont{\the\glslongtok}},
7249     plural={\glslongfont{\the\glslongpltok}},%
7250     description={\the\glslongtok}%

```

```

7251 }%
7252 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7253   \glsssetattribute{\the\glslabeltok}{regular}{true}}%
7254 }%
7255 {%
7256   \GlsXtrUseAbbrStyleFmts{long-desc}%
7257 }

```

long-noshort Provide a synonym that matches similar styles.

```
7258 \letabbreviationstyle{long-noshort}{long}
```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```

7259 \newabbreviationstyle{long-noshort-noreg}%
7260 {%
7261   \GlsXtrUseAbbrStyleSetup{long-noshort}%
7262   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7263     \glshasattribute{\the\glslabeltok}{regular}%
7264     {%
7265       \glsssetattribute{\the\glslabeltok}{regular}{false}%
7266     }%
7267     {}%
7268   }%
7269 }%
7270 {%
7271   \GlsXtrUseAbbrStyleFmts{long-noshort}%
7272 }

```

1.6.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

\glsxtrscfont Maintained for backward-compatibility.

```
7273 \newcommand*{\glsxtrscfont}[1]{\textsc{#1}}
```

\glsabbrvscfont Added for consistent naming.

```
7274 \newcommand*{\glsabbrvscfont}{\glsxtrscfont}
```

sxtrfirstscfont Maintained for backward-compatibility.

```
7275 \newcommand*{\glxsxtrfirstscfont}[1]{\glsabbrvscfont{#1}}
```

irstabbrvscfont Added for consistent naming.

```
7276 \newcommand*{\glsfirstabbrvscfont}{\glxsxtrfirstscfont}
```

and for the default short form suffix:

\glsxtrscsuffix

```
7277 \newcommand*{\glsxtrscsuffix}{\glstextup{\glsxtrabbrvpluralsuffix}}
```

long-short-sc

```
7278 \newabbreviationstyle{long-short-sc}%
7279 {%
7280   \renewcommand*{\CustomAbbreviationFields}{%
7281     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7282     sort={\the\glsshorttok},
7283     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7284       \protect\glsxtrfullsep{\the\glslabeltok}%
7285       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7286     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7287       \protect\glsxtrfullsep{\the\glslabeltok}%
7288       \glsxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7289     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},%
7290     description={\the\glslongtok}}%
7291   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7292     \glshasattribute{\the\glslabeltok}{regular}%
7293     {%
7294       \glissetattribute{\the\glslabeltok}{regular}{false}%
7295     }%
7296   }%
7297 }%
7298 }%
7299 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7300 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrscsuffix}%
7301 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7302 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
```

Use the default long fonts.

```
7303 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7304 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
7305 \renewcommand*{\glsxtrfullformat}[2]{%
7306   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7307   \ifglsxtrininsertinside\else##2\fi
7308   \glsxtrfullsep{##1}%
7309   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7310 }%
7311 \renewcommand*{\glsxtrfullplformat}[2]{%
7312   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
7313   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7314   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7315 }%
7316 \renewcommand*{\Glsxtrfullformat}[2]{%
7317   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7318   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7319   \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7320 }%
```

```

7321 \renewcommand*{\Glsxtrfullplformat}[2]{%
7322   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7323   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7324   \glxtrparen{\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7325 }%
7326 }

```

g-short-sc-desc

```

7327 \newabbreviationstyle{long-short-sc-desc}%
7328 {%
7329   \renewcommand*{\CustomAbbreviationFields}{%
7330     name={\glxtrlongshortdescname},
7331     sort={\glxtrlongshortdescsort},%
7332     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7333       \protect\glxtrfullsep{\the\glslabeltok}%
7334       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7335     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7336       \protect\glxtrfullsep{\the\glslabeltok}%
7337       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7338     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7339     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7340   }%

```

Unset the regular attribute if it has been set.

```

7341 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7342   \glshasattribute{\the\glslabeltok}{regular}%
7343   {%
7344     \glissetattribute{\the\glslabeltok}{regular}{false}%
7345   }%
7346   {}%
7347 }%
7348 }%
7349 {%

```

As long-short-sc style:

```

7350 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7351 }

```

Now the short (long) version

```

7352 \newabbreviationstyle{short-sc-long}%
7353 {%
7354   \renewcommand*{\CustomAbbreviationFields}{%
7355     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7356     sort={\the\glsshorttok},
7357     description={\the\glslongtok},%
7358     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7359       \protect\glxtrfullsep{\the\glslabeltok}%
7360       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7361     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7362       \protect\glxtrfullsep{\the\glslabeltok}%

```

```

7363     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7364     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

7365 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7366     \glshasattribute{\the\glslabeltok}{regular}%
7367     {%
7368         \glssetattribute{\the\glslabeltok}{regular}{false}%
7369     }%
7370     }%
7371 }%
7372 }%
7373 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7374 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7375 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7376 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7377 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7378 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7379 \renewcommand*{\glxtrfullformat}[2]{%
7380     \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7381     \ifglxtrinsertinside\else##2\fi
7382     \glxtrfullsep{##1}%
7383     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7384 }%
7385 \renewcommand*{\glxtrfullplformat}[2]{%
7386     \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7387     \ifglxtrinsertinside\else##2\fi
7388     \glxtrfullsep{##1}%
7389     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7390 }%
7391 \renewcommand*{\Glsxtrfullformat}[2]{%
7392     \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7393     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7394     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7395 }%
7396 \renewcommand*{\Glsxtrfullplformat}[2]{%
7397     \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7398     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7399     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7400 }%
7401 }

```

As before but user provides description

```

7402 \newabbreviationstyle{short-sc-long-desc}%
7403 {%
7404     \renewcommand*{\CustomAbbreviationFields}{%

```

```

7405   name={\glxtrshortlongdescname},
7406   sort={\glxtrshortlongdescsort},
7407   first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7408     \protect\glxtrfullsep{\the\glslabeltok}%
7409     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7410   firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7411     \protect\glxtrfullsep{\the\glslabeltok}%
7412     \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7413   text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7414   plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7415 }%

```

Unset the regular attribute if it has been set.

```

7416 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7417   \glshasattribute{\the\glslabeltok}{regular}%
7418   {%
7419     \glissetattribute{\the\glslabeltok}{regular}{false}%
7420   }%
7421   {}%
7422 }%
7423 }%
7424 {%

```

As short-sc-long style:

```

7425   \GlsXtrUseAbbrStyleFmts{short-sc-long}%
7426 }

```

short-sc

```

7427 \newabbreviationstyle{short-sc}%
7428 {%
7429   \renewcommand*{\CustomAbbreviationFields}{%
7430     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7431     sort={\the\glsshorttok},
7432     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7433     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7434     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7435     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7436     description={\the\glslongtok}}%
7437   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7438     \glissetattribute{\the\glslabeltok}{regular}{true}}%
7439 }%
7440 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7441 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7442 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvscfont{##1}}%
7443 \renewcommand*{\glsfirstabbrvfont[1]}{\glsfirstabbrvscfont{##1}}%
7444 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7445 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.


```

7446 \renewcommand*{\glxtrinlinefullformat}[2]{%
7447   \protect\glsfirstabbrvscfont{\glsaccesssshort{##1}%
7448     \ifglxtrininsertinside##2\fi}%
7449   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7450   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7451 }%
7452 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7453   \protect\glsfirstabbrvscfont{\glsaccesssshortpl{##1}%
7454     \ifglxtrininsertinside##2\fi}%
7455   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7456   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7457 }%
7458 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7459   \protect\glsfirstabbrvscfont{\Glsaccesssshort{##1}%
7460     \ifglxtrininsertinside##2\fi}%
7461   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7462   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7463 }%
7464 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7465   \protect\glsfirstabbrvscfont{\Glsaccesssshortpl{##1}%
7466     \ifglxtrininsertinside##2\fi}%
7467   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7468   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7469 }%

```

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.

```

7470 \renewcommand*{\glxtrfullformat}[2]{%
7471   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglxtrininsertinside##2\fi}%
7472   \ifglxtrininsertinside\else##2\fi
7473 }%
7474 \renewcommand*{\glxtrfullplformat}[2]{%
7475   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglxtrininsertinside##2\fi}%
7476   \ifglxtrininsertinside\else##2\fi
7477 }%
7478 \renewcommand*{\Glsxtrfullformat}[2]{%
7479   \glsfirstabbrvscfont{\glsaccesssshort{##1}\ifglxtrininsertinside##2\fi}%
7480   \ifglxtrininsertinside\else##2\fi
7481 }%
7482 \renewcommand*{\Glsxtrfullplformat}[2]{%
7483   \glsfirstabbrvscfont{\glsaccesssshortpl{##1}\ifglxtrininsertinside##2\fi}%
7484   \ifglxtrininsertinside\else##2\fi
7485 }%
7486 }

```

short-sc-nolong

```
7487 \letabbreviationstyle{short-sc-nolong}{short-sc}
```

short-sc-desc

```

7488 \newabbreviationstyle{short-sc-desc}%
7489 {%
7490   \renewcommand*{\CustomAbbreviationFields}{%
7491     name={\glxtrshortdescname},
7492     sort={\the\glsshorttok},
7493     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7494     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7495     text={\protect\glssabbrvscfont{\the\glsshorttok}},
7496     plural={\protect\glssabbrvscfont{\the\glsshortpltok}},
7497     description={\the\glslongtok}}%
7498   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7499     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7500 }%
7501 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7502   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7503   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
7504   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7505   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7506   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7507   \renewcommand*{\glxtrininlinefullformat}[2]{%
7508     \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7509     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7510     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7511   }%
7512   \renewcommand*{\glxtrininlinefullplformat}[2]{%
7513     \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7514     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7515     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7516   }%
7517   \renewcommand*{\Glsxtrininlinefullformat}[2]{%
7518     \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7519     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7520     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7521   }%
7522   \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
7523     \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7524     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7525     \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7526   }%

```

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.

```

7527   \renewcommand*{\glxtrfullformat}[2]{%
7528     \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7529     \ifglxtrininsertinside\else##2\fi
7530   }%
7531   \renewcommand*{\glxtrfullplformat}[2]{%

```

```

7532 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7533 \ifglxtrinsertinside\else##2\fi
7534 }%
7535 \renewcommand*{\Glsxtrfullformat}[2]{%
7536 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7537 \ifglxtrinsertinside\else##2\fi
7538 }%
7539 \renewcommand*{\Glsxtrfullplformat}[2]{%
7540 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7541 \ifglxtrinsertinside\else##2\fi
7542 }%
7543 }

```

-sc-nolong-desc

```

7544 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}

```

nolong-short-sc

```

7545 \newabbreviationstyle{nolong-short-sc}%
7546 {%
7547 \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
7548 }%
7549 {%
7550 \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7551 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7552 \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
7553 \ifglxtrinsertinside##2\fi}%
7554 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7555 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7556 }%
7557 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7558 \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
7559 \ifglxtrinsertinside##2\fi}%
7560 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7561 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7562 }%
7563 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7564 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
7565 \ifglxtrinsertinside##2\fi}%
7566 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7567 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7568 }%
7569 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7570 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
7571 \ifglxtrinsertinside##2\fi}%
7572 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7573 \glsxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7574 }%
7575 }

```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7576 \newabbreviationstyle{long-noshort-sc}%
7577 {%
7578   \renewcommand*{\CustomAbbreviationFields}{%
7579     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7580     sort={\the\glsshorttok},
7581     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7582     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7583     text={\protect\glslongdefaultfont{\the\glslongtok}},
7584     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7585     description={\the\glslongtok}%
7586   }%
7587   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7588     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7589 }%
7590 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7591 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7592 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7593 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7594 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7595 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7596 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7597   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7598   \ifglxtrininsertinside \else##2\fi
7599 }%
7600 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7601   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7602   \ifglxtrininsertinside \else##2\fi
7603 }%
7604 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7605   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
7606   \ifglxtrininsertinside \else##2\fi
7607 }%
7608 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7609   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
7610   \ifglxtrininsertinside \else##2\fi
7611 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7612 \renewcommand*{\glxtrininlinefullformat}[2]{%
7613   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
7614   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7615   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7616 }%
7617 \renewcommand*{\glxtrininlinefullplformat}[2]{%

```

```

7618 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7619 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7620 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7621 }%
7622 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7623 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7624 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7625 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7626 }%
7627 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7628 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7629 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7630 \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7631 }%

```

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.

```

7632 \renewcommand*{\glxtrfullformat}[2]{%
7633 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7634 \ifglxtrinsertinside\else##2\fi
7635 }%
7636 \renewcommand*{\glxtrfullplformat}[2]{%
7637 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7638 \ifglxtrinsertinside\else##2\fi
7639 }%
7640 \renewcommand*{\Glsxtrfullformat}[2]{%
7641 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7642 \ifglxtrinsertinside\else##2\fi
7643 }%
7644 \renewcommand*{\Glsxtrfullplformat}[2]{%
7645 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7646 \ifglxtrinsertinside\else##2\fi
7647 }%
7648 }

```

long-sc Backward compatibility:

```

7649 \@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`.

```

7650 \newabbreviationstyle{long-noshort-sc-desc}%
7651 {%
7652 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7653 }%
7654 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7655 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7656 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7657 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%

```

```

7658 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7659 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7660 \renewcommand*\glsxtrsubsequentfmt}[2]{%
7661   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside ##2\fi}%
7662   \ifglsxtrinsertinside \else##2\fi
7663 }%
7664 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
7665   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside ##2\fi}%
7666   \ifglsxtrinsertinside \else##2\fi
7667 }%
7668 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
7669   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside ##2\fi}%
7670   \ifglsxtrinsertinside \else##2\fi
7671 }%
7672 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
7673   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside ##2\fi}%
7674   \ifglsxtrinsertinside \else##2\fi
7675 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7676 \renewcommand*\glsxtrinlinefullformat}[2]{%
7677   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7678   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7679   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7680 }%
7681 \renewcommand*\glsxtrinlinefullplformat}[2]{%
7682   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7683   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7684   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7685 }%
7686 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7687   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7688   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7689   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7690 }%
7691 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7692   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7693   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7694   \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7695 }%

```

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.

```

7696 \renewcommand*\glsxtrfullformat}[2]{%
7697   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7698   \ifglsxtrinsertinside\else##2\fi
7699 }%
7700 \renewcommand*\glsxtrfullplformat}[2]{%
7701   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%

```

```

7702 \ifglxtrinsertinside\else##2\fi
7703 }%
7704 \renewcommand*{\Glsxtrfullformat}[2]{%
7705 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7706 \ifglxtrinsertinside\else##2\fi
7707 }%
7708 \renewcommand*{\Glsxtrfullplformat}[2]{%
7709 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7710 \ifglxtrinsertinside\else##2\fi
7711 }%
7712 }

```

long-desc-sc Backward compatibility:

```

7713 \@glxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7714 \newabbreviationstyle{short-sc-footnote}%
7715 {%
7716 \renewcommand*{\CustomAbbreviationFields}{%
7717 name={\protect\glsabbrvscfont{\the\glsshorttok}},
7718 sort={\the\glsshorttok},
7719 description={\the\glslongtok},%
7720 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%
7721 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7722 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7723 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}}%
7724 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7725 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7726 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.

```

7727 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7728 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7729 \glsattribute{\the\glslabeltok}{regular}%
7730 {%
7731 \glssetattribute{\the\glslabeltok}{regular}{false}%
7732 }%
7733 {}%
7734 }%
7735 }%
7736 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7737 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7738 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7739 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7740 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7741 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7742 \renewcommand*{\glxtrfullformat}[2]{%
7743   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7744   \ifglxtrininsertinside\else##2\fi
7745   \protect\glxtrabbrvfootnote{##1}%
7746   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7747 }%
7748 \renewcommand*{\glxtrfullplformat}[2]{%
7749   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7750   \ifglxtrininsertinside\else##2\fi
7751   \protect\glxtrabbrvfootnote{##1}%
7752   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7753 }%
7754 \renewcommand*{\Glsxtrfullformat}[2]{%
7755   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7756   \ifglxtrininsertinside\else##2\fi
7757   \protect\glxtrabbrvfootnote{##1}%
7758   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7759 }%
7760 \renewcommand*{\Glsxtrfullplformat}[2]{%
7761   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7762   \ifglxtrininsertinside\else##2\fi
7763   \protect\glxtrabbrvfootnote{##1}%
7764   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7765 }%

```

The first use full form and the inline full form use the short (long) style.

```

7766 \renewcommand*{\glxtrinlinefullformat}[2]{%
7767   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7768   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7769   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7770 }%
7771 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7772   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7773   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7774   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7775 }%
7776 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7777   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7778   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7779   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7780 }%
7781 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7782   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7783   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7784   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7785 }%
7786 }

```


footnote-sc Backward compatibility:

```
7787 \@glxtr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
7788 \newabbreviationstyle{short-sc-postfootnote}%
7789 {%
7790   \renewcommand*{\CustomAbbreviationFields}{%
7791     name={\protect\glsabbrvscfont{\the\glsshorttok}},
7792     sort={\the\glsshorttok},
7793     description={\the\glslongtok},%
7794     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7795     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7796     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7797 }
```

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.

```
7797 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7798   \csdef{glxtrpostlink\glscategorylabel}{%
7799     \glxtrifwasfirstuse
7800     {%
7801       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7802         {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
7803       }%
7804     }%
7805   }%
7806   \glshasattribute{\the\glslabeltok}{regular}%
7807   {%
7808     \glissetattribute{\the\glslabeltok}{regular}{false}%
7809   }%
7810 }
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
7801   \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
7802     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
7803   }%
7804   }%
7805   }%
7806   \glshasattribute{\the\glslabeltok}{regular}%
7807   {%
7808     \glissetattribute{\the\glslabeltok}{regular}{false}%
7809   }%
7810 }
```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```
7812 \renewcommand*{\glxtrsetupfulldefs}{%
7813   \let\glxtrifwasfirstuse\@secondoftwo
7814 }%
7815 }%
7816 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7817 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7818 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7819 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7820 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7821 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```

7822 \renewcommand*{\glxtrfullformat}[2]{%
7823   \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7824   \ifglxtrinsertinside\else##2\fi
7825 }%
7826 \renewcommand*{\glxtrfullplformat}[2]{%
7827   \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7828   \ifglxtrinsertinside\else##2\fi
7829 }%
7830 \renewcommand*{\Glsxtrfullformat}[2]{%
7831   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7832   \ifglxtrinsertinside\else##2\fi
7833 }%
7834 \renewcommand*{\Glsxtrfullplformat}[2]{%
7835   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7836   \ifglxtrinsertinside\else##2\fi
7837 }%

```

The first use full form and the inline full form use the short (long) style.

```

7838 \renewcommand*{\glxtrininlinefullformat}[2]{%
7839   \glsfirstabbrvscfont{\glaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7840   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7841   \glxtrparen{\glsfirstlongfootnotefont{\glaccesslong{##1}}}%
7842 }%
7843 \renewcommand*{\glxtrininlinefullplformat}[2]{%
7844   \glsfirstabbrvscfont{\glaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7845   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7846   \glxtrparen{\glsfirstlongfootnotefont{\glaccesslongpl{##1}}}%
7847 }%
7848 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
7849   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7850   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7851   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7852 }%
7853 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
7854   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7855   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7856   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7857 }%
7858 }

```

postfootnote-sc Backward compatibility:

```

7859 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

1.6.4 Predefined Styles (Fake Small Capitals)

These styles require the `relsize` package, which must be loaded by the user. These styles all use:

`\glxtrsmfont` Maintained for backward compatibility.

```
7860 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}
```

`\glsabbrvsmfont` Added for consistent naming.

```
7861 \newcommand*{\glsabbrvsmfont}{\glxtrsmfont}
```

`\glxtrfirstsmfont` Maintained for backward compatibility.

```
7862 \newcommand*{\glxtrfirstsmfont}[1]{\glsabbrvsmfont{#1}}
```

`\glxtrfirstabbrvsmfont` Added for consistent naming.

```
7863 \newcommand*{\glxtrfirstabbrvsmfont}{\glxtrfirstsmfont}
```

and for the default short form suffix:

`\glxtrsmssuffix`

```
7864 \newcommand*{\glxtrsmssuffix}{\glxtrabbrvpluralsuffix}
```

long-short-sm

```
7865 \newabbreviationstyle{long-short-sm}%
7866 {%
7867   \renewcommand*{\CustomAbbreviationFields}{%
7868     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
7869     sort={\the\glsshorttok},
7870     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7871       \protect\glxtrfullsep{\the\glslabeltok}%
7872       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7873     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7874       \protect\glxtrfullsep{\the\glslabeltok}%
7875       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7876     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},%
7877     description={\the\glslongtok}}%
7878   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7879     \glshasattribute{\the\glslabeltok}{regular}%
7880     {%
7881       \glissetattribute{\the\glslabeltok}{regular}{false}%
7882     }%
7883   }%
7884 }%
7885 }%
7886 {%
7887   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
7888   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
7889   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmssuffix}%
7890   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7891   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

Use the default long fonts.

The first use full form and the inline form are the same for this style.

```

7892 \renewcommand*{\glxtrfullformat}[2]{%
7893   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7894   \ifglxtrinsertinside\else##2\fi
7895   \glxtrfullsep{##1}%
7896   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7897 }%
7898 \renewcommand*{\glxtrfullplformat}[2]{%
7899   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7900   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7901   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7902 }%
7903 \renewcommand*{\Glsxtrfullformat}[2]{%
7904   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7905   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7906   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
7907 }%
7908 \renewcommand*{\Glsxtrfullplformat}[2]{%
7909   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7910   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7911   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
7912 }%
7913 }

```

g-short-sm-desc

```

7914 \newabbreviationstyle{long-short-sm-desc}%
7915 {%
7916   \renewcommand*{\CustomAbbreviationFields}{%
7917     name={\glxtrlongshortdescname},
7918     sort={\glxtrlongshortdescsort},%
7919     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7920       \protect\glxtrfullsep{\the\glslabeltok}%
7921       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
7922     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7923       \protect\glxtrfullsep{\the\glslabeltok}%
7924       \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
7925     text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
7926     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
7927   }%

```

Unset the regular attribute if it has been set.

```

7928 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7929   \glshasattribute{\the\glslabeltok}{regular}%
7930   {%
7931     \glssetattribute{\the\glslabeltok}{regular}{false}%
7932   }%
7933   {}%
7934 }%
7935 }%
7936 {%

```

As long-short-sm style:

```
7937 \GlsXtrUseAbbrStyleFmts{long-short-sm}%  
7938 }
```

short-sm-long Now the short (long) version

```
7939 \newabbreviationstyle{short-sm-long}%  
7940 {%  
7941   \renewcommand*{\CustomAbbreviationFields}{%  
7942     name={\protect\glsabbrvsmfont{\the\glsshorttok}},  
7943     sort={\the\glsshorttok},  
7944     description={\the\glslongtok},%  
7945     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}%  
7946     \protect\glsxtrfullsep{\the\glslabeltok}%  
7947     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%  
7948     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}%  
7949     \protect\glsxtrfullsep{\the\glslabeltok}%  
7950     \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%  
7951     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%  
7952 }
```

Unset the regular attribute if it has been set.

```
7952 \renewcommand*{\GlsXtrPostNewAbbreviation}{%  
7953   \glsattribute{\the\glslabeltok}{regular}%  
7954   {%  
7955     \glssetattribute{\the\glslabeltok}{regular}{false}%  
7956   }%  
7957   {}%  
7958 }%  
7959 }%  
7960 {%  
7961   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%  
7962   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%  
7963   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%  
7964   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%  
7965   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%  
7966 }
```

The first use full form and the inline full form are the same for this style.

```
7966 \renewcommand*\glsxtrfullformat[2]{%  
7967   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%  
7968   \ifglsxtrininsertinside\else##2\fi  
7969   \glsxtrfullsep{##1}%  
7970   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%  
7971 }%  
7972 \renewcommand*\glsxtrfullplformat[2]{%  
7973   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%  
7974   \ifglsxtrininsertinside\else##2\fi  
7975   \glsxtrfullsep{##1}%  
7976   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%  
7977 }%  
7978 \renewcommand*\Glsxtrfullformat[2]{%  
7979   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%  
7980 }
```

```

7980 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7981 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7982 }%
7983 \renewcommand*{\Glsxtrfullplformat}[2]{%
7984 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7985 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7986 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7987 }%
7988 }

```

rt-sm-long-desc As before but user provides description

```

7989 \newabbreviationstyle{short-sm-long-desc}%
7990 {%
7991 \renewcommand*{\CustomAbbreviationFields}{%
7992 name={\glxtrshortlongdescname},
7993 sort={\glxtrshortlongdescsort},
7994 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
7995 \protect\glxtrfullsep{\the\glslabeltok}%
7996 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7997 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
7998 \protect\glxtrfullsep{\the\glslabeltok}%
7999 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8000 text={\protect\glssabbrvsmfont{\the\glsshorttok}},%
8001 plural={\protect\glssabbrvsmfont{\the\glsshortpltok}}}%
8002 }%

```

Unset the regular attribute if it has been set.

```

8003 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8004 \glshasattribute{\the\glslabeltok}{regular}%
8005 {%
8006 \glissetattribute{\the\glslabeltok}{regular}{false}%
8007 }%
8008 {}%
8009 }%
8010 }%
8011 {%

```

As short-sm-long style:

```

8012 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8013 }

```

short-sm

```

8014 \newabbreviationstyle{short-sm}%
8015 {%
8016 \renewcommand*{\CustomAbbreviationFields}{%
8017 name={\protect\glssabbrvsmfont{\the\glsshorttok}},
8018 sort={\the\glsshorttok},
8019 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8020 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8021 text={\protect\glssabbrvsmfont{\the\glsshorttok}},

```

```

8022 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8023 description={\the\glslongtok}}%
8024 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8025   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8026 }%
8027 {%
8028   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8029   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8030   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsmsuffix}%
8031   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8032   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8033 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8034   \protect\glsfirstabbrvsmfont{\glsaccesssshort{##1}}%
8035   \ifglsxtrininsertinside##2\fi}%
8036   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8037   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8038 }%
8039 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8040   \protect\glsfirstabbrvsmfont{\glsaccesssshortpl{##1}}%
8041   \ifglsxtrininsertinside##2\fi}%
8042   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8043   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8044 }%

8045 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8046   \protect\glsfirstabbrvsmfont{\Glsaccesssshort{##1}}%
8047   \ifglsxtrininsertinside##2\fi}%
8048   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8049   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8050 }%
8051 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8052   \protect\glsfirstabbrvsmfont{\Glsaccesssshortpl{##1}}%
8053   \ifglsxtrininsertinside##2\fi}%
8054   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8055   \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8056 }%

```

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.

```

8057 \renewcommand*{\glsxtrfullformat}[2]{%
8058   \glsfirstabbrvsmfont{\glsaccesssshort{##1}}\ifglsxtrininsertinside##2\fi}%
8059   \ifglsxtrininsertinside\else##2\fi
8060 }%
8061 \renewcommand*{\glsxtrfullplformat}[2]{%
8062   \glsfirstabbrvsmfont{\glsaccesssshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8063   \ifglsxtrininsertinside\else##2\fi
8064 }%
8065 \renewcommand*{\Glsxtrfullformat}[2]{%

```

```

8066 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8067 \ifglxtrinsertinside\else##2\fi
8068 }%
8069 \renewcommand*{\Glsxtrfullplformat}[2]{%
8070 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8071 \ifglxtrinsertinside\else##2\fi
8072 }%
8073 }

```

short-sm-nolong

```

8074 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

8075 \newabbreviationstyle{short-sm-desc}%
8076 {%
8077 \renewcommand*{\CustomAbbreviationFields}{%
8078 name={\glsxtrshortdescname},
8079 sort={\the\glsshorttok},
8080 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8081 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8082 text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8083 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8084 description={\the\glslongtok}}%
8085 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8086 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8087 }%
8088 {%
8089 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8090 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8091 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8092 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8093 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8094 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8095 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8096 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8097 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8098 }%
8099 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8100 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8101 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8102 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8103 }%
8104 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8105 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8106 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8107 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8108 }%
8109 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

```



```

8110 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8111 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8112 \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8113 }%

```

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.

```

8114 \renewcommand*{\glxtrfullformat}[2]{%
8115 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8116 \ifglxtrinsertinside\else##2\fi
8117 }%
8118 \renewcommand*{\glxtrfullplformat}[2]{%
8119 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8120 \ifglxtrinsertinside\else##2\fi
8121 }%
8122 \renewcommand*{\Glsxtrfullformat}[2]{%
8123 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8124 \ifglxtrinsertinside\else##2\fi
8125 }%
8126 \renewcommand*{\Glsxtrfullplformat}[2]{%
8127 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8128 \ifglxtrinsertinside\else##2\fi
8129 }%
8130 }

```

-sm-nolong-desc

```

8131 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}

```

nolong-short-sm

```

8132 \newabbreviationstyle{nolong-short-sm}%
8133 {%
8134 \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8135 }%
8136 {%
8137 \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8138 \renewcommand*{\glxtrinlinefullformat}[2]{%
8139 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8140 \ifglxtrinsertinside##2\fi}%
8141 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8142 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8143 }%
8144 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8145 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8146 \ifglxtrinsertinside##2\fi}%
8147 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8148 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8149 }%
8150 \renewcommand*{\Glsxtrinlinefullformat}[2]{%

```

```

8151 \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8152 \ifglxtrinsertinside##2\fi}%
8153 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8154 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8155 }%
8156 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8157 \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8158 \ifglxtrinsertinside##2\fi}%
8159 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8160 \glxtrparen{\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8161 }%
8162 }

```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

8163 \newabbreviationstyle{long-noshort-sm}%
8164 {%
8165 \renewcommand*{\CustomAbbreviationFields}{%
8166 name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8167 sort={\the\glsshorttok},
8168 first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8169 firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8170 text={\protect\glslongdefaultfont{\the\glslongtok}},
8171 plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8172 description={\the\glslongtok}%
8173 }%
8174 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8175 \glsssetattribute{\the\glslabeltok}{regular}{true}}%
8176 }%
8177 {%
8178 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8179 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8180 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8181 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8182 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8183 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8184 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8185 \ifglxtrinsertinside \else##2\fi
8186 }%
8187 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8188 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8189 \ifglxtrinsertinside \else##2\fi
8190 }%
8191 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8192 \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8193 \ifglxtrinsertinside \else##2\fi
8194 }%
8195 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```

```

8196 \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8197 \ifglxtrinsertinside \else##2\fi
8198 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8199 \renewcommand*{\glxtrinlinefullformat}[2]{%
8200 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8201 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8202 \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8203 }%
8204 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8205 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8206 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8207 \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8208 }%
8209 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8210 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8211 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8212 \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8213 }%
8214 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8215 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8216 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8217 \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8218 }%

```

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.

```

8219 \renewcommand*{\glxtrfullformat}[2]{%
8220 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8221 \ifglxtrinsertinside\else##2\fi
8222 }%
8223 \renewcommand*{\glxtrfullplformat}[2]{%
8224 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8225 \ifglxtrinsertinside\else##2\fi
8226 }%
8227 \renewcommand*{\Glsxtrfullformat}[2]{%
8228 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8229 \ifglxtrinsertinside\else##2\fi
8230 }%
8231 \renewcommand*{\Glsxtrfullplformat}[2]{%
8232 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8233 \ifglxtrinsertinside\else##2\fi
8234 }%
8235 }

```

long-sm Backward compatibility:

```

8236 \@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.

```

8237 \newabbreviationstyle{long-noshort-sm-desc}%
8238 {%
8239   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8240 }%
8241 {%
8242   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8243   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8244   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8245   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8246   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8247 \renewcommand*\glsxtrsubsequentfmt[2]{%
8248   \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8249   \ifglsxtrininsertinside \else##2\fi
8250 }%
8251 \renewcommand*\glsxtrsubsequentplfmt[2]{%
8252   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8253   \ifglsxtrininsertinside \else##2\fi
8254 }%
8255 \renewcommand*\Glsxtrsubsequentfmt[2]{%
8256   \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8257   \ifglsxtrininsertinside \else##2\fi
8258 }%
8259 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
8260   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8261   \ifglsxtrininsertinside \else##2\fi
8262 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8263 \renewcommand*\glsxtrinlinefullformat[2]{%
8264   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8265   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8266   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8267 }%
8268 \renewcommand*\glsxtrinlinefullplformat[2]{%
8269   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8270   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8271   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8272 }%
8273 \renewcommand*\Glsxtrinlinefullformat[2]{%
8274   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8275   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8276   \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8277 }%
8278 \renewcommand*\Glsxtrinlinefullplformat[2]{%
8279   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8280   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%

```

```

8281 \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8282 }%

```

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.

```

8283 \renewcommand*{\glxtrfullformat}[2]{%
8284   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8285   \ifglxtrininsertinside\else##2\fi
8286 }%
8287 \renewcommand*{\glxtrfullplformat}[2]{%
8288   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8289   \ifglxtrininsertinside\else##2\fi
8290 }%
8291 \renewcommand*{\Glsxtrfullformat}[2]{%
8292   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8293   \ifglxtrininsertinside\else##2\fi
8294 }%
8295 \renewcommand*{\Glsxtrfullplformat}[2]{%
8296   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8297   \ifglxtrininsertinside\else##2\fi
8298 }%
8299 }

```

long-desc-sm Backward compatibility:

```

8300 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```

8301 \newabbreviationstyle{short-sm-footnote}%
8302 {%
8303   \renewcommand*{\CustomAbbreviationFields}{%
8304     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8305     sort={\the\glsshorttok},
8306     description={\the\glslongtok},%
8307     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8308       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8309       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8310     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8311       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8312       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8313     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

8314 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8315   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8316   \glsattribute{\the\glslabeltok}{regular}%
8317   {%
8318     \glssetattribute{\the\glslabeltok}{regular}{false}%
8319   }%
8320 }%

```

```

8321 }%
8322 }%
8323 {%
8324 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8325 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8326 \renewcommand*\abbrvpluralsuffix{\protect\glsxrtrmsuffix}%
8327 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8328 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

8329 \renewcommand*\glsxtrfullformat[2]{%
8330 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8331 \ifglsxtrininsertinside\else##2\fi
8332 \protect\glsxtrabbrvfootnote{##1}%
8333 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8334 }%
8335 \renewcommand*\glsxtrfullplformat[2]{%
8336 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8337 \ifglsxtrininsertinside\else##2\fi
8338 \protect\glsxtrabbrvfootnote{##1}%
8339 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8340 }%
8341 \renewcommand*\Glsxtrfullformat[2]{%
8342 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8343 \ifglsxtrininsertinside\else##2\fi
8344 \protect\glsxtrabbrvfootnote{##1}%
8345 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8346 }%
8347 \renewcommand*\Glsxtrfullplformat[2]{%
8348 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8349 \ifglsxtrininsertinside\else##2\fi
8350 \protect\glsxtrabbrvfootnote{##1}%
8351 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8352 }%

```

The first use full form and the inline full form use the short (long) style.

```

8353 \renewcommand*\glsxtrininlinefullformat[2]{%
8354 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8355 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8356 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8357 }%
8358 \renewcommand*\glsxtrininlinefullplformat[2]{%
8359 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8360 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8361 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8362 }%
8363 \renewcommand*\Glsxtrininlinefullformat[2]{%
8364 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8365 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8366 \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%

```

```

8367 }%
8368 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8369   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8370   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8371   \glxtrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
8372 }%
8373 }

```

footnote-sm Backward compatibility:

```

8374 \@glxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

sm-postfootnote

```

8375 \newabbreviationstyle{short-sm-postfootnote}%
8376 {%
8377   \renewcommand*{\CustomAbbreviationFields}{%
8378     name={\protect\glsabbrvsmfont{\the\glsshorttok}},
8379     sort={\the\glsshorttok},
8380     description={\the\glslongtok},%
8381     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8382     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8383     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.

```

8384 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8385   \csdef{glxtrpostlink\glscategorylabel}{%
8386     \glxtrifwasfirstuse
8387     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8388       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8389       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
8390     }%
8391   }%
8392 }%
8393 \glshasattribute{\the\glslabeltok}{regular}%
8394 {%
8395   \glissetattribute{\the\glslabeltok}{regular}{false}%
8396 }%
8397 {}%
8398 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

8399 \renewcommand*{\glxtrsetupfulldefs}{%
8400   \let\glxtrifwasfirstuse\@secondoftwo
8401 }%
8402 }%
8403 {%

```

```

8404 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8405 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8406 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8407 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8408 \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8409 \renewcommand*\glsxtrfullformat}[2]{%
8410   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8411   \ifglsxtrininsertinside\else##2\fi
8412 }%
8413 \renewcommand*\glsxtrfullplformat}[2]{%
8414   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8415   \ifglsxtrininsertinside\else##2\fi
8416 }%
8417 \renewcommand*\Glsxtrfullformat}[2]{%
8418   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8419   \ifglsxtrininsertinside\else##2\fi
8420 }%
8421 \renewcommand*\Glsxtrfullplformat}[2]{%
8422   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8423   \ifglsxtrininsertinside\else##2\fi
8424 }%

```

The first use full form and the inline full form use the short (long) style.

```

8425 \renewcommand*\glsxtrinlinefullformat}[2]{%
8426   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8427   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8428   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8429 }%
8430 \renewcommand*\glsxtrinlinefullplformat}[2]{%
8431   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8432   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8433   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8434 }%
8435 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8436   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8437   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8438   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8439 }%
8440 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8441   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8442   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8443   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8444 }%
8445 }

```

postfootnote-sm Backward compatibility:

```

8446 \@glsxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```


1.6.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```
8447 \newcommand*{\glsabbrvemfont}[1]{\emph{#1}}%
```

`\glsfirstabbrvemfont`

```
8448 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{#1}}%
```

The default short form suffix:

`\glsxtremsuffix`

```
8449 \newcommand*{\glsxtremsuffix}{\glsxtrabbrvpluralsuffix}
```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```
8450 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{#1}}%
```

`\glslongemfont` Only used by the “long-em” styles.

```
8451 \newcommand*{\glslongemfont}[1]{\emph{#1}}%
```

`\glslong-short-em` The long form is just set in the default long font.

```
8452 \newabbreviationstyle{long-short-em}%
8453 {%
8454   \renewcommand*{\CustomAbbreviationFields}{%
8455     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8456     sort={\the\glsshorttok},
8457     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}}%
8458     \protect\glsxtrfullsep{\the\glslabeltok}}%
8459     \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8460     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}}%
8461     \protect\glsxtrfullsep{\the\glslabeltok}}%
8462     \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8463     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8464     description={\the\glslongtok}}%
8465   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8466     \glshasattribute{\the\glslabeltok}{regular}}%
8467     {%
8468       \glssetattribute{\the\glslabeltok}{regular}{false}}%
8469     }%
8470   }%
8471 }%
8472 }%
8473 {%
8474   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8475   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8476   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%

```

Use the default long fonts.

```
8477 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8478 \renewcommand*\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
8479 \renewcommand*\glsxtrfullformat}[2]{%
8480   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8481   \ifglsxtrinsertinside\else##2\fi
8482   \glsxtrfullsep{##1}%
8483   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8484 }%
8485 \renewcommand*\glsxtrfullplformat}[2]{%
8486   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8487   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8488   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8489 }%
8490 \renewcommand*\Glsxtrfullformat}[2]{%
8491   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8492   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8493   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8494 }%
8495 \renewcommand*\Glsxtrfullplformat}[2]{%
8496   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
8497   \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8498   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8499 }%
8500 }
```

g-short-em-desc

```
8501 \newabbreviationstyle{long-short-em-desc}%
8502 {%
8503   \renewcommand*\CustomAbbreviationFields{%
8504     name={\glsxtrlongshortdescname},
8505     sort={\glsxtrlongshortdescsort},%
8506     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8507       \protect\glsxtrfullsep{\the\glslabeltok}%
8508       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8509     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8510       \protect\glsxtrfullsep{\the\glslabeltok}%
8511       \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8512     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8513     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8514   }%
```

Unset the regular attribute if it has been set.

```
8515 \renewcommand*\GlsXtrPostNewAbbreviation{%
8516   \glshasattribute{\the\glslabeltok}{regular}%
8517   {%
8518     \glssetattribute{\the\glslabeltok}{regular}{false}%
8519   }%
```

```

8520   {}%
8521   }%
8522 }%
8523 {%

```

As long-short-em style:

```

8524   \GlsXtrUseAbbrStyleFmts{long-short-em}%
8525 }

```

ong-em-short-em

```

8526 \newabbreviationstyle{long-em-short-em}%
8527 {%

```

\glslongemfont is used in the description since \glsdesc doesn't set the style.

```

8528 \renewcommand*{\CustomAbbreviationFields}{%
8529   name={\protect\glsabbrvemfont{\the\glsshorttok}},
8530   sort={\the\glsshorttok},
8531   first={\protect\glsfirstlongemfont{\the\glslongtok}}%
8532   \protect\glsxtrfullsep{\the\glslabeltok}}%
8533   \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8534   firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}}%
8535   \protect\glsxtrfullsep{\the\glslabeltok}}%
8536   \glsxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%

8537   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8538   description={\protect\glslongemfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

8539 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8540   \glsattribute{\the\glslabeltok}{regular}%
8541   {%
8542     \glssetattribute{\the\glslabeltok}{regular}{false}%
8543     }%
8544   }%
8545 }%
8546 }%
8547 {%
8548 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8549 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8550 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8551 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8552 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8553 \renewcommand*{\glsxtrfullformat}[2]{%
8554   \glsfirstlongemfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
8555   \ifglsxtrinsertinside\else##2\fi
8556   \glsxtrfullsep{##1}}%
8557   \glsxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8558 }%
8559 \renewcommand*{\glsxtrfullplformat}[2]{%

```

```

8560 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8561 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8562 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8563 }%
8564 \renewcommand*{\Glsxtrfullformat}[2]{%
8565 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8566 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8567 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8568 }%
8569 \renewcommand*{\Glsxtrfullplformat}[2]{%
8570 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8571 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8572 \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8573 }%
8574 }

```

m-short-em-desc

```

8575 \newabbreviationstyle{long-em-short-em-desc}%
8576 {%
8577 \renewcommand*{\CustomAbbreviationFields}{%
8578 name={\glxtrlongshortdescname},
8579 sort={\glxtrlongshortdescsort},%
8580 first={\protect\glsfirstlongemfont{\the\glslongtok}%
8581 \protect\glxtrfullsep{\the\glslabeltok}%
8582 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8583 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}%
8584 \protect\glxtrfullsep{\the\glslabeltok}%
8585 \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8586 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8587 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8588 }%

```

Unset the regular attribute if it has been set.

```

8589 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8590 \glshasattribute{\the\glslabeltok}{regular}%
8591 {%
8592 \glssetattribute{\the\glslabeltok}{regular}{false}%
8593 }%
8594 {}%
8595 }%
8596 }%
8597 {%
8598 \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8599 }

```

short-em-long Now the short (long) version

```

8600 \newabbreviationstyle{short-em-long}%
8601 {%
8602 \renewcommand*{\CustomAbbreviationFields}{%
8603 name={\protect\glsabbrvemfont{\the\glsshorttok}},

```

```

8604     sort={\the\glsshorttok},
8605     description={\the\glslongtok},%
8606     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8607       \protect\glsxtrfullsep{\the\glslabeltok}%
8608       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8609     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8610       \protect\glsxtrfullsep{\the\glslabeltok}%
8611       \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8612     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8613 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8614   \glsattribute{\the\glslabeltok}{regular}%
8615   {%
8616     \glssetattribute{\the\glslabeltok}{regular}{false}%
8617   }%
8618   {}%
8619 }%
8620}%
8621{%

```

Mostly as short-long style:

```

8622 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrsuffix}%
8623 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8624 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8625 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8626 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8627 \renewcommand*{\glsxtrfullformat}[2]{%
8628   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8629   \ifglxtrinsertinside\else##2\fi
8630   \glsxtrfullsep{##1}%
8631   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8632 }%
8633 \renewcommand*{\glsxtrfullplformat}[2]{%
8634   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8635   \ifglxtrinsertinside\else##2\fi
8636   \glsxtrfullsep{##1}%
8637   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8638 }%
8639 \renewcommand*{\Glsxtrfullformat}[2]{%
8640   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8641   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8642   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8643 }%
8644 \renewcommand*{\Glsxtrfullplformat}[2]{%
8645   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8646   \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8647   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8648 }%

```

8649 }

rt-em-long-desc As before but user provides description

```
8650 \newabbreviationstyle{short-em-long-desc}%
8651 {%
8652   \renewcommand*{\CustomAbbreviationFields}{%
8653     name={\glstrshortlongdescname},
8654     sort={\glstrshortlongdescsort},
8655     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8656       \protect\glstrfullsep{\the\glslabeltok}%
8657       \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8658     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8659       \protect\glstrfullsep{\the\glslabeltok}%
8660       \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8661     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8662     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8663   }%
```

Unset the regular attribute if it has been set.

```
8664   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8665     \glshasattribute{\the\glslabeltok}{regular}%
8666     {%
8667       \glissetattribute{\the\glslabeltok}{regular}{false}%
8668     }%
8669   }%
8670 }%
8671 }%
8672 {%
8673   \GlsXtrUseAbbrStyleFmts{short-em-long}%
8674 }
```

hort-em-long-em

```
8675 \newabbreviationstyle{short-em-long-em}%
8676 {%
8677   \glslongemfont is used in the description since \glsdesc doesn't set the style.
8678   \renewcommand*{\CustomAbbreviationFields}{%
8679     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8680     sort={\the\glsshorttok},
8681     description={\protect\glslongemfont{\the\glslongtok}},%
8682     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8683       \protect\glstrfullsep{\the\glslabeltok}%
8684       \glstrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8685     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8686       \protect\glstrfullsep{\the\glslabeltok}%
8687       \glstrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8688     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}%
8689   }%
```

Unset the regular attribute if it has been set.

```

8688 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8689   \glshasattribute{\the\glslabeltok}{regular}}%
8690   {%
8691     \glissetattribute{\the\glslabeltok}{regular}{false}%
8692   }%
8693   {}%
8694 }%
8695 }%
8696 {%
8697 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
8698 \renewcommand*{\glssabbrvfnt}[1]{\glssabbrvemfont{##1}}%
8699 \renewcommand*{\glssfirstabbrvfnt}[1]{\glssfirstabbrvemfont{##1}}%
8700 \renewcommand*{\glssfirstlongfnt}[1]{\glssfirstlongemfont{##1}}%
8701 \renewcommand*{\glsslongfnt}[1]{\glsslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8702 \renewcommand*{\glssxtrfullformat}[2]{%
8703   \glssfirstabbrvemfont{\glssaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8704   \ifglssxtrininsertinside\else##2\fi
8705   \glssxtrfullsep{##1}}%
8706   \glssxtrparen{\glssfirstlongemfont{\glssaccesslong{##1}}}%
8707 }%
8708 \renewcommand*{\glssxtrfullplformat}[2]{%
8709   \glssfirstabbrvemfont{\glssaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8710   \ifglssxtrininsertinside\else##2\fi
8711   \glssxtrfullsep{##1}}%
8712   \glssxtrparen{\glssfirstlongemfont{\glssaccesslongpl{##1}}}%
8713 }%
8714 \renewcommand*{\Glsxtrfullformat}[2]{%
8715   \glssfirstabbrvemfont{\Glsaccessshort{##1}\ifglssxtrininsertinside##2\fi}%
8716   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8717   \glssxtrparen{\glssfirstlongemfont{\glssaccesslong{##1}}}%
8718 }%
8719 \renewcommand*{\Glsxtrfullplformat}[2]{%
8720   \glssfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglssxtrininsertinside##2\fi}%
8721   \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8722   \glssxtrparen{\glssfirstlongemfont{\glssaccesslongpl{##1}}}%
8723 }%
8724 }

```

em-long-em-desc

```

8725 \newabbreviationstyle{short-em-long-em-desc}%
8726 {%
8727 \renewcommand*{\CustomAbbreviationFields}{%
8728   name={\glssxtrshortlongdescname},%
8729   sort={\glssxtrshortlongdescsort},%
8730   first={\protect\glssfirstabbrvemfont{\the\glssshorttok}}%
8731   \protect\glssxtrfullsep{\the\glslabeltok}}%
8732   \glssxtrparen{\protect\glssfirstlongemfont{\the\glsslongtok}}},%
8733   firstplural={\protect\glssfirstabbrvemfont{\the\glssshortpltok}}%

```

```

8734 \protect\glxtrfullsep{\the\glslabeltok}%
8735 \glxtrparen{\protect\glslfirstlongemfont{\the\glslongpltok}}},%
8736 text={\protect\glssabrvemfont{\the\glsshorttok}},%
8737 plural={\protect\glssabrvemfont{\the\glsshortpltok}}}%
8738 }%

```

Unset the regular attribute if it has been set.

```

8739 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8740 \glshasattribute{\the\glslabeltok}{regular}%
8741 {%
8742 \glissetattribute{\the\glslabeltok}{regular}{false}%
8743 }%
8744 {}%
8745 }%
8746 }%
8747 {%
8748 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8749 }

```

short-em

```

8750 \newabbreviationstyle{short-em}%
8751 {%
8752 \renewcommand*{\CustomAbbreviationFields}{%
8753 name={\protect\glssabrvemfont{\the\glsshorttok}},
8754 sort={\the\glsshorttok},
8755 first={\protect\glslfirstabrvemfont{\the\glsshorttok}},
8756 firstplural={\protect\glslfirstabrvemfont{\the\glsshortpltok}},
8757 text={\protect\glssabrvemfont{\the\glsshorttok}},
8758 plural={\protect\glssabrvemfont{\the\glsshortpltok}},
8759 description={\the\glslongtok}}%
8760 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8761 \glissetattribute{\the\glslabeltok}{regular}{true}}%
8762 }%
8763 {%
8764 \renewcommand*{\abbrvpluralsuffix}{\protect\glstxtremsuffix}%
8765 \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
8766 \renewcommand*{\glslfirstabrvfont}[1]{\glslfirstabrvemfont{##1}}%
8767 \renewcommand*{\glslfirstlongfont}[1]{\glslfirstlongdefaultfont{##1}}%
8768 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8769 \renewcommand*{\glxtrinlinefullformat}[2]{%
8770 \protect\glslfirstabrvemfont{\glssaccessshort{##1}}%
8771 \ifglxtrininsertinside##2\fi}%
8772 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8773 \glxtrparen{\glslfirstlongdefaultfont{\glssaccesslong{##1}}}%
8774 }%
8775 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8776 \protect\glslfirstabrvemfont{\glssaccessshortpl{##1}}%
8777 \ifglxtrininsertinside##2\fi}%

```



```

8778 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8779 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8780 }%

8781 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8782 \protect\glsfirstabbrvemfont{\Glsaccessshort{##1}%
8783 \ifglxtrinsertinside##2\fi}%
8784 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8785 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
8786 }%
8787 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8788 \protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}%
8789 \ifglxtrinsertinside##2\fi}%
8790 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8791 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8792 }%

```

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.

```

8793 \renewcommand*{\glxtrfullformat}[2]{%
8794 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8795 \ifglxtrinsertinside\else##2\fi
8796 }%
8797 \renewcommand*{\glxtrfullplformat}[2]{%
8798 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8799 \ifglxtrinsertinside\else##2\fi
8800 }%
8801 \renewcommand*{\Glsxtrfullformat}[2]{%
8802 \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8803 \ifglxtrinsertinside\else##2\fi
8804 }%
8805 \renewcommand*{\Glsxtrfullplformat}[2]{%
8806 \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8807 \ifglxtrinsertinside\else##2\fi
8808 }%
8809 }

```

short-em-nolong

```

8810 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8811 \newabbreviationstyle{short-em-desc}%
8812 {%
8813 \renewcommand*{\CustomAbbreviationFields}{%
8814 name={\glxtrshortdescname},
8815 sort={\the\glssshorttok},
8816 first={\protect\glsfirstabbrvemfont{\the\glssshorttok}},
8817 firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}},
8818 text={\protect\glssabbrvemfont{\the\glssshorttok}},

```

```

8819 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8820 description={\the\glslongtok}}%
8821 \renewcommand*\GlsXtrPostNewAbbreviation}{%
8822   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8823 }%
8824 {%
8825   \renewcommand*\abbrvpluralsuffix{\protect\glxtremsuffix}%
8826   \renewcommand*\glsabbrvfont[1]{\glsabbrvemfont{##1}}%
8827   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}%
8828   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8829   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8830 \renewcommand*\glsxtrinlinefullformat}[2]{%
8831   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8832   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8833   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8834 }%
8835 \renewcommand*\glsxtrinlinefullplformat}[2]{%
8836   \glsfirstabbrvemfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8837   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8838   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8839 }%
8840 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8841   \glsfirstabbrvemfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8842   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8843   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8844 }%
8845 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8846   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8847   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8848   \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8849 }%

```

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.

```

8850 \renewcommand*\glsxtrfullformat}[2]{%
8851   \glsfirstabbrvemfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8852   \ifglsxtrininsertinside\else##2\fi
8853 }%
8854 \renewcommand*\glsxtrfullplformat}[2]{%
8855   \glsfirstabbrvemfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8856   \ifglsxtrininsertinside\else##2\fi
8857 }%
8858 \renewcommand*\Glsxtrfullformat}[2]{%
8859   \glsfirstabbrvemfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
8860   \ifglsxtrininsertinside\else##2\fi
8861 }%
8862 \renewcommand*\Glsxtrfullplformat}[2]{%
8863   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

8864     \ifglxtrinsertinside\else##2\fi
8865 }%
8866 }

```

-em-nolong-desc

```

8867 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8868 \newabbreviationstyle{nolong-short-em}%
8869 {%
8870   \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8871 }%
8872 {%
8873   \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8874 \renewcommand*{\glxtrinlinefullformat}[2]{%
8875   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8876     \ifglxtrinsertinside##2\fi}%
8877   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8878   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8879 }%
8880 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8881   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8882     \ifglxtrinsertinside##2\fi}%
8883   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8884   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8885 }%
8886 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8887   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8888     \ifglxtrinsertinside##2\fi}%
8889   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8890   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8891 }%
8892 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8893   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8894     \ifglxtrinsertinside##2\fi}%
8895   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8896   \glxtrparen{\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8897 }%
8898 }

```

long-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8899 \newabbreviationstyle{long-noshort-em}%
8900 {%
8901   \renewcommand*{\CustomAbbreviationFields}{%
8902     name={\protect\glsabbrvemfont{\the\glsshorttok}},
8903     sort={\the\glsshorttok},
8904     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},

```

```

8905     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8906     text={\protect\glslongdefaultfont{\the\glslongtok}},
8907     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
8908     description={\the\glslongtok}%
8909 }%
8910 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8911   \glssetattribute{\the\glslabeltok}{regular}{true}}%
8912 }%
8913 {%
8914   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8915   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8916   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8917   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8918   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8919   \renewcommand*{\glsxtrsubsequentfmt}[2]{%
8920     \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8921     \ifglsxtrininsertinside \else##2\fi
8922   }%
8923   \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
8924     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8925     \ifglsxtrininsertinside \else##2\fi
8926   }%
8927   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8928     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8929     \ifglsxtrininsertinside \else##2\fi
8930   }%
8931   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8932     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8933     \ifglsxtrininsertinside \else##2\fi
8934   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8935   \renewcommand*{\glsxtrinlinefullformat}[2]{%
8936     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8937     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8938     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8939   }%
8940   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8941     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8942     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8943     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
8944   }%
8945   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8946     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8947     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8948     \glsxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
8949   }%
8950   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%

```

```

8951 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8952 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8953 \glxtrparen{\protect\glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
8954 }%

```

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.

```

8955 \renewcommand*{\glxtrfullformat}[2]{%
8956 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8957 \ifglxtrinsertinside\else##2\fi
8958 }%
8959 \renewcommand*{\glxtrfullplformat}[2]{%
8960 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8961 \ifglxtrinsertinside\else##2\fi
8962 }%
8963 \renewcommand*{\Glsxtrfullformat}[2]{%
8964 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8965 \ifglxtrinsertinside\else##2\fi
8966 }%
8967 \renewcommand*{\Glsxtrfullplformat}[2]{%
8968 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8969 \ifglxtrinsertinside\else##2\fi
8970 }%
8971 }

```

long-em Backward compatibility:

```

8972 \@glxtr@deprecated@abbrstyle{long-em}{long-noshort-em}

```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```

8973 \newabbreviationstyle{long-em-noshort-em}%
8974 {%
8975 \renewcommand*{\CustomAbbreviationFields}{%
8976 name={\protect\glsabbrvemfont{\the\glsshorttok}},
8977 sort={\the\glsshorttok},
8978 first={\protect\glsfirstlongemfont{\the\glslongtok}},
8979 firstplural={\protect\glsfirstlongemfont{\the\glslongpltok}},
8980 text={\protect\glslongemfont{\the\glslongtok}},
8981 plural={\protect\glslongemfont{\the\glslongpltok}},%
8982 description={\protect\glslongemfont{\the\glslongtok}}}%
8983 }%
8984 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8985 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8986 }%
8987 {%
8988 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8989 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8990 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8991 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8992 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8993 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8994   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8995   \ifglxtrinsertinside \else##2\fi
8996 }%
8997 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8998   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8999   \ifglxtrinsertinside \else##2\fi
9000 }%
9001 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9002   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9003   \ifglxtrinsertinside \else##2\fi
9004 }%
9005 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9006   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9007   \ifglxtrinsertinside \else##2\fi
9008 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9009 \renewcommand*{\glxtrinlinefullformat}[2]{%
9010   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9011   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9012   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9013 }%
9014 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9015   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9016   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9017   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9018 }%
9019 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9020   \glslongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9021   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9022   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9023 }%
9024 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9025   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9026   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9027   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9028 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

9029 \renewcommand*{\glxtrfullformat}[2]{%
9030   \glslongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9031   \ifglxtrinsertinside\else##2\fi
9032 }%
9033 \renewcommand*{\glxtrfullplformat}[2]{%
9034   \glslongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9035   \ifglxtrinsertinside\else##2\fi
9036 }%

```

```

9037 \renewcommand*{\Glsxtrfullformat}[2]{%
9038   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9039   \ifglxtrinsertinside\else##2\fi
9040 }%
9041 \renewcommand*{\Glsxtrfullplformat}[2]{%
9042   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9043   \ifglxtrinsertinside\else##2\fi
9044 }%
9045 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9046 \newabbreviationstyle{long-em-noshort-em-noreg}%
9047 {%
9048   \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%
   Unset the regular attribute if it has been set.
9049   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9050     \glshasattribute{\the\glslabeltok}{regular}%
9051     {%
9052       \glissetattribute{\the\glslabeltok}{regular}{false}%
9053     }%
9054   }%
9055 }%
9056 }%
9057 {%
9058   \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9059 }

```

`noshort-em-desc` The emphasized font will only be used if the short form is explicitly invoked through commands like `\glsshort`.

```

9060 \newabbreviationstyle{long-noshort-em-desc}%
9061 {%
9062   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9063 }%
9064 {%
9065   \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9066   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9067   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9068   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9069   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
   The format for subsequent use (not used when the regular attribute is set).
9070   \renewcommand*{\glxtrsubsequentfmt}[2]{%
9071     \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9072     \ifglxtrinsertinside \else##2\fi
9073   }%
9074   \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9075     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9076     \ifglxtrinsertinside \else##2\fi
9077   }%

```

```

9078 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9079   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9080   \ifglxtrinsertinside \else##2\fi
9081 }%
9082 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9083   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9084   \ifglxtrinsertinside \else##2\fi
9085 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9086 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9087   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9088   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
9089   \Glsxtrparen{\protect\Glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9090 }%
9091 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9092   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9093   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
9094   \Glsxtrparen{\protect\Glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9095 }%
9096 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9097   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9098   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
9099   \Glsxtrparen{\protect\Glsfirstabbrvemfont{\Glsaccessshort{##1}}}%
9100 }%
9101 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9102   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9103   \ifglxtrinsertinside\else##2\fi\Glsxtrfullsep{##1}%
9104   \Glsxtrparen{\protect\Glsfirstabbrvemfont{\Glsaccessshortpl{##1}}}%
9105 }%

```

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.

```

9106 \renewcommand*{\Glsxtrfullformat}[2]{%
9107   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9108   \ifglxtrinsertinside\else##2\fi
9109 }%
9110 \renewcommand*{\Glsxtrfullplformat}[2]{%
9111   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9112   \ifglxtrinsertinside\else##2\fi
9113 }%
9114 \renewcommand*{\Glsxtrfullformat}[2]{%
9115   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9116   \ifglxtrinsertinside\else##2\fi
9117 }%
9118 \renewcommand*{\Glsxtrfullplformat}[2]{%
9119   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9120   \ifglxtrinsertinside\else##2\fi
9121 }%
9122 }

```


long-desc-em Backward compatibility:

```
9123 \@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.

```
9124 \newabbreviationstyle{long-em-noshort-em-desc}%
9125 {%
9126   \renewcommand*{\CustomAbbreviationFields}{%
9127     name={\protect\protect\glslongemfont{\the\glslongtok}},
9128     sort={\the\glslongtok},
9129     first={\protect\glslfirstlongemfont{\the\glslongtok}},
9130     firstplural={\protect\glslfirstlongemfont{\the\glslongpltok}},
9131     text={\glslongemfont{\the\glslongtok}},
9132     plural={\glslongemfont{\the\glslongpltok}}}%
9133   }%
9134   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9135     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9136 }%
9137 {%
9138   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtrmsuffix}%
9139   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9140   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
9141   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongemfont{##1}}%
9142   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
9143 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9144   \glslongemfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9145   \ifglsxtrininsertinside \else##2\fi
9146 }%
9147 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9148   \glslongemfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9149   \ifglsxtrininsertinside \else##2\fi
9150 }%
9151 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9152   \glslongemfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9153   \ifglsxtrininsertinside \else##2\fi
9154 }%
9155 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9156   \glslongemfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9157   \ifglsxtrininsertinside \else##2\fi
9158 }%
```

The inline full form displays the long format followed by the short form in parentheses.

```
9159 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9160   \glssfirstlongemfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9161   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9162   \glsxtrparen{\protect\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
9163 }%
9164 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
```

```

9165 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9166 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9167 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9168 }%
9169 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9170 \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9171 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9172 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9173 }%
9174 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9175 \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9176 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9177 \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9178 }%

```

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.

```

9179 \renewcommand*{\glxtrfullformat}[2]{%
9180 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9181 \ifglxtrinsertinside\else##2\fi
9182 }%
9183 \renewcommand*{\glxtrfullplformat}[2]{%
9184 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9185 \ifglxtrinsertinside\else##2\fi
9186 }%
9187 \renewcommand*{\Glsxtrfullformat}[2]{%
9188 \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9189 \ifglxtrinsertinside\else##2\fi
9190 }%
9191 \renewcommand*{\Glsxtrfullplformat}[2]{%
9192 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9193 \ifglxtrinsertinside\else##2\fi
9194 }%
9195 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9196 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9197 {%
9198 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%

```

Unset the regular attribute if it has been set.

```

9199 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9200 \glshasattribute{\the\glslabeltok}{regular}%
9201 {%
9202 \glssetattribute{\the\glslabeltok}{regular}{false}%
9203 }%
9204 {}%
9205 }%
9206 }%
9207 {%

```

```

9208 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9209 }

```

ort-em-footnote

```

9210 \newabbreviationstyle{short-em-footnote}%
9211 {%
9212 \renewcommand*{\CustomAbbreviationFields}{%
9213   name={\protect\glsabbrvemfont{\the\glsshorttok}},
9214   sort={\the\glsshorttok},
9215   description={\the\glslongtok},%
9216   first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
9217     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
9218     {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9219   firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
9220     \protect\glsxtrabbrvfootnote{\the\glslabeltok}%
9221     {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9222   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.

```

9223 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9224 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9225 \glsattribute{\the\glslabeltok}{regular}%
9226 {%
9227 \glssetattribute{\the\glslabeltok}{regular}{false}%
9228 }%
9229 {}%
9230 }%
9231 }%
9232 {%
9233 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
9234 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9235 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9236 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9237 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9238 \renewcommand*{\glsxtrfullformat}[2]{%
9239 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9240 \ifglsxtrininsertinside\else##2\fi
9241 \protect\glsxtrabbrvfootnote{##1}%
9242 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9243 }%
9244 \renewcommand*{\glsxtrfullplformat}[2]{%
9245 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9246 \ifglsxtrininsertinside\else##2\fi
9247 \protect\glsxtrabbrvfootnote{##1}%
9248 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9249 }%
9250 \renewcommand*{\Glsxtrfullformat}[2]{%

```

```

9251 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9252 \ifglxtrinsertinside\else##2\fi
9253 \protect\glxtrabbrvfootnote{##1}%
9254 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9255 }%
9256 \renewcommand*{\Glsxtrfullplformat}[2]{%
9257 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9258 \ifglxtrinsertinside\else##2\fi
9259 \protect\glxtrabbrvfootnote{##1}%
9260 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9261 }%

```

The first use full form and the inline full form use the short (long) style.

```

9262 \renewcommand*{\glxtrinlinefullformat}[2]{%
9263 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9264 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9265 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9266 }%
9267 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9268 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9269 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9270 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9271 }%
9272 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9273 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9274 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9275 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9276 }%
9277 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9278 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9279 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9280 \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9281 }%
9282 }

```

footnote-em Backward compatibility:

```

9283 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9284 \newabbreviationstyle{short-em-postfootnote}%
9285 {%
9286 \renewcommand*{\CustomAbbreviationFields}{%
9287 name={\protect\glsabbrvemfont{\the\glsshorttok}},
9288 sort={\the\glsshorttok},
9289 description={\the\glslongtok},%
9290 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
9291 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
9292 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```
9293 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9294   \csdef{glxstrpostlink\glscategorylabel}{%
9295     \glxstrifwasfirstuse
9296     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
9297       \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
9298       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
9299   }%
9300 }%
9301 }%
9302 \glshasattribute{\the\glslabeltok}{regular}%
9303 {%
9304   \glissetattribute{\the\glslabeltok}{regular}{false}%
9305   }%
9306   }%
9307 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
9308 \renewcommand*{\glxtrsetupfulldefs}{%
9309   \let\glxstrifwasfirstuse\@secondoftwo
9310 }%
9311 }%
9312 {%
9313 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9314 \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
9315 \renewcommand*{\glfirstabbrvfont}[1]{\glfirstabbrvemfont{##1}}%
9316 \renewcommand*{\glfirstlongfont}[1]{\glfirstlongfootnotefont{##1}}%
9317 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%
```

The full format displays the short form. The long form is deferred.

```
9318 \renewcommand*{\glxtrfullformat}[2]{%
9319   \glsfirstabbrvemfont{\glaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9320   \ifglxtrininsertinside\else##2\fi
9321 }%
9322 \renewcommand*{\glxtrfullplformat}[2]{%
9323   \glsfirstabbrvemfont{\glaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9324   \ifglxtrininsertinside\else##2\fi
9325 }%
9326 \renewcommand*{\Glsxtrfullformat}[2]{%
9327   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9328   \ifglxtrininsertinside\else##2\fi
9329 }%
9330 \renewcommand*{\Glsxtrfullplformat}[2]{%
9331   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9332   \ifglxtrininsertinside\else##2\fi
```

9333 }%

The first use full form and the inline full form use the short (long) style.

```
9334 \renewcommand*\glxstrinlinefullformat}[2]{%
9335   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
9336   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9337   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9338 }%
9339 \renewcommand*\glxstrinlinefullplformat}[2]{%
9340   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
9341   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9342   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9343 }%
9344 \renewcommand*\Glsxtrinlinefullformat}[2]{%
9345   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
9346   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9347   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9348 }%
9349 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
9350   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
9351   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9352   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9353 }%
9354 }
```

postfootnote-em Backward compatibility:

```
9355 \@glxstr@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:

glxstruserfield Default is the useri field.

```
9356 \newcommand*\glxstruserfield}{useri}
```

glxstruserparen 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.

```
9357 \ifdef\glscurrentfieldvalue
9358 {
9359   \newcommand*\glxstruserparen}[2]{%
9360     \glxstrfullsep{#2}%
9361     \glxstrparen
9362     {#1\ifglshasfield{\glxstruserfield}{#2}{, \glscurrentfieldvalue}}}%
9363   }
9364 }
9365 {
9366   \newcommand*\glxstruserparen}[2]{%
```

```

9367 \glxtrfullsep{#2}%
9368 \glxtrparen
9369 {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9370 }
9371 }

```

Font used for short form:

lsabbrvuserfont

```
9372 \newcommand*{\glabbrvuserfont}[1]{\glabbrvdefaultfont{#1}}
```

Font used for short form on first use:

stabbrvuserfont

```
9373 \newcommand*{\glfirstabbrvuserfont}[1]{\glabbrvuserfont{#1}}
```

Font used for long form:

glslonguserfont

```
9374 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

rstlonguserfont

```
9375 \newcommand*{\glfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

lsxtrusersuffix

```
9376 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

long-short-user

```

9377 \newabbreviationstyle{long-short-user}%
9378 {%
9379 \renewcommand*{\CustomAbbreviationFields}{%
9380 name={\protect\glabbrvuserfont{\the\glsshorttok}},
9381 sort={\the\glsshorttok},
9382 first={\protect\glfirstlonguserfont{\the\glslongtok}}%
9383 \protect\glxtruserparen{\protect\glfirstabbrvuserfont{\the\glsshorttok}}%
9384 {\the\glslabeltok}},%
9385 firstplural={\protect\glfirstlonguserfont{\the\glslongpltok}}%
9386 \protect\glxtruserparen
9387 {\protect\glfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9388 plural={\protect\glabbrvuserfont{\the\glsshortpltok}},%
9389 description={\protect\glslonguserfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9390 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9391 \glshasattribute{\the\glslabeltok}{regular}%
9392 }%

```

```

9393     \glsetattribute{\the\glslabelltok}{regular}{false}%
9394 }%
9395 {}%
9396 }%
9397 }%
9398 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9399 \renewcommand*\abbrvpluralsuffix{\glxtrusersuffix}%
9400 \renewcommand*\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9401 \renewcommand*\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9402 \renewcommand*\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9403 \renewcommand*\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9404 \renewcommand*\glxtrfullformat}[2]{%
9405   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9406   \ifglxtrinsertinside\else##2\fi
9407   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9408 }%
9409 \renewcommand*\glxtrfullplformat}[2]{%
9410   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9411   \ifglxtrinsertinside\else##2\fi
9412   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9413 }%
9414 \renewcommand*\Glsxtrfullformat}[2]{%
9415   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9416   \ifglxtrinsertinside\else##2\fi
9417   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9418 }%
9419 \renewcommand*\Glsxtrfullplformat}[2]{%
9420   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9421   \ifglxtrinsertinside\else##2\fi
9422   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9423 }%
9424 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9425 \newabbreviationstyle{long-postshort-user}%
9426 {%
9427   \renewcommand*\CustomAbbreviationFields{%
9428     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9429     sort={\the\glsshorttok},
9430     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9431     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9432     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9433     description={\protect\glslonguserfont{\the\glslongtok}}}%
9434 \renewcommand*\GlsXtrPostNewAbbreviation{%
9435   \csdef{glxtrpostlink\glscategorylabel}{%

```



```

9436     \glxtrifwasfirstuse
9437     {%
9438         \glxtruserparen
9439         {\glsfirstabbrvuserfont{\gl Sentryshort{\glslabel}}}%
9440         {\glslabel}%
9441     }%
9442     {}%
9443 }%
9444 \glshasattribute{\the\glslabeltok}{regular}%
9445 {%
9446     \glissetattribute{\the\glslabeltok}{regular}{false}%
9447 }%
9448 {}%
9449 }%
9450 }%
9451 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9452 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9453 \renewcommand*{\glabbrvfont}[1]{\glabbrvuserfont{##1}}%
9454 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9455 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9456 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9457 \renewcommand*{\glxtrfullformat}[2]{%
9458     \glsfirstlonguserfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9459     \ifglxtrininsertinside\else##2\fi
9460 }%
9461 \renewcommand*{\glxtrfullplformat}[2]{%
9462     \glsfirstlonguserfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9463     \ifglxtrininsertinside\else##2\fi
9464 }%
9465 \renewcommand*{\Glsxtrfullformat}[2]{%
9466     \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9467     \ifglxtrininsertinside\else##2\fi
9468 }%
9469 \renewcommand*{\Glsxtrfullplformat}[2]{%
9470     \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9471     \ifglxtrininsertinside\else##2\fi
9472 }%

```

In-line format:

```

9473 \renewcommand*{\glxtrininlinefullformat}[2]{%
9474     \glsfirstlonguserfont{\glaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9475     \ifglxtrininsertinside\else##2\fi
9476     \glxtruserparen{\glsfirstabbrvuserfont{\glaccessshort{##1}}}{##1}%
9477 }%
9478 \renewcommand*{\glxtrininlinefullplformat}[2]{%
9479     \glsfirstlonguserfont{\glaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

9480 \ifglxtrinsertinside\else##2\fi
9481 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9482 }%
9483 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9484 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9485 \ifglxtrinsertinside\else##2\fi
9486 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshort{##1}}}{##1}%
9487 }%
9488 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9489 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9490 \ifglxtrinsertinside\else##2\fi
9491 \glxtruserparen{\glsfirstabbrvuserfont{\glssaccessshortpl{##1}}}{##1}%
9492 }%
9493 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9494 \newabbreviationstyle{long-postshort-user-desc}%
9495 {%
9496 \renewcommand*{\CustomAbbreviationFields}{%
9497 name={\protect\glslonguserfont{\the\glslongtok}}%
9498 \protect\glxtruserparen
9499 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}},
9500 sort={\the\glslongtok},
9501 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9502 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9503 text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9504 plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
9505 }%
9506 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9507 \csdef{glxtrpostlink\glscategorylabel}{%
9508 \glxtrifwasfirstuse
9509 {%
9510 \glxtruserparen
9511 {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9512 {\glslabel}}%
9513 }%
9514 }%
9515 }%
9516 \glshasattribute{\the\glslabeltok}{regular}%
9517 {%
9518 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
9519 }%
9520 {}%
9521 }%
9522 }%
9523 {%
9524 \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9525 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9526 \newabbreviationstyle{short-postlong-user}%
9527 {%
9528   \renewcommand*{\CustomAbbreviationFields}{%
9529     name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9530     sort={\the\glsshorttok},
9531     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9532     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

9533     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9534     description={\protect\glslonguserfont{\the\glslongtok}}}%
9535 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9536   \csdef{glsxtrpostlink\glscategorylabel}{%
9537     \glsxtrifwasfirstuse
9538     {%
9539       \glsxtruserparen
9540       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9541       {\glslabel}%
9542     }%
9543     }%
9544   }%
9545   \glsattribute{\the\glslabeltok}{regular}%
9546   {%
9547     \glssetattribute{\the\glslabeltok}{regular}{false}%
9548   }%
9549   }%
9550 }%
9551 }%
9552 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9553 \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9554 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9555 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9556 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9557 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9558 \renewcommand*{\glsxtrfullformat}[2]{%
9559   \glsfirstabbrvuserfont{\glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9560   \ifglsxtrininsertinside\else##2\fi
9561 }%
9562 \renewcommand*{\glsxtrfullplformat}[2]{%
9563   \glsfirstabbrvuserfont{\glsaccessshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
9564   \ifglsxtrininsertinside\else##2\fi
9565 }%
9566 \renewcommand*{\Glsxtrfullformat}[2]{%
9567   \glsfirstabbrvuserfont{\Glsaccessshort{##1}}\ifglsxtrininsertinside##2\fi}%
9568   \ifglsxtrininsertinside\else##2\fi
9569 }%

```

```

9570 \renewcommand*{\Glsxtrfullplformat}[2]{%
9571   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9572   \ifglxtrinsertinside\else##2\fi
9573 }%

```

In-line format:

```

9574 \renewcommand*{\glxtrinlinefullformat}[2]{%
9575   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9576   \ifglxtrinsertinside\else##2\fi
9577   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9578 }%
9579 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9580   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9581   \ifglxtrinsertinside\else##2\fi
9582   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9583 }%
9584 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9585   \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
9586   \ifglxtrinsertinside\else##2\fi
9587   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9588 }%
9589 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9590   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
9591   \ifglxtrinsertinside\else##2\fi
9592   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9593 }%
9594 }

```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```

9595 \newabbreviationstyle{short-postlong-user-desc}%
9596 {%
9597   \renewcommand*{\CustomAbbreviationFields}{%
9598     name={\protect\glsabbrvuserfont{\the\glsshorttok}%
9599           \protect\glxtruserparen
9600             {\protect\glslonguserfont{\the\glslongpltok}}}%
9601             {\the\glslabeltok}},
9602     sort={\the\glsshorttok},
9603     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9604     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9605     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9606     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
9607 }%
9608 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9609   \csdef{glxtrpostlink\glscategorylabel}{%
9610     \glxtrifwasfirstuse
9611     {%
9612       \glxtruserparen
9613         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9614         {\glslabel}%

```

```

9615     }%
9616     {}%
9617     }%
9618     \glshasattribute{\the\glslabeltok}{regular}%
9619     {%
9620         \glissetattribute{\the\glslabeltok}{regular}{false}%
9621     }%
9622     {}%
9623 }%
9624 }%
9625 {%
9626     \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9627 }

```

short-user-desc

```

9628 \newabbreviationstyle{long-short-user-desc}%
9629 {%
9630     \renewcommand*{\CustomAbbreviationFields}{%
9631         name={\glsxtrlongshortdescname},
9632         sort={\glsxtrlongshortdescsort},%
9633         first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9634             \protect\glsxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9635             {\the\glslabeltok}},%
9636         firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9637             \protect\glsxtruserparen
9638             {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9639         text={\protect\glsabbrvfont{\the\glsshorttok}},%
9640         plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9641     }%

```

Unset the regular attribute if it has been set.

```

9642     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9643         \glshasattribute{\the\glslabeltok}{regular}%
9644         {%
9645             \glissetattribute{\the\glslabeltok}{regular}{false}%
9646         }%
9647         {}%
9648     }%
9649 }%
9650 {%
9651     \GlsXtrUseAbbrStyleFmts{long-short-user}%
9652 }

```

short-long-user

```

9653 \newabbreviationstyle{short-long-user}%
9654 {%
9655     \glslonguserfont is used in the description since \glsdesc doesn't set the style.
9656     \renewcommand*{\CustomAbbreviationFields}{%

```

```

9656   name={\protect\glsabbrvuserfont{\the\glsshorttok}},
9657   sort={\the\glsshorttok},
9658   description={\protect\glslonguserfont{\the\glslongtok}},%
9659   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}}%
9660     \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}%
9661     {\the\glslabeltok}},%
9662   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}%
9663     \protect\glsxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}%
9664     {\the\glslabeltok}},%
9665   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9666   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9667     \glshasattribute{\the\glslabeltok}{regular}%
9668     {%
9669       \glssetattribute{\the\glslabeltok}{regular}{false}%
9670     }%
9671   }%
9672 }%
9673 }%
9674 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9675   \renewcommand*{\abbrvpluralsuffix}{\glsxtrusersuffix}%
9676   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9677   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9678   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9679   \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9680   \renewcommand*{\glsxtrfullformat}[2]{%
9681     \glsfirstabbrvuserfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9682     \ifglsxtrinsertinside\else##2\fi
9683     \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9684   }%
9685   \renewcommand*{\glsxtrfullplformat}[2]{%
9686     \glsfirstabbrvuserfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9687     \ifglsxtrinsertinside\else##2\fi
9688     \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9689   }%
9690   \renewcommand*{\Glsxtrfullformat}[2]{%
9691     \glsfirstabbrvuserfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
9692     \ifglsxtrinsertinside\else##2\fi
9693     \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslong{##1}}}{##1}%
9694   }%
9695   \renewcommand*{\Glsxtrfullplformat}[2]{%
9696     \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
9697     \ifglsxtrinsertinside\else##2\fi
9698     \glsxtruserparen{\glsfirstlonguserfont{\glsaccesslongpl{##1}}}{##1}%
9699   }%

```

```
9700 }
```

-long-user-desc

```
9701 \newabbreviationstyle{short-long-user-desc}%
9702 {%
9703   \renewcommand*{\CustomAbbreviationFields}{%
9704     name={\glxtrshortlongdescname},
9705     sort={\glxtrshortlongdescsort},%

9706     first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9707       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9708     {\the\glslabeltok}},%
9709     firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9710       \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9711     {\the\glslabeltok}},%
9712     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9713     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9714   }%
```

Unset the regular attribute if it has been set.

```
9715   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9716     \glshasattribute{\the\glslabeltok}{regular}%
9717     {%
9718       \glissetattribute{\the\glslabeltok}{regular}{false}%
9719     }%
9720   }%
9721 }%
9722 }%
9723 {%
9724   \GlsXtrUseAbbrStyleFmts{short-long-user}%
9725 }
```

1.6.7 Predefined Styles (Hyphen)

These styles are designed to work with the markwords attribute. They check if the inserted material (provided by the final optional argument of commands like `\gls`) starts with a hyphen. If it does, the insert is added to the parenthetical material. Note that commands like `\glxtrlong` set `\glsinsert` to empty with the entire link-text stored in `\glscustomtext`.

`\trifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glsinsert` so check for that and expand.

```
9726 \newrobustcmd*{\glxtrifhyphenstart}[3]{%
9727   \ifx\glsinsert#1\relax
9728     \expandafter@\glxtrifhyphenstart#1\relax\relax
9729     \@end@glxtrifhyphenstart{#2}{#3}%
9730   \else
9731     \@glxtrifhyphenstart#1\relax\relax\@end@glxtrifhyphenstart{#2}{#3}%
9732   \fi
9733 }
```

trifhyphenstart

```
9734 \def\@glxtrifhyphenstart#1#2\@end@glxtrifhyphenstart#3#4{%  
9735   \ifx-#1\relax#3\else #4\fi  
9736 }
```

rlonghyphenshort

```
\glxtrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
9737 \newcommand*\glxtrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9738   {%
```

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.

```
9739   \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{ }%  
9740   \glsfirstlonghyphenfont{#2\ifglxtrininsertinside{#4}\fi}%  
9741   \ifglxtrininsertinside\else{#4}\fi  
9742   \glxtrfullsep{#1}%  
9743   \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrininsertinside{#4}\fi}%  
9744     \ifglxtrininsertinside\else{#4}\fi}%  
9745   }%  
9746 }
```

abbrvhyphenfont

```
9747 \newcommand*\glsabbrvhyphenfont{\glsabbrvdefaultfont}%
```

abbrvhyphenfont

```
9748 \newcommand*\glsfirstabbrvhyphenfont{\glsabbrvhyphenfont}%
```

slonghyphenfont

```
9749 \newcommand*\glslonghyphenfont{\glslongdefaultfont}%
```

tlonghyphenfont

```
9750 \newcommand*\glsfirstlonghyphenfont{\glslonghyphenfont}%
```

The default short form suffix:

xtrhyphensuffix

```
9751 \newcommand*\glxtrhyphensuffix{\glxtrabbrvpluralsuffix}
```

en-short-hyphen

Designed for use with the `markwords` attribute.

```
9752 \newabbreviationstyle{long-hyphen-short-hyphen}%  
9753 {%
```



```

9754 \renewcommand*{\CustomAbbreviationFields}{%
9755   name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9756   sort={\the\glsshorttok},
9757   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
9758     \protect\glsxtrfullsep{\the\glslabeltok}%
9759     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9760   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
9761     \protect\glsxtrfullsep{\the\glslabeltok}%
9762     \glsxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9763   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9764   description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9765 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9766   \glshasattribute{\the\glslabeltok}{regular}%
9767   {%
9768     \glssetattribute{\the\glslabeltok}{regular}{false}%
9769   }%
9770   {}%
9771 }%
9772 }%
9773 {%
9774 \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
9775 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9776 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9777 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9778 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9779 \renewcommand*{\glsxtrfullformat}[2]{%
9780   \glsxtrlonghyphenshort{##1}{\glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9781 }%
9782 \renewcommand*{\glsxtrfullplformat}[2]{%
9783   \glsxtrlonghyphenshort{##1}{\glsaccesslongpl{##1}}%
9784   {\glsaccessshortpl{##1}}{##2}%
9785 }%
9786 \renewcommand*{\Glsxtrfullformat}[2]{%
9787   \glsxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glsaccessshort{##1}}{##2}%
9788 }%
9789 \renewcommand*{\Glsxtrfullplformat}[2]{%
9790   \glsxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9791   {\glsaccessshortpl{##1}}{##2}%
9792 }%
9793 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9794 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9795 {%
9796 \renewcommand*{\CustomAbbreviationFields}{%
9797   name={\glsxtrlongshortdescname},

```

```

9798 sort={\glxtrlongshortdescsort},
9799 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
9800 \protect\glxtrfullsep{\the\glslabeltok}%
9801 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9802 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
9803 \protect\glxtrfullsep{\the\glslabeltok}%
9804 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9805 text={\protect\glssabbvrhyphenfont{\the\glsshorttok}},%
9806 plural={\protect\glssabbvrhyphenfont{\the\glsshortpltok}}%
9807 }%

```

Unset the regular attribute if it has been set.

```

9808 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9809 \glshasattribute{\the\glslabeltok}{regular}%
9810 {%
9811 \glissetattribute{\the\glslabeltok}{regular}{false}%
9812 }%
9813 {}%
9814 }%
9815 }%
9816 {%
9817 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9818 }

```

onghyphennoshort `\glxtrlonghyphennoshort{<label>}{<long>}{<insert>}`

```

9819 \newcommand*{\glxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```

9820 {%

```

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.

```

9821 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
9822 \glsfirstlonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
9823 \ifglxtrininsertinside\else{#3}\fi
9824 }%
9825 }

```

hort-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.

```

9826 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
9827 {%
9828 \renewcommand*{\CustomAbbreviationFields}{%

```

```

9829   name={\protect\protect\glslonghyphenfont{\the\glslongtok}},
9830   sort={\expandonce\glsxtrorglong},
9831   first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9832   firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9833   plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
9834 }%

```

Unset the regular attribute if it has been set.

```

9835 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9836   \glsattribute{\the\glslabeltok}{regular}%
9837   {%
9838     \glssetattribute{\the\glslabeltok}{regular}{false}%
9839   }%
9840 }%
9841 }%
9842 }%
9843 {%
9844   \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9845 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
9846 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
9847 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
9848 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9849 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9850 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9851   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9852 }%
9853 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9854   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9855 }%
9856 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9857   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9858 }%
9859 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9860   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9861 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9862 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9863   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9864   \glsxtrfullsep{##1}%
9865   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9866 }%
9867 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9868   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9869   \glsxtrfullsep{##1}%
9870   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
9871 }%

```

```

9872 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9873   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9874   \glsxtrfullsep{##1}%
9875   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
9876 }%
9877 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9878   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9879   \glsxtrfullsep{##1}%
9880   \glsxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
9881 }%

```

The first use full form only displays the long form.

```

9882 \renewcommand*{\glsxtrfullformat}[2]{%
9883   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9884 }%
9885 \renewcommand*{\glsxtrfullplformat}[2]{%
9886   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9887 }%
9888 \renewcommand*{\Glsxtrfullformat}[2]{%
9889   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9890 }%
9891 \renewcommand*{\Glsxtrfullplformat}[2]{%
9892   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9893 }%
9894 }

```

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.

```

9895 \newabbreviationstyle{long-hyphen-noshort-noreg}%
9896 {%
9897   \renewcommand*{\CustomAbbreviationFields}{%
9898     name={\protect\glsabbrvfont{\the\glsshorttok}},
9899     sort={\the\glsshorttok},
9900     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9901     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9902     text={\protect\glslonghyphenfont{\the\glslongtok}},%
9903     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
9904     description={\the\glslongtok}%
9905   }%

```

Unset the regular attribute if it has been set.

```

9906 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9907   \glshasattribute{\the\glslabeltok}{regular}%
9908   {%
9909     \glissetattribute{\the\glslabeltok}{regular}{false}%
9910   }%
9911   {}%
9912 }%
9913 }%

```

```

9914 {%
9915   \GlsXtrUseAbbrStyleFmts{long-desc}%
9916 }

```

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.

```

9917 \newcommand*{\glsxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

9918 {%
9919   \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
9920   \glsfirstlonghyphenfont{#1}%
9921 }%
9922 }

```

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.

```

9923 \newcommand*{\glsxtrposthyphenshort}[2]{%
9924 {%
9925   \glsxtrifhyphenstart{#2}{\def\glsxtrwordsep{-}}{}%
9926   \ifglsxtrininsertinside{\glsfirstlonghyphenfont{#2}}\else{#2}\fi
9927   \glsxtrfullsep{#1}%
9928   \glsxtrparen
9929   {\glsfirstabbrvhyphenfont{\glsentryshort{#1}}\ifglsxtrininsertinside{#2}\fi}%
9930   \ifglsxtrininsertinside\else{#2}\fi
9931 }%
9932 }%
9933 }

```

hyphensubsequent `\glsxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```

9934 \newcommand*{\glsxtrposthyphensubsequent}[2]{%
9935   \glsabbrvfont{\ifglsxtrininsertinside {#2}\fi}%
9936   \ifglsxtrininsertinside \else{#2}\fi
9937 }

```

ostshort-hyphen Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

9938 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
9939 {%
9940   \renewcommand*{\CustomAbbreviationFields}{%
9941     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
9942     sort={\the\glsshorttok},
9943     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
9944     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
9945     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9946     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
9947   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9948     \csdef{glsxtrpostlink\glscategorylabel}{%
9949       \glsxtrifwasfirstuse
9950       {%
9951         \glsxtrposthyphenshort{\glslabel}{\glsinsert}%
9952       }%
9953     }%

```

Put the insertion into the post-link:

```

9954       \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
9955     }%
9956   }%
9957   \glsattribute{\the\glslabeltok}{regular}%
9958   {%
9959     \glssetAttribute{\the\glslabeltok}{regular}{false}%
9960   }%
9961   {}%
9962 }%
9963 }%
9964 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9965 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
9966 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
9967 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
9968 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
9969 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

9970 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9971   \glsabbrvfont{\glsaccessshort{##1}}%
9972 }%
9973 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9974   \glsabbrvfont{\glsaccessshortpl{##1}}%
9975 }%
9976 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9977   \glsabbrvfont{\Glsaccessshort{##1}}%
9978 }%
9979 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%

```

```

9980 \glsabbrvfont{\Glsaccessshortpl{##1}}%
9981 }%
    First use full form:
9982 \renewcommand*{\glsxtrfullformat}[2]{%
9983 \glsxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
9984 }%
9985 \renewcommand*{\glsxtrfullplformat}[2]{%
9986 \glsxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
9987 }%
9988 \renewcommand*{\Glsxtrfullformat}[2]{%
9989 \glsxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
9990 }%
9991 \renewcommand*{\Glsxtrfullplformat}[2]{%
9992 \glsxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
9993 }%

```

In-line format.

```

9994 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9995 \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
9996 \ifglsxtrininsertinside{##2}\fi}%
9997 \ifglsxtrininsertinside \else{##2}\fi
9998 }%
9999 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10000 \glsfirstlonghyphenfont{\glsaccesslongpl{##1}}%
10001 \ifglsxtrininsertinside{##2}\fi}%
10002 \ifglsxtrininsertinside \else{##2}\fi
10003 }%
10004 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10005 \glsfirstlonghyphenfont{\Glsaccesslong{##1}}%
10006 \ifglsxtrininsertinside{##2}\fi}%
10007 \ifglsxtrininsertinside \else{##2}\fi
10008 }%
10009 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10010 \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}}%
10011 \ifglsxtrininsertinside{##2}\fi}%
10012 \ifglsxtrininsertinside \else{##2}\fi
10013 }%
10014 }

```

`ort-hyphen-desc` Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10015 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10016 {%
10017 \renewcommand*{\CustomAbbreviationFields}{%
10018 name={\glsxtrlongshortdescname},
10019 sort={\glsxtrlongshortdescsort},%
10020 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10021 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10022 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10023 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%

```

```

10024 }%
10025 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10026   \csdef{glstrpostlink\glscategorylabel}{%
10027     \glstrifwasfirstuse
10028     {%
10029       \glstrposthyphenshort{\glslabel}{\glinsert}%
10030     }%
10031     {%

```

Put the insertion into the post-link:

```

10032       \glstrposthyphensubsequent{\glslabel}{\glinsert}%
10033     }%
10034   }%
10035   \glshasattribute{\the\glslabeltok}{regular}%
10036   {%
10037     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10038   }%
10039   {}%
10040 }%
10041 }%
10042 {%
10043   \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10044 }

```

rshorthyphenlong

```
\glstrshorthyphenlong{<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.

```
10045 \newcommand*{\glstrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10046 {%
```

If *<insert>* starts with a hyphen, redefine `\glstrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.)

```

10047   \glstrifhyphenstart{#4}{\def\glstrwordsep{-}}{%
10048   \glsfirstabbrvhyphenfont{#2\ifglstrinsertinside{#4}\fi}%
10049   \ifglstrinsertinside\else{#4}\fi
10050   \glstrfullsep{#1}%
10051   \glstrparen{\glsfirstlonghyphenfont{#3\ifglstrinsertinside{#4}\fi}%
10052   \ifglstrinsertinside\else{#4}\fi}%
10053 }%
10054 }

```

hen-long-hyphen Designed for use with the markwords attribute.

```
10055 \newabbreviationstyle{short-hyphen-long-hyphen}%
```



```

10056 {%
10057   \renewcommand*{\CustomAbbreviationFields}{%
10058     name={\protect\glsabbrvhyphenfont{\the\glsshorttok}},
10059     sort={\the\glsshorttok},
10060     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10061       \protect\glsxtrfullsep{\the\glslabeltok}%
10062       \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10063     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10064       \protect\glsxtrfullsep{\the\glslabeltok}%
10065       \glsxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10066     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
10067     description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

10068   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10069     \glsasattribute{\the\glslabeltok}{regular}%
10070     {%
10071       \glssetattribute{\the\glslabeltok}{regular}{false}%
10072     }%
10073   }%
10074 }%
10075 }%
10076 {%
10077   \renewcommand*{\abbrvpluralsuffix}{\glsxtrhyphensuffix}%
10078   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10079   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10080   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10081   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10082   \renewcommand*{\glsxtrfullformat}[2]{%
10083     \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10084   }%
10085   \renewcommand*{\glsxtrfullplformat}[2]{%
10086     \glsxtrshorthyphenlong{##1}%
10087     {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10088   }%
10089   \renewcommand*{\Glsxtrfullformat}[2]{%
10090     \glsxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10091   }%
10092   \renewcommand*{\Glsxtrfullplformat}[2]{%
10093     \glsxtrshorthyphenlong{##1}%
10094     {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10095   }%
10096 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10097 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10098 {%
10099   \renewcommand*{\CustomAbbreviationFields}{%

```

```

10100   name={\glxtrshortlongdescname},
10101   sort={\glxtrshortlongdescsort},
10102   first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10103     \protect\glxtrfullsep{\the\glslabeltok}%
10104     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10105   firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10106     \protect\glxtrfullsep{\the\glslabeltok}%
10107     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10108   text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10109   plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
10110 }%

```

Unset the regular attribute if it has been set.

```

10111 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10112   \glshasattribute{\the\glslabeltok}{regular}%
10113   {%
10114     \glissetattribute{\the\glslabeltok}{regular}{false}%
10115   }%
10116   {}%
10117 }%
10118 }%
10119 {%
10120   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10121 }

```

lsxtrshorthyphen

```
\glxtrshorthyphen{<short>}{<label>}{<insert>}
```

Used by short-hyphen-postlong-hyphen. The *<insert>* is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```
10122 \newcommand*{\glxtrshorthyphen}[3]{%
```

Grouping is needed to localise the redefinitions.

```

10123   {%
10124     \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}}%
10125     \glsfirstabbrvhyphenfont{#1}%
10126   }%
10127 }

```

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.

```

10128 \newcommand*{\glxtrposthyphenlong}[2]{%
10129   {%

```

```

10130 \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}%
10131 \ifglxtrininsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10132 \glxtrfullsep{#1}%
10133 \glxtrparen
10134 {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglxtrininsertinside{#2}\fi}%
10135 \ifglxtrininsertinside\else{#2}\fi
10136 }%
10137 }%
10138 }

```

postlong-hyphen Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10139 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10140 {%
10141 \renewcommand*{\CustomAbbreviationFields}{%
10142 name={\protect\glssabbrvhyphenfont{\the\glssshorttok}},
10143 sort={\the\glssshorttok},
10144 first={\protect\glsfirstabbrvhyphenfont{\the\glssshorttok}},%
10145 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glssshortpltok}},%
10146 plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}},%
10147 description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10148 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10149 \csdef{glxtrpostlink\glscategorylabel}{%
10150 \glxtrifwasfirstuse
10151 {%
10152 \glxtrposthyphenlong{\glslabel}{\glsinsert}%
10153 }%
10154 }%

```

Put the insertion into the post-link:

```

10155 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10156 }%
10157 }%
10158 \glshasattribute{\the\glslabeltok}{regular}%
10159 {%
10160 \glissetattribute{\the\glslabeltok}{regular}{false}%
10161 }%
10162 {}%
10163 }%
10164 }%
10165 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10166 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
10167 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvhyphenfont{##1}}%
10168 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10169 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10170 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10171 \renewcommand*{\glxtrsubsequentfmt}[2]{%
10172   \glsabbrvfont{\glsaccessshort{##1}}}%
10173 }%
10174 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
10175   \glsabbrvfont{\glsaccessshortpl{##1}}}%
10176 }%
10177 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10178   \glsabbrvfont{\Glsaccessshort{##1}}}%
10179 }%
10180 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10181   \glsabbrvfont{\Glsaccessshortpl{##1}}}%
10182 }%

```

First use full form:

```

10183 \renewcommand*{\glxtrfullformat}[2]{%
10184   \glxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10185 }%
10186 \renewcommand*{\glxtrfullplformat}[2]{%
10187   \glxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10188 }%
10189 \renewcommand*{\Glsxtrfullformat}[2]{%
10190   \glxtrshorthyphen{\Glsaccessshort{##1}}{##1}{##2}%
10191 }%
10192 \renewcommand*{\Glsxtrfullplformat}[2]{%
10193   \glxtrshorthyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10194 }%

```

In-line format. Commands like `\glxtrfull` set `\glsinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10195 \renewcommand*{\glxtrinlinefullformat}[2]{%
10196   \glsfirstabbrvhyphenfont{\glsaccessshort{##1}}%
10197   \ifglxtrininsertinside{##2}\fi}%
10198   \ifglxtrininsertinside \else{##2}\fi
10199 }%
10200 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10201   \glsfirstabbrvhyphenfont{\glsaccessshortpl{##1}}%
10202   \ifglxtrininsertinside{##2}\fi}%
10203   \ifglxtrininsertinside \else{##2}\fi
10204 }%
10205 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10206   \glsfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10207   \ifglxtrininsertinside{##2}\fi}%
10208   \ifglxtrininsertinside \else{##2}\fi
10209 }%
10210 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10211   \glsfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10212   \ifglxtrininsertinside{##2}\fi}%
10213   \ifglxtrininsertinside \else{##2}\fi
10214 }%
10215 }

```

ong-hyphen-desc Like short-hyphen-postlong-hyphen but the description must be supplied by the user.

```
10216 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10217 {%
10218   \renewcommand*{\CustomAbbreviationFields}{%
10219     name={\glstrshortlongdescname},
10220     sort={\glstrshortlongdescsort},%
10221     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10222     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10223     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10224     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10225 }%
10226 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10227   \csdef{glstrpostlink\glscategorylabel}{%
10228     \glstrifwasfirstuse
10229     {%
10230       \glstrposthyphenlong{\glslabel}{\glsinsert}%
10231     }%
10232   }%
```

Put the insertion into the post-link:

```
10233     \glstrposthyphensubsequent{\glslabel}{\glsinsert}%
10234   }%
10235 }%
10236 \glshasattribute{\the\glslabeltok}{regular}%
10237 {%
10238   \glissetattribute{\the\glslabeltok}{regular}{false}%
10239 }%
10240 {}%
10241 }%
10242 }%
10243 {%
10244   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10245 }
```

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

```
10246 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%
```

stabbrvonlyfont

```
10247 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%
```

glslongonlyfont

```
10248 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
10249 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

lsxtronlysuffix

```
10250 \newcommand*{\glxtronlysuffix}{\glxtrabbrvpluralsuffix}
```

only-short-only

```
10251 \newabbreviationstyle{long-only-short-only}%
10252 {%
10253   \renewcommand*{\CustomAbbreviationFields}{%
10254     name={\protect\glsabbrvonlyfont{\the\glsshorttok}},
10255     sort={\the\glsshorttok},
10256     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10257     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10258     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10259     description={\protect\glslongonlyfont{\the\glslongtok}}}%
```

Unset the regular attribute if it has been set.

```
10260 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10261   \glshasattribute{\the\glslabeltok}{regular}%
10262   {%
10263     \glissetattribute{\the\glslabeltok}{regular}{false}%
10264   }%
10265   {}%
10266 }%
10267 }%
10268 {%
10269   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtronlysuffix}%
10270   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
10271   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10272   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10273   \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
10274 \renewcommand*{\glxtrfullformat}[2]{%
10275   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10276   \ifglxtrininsertinside\else##2\fi
10277 }%
10278 \renewcommand*{\glxtrfullplformat}[2]{%
10279   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10280   \ifglxtrininsertinside\else##2\fi
10281 }%
10282 \renewcommand*{\Glsxtrfullformat}[2]{%
10283   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
10284   \ifglxtrininsertinside\else##2\fi
10285 }%
10286 \renewcommand*{\Glsxtrfullplformat}[2]{%
10287   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
10288   \ifglxtrininsertinside\else##2\fi
10289 }%
```

The inline full form does show the short form.

```

10290 \renewcommand*{\glxtrinlinefullformat}[2]{%
10291   \glsfirstlongonlyfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10292   \ifglxtrinsertinside\else##2\fi
10293   \glxtrfullsep{##1}%
10294   \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshort{##1}}}%
10295 }%
10296 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10297   \glsfirstlongonlyfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10298   \ifglxtrinsertinside\else##2\fi
10299   \glxtrfullsep{##1}%
10300   \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10301 }%
10302 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10303   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
10304   \ifglxtrinsertinside\else##2\fi
10305   \glxtrfullsep{##1}%
10306   \glxtrparen{\protect\glsfirstabbrvonlyfont{\glsaccessshortpl{##1}}}%
10307 }%
10308 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10309   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10310   \ifglxtrinsertinside\else##2\fi
10311   \glxtrfullsep{##1}%
10312   \glxtrparen{\protect\glsfirstabbrvonlyfont{\Glsaccessshortpl{##1}}}%
10313 }%
10314 }

```

xtronlydescsort

```

10315 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

10316 \newcommand*{\glxtronlydescname}{%
10317   \protect\glslongfont{\the\glslongtok}}%
10318 }

```

short-only-desc

```

10319 \newabbreviationstyle{long-only-short-only-desc}%
10320 {%
10321   \renewcommand*{\CustomAbbreviationFields}{%
10322     name={\glxtronlydescname},
10323     sort={\glxtronlydescsort},%
10324     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10325     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10326     text={\protect\glsabbrvonlyfont{\the\glsshorttok}},%
10327     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}}}%
10328 }%

```

Unset the regular attribute if it has been set.

```

10329 \renewcommand*{\GlsXtrPostNewAbbreviation}{%

```

```

10330 \glshasattribute{\the\glslabeltok}{regular}%
10331 {%
10332 \glissetattribute{\the\glslabeltok}{regular}{false}%
10333 }%
10334 {}%
10335 }%
10336 }%
10337 {%
10338 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10339 }

```

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 `\glsabrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The \TeX string case can now use `\glstrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to false, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that glossaries automatically loads `textcase`, so the label can be protected from case change with `textcase`'s `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
10340 \let\@glstr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```

10341 \renewcommand*{\markright}[1]{%
10342 \glstrmarkhook
10343 \@glstr@org@markright{\@glstrinmark#1\@glstrnotinmark}%
10344 \glstrrestoremarkhook
10345 }

```

`\markboth` Save original definition:


```
10346 \let\@glxstr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
10347 \renewcommand*{\markboth}[2]{%
10348   \glxstrmarkhook
10349   \@glxstr@org@markboth
10350   {\@glxstrinmark#1\@glxstrnotinmark}%
10351   {\@glxstrinmark#2\@glxstrnotinmark}%
10352   \glxstrrestoremarkhook
10353 }
```

Also do this for \@starttoc

\@starttoc Save original definition:

```
10354 \let\@glxstr@org@@starttoc\@starttoc
```

Redefine:

```
10355 \renewcommand*{\@starttoc}[1]{%
10356   \glxstrmarkhook
10357   \@glxstrinmark
10358   \@glxstr@org@@starttoc{#1}%
10359   \@glxstrnotinmark
10360   \glxstrrestoremarkhook
10361 }
```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```
10362 \newcommand*{\glxstrRevertMarks}{%
10363   \let\markright\@glxstr@org@markright
10364   \let\markboth\@glxstr@org@markboth
10365   \let\@starttoc\@glxstr@org@@starttoc
10366 }
```

\glxstrifinmark

```
10367 \newcommand*{\glxstrifinmark}[2]{#2}
```

\@glxstrinmark

```
10368 \newrobustcmd*{\@glxstrinmark}{%
10369   \let\glxstrifinmark\@firstoftwo
10370 }
```

glxstrnotinmark

```
10371 \newrobustcmd*{\@glxstrnotinmark}{%
10372   \let\glxstrifinmark\@secondoftwo
10373 }
```

eorpdforheading

```
10374 \ifdef\texorpdfstring
10375 {
```

```

10376 \newcommand*{\glxstrtitleorpdforheading}[3]{\texorpdfstring{#1}{#2}}
10377 }
10378 {
10379 \newcommand*{\glxstrtitleorpdforheading}[3]{#1}
10380 }

```

\glxstrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

10381 \newcommand*{\glxstrmarkhook}{%

```

Save current definitions:

```

10382 \let\@glxstr@org@MakeUppercase\MakeUppercase
10383 \let\@glxstr@org@glxstrtitleorpdforheading\glxstrtitleorpdforheading
10384 \let\@glxstr@org@glxstrtitleshort\glxstrtitleshort
10385 \let\@glxstr@org@glxstrtitleshortpl\glxstrtitleshortpl
10386 \let\@glxstr@org@Glsxstrtitleshort\Glsxstrtitleshort
10387 \let\@glxstr@org@Glsxstrtitleshortpl\Glsxstrtitleshortpl
10388 \let\@glxstr@org@glxstrtitlename\glxstrtitlename
10389 \let\@glxstr@org@Glsxstrtitlename\Glsxstrtitlename
10390 \let\@glxstr@org@glxstrtitletext\glxstrtitletext
10391 \let\@glxstr@org@Glsxstrtitletext\Glsxstrtitletext
10392 \let\@glxstr@org@glxstrtitleplural\glxstrtitleplural
10393 \let\@glxstr@org@Glsxstrtitleplural\Glsxstrtitleplural
10394 \let\@glxstr@org@glxstrtitlefirst\glxstrtitlefirst
10395 \let\@glxstr@org@Glsxstrtitlefirst\Glsxstrtitlefirst
10396 \let\@glxstr@org@glxstrtitlefirstplural\glxstrtitlefirstplural
10397 \let\@glxstr@org@Glsxstrtitlefirstplural\Glsxstrtitlefirstplural
10398 \let\@glxstr@org@glxstrtitlelong\glxstrtitlelong
10399 \let\@glxstr@org@glxstrtitlelongpl\glxstrtitlelongpl
10400 \let\@glxstr@org@Glsxstrtitlelong\Glsxstrtitlelong
10401 \let\@glxstr@org@Glsxstrtitlelongpl\Glsxstrtitlelongpl
10402 \let\@glxstr@org@glxstrtitlefull\glxstrtitlefull
10403 \let\@glxstr@org@glxstrtitlefullpl\glxstrtitlefullpl
10404 \let\@glxstr@org@Glsxstrtitlefull\Glsxstrtitlefull
10405 \let\@glxstr@org@Glsxstrtitlefullpl\Glsxstrtitlefullpl

```

New definitions

```

10406 \let\glxstrifinmark\@firstoftwo
10407 \let\MakeUppercase\MakeTextUppercase
10408 \let\glxstrtitleorpdforheading\@thirdofthree
10409 \let\glxstrtitleshort\glxstrheadshort
10410 \let\glxstrtitleshortpl\glxstrheadshortpl
10411 \let\Glsxstrtitleshort\Glsxstrheadshort
10412 \let\Glsxstrtitleshortpl\Glsxstrheadshortpl
10413 \let\glxstrtitlename\glxstrheadname
10414 \let\Glsxstrtitlename\Glsxstrheadname
10415 \let\glxstrtitletext\glxstrheadtext
10416 \let\Glsxstrtitletext\Glsxstrheadtext
10417 \let\glxstrtitleplural\glxstrheadplural
10418 \let\Glsxstrtitleplural\Glsxstrheadplural

```

```

10419 \let\glsxtrtitlefirst\glsxtrheadfirst
10420 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10421 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
10422 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10423 \let\glsxtrtitlelong\glsxtrheadlong
10424 \let\glsxtrtitlelongpl\glsxtrheadlongpl
10425 \let\Glsxtrtitlelong\Glsxtrheadlong
10426 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10427 \let\glsxtrtitlefull\glsxtrheadfull
10428 \let\glsxtrtitlefullpl\glsxtrheadfullpl
10429 \let\Glsxtrtitlefull\Glsxtrheadfull
10430 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10431 }

```

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.)

```

10432 \newcommand*{\glsxtrrestoremarkhook}{%
10433 \let\glsxtrifinmark\@secondoftwo
10434 \let\MakeUppercase\@glsxtr@org@MakeUppercase
10435 \let\glsxtrtitleorpdforheading\@glsxtr@org@glsxtrtitleorpdforheading
10436 \let\glsxtrtitleshort\@glsxtr@org@glsxtrtitleshort
10437 \let\glsxtrtitleshortpl\@glsxtr@org@glsxtrtitleshortpl
10438 \let\Glsxtrtitleshort\@glsxtr@org@Glsxtrtitleshort
10439 \let\Glsxtrtitleshortpl\@glsxtr@org@Glsxtrtitleshortpl
10440 \let\glsxtrtitlename\@glsxtr@org@glsxtrtitlename
10441 \let\Glsxtrtitlename\@glsxtr@org@Glsxtrtitlename
10442 \let\glsxtrtitletext\@glsxtr@org@glsxtrtitletext
10443 \let\Glsxtrtitletext\@glsxtr@org@Glsxtrtitletext
10444 \let\glsxtrtitleplural\@glsxtr@org@glsxtrtitleplural
10445 \let\Glsxtrtitleplural\@glsxtr@org@Glsxtrtitleplural
10446 \let\glsxtrtitlefirst\@glsxtr@org@glsxtrtitlefirst
10447 \let\Glsxtrtitlefirst\@glsxtr@org@Glsxtrtitlefirst
10448 \let\glsxtrtitlefirstplural\@glsxtr@org@glsxtrtitlefirstplural
10449 \let\Glsxtrtitlefirstplural\@glsxtr@org@Glsxtrtitlefirstplural
10450 \let\glsxtrtitlelong\@glsxtr@org@glsxtrtitlelong
10451 \let\glsxtrtitlelongpl\@glsxtr@org@glsxtrtitlelongpl
10452 \let\Glsxtrtitlelong\@glsxtr@org@Glsxtrtitlelong
10453 \let\Glsxtrtitlelongpl\@glsxtr@org@Glsxtrtitlelongpl
10454 \let\glsxtrtitlefull\@glsxtr@org@glsxtrtitlefull
10455 \let\glsxtrtitlefullpl\@glsxtr@org@glsxtrtitlefullpl
10456 \let\Glsxtrtitlefull\@glsxtr@org@Glsxtrtitlefull
10457 \let\Glsxtrtitlefullpl\@glsxtr@org@Glsxtrtitlefullpl
10458 }

```

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.

```

10459 \newcommand*{\glsxtrheadshort}[1]{%
10460   \protect\NoCaseChange
10461   {%
10462     \glsifattribute{#1}{headuc}{true}%
10463     {%
10464       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10465     }%
10466     {%
10467       \glsxtrshort[noindex,hyper=false]{#1}[]%
10468     }%
10469   }%
10470 }
```

`lsxtrtitleshort` Command to display short form of abbreviation in section title and table of contents.

```

10471 \newrobustcmd*{\glsxtrtitleshort}[1]{%
10472   \glsxtrshort[noindex,hyper=false]{#1}[]%
10473 }
```

`sxtrheadshortpl` Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10474 \newcommand*{\glsxtrheadshortpl}[1]{%
10475   \protect\NoCaseChange
10476   {%
10477     \glsifattribute{#1}{headuc}{true}%
10478     {%
10479       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10480     }%
10481     {%
10482       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10483     }%
10484   }%
10485 }
```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents.

```

10486 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10487   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10488 }
```

`Glsxtrheadshort` Command used to display short form in the page header with the first letter converted to upper case.

```

10489 \newcommand*{\Glsxtrheadshort}[1]{%
10490   \protect\NoCaseChange
10491   {%
10492     \glsifattribute{#1}{headuc}{true}%
10493     {%
10494       \GLSxtrshort[noindex,hyper=false]{#1}[]%

```

```

10495 }%
10496 {%
10497 \Glsxtrshort[noindex,hyper=false]{#1}[]%
10498 }%
10499 }%
10500 }

```

lsxtrtitleshort Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10501 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10502 \Glsxtrshort[noindex,hyper=false]{#1}[]%
10503 }

```

sxtrheadshortpl Command used to display plural short form in the page header with the first letter converted to upper case.

```

10504 \newcommand*{\Glsxtrheadshortpl}[1]{%
10505 \protect\NoCaseChange
10506 {%
10507 \glsifattribute{#1}{headuc}{true}%
10508 {%
10509 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10510 }%
10511 {%
10512 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10513 }%
10514 }%
10515 }

```

xtrtitleshortpl Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10516 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10517 \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10518 }

```

\glsxtrheadname As above but for the name value.

```

10519 \newcommand*{\glsxtrheadname}[1]{%
10520 \protect\NoCaseChange
10521 {%
10522 \glsifattribute{#1}{headuc}{true}%
10523 {%
10524 \GlsSname[noindex,hyper=false]{#1}[]%
10525 }%
10526 {%
10527 \glsname[noindex,hyper=false]{#1}[]%
10528 }%
10529 }%
10530 }

```

`glsxtrtitlename` Command to display name value in section title and table of contents.

```
10531 \newrobustcmd*{\glsxtrtitlename}[1]{%
10532   \glsname[noindex,hyper=false]{#1}[]%
10533 }
```

`\Glsxtrheadname` First letter converted to upper case

```
10534 \newcommand*{\Glsxtrheadname}[1]{%
10535   \protect\NoCaseChange
10536   {%
10537     \glsifattribute{#1}{headuc}{true}%
10538     {%
10539       \GLSname[noindex,hyper=false]{#1}[]%
10540     }%
10541     {%
10542       \Glsname[noindex,hyper=false]{#1}[]%
10543     }%
10544   }%
10545 }
```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
10546 %\changes{1.21}{2017-11-03}{new}
10547 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10548   \Glsname[noindex,hyper=false]{#1}[]%
10549 }
```

`\glsxtrheadtext` As above but for the text value.

```
10550 \newcommand*{\glsxtrheadtext}[1]{%
10551   \protect\NoCaseChange
10552   {%
10553     \glsifattribute{#1}{headuc}{true}%
10554     {%
10555       \GLStext[noindex,hyper=false]{#1}[]%
10556     }%
10557     {%
10558       \glstext[noindex,hyper=false]{#1}[]%
10559     }%
10560   }%
10561 }
```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```
10562 \newrobustcmd*{\glsxtrtitletext}[1]{%
10563   \glstext[noindex,hyper=false]{#1}[]%
10564 }
```

`\Glsxtrheadtext` First letter converted to upper case

```
10565 \newcommand*{\Glsxtrheadtext}[1]{%
10566   \protect\NoCaseChange
```

```

10567 {%
10568   \glsifattribute{#1}{headuc}{true}%
10569   {%
10570     \GLStext[noindex,hyper=false]{#1}[]%
10571   }%
10572   {%
10573     \GLStext[noindex,hyper=false]{#1}[]%
10574   }%
10575 }%
10576 }

```

Glsxtrtitletext Command to display text value in section title and table of contents with the first letter changed to upper case.

```

10577 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10578   \GLStext[noindex,hyper=false]{#1}[]%
10579 }

```

lsxtrheadplural As above but for the plural value.

```

10580 \newcommand*{\lsxtrheadplural}[1]{%
10581   \protect\NoCaseChange
10582   {%
10583     \glsifattribute{#1}{headuc}{true}%
10584     {%
10585       \GLSplural[noindex,hyper=false]{#1}[]%
10586     }%
10587     {%
10588       \glsplural[noindex,hyper=false]{#1}[]%
10589     }%
10590   }%
10591 }

```

sxtrtitleplural Command to display plural value in section title and table of contents.

```

10592 \newrobustcmd*{\gsxtrtitleplural}[1]{%
10593   \glsplural[noindex,hyper=false]{#1}[]%
10594 }

```

lsxtrheadplural Convert first letter to upper case.

```

10595 \newcommand*{\Glsxtrheadplural}[1]{%
10596   \protect\NoCaseChange
10597   {%
10598     \glsifattribute{#1}{headuc}{true}%
10599     {%
10600       \GLSplural[noindex,hyper=false]{#1}[]%
10601     }%
10602     {%
10603       \Glsplural[noindex,hyper=false]{#1}[]%
10604     }%
10605   }%
10606 }

```

`lsxtrtitleplural` Command to display plural value in section title and table of contents with the first letter changed to upper case.

```
10607 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10608   \Glsplural[noindex,hyper=false]{#1}[]%
10609 }
```

`glsxtrheadfirst` As above but for the first value.

```
10610 \newcommand*{\glsxtrheadfirst}[1]{%
10611   \protect\NoCaseChange
10612   {%
10613     \glsifattribute{#1}{headuc}{true}%
10614     {%
10615       \GLSfirst[noindex,hyper=false]{#1}[]%
10616     }%
10617     {%
10618       \glsfirst[noindex,hyper=false]{#1}[]%
10619     }%
10620   }%
10621 }
```

`lsxtrtitlefirst` Command to display first value in section title and table of contents.

```
10622 \newrobustcmd*{\glsxtrtitlefirst}[1]{%
10623   \glsfirst[noindex,hyper=false]{#1}[]%
10624 }
```

`Glsxtrheadfirst` First letter converted to upper case

```
10625 \newcommand*{\Glsxtrheadfirst}[1]{%
10626   \protect\NoCaseChange
10627   {%
10628     \glsifattribute{#1}{headuc}{true}%
10629     {%
10630       \GLSfirst[noindex,hyper=false]{#1}[]%
10631     }%
10632     {%
10633       \Glsfirst[noindex,hyper=false]{#1}[]%
10634     }%
10635   }%
10636 }
```

`lsxtrtitlefirst` Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10637 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10638   \Glsfirst[noindex,hyper=false]{#1}[]%
10639 }
```

`headfirstplural` As above but for the firstplural value.

```
10640 \newcommand*{\glsxtrheadfirstplural}[1]{%
10641   \protect\NoCaseChange
```



```

10642 {%
10643   \glsifattribute{#1}{headuc}{true}%
10644   {%
10645     \GLSfirstplural[noindex,hyper=false]{#1}[]%
10646   }%
10647   {%
10648     \glsfirstplural[noindex,hyper=false]{#1}[]%
10649   }%
10650 }%
10651 }

```

`\titlefirstplural` Command to display firstplural value in section title and table of contents.

```

10652 \newrobustcmd*{\glsxtrtitlefirstplural}[1]{%
10653   \glsfirstplural[noindex,hyper=false]{#1}[]%
10654 }

```

`\headfirstplural` First letter converted to upper case

```

10655 \newcommand*{\Glsxtrheadfirstplural}[1]{%
10656   \protect\NoCaseChange
10657   {%
10658     \glsifattribute{#1}{headuc}{true}%
10659     {%
10660       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10661     }%
10662     {%
10663       \Glsfirstplural[noindex,hyper=false]{#1}[]%
10664     }%
10665   }%
10666 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10667 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10668   \Glsfirstplural[noindex,hyper=false]{#1}[]%
10669 }

```

`\glsxtrheadlong` Command used to display long form in the page header.

```

10670 \newcommand*{\glsxtrheadlong}[1]{%
10671   \protect\NoCaseChange
10672   {%
10673     \glsifattribute{#1}{headuc}{true}%
10674     {%
10675       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10676     }%
10677     {%
10678       \glsxtrlong[noindex,hyper=false]{#1}[]%
10679     }%
10680   }%
10681 }

```

`glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```
10682 \newrobustcmd*{\glsxtrtitlelong}[1]{%
10683   \glsxtrlong[noindex,hyper=false]{#1}[]%
10684 }
```

`lslxtrheadlongpl` 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.

```
10685 \newcommand*{\glsxtrheadlongpl}[1]{%
10686   \protect\NoCaseChange
10687   {%
10688     \glsifattribute{#1}{headuc}{true}%
10689     {%
10690       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10691     }%
10692     {%
10693       \glsxtrlongpl[noindex,hyper=false]{#1}[]%
10694     }%
10695   }%
10696 }
```

`sxtrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```
10697 \newrobustcmd*{\glsxtrtitlelongpl}[1]{%
10698   \glsxtrlongpl[noindex,hyper=false]{#1}[]%
10699 }
```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```
10700 \newcommand*{\Glsxtrheadlong}[1]{%
10701   \protect\NoCaseChange
10702   {%
10703     \glsifattribute{#1}{headuc}{true}%
10704     {%
10705       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10706     }%
10707     {%
10708       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10709     }%
10710   }%
10711 }
```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10712 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10713   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10714 }
```

`\lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```
10715 \newcommand*{\Glsxtrheadlongpl}[1]{%
10716   \protect\NoCaseChange
10717   {%
10718     \glsifattribute{#1}{headuc}{true}%
10719     {%
10720       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10721     }%
10722     {%
10723       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10724     }%
10725   }%
10726 }
```

`\sxttrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10727 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10728   \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10729 }
```

`\glsxtrheadfull` Command used to display full form in the page header.

```
10730 \newcommand*{\glsxtrheadfull}[1]{%
10731   \protect\NoCaseChange
10732   {%
10733     \glsifattribute{#1}{headuc}{true}%
10734     {%
10735       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10736     }%
10737     {%
10738       \glsxtrfull[noindex,hyper=false]{#1}[]%
10739     }%
10740   }%
10741 }
```

`\glsxtrtitlefull` Command to display full form of abbreviation in section title and table of contents.

```
10742 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10743   \glsxtrfull[noindex,hyper=false]{#1}[]%
10744 }
```

`\lsxtrheadfullpl` Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```
10745 \newcommand*{\glsxtrheadfullpl}[1]{%
10746   \protect\NoCaseChange
10747   {%
10748     \glsifattribute{#1}{headuc}{true}%
10749     {%
```

```

10750      \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10751    }%
10752    {%
10753      \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10754    }%
10755  }%
10756 }

```

sxtrtitlefullpl Command to display plural full form of abbreviation in section title and table of contents.

```

10757 \newrobustcmd*{\glsxtrtitlefullpl}[1]{%
10758   \glsxtrfullpl[noindex,hyper=false]{#1}[]%
10759 }

```

\Glsxtrheadfull Command used to display full form in the page header with the first letter converted to upper case.

```

10760 \newcommand*{\Glsxtrheadfull}[1]{%
10761   \protect\NoCaseChange
10762   {%
10763     \glsifattribute{#1}{headuc}{true}%
10764     {%
10765       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10766     }%
10767     {%
10768       \Glsxtrfull[noindex,hyper=false]{#1}[]%
10769     }%
10770   }%
10771 }

```

Glsxtrtitlefull Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10772 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10773   \Glsxtrfull[noindex,hyper=false]{#1}[]%
10774 }

```

lgsxtrheadfullpl Command used to display plural full form in the page header with the first letter converted to upper case.

```

10775 \newcommand*{\lgsxtrheadfullpl}[1]{%
10776   \protect\NoCaseChange
10777   {%
10778     \glsifattribute{#1}{headuc}{true}%
10779     {%
10780       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10781     }%
10782     {%
10783       \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10784     }%
10785   }%
10786 }

```

`sxtrtitlefullpl` Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10787 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10788   \Glsxtrfullpl[noindex,hyper=false]{#1}[]%
10789 }
```

`\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.

```
10790 \ifdef\texorpdfstring
10791 {
10792   \newcommand*{\glsfmtshort}[1]{%
10793     \texorpdfstring
10794       {\Glsxtrtitleshort{#1}}%
10795       {\Glsentryshort{#1}}%
10796   }
10797 }
10798 {
10799   \newcommand*{\glsfmtshort}[1]{%
10800     \Glsxtrtitleshort{#1}}
10801 }
```

Similarly for the plural version.

`\glsfmtshortpl`

```
10802 \ifdef\texorpdfstring
10803 {
10804   \newcommand*{\glsfmtshortpl}[1]{%
10805     \texorpdfstring
10806       {\Glsxtrtitleshortpl{#1}}%
10807       {\Glsentryshortpl{#1}}%
10808   }
10809 }
10810 {
10811   \newcommand*{\glsfmtshortpl}[1]{%
10812     \Glsxtrtitleshortpl{#1}}
10813 }
```

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).

```
10814 \ifdef\texorpdfstring
10815 {
10816   \newcommand*{\Glsfmtshort}[1]{%
10817     \texorpdfstring
10818       {\Glsxtrtitleshort{#1}}%
10819       {\Glsentryshort{#1}}%
10820   }
10821 }
```

```

10822 {
10823   \newcommand*{\Glsfmtshort}[1]{%
10824     \Glsxtrtitleshort{#1}}
10825 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10826 \ifdef\texorpdfstring
10827 {
10828   \newcommand*{\Glsfmtshortpl}[1]{%
10829     \texorpdfstring
10830     {\Glsxtrtitleshortpl{#1}}%
10831     {\glsentryshortpl{#1}}%
10832   }
10833 }
10834 {
10835   \newcommand*{\Glsfmtshortpl}[1]{%
10836     \Glsxtrtitleshortpl{#1}}
10837 }

```

`\glsfmtname` As above but for the name value.

```

10838 \ifdef\texorpdfstring
10839 {
10840   \newcommand*{\glsfmtname}[1]{%
10841     \texorpdfstring
10842     {\glsxtrtitlename{#1}}%
10843     {\glsentryname{#1}}%
10844   }
10845 }
10846 {
10847   \newcommand*{\glsfmtname}[1]{%
10848     \glsxtrtitlename{#1}}
10849 }

```

`\Glsfmtname` First letter converted to upper case.

```

10850 \ifdef\texorpdfstring
10851 {
10852   \newcommand*{\Glsfmtname}[1]{%
10853     \texorpdfstring
10854     {\Glsxtrtitlename{#1}}%
10855     {\glsentryname{#1}}%
10856   }
10857 }
10858 {
10859   \newcommand*{\Glsfmtname}[1]{%
10860     \Glsxtrtitlename{#1}}
10861 }

```

`\glsfmttext` As above but for the text value.

```

10862 \ifdef\texorpdfstring

```

```

10863 {
10864   \newcommand*{\glsfmttext}[1]{%
10865     \texorpdfstring
10866     {\glsxtrtitletext{#1}}%
10867     {\glsentrytext{#1}}%
10868   }
10869 }
10870 {
10871   \newcommand*{\glsfmttext}[1]{%
10872     \glsxtrtitletext{#1}}
10873 }

```

`\Glsfmttext` First letter converted to upper case.

```

10874 \ifdef\texorpdfstring
10875 {
10876   \newcommand*{\Glsfmttext}[1]{%
10877     \texorpdfstring
10878     {\Glsxtrtitletext{#1}}%
10879     {\glsentrytext{#1}}%
10880   }
10881 }
10882 {
10883   \newcommand*{\Glsfmttext}[1]{%
10884     \Glsxtrtitletext{#1}}
10885 }

```

`\glsfmtplural` As above but for the plural value.

```

10886 \ifdef\texorpdfstring
10887 {
10888   \newcommand*{\glsfmtplural}[1]{%
10889     \texorpdfstring
10890     {\glsxtrtitleplural{#1}}%
10891     {\glsentryplural{#1}}%
10892   }
10893 }
10894 {
10895   \newcommand*{\glsfmtplural}[1]{%
10896     \glsxtrtitleplural{#1}}
10897 }

```

`\Glsfmtplural` First letter converted to upper case.

```

10898 \ifdef\texorpdfstring
10899 {
10900   \newcommand*{\Glsfmtplural}[1]{%
10901     \texorpdfstring
10902     {\Glsxtrtitleplural{#1}}%
10903     {\glsentryplural{#1}}%
10904   }
10905 }

```

```

10906 {
10907   \newcommand*{\Glsfmtplural}[1]{%
10908     \Glsxtrtitleplural{#1}}
10909 }

```

`\glsfmtfirst` As above but for the first value.

```

10910 \ifdef\texorpdfstring
10911 {
10912   \newcommand*{\glsfmtfirst}[1]{%
10913     \texorpdfstring
10914       {\Glsxtrtitlefirst{#1}}%
10915       {\Glsentryfirst{#1}}%
10916   }
10917 }
10918 {
10919   \newcommand*{\glsfmtfirst}[1]{%
10920     \Glsxtrtitlefirst{#1}}
10921 }

```

`\Glsfmtfirst` First letter converted to upper case.

```

10922 \ifdef\texorpdfstring
10923 {
10924   \newcommand*{\Glsfmtfirst}[1]{%
10925     \texorpdfstring
10926       {\Glsxtrtitlefirst{#1}}%
10927       {\Glsentryfirst{#1}}%
10928   }
10929 }
10930 {
10931   \newcommand*{\Glsfmtfirst}[1]{%
10932     \Glsxtrtitlefirst{#1}}
10933 }

```

`\glsfmtfirstpl` As above but for the firstplural value.

```

10934 \ifdef\texorpdfstring
10935 {
10936   \newcommand*{\glsfmtfirstpl}[1]{%
10937     \texorpdfstring
10938       {\Glsxtrtitlefirstplural{#1}}%
10939       {\Glsentryfirstplural{#1}}%
10940   }
10941 }
10942 {
10943   \newcommand*{\glsfmtfirstpl}[1]{%
10944     \Glsxtrtitlefirstplural{#1}}
10945 }

```

`\Glsfmtfirstpl` First letter converted to upper case.

```

10946 \ifdef\texorpdfstring

```



```

10947 {
10948   \newcommand*{\Glsfmtfirstpl}[1]{%
10949     \texorpdfstring
10950     {\Glsxtrtitlefirstplural{#1}}%
10951     {\glsentryfirstplural{#1}}%
10952   }
10953 }
10954 {
10955   \newcommand*{\Glsfmtfirstpl}[1]{%
10956     \Glsxtrtitlefirstplural{#1}}
10957 }

```

`\glsfmtlong` As above but for the long value.

```

10958 \ifdef\texorpdfstring
10959 {
10960   \newcommand*{\glsfmtlong}[1]{%
10961     \texorpdfstring
10962     {\glsxtrtitlelong{#1}}%
10963     {\glsentrylong{#1}}%
10964   }
10965 }
10966 {
10967   \newcommand*{\glsfmtlong}[1]{%
10968     \glsxtrtitlelong{#1}}
10969 }

```

`\Glsfmtlong` First letter converted to upper case.

```

10970 \ifdef\texorpdfstring
10971 {
10972   \newcommand*{\Glsfmtlong}[1]{%
10973     \texorpdfstring
10974     {\Glsxtrtitlelong{#1}}%
10975     {\glsentrylong{#1}}%
10976   }
10977 }
10978 {
10979   \newcommand*{\Glsfmtlong}[1]{%
10980     \Glsxtrtitlelong{#1}}
10981 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

10982 \ifdef\texorpdfstring
10983 {
10984   \newcommand*{\glsfmtlongpl}[1]{%
10985     \texorpdfstring
10986     {\glsxtrtitlelongpl{#1}}%
10987     {\glsentrylongpl{#1}}%
10988   }
10989 }

```

```

10990 {
10991   \newcommand*{\Glsfmtlongpl}[1]{%
10992     \Glsxtrtitlelongpl{#1}}
10993 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

10994 \ifdef\teorpdfstring
10995 {
10996   \newcommand*{\Glsfmtlongpl}[1]{%
10997     \teorpdfstring
10998       {\Glsxtrtitlelongpl{#1}}%
10999     {\Glsentrylongpl{#1}}%
11000   }
11001 }
11002 {
11003   \newcommand*{\Glsfmtlongpl}[1]{%
11004     \Glsxtrtitlelongpl{#1}}
11005 }

```

`\Glsfmtfull` In-line full format.

```

11006 \ifdef\teorpdfstring
11007 {
11008   \newcommand*{\Glsfmtfull}[1]{%
11009     \teorpdfstring
11010       {\Glsxtrtitlefull{#1}}%
11011     {\Glsxtrinlinefullformat{#1}-{}}%
11012   }
11013 }
11014 {
11015   \newcommand*{\Glsfmtfull}[1]{%
11016     \Glsxtrtitlefull{#1}}
11017 }

```

`\Glsfmtfull` First letter converted to upper case.

```

11018 \ifdef\teorpdfstring
11019 {
11020   \newcommand*{\Glsfmtfull}[1]{%
11021     \teorpdfstring
11022       {\Glsxtrtitlefull{#1}}%
11023     {\Glsxtrinlinefullformat{#1}-{}}%
11024   }
11025 }
11026 {
11027   \newcommand*{\Glsfmtfull}[1]{%
11028     \Glsxtrtitlefull{#1}}
11029 }

```

`\Glsfmtfullpl` In-line full plural format.

```

11030 \ifdef\teorpdfstring

```

```

11031 {
11032   \newcommand*{\glsfmtfullpl}[1]{%
11033     \texorpdfstring
11034     {\glsxtrtitlefullpl{#1}}%
11035     {\glsxtrinlinefullplformat{#1}{}}%
11036   }
11037 }
11038 {
11039   \newcommand*{\glsfmtfullpl}[1]{%
11040     \glsxtrtitlefullpl{#1}}
11041 }

```

`\Glsfmtfullpl` First letter converted to upper case.

```

11042 \ifdef\texorpdfstring
11043 {
11044   \newcommand*{\Glsfmtfullpl}[1]{%
11045     \texorpdfstring
11046     {\Glsxtrtitlefullpl{#1}}%
11047     {\Glsxtrinlinefullplformat{#1}{}}%
11048   }
11049 }
11050 {
11051   \newcommand*{\Glsfmtfullpl}[1]{%
11052     \Glsxtrtitlefullpl{#1}}
11053 }

```

1.8 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

11054 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11055   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
11056 }

```

`sariesExtraLang`

```

11057 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11058   \ProvidesFile{glossariesxtr-#1.ldf}%
11059 }

```

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.)

```

11060 \@ifpackageloaded{tracklang}
11061 {%
11062   \AnyTrackedLanguages

```

```

11063 {%
11064   \ForEachTrackedDialect{\this@dialect}{%
11065     \IfTrackedLanguageFileExists{\this@dialect}%
11066     {glossariesxtr-}% prefix
11067     {.ldf}%
11068     {%
11069       \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11070     }%
11071     {%
11072     }%
11073   }%
11074 }%
11075 {}%
11076 }
11077 {}

```

Load glossaries-extra-stylemods if required.

```
11078 \@glsxtr@redefstyles
```

and set the style:

```
11079 \@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:

```
11080 \NeedsTeXFormat{LaTeX2e}
11081 \ProvidesPackage{glossaries-extra-stylemods}[2017/11/08 v1.22 (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`

```
11082 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
11083 \DeclareOption{all}{%
11084   \appto\@glxtr@loadstyles{%
11085     \RequirePackage{glossary-inline}%
11086     \RequirePackage{glossary-list}%
11087     \RequirePackage{glossary-tree}%
11088     \RequirePackage{glossary-mcols}%
11089     \RequirePackage{glossary-long}%
11090     \RequirePackage{glossary-longragged}%
11091     \RequirePackage{glossary-longbooktabs}%
11092     \RequirePackage{glossary-super}%
11093     \RequirePackage{glossary-superragged}%
11094     \RequirePackage{glossary-bookindex}%
11095   }
11096 }

11097 \DeclareOption*{%
11098   \IfFileExists{glossary-\CurrentOption.sty}
11099   {\eappto\@glxtr@loadstyles{%
11100     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
11101   }%
11102   {%
11103     \PackageError{glossaries-extra-styles}%

```

```

11104     {Unknown option ‘\CurrentOption’}{}%
11105   }%
11106 }

```

Process the package options:

```
11107 \ProcessOptions
```

Load the required packages:

```
11108 \@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`.

```
11109 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

11110 \providecommand{\renewglossarystyle}[2]{%
11111   \ifcsundef{@glsstyle@#1}%
11112   {%
11113     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
11114   }%
11115   {%
11116     \csdef{@glsstyle@#1}{#2}%
11117   }%
11118 }

```

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.

```

11119 \ifdef{@glsstyle@listdotted}
11120 {%
11121   \renewglossarystyle{listdotted}{%
11122     \setglossarystyle{list}%
11123     \renewcommand*{\glossentry}[2]{%
11124       \item[]\makebox[\glslistdottedwidth][l]{%
11125         \glsentryitem{##1}%
11126         \glstarget{##1}{\glossentryname{##1}}%
11127         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11128         \glossentrydesc{##1}\glspostdescription}%
11129     \renewcommand*{\subglossentry}[3]{%
11130       \item[]\makebox[\glslistdottedwidth][l]{%
11131         \glssubentryitem{##2}%
11132         \glstarget{##2}{\glossentryname{##2}}%
11133         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
11134         \glossentrydesc{##2}\glspostdescription}%
11135   }

```

```
11136 }
11137 {%
```

Assume the style isn't required if it hasn't already been defined.

```
11138 }
```

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.

```
11139 \ifdef{\@glsstyle@list}
11140 {%
```

listprelocation Space before number list for top-level entries.

```
11141 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
11142 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
11143 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
11144 \renewglossarystyle{list}{%
11145   \renewenvironment{theglossary}%
11146     {\begin{description}}{\end{description}}%
11147   \renewcommand*\{glossaryheader}{}%
11148   \renewcommand*\{glsgroupheading}[1]{}%
11149   \renewcommand*\{glossentry}[2]{%
11150     \item[\glsentryitem{##1}%
11151       \glstarget{##1}{\glossentryname{##1}}]
11152     \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11153   \renewcommand*\{subglossentry}[3]{%
11154     \glssubentryitem{##2}%
11155     \glstarget{##2}{\strut}\space
11156     \glossentrydesc{##2}\glspostdescription
11157     \glslistchildprelocation ##3\glslistchildpostlocation}%
11158   \renewcommand*\{glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11159 }
11160 }
11161 {}
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
11162 \ifdef{\@glsstyle@altlist}
11163 {%
11164   \renewglossarystyle{altlist}{%
11165     \setglossarystyle{list}%
11166     \renewcommand*\{glossentry}[2]{%
11167       \item[\glsentryitem{##1}%
```

```

11168      \glstarget{##1}{\glossentryname{##1}}}%
11169      \mbox{}\par\nobreak\@afterheading
11170      \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
11171      \renewcommand{\subglossentry}[3]{%
11172        \par
11173        \glssubentryitem{##2}%
11174        \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11175        \glslistchildprelocation ##3}%
11176    }
11177 }
11178 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

11179 \ifdef{\@glsstyle@listgroup}
11180 {%
11181   \renewglossarystyle{listgroup}{%
11182     \setglossarystyle{list}%
11183     \renewcommand*\glsgroupheading[1]{%
11184       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11185       \mbox{}\par\nobreak\@afterheading
11186     }%
11187   }
11188 }
11189 {}

```

Similarly for listhypergroup.

```

11190 \ifdef{\@glsstyle@listhypergroup}
11191 {%
11192   \renewglossarystyle{listhypergroup}{%
11193     \setglossarystyle{list}%
11194     \renewcommand*\glossaryheader{%
11195       \glslistnavigationitem{\glsnavigation}}%
11196     \renewcommand*\glsgroupheading[1]{%
11197       \item[\glslistgroupheaderfmt
11198         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
11199       \mbox{}\par\nobreak\@afterheading
11200     }%
11201   }
11202 }
11203 {}

```

Similarly for altlistgroup.

```

11204 \ifdef{\@glsstyle@altlistgroup}
11205 {%
11206   \renewglossarystyle{altlistgroup}{%
11207     \setglossarystyle{altlist}%
11208     \renewcommand*\glsgroupheading[1]{%
11209       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
11210       \mbox{}\par\nobreak\@afterheading
11211     }%
11212   }

```



```

11213 }
11214 {}

    Similarly for altlisthypergroup.
11215 \ifdef{\@glsstyle@altlisthypergroup}
11216 {%
11217   \renewglossarystyle{altlisthypergroup}{%
11218     \setglossarystyle{altlist}%
11219     \renewcommand*\{glossaryheader\}{%
11220       \glslistnavigationitem{\glsnavigation}}%
11221     \renewcommand*\{glsgroupheading}[1]{%
11222       \item[\glslistgroupheaderfmt
11223         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
11224       \mbox{}\par\nobreak\@afterheading
11225     }%
11226   }
11227 }
11228 {}

```

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`.

```

11229 \ifcsdef{@glsstyle@long}
11230 {%
11231   \renewglossarystyle{long}{%
11232     \renewenvironment{theglossary}%
11233       {\begin{longtable}{lp{\glsdescwidth}}}%
11234       {\end{longtable}}%
11235     \renewcommand*\{glossaryheader\}{}%
11236     \renewcommand*\{glsgroupheading}[1]{}%
11237     \renewcommand{\glossentry}[2]{%
11238       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11239       \glossentrydesc{##1}\glspostdescription
11240       \glstrprelocation ##2\tabularnewline
11241     }%
11242     \renewcommand{\subglossentry}[3]{%
11243       &
11244       \glssubentryitem{##2}%
11245       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11246       \glstrprelocation ##3\tabularnewline
11247     }%
11248     \ifglsnogroupskip
11249       \renewcommand*\{glsgroupskip\}{}%
11250     \else
11251       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
11252     \fi
11253   }

```

```
11254 }
11255 {}
```

Three column style:

```
11256 \ifcsdef{@glsstyle@long3col}
11257 {%
11258   \renewglossarystyle{long3col}{%
11259     \renewenvironment{theglossary}%
11260       {\begin{longtable}{lp{@glsdescwidth}p{@glspagelistwidth}}}%
11261       {\end{longtable}}}%
11262     \renewcommand*{\glossaryheader}{}%
11263     \renewcommand*{\glsgroupheading}[1]{}%
11264     \renewcommand{\glossentry}[2]{%
11265       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11266       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11267     }%
11268     \renewcommand{\subglossentry}[3]{%
11269       &
11270       \glssubentryitem{##2}%
11271       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11272       ##3\tabularnewline
11273     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
11274   \ifglsnogroupskip
11275     \renewcommand*{\glsgroupskip}{}%
11276   \else
11277     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11278   \fi
11279 }
11280 }
11281 {}
```

Four column style:

```
11282 \ifcsdef{@glsstyle@long4col}
11283 {%
11284   \renewglossarystyle{long4col}{%
11285     \renewenvironment{theglossary}%
11286       {\begin{longtable}{llll}}%
11287       {\end{longtable}}}%
11288     \renewcommand*{\glossaryheader}{}%
11289     \renewcommand*{\glsgroupheading}[1]{}%
11290     \renewcommand{\glossentry}[2]{%
11291       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11292       \glossentrydesc{##1}\glspostdescription &
11293       \glossentrysymbol{##1} &
11294       ##2\tabularnewline
11295     }%
11296     \renewcommand{\subglossentry}[3]{%
11297       &
```

```

11298     \glssubentryitem{##2}%
11299     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11300     \glossentrysymbol{##2} & ##3\tabularnewline
11301 }%

11302 \ifglsnogroupskip
11303     \renewcommand*{\glsgroupskip}{}%
11304 \else
11305     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11306 \fi
11307 }
11308 }
11309 {}

```

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`.

```

11310 \ifcsdef{@glstyle@longragged}
11311 {%
11312     \renewglossarystyle{longragged}{%
11313         \renewenvironment{theglossary}%
11314             {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}}}%
11315             {\end{longtable}}%
11316         \renewcommand*{\glossaryheader}{}%
11317         \renewcommand*{\glsgroupheading}[1]{}%
11318         \renewcommand{\glossentry}[2]{%
11319             \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11320             \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
11321             \tabularnewline
11322         }%
11323         \renewcommand{\subglossentry}[3]{%
11324             &
11325             \glssubentryitem{##2}%
11326             \glstarget{##2}{\strut}\glossentrydesc{##2}%
11327             \glspostdescription\glxtrprelocation ##3%
11328             \tabularnewline
11329         }%
11330     \ifglsnogroupskip
11331         \renewcommand*{\glsgroupskip}{}%
11332     \else
11333         \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11334     \fi
11335 }
11336 }

```

11337 {}

Three and four column styles don't use \glstrprelocation since the number list is in its own column.

```
11338 \ifcsdef{@glsstyle@longragged3col}
11339 {%
11340   \renewglossarystyle{longragged3col}{%
11341     \renewenvironment{theglossary}%
11342       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
11343         >{\raggedright}p{\glspagelistwidth}}}%
11344       {\end{longtable}}}%
11345     \renewcommand*{\glossaryheader}{}%
11346     \renewcommand*{\glsgroupheading}[1]{}%
11347     \renewcommand{\glossentry}[2]{%
11348       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11349       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11350     }%
11351     \renewcommand{\subglossentry}[3]{%
11352       &
11353       \glssubentryitem{##2}%
11354       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11355       ##3\tabularnewline
11356     }%
11357   \ifglsgroupskip
11358     \renewcommand*{\glsgroupskip}{}%
11359   \else
11360     \renewcommand*{\glsgroupskip}{& \&\tabularnewline}%
11361   \fi
11362 }
11363 }
11364 {}
```

Four column style:

```
11365 \ifcsdef{@glsstyle@altlongragged4col}
11366 {%
11367   \renewglossarystyle{altlongragged4col}{%
11368     \renewenvironment{theglossary}%
11369       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
11370         >{\raggedright}p{\glspagelistwidth}}}%
11371       {\end{longtable}}}%
11372     \renewcommand*{\glossaryheader}{}%
11373     \renewcommand*{\glsgroupheading}[1]{}%
11374     \renewcommand{\glossentry}[2]{%
11375       \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11376       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
11377       ##2\tabularnewline
11378     }%
11379     \renewcommand{\subglossentry}[3]{%
11380       &
```

```

11381      \glssubentryitem{##2}%
11382      \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11383      \glossentrysymbol{##2} & ##3\tabularnewline
11384  }%

11385  \ifglsnogroupskip
11386      \renewcommand*{\glsgroupskip}{}%
11387  \else
11388      \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
11389  \fi
11390  }
11391  }
11392  {}

```

2.5 Supertabular Styles

The three and four column styles require adjustment to add the post-description hook. The two column styles need the hard-coded `\space` changed to `\glxtrprelocation`.

```

11393 \ifcsdef{@glstyle@super}
11394 {%
11395   \renewglossarystyle{super}{%
11396     \renewenvironment{theglossary}%
11397       {\tablehead{}\tabletail}%
11398       \begin{supertabular}{lp{\glsdescwidth}}%
11399       {\end{supertabular}}%
11400   \renewcommand*{\glossaryheader}{}%
11401   \renewcommand*{\glsgroupheading}[1]{}%
11402   \renewcommand{\glossentry}[2]{%
11403     \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11404     \glossentrydesc{##1}\glspostdescription
11405     \glxtrprelocation ##2\tabularnewline
11406   }%
11407   \renewcommand{\subglossentry}[3]{%
11408     &
11409     \glssubentryitem{##2}%
11410     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11411     \glxtrprelocation ##3\tabularnewline
11412   }%
11413   \ifglsnogroupskip
11414       \renewcommand*{\glsgroupskip}{}%
11415   \else
11416       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11417   \fi
11418  }
11419  }
11420  {}

```

Three column style:

```

11421 \ifcsdef{@glstyle@super3col}

```

```

11422 {%
11423   \renewglossarystyle{super3col}{%
11424     \renewenvironment{theglossary}%
11425       {\tablehead{}\tabletail{}}%
11426       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
11427     {\end{supertabular}}%
11428     \renewcommand*{\glossaryheader}{}%
11429     \renewcommand*{\glsgroupheading}[1]{}%
11430     \renewcommand{\glossentry}[2]{%
11431       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11432       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
11433     }%
11434     \renewcommand{\subglossentry}[3]{%
11435       &
11436       \glssubentryitem{##2}%
11437       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11438       ##3\tabularnewline
11439     }%

11440     \ifglsgnogroupskip
11441       \renewcommand*{\glsgroupskip}{}%
11442     \else
11443       \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
11444     \fi
11445   }
11446 }
11447 {}

```

Four column styles:

```

11448 \ifcsdef{@glstyle@super4col}
11449 {%
11450   \renewglossarystyle{super4col}{%
11451     \renewenvironment{theglossary}%
11452       {\tablehead{}\tabletail{}}%
11453       \begin{supertabular}{llll}}{%
11454     \end{supertabular}}%
11455     \renewcommand*{\glossaryheader}{}%
11456     \renewcommand*{\glsgroupheading}[1]{}%
11457     \renewcommand{\glossentry}[2]{%
11458       \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11459       \glossentrydesc{##1}\glspostdescription &
11460       \glossentrysymbol{##1} & ##2\tabularnewline
11461     }%
11462     \renewcommand{\subglossentry}[3]{%
11463       &
11464       \glssubentryitem{##2}%
11465       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11466       \glossentrysymbol{##2} & ##3\tabularnewline
11467     }%

```

```

11468 \ifglsnogroupskip
11469 \renewcommand*{\glsgroupskip}{}%
11470 \else
11471 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11472 \fi
11473 }
11474 }
11475 {}

```

2.6 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`.

```

11476 \ifcsdef{@glstyle@superragged}
11477 {%
11478 \renewglossarystyle{superragged}{%
11479 \renewenvironment{theglossary}%
11480 {\tablehead{}}\tabletail{}}%
11481 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
11482 {\end{supertabular}}%
11483 \renewcommand*{\glossaryheader}{}%
11484 \renewcommand*{\glsgroupheading}[1]{}%
11485 \renewcommand{\glossentry}[2]{%
11486 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11487 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
11488 \tabularnewline
11489 }%
11490 \renewcommand{\subglossentry}[3]{%
11491 &
11492 \glssubentryitem{##2}%
11493 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
11494 \glstrprelocation ##3%
11495 \tabularnewline
11496 }%
11497 \ifglsnogroupskip
11498 \renewcommand*{\glsgroupskip}{}%
11499 \else
11500 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
11501 \fi
11502 }
11503 }
11504 {}

```

Three column style:

```

11505 \ifcsdef{@glstyle@superragged3col}
11506 {%
11507 \renewglossarystyle{superragged3col}{%

```

```

11508 \renewenvironment{theglossary}%
11509     {\tablehead{}\tabletail{}}%
11510     \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
11511       >{\raggedright}p{\glspagelistwidth}}}%
11512     {\end{supertabular}}%
11513 \renewcommand*{\glossaryheader}{}%
11514 \renewcommand*{\glsgroupheading}[1]{}%
11515 \renewcommand{\glossentry}[2]{%
11516     \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11517     \glossentrydesc{##1}\glspostdescription &
11518     ##2\tabularnewline
11519 }%
11520 \renewcommand{\subglossentry}[3]{%
11521     &
11522     \glssubentryitem{##2}%
11523     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11524     ##3\tabularnewline
11525 }%

11526 \ifglsnogroupskip
11527     \renewcommand*{\glsgroupskip}{}%
11528 \else
11529     \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
11530 \fi
11531 }
11532 }
11533 {}

```

Four columns:

```

11534 \ifcsdef{@glsstyle@altsuperragged4col}
11535 {%
11536     \renewglossarystyle{altsuperragged4col}{%
11537         \renewenvironment{theglossary}%
11538             {\tablehead{}\tabletail{}}%
11539             \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
11540               >{\raggedright}p{\glspagelistwidth}}}%
11541             {\end{supertabular}}%
11542         \renewcommand*{\glossaryheader}{}%
11543         \renewcommand{\glossentry}[2]{%
11544             \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
11545             \glossentrydesc{##1}\glspostdescription &
11546             \glossentrysymbol{##1} & ##2\tabularnewline
11547         }%
11548         \renewcommand{\subglossentry}[3]{%
11549             &
11550             \glssubentryitem{##2}%
11551             \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
11552             \glossentrysymbol{##2} & ##3\tabularnewline
11553         }%

```



```

11554 \ifglsnogroupskip
11555 \renewcommand*{\glsgroupskip}{}%
11556 \else
11557 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
11558 \fi
11559 }
11560 }
11561 {}

```

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`.

```

11562 \ifdef{\@glsstyle@inline}
11563 {%
11564 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
11565 \renewcommand*{\glsinlinedescformat}[3]{%
11566 \space#1\glxtrpostdescription}
11567 \renewcommand*{\glsinlinesubdescformat}[3]{%
11568 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

11569 }
11570 {}

```

2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

11571 \ifdef{\@glsstyle@index}
11572 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

11573 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

11574 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

11575 \renewglossarystyle{index}{%
11576 \renewenvironment{theglossary}%
11577 {\setlength{\parindent}{0pt}%
11578 \setlength{\parskip}{0pt plus 0.3pt}%
11579 \let\item\glstreeitem
11580 \let\subitem\glstreesubitem

```

```

11581     \let\subsubitem\glstreesubsubitem
11582   }%
11583 {\par}%
11584   \renewcommand*\{\glossaryheader}\{%
11585   \renewcommand*\{\glsgroupheading}[1]{\%
11586   \renewcommand*\{\glossentry}[2]{\%
11587     \item\glstentryitem{##1}%
11588     \glstreenamefmt{\glstarget{##1}\glossentryname{##1}}}%
11589     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{\}%
11590     \glstreepredesc \glossentrydesc{##1}\glspostdescription
11591     \glstreeprelocation ##2%
11592   }%
11593   \renewcommand{\subglossentry}[3]{\%
11594     \ifcase##1\relax
11595       \item
11596     \or
11597       \subitem
11598       \glssubentryitem{##2}%
11599     \else
11600       \subsubitem
11601     \fi
11602     \glstreenamefmt{\glstarget{##2}\glossentryname{##2}}}%
11603     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{\}%
11604     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11605     \glstreechildprelocation ##3%
11606   }%
11607   \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11608 }
11609 }
11610 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

11611 \ifdef{\@glsstyle@indexgroup}
11612 {\%
11613   \renewglossarystyle{indexgroup}{\%
11614     \setglossarystyle{index}%
11615     \renewcommand*\{\glsgroupheading}[1]{\%
11616       \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}}%
11617       \nopagebreak\indexspace
11618       \nobreak\@afterheading
11619     }%
11620   }
11621 }
11622 {}

```

Similarly for indexhypergroup.

```

11623 \ifdef{\@glsstyle@indexhypergroup}
11624 {\%
11625   \renewglossarystyle{indexhypergroup}{\%
11626     \setglossarystyle{index}%

```

```

11627 \renewcommand*{\glossaryheader}{%
11628     \item\glstreenavigationfmt{\glsnavigation}%
11629     \nobreak\@afterheading\indexspace}%
11630 \renewcommand*{\glsgroupheading}[1]{%
11631     \item\glstreegroupheaderfmt
11632     {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
11633     \nopagebreak\indexspace
11634     \nobreak\@afterheading}%
11635 }%
11636 }
11637 {}

```

Adjust tree style to remove hard coded space before number list.

```

11638 \ifdef{\@glsstyle@tree}
11639 {%
11640 \renewglossarystyle{tree}{%
11641 \renewenvironment{theglossary}%
11642 {\setlength{\parindent}{0pt}%
11643 \setlength{\parskip}{0pt plus 0.3pt}}%
11644 {}%
11645 \renewcommand*{\glossaryheader}{}%
11646 \renewcommand*{\glsgroupheading}[1]{}%
11647 \renewcommand{\glossentry}[2]{%
11648     \hangindent0pt\relax
11649     \parindent0pt\relax
11650     \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11651     \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11652     \glstreepredesc\glossentrydesc{##1}\glspostdescription
11653     \glstreeprelocation##2\par
11654 }%
11655 \renewcommand{\subglossentry}[3]{%
11656     \hangindent##1\glstreeindent\relax
11657     \parindent##1\glstreeindent\relax
11658     \ifnum##1=1\relax
11659         \glssubentryitem{##2}%
11660     \fi
11661     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
11662     \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}{}%
11663     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
11664     \glstreechildprelocation ##3\par
11665 }%
11666 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11667 }%
11668 }
11669 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

11670 \ifdef{\@glsstyle@treegroup}
11671 {%
11672 \renewglossarystyle{treegroup}{%

```

```

11673 \setglossarystyle{tree}%
11674 \renewcommand{\glsgroupheading}[1]{\par
11675 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
11676 \nopagebreak\indexspace\nobreak\@afterheading}%
11677 }
11678 }
11679 {}

```

Similarly for treehypergroup

```

11680 \ifdef{\@glsstyle@treehypergroup}
11681 {%
11682 \renewglossarystyle{treehypergroup}{%
11683 \setglossarystyle{tree}%
11684 \renewcommand*\glossaryheader{%
11685 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11686 \nobreak\@afterheading\indexspace}%
11687 \renewcommand*\glsgroupheading}[1]{%
11688 \par\noindent
11689 \glstreegroupheaderfmt
11690 {\glssnavhypertarget{##1}{\glsgrouptitle{##1}}}\par
11691 \nopagebreak\indexspace\nobreak\@afterheading}%
11692 }
11693 }
11694 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

11695 \ifdef{\@glsstyle@treenoname}
11696 {%
11697 \renewglossarystyle{treenoname}{%
11698 \renewenvironment{theglossary}%
11699 {\setlength{\parindent}{0pt}%
11700 \setlength{\parskip}{0pt plus 0.3pt}}%
11701 {}%
11702 \renewcommand*\glossaryheader{}%
11703 \renewcommand*\glsgroupheading}[1]{}%
11704 \renewcommand{\glossentry}[2]{%
11705 \hangindent0pt\relax
11706 \parindent0pt\relax
11707 \glssentryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
11708 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}{}%
11709 \glstreepredesc\glossentrydesc{##1}\glspostdescription
11710 \glstreeprelocation##2\par
11711 }%
11712 \renewcommand{\subglossentry}[3]{%
11713 \hangindent##1\glstreeindent\relax
11714 \parindent##1\glstreeindent\relax
11715 \ifnum##1=1\relax
11716 \glssubentryitem{##2}%
11717 \fi
11718 \glstarget{##2}{\strut}%

```

```

11719     \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
11720 }%
11721 \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
11722 }
11723 }
11724 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

11725 \ifdef{\@glsstyle@treenonamegroup}
11726 {%
11727   \renewglossarystyle{treenonamegroup}{%
11728     \setglossarystyle{treenoname}%
11729     \renewcommand{\glsgroupheading}[1]{\par
11730       \noindent\glstreegroupheaderfmt
11731       {\glsgrouptitle{##1}}}%
11732     \nopagebreak\indexspace\nobreak\@afterheading
11733   }%
11734 }
11735 }
11736 {}

```

Similarly for `treenonamehypergroup`

```

11737 \ifdef{\@glsstyle@treenonamehypergroup}
11738 {%
11739   \renewglossarystyle{treenonamehypergroup}{%
11740     \setglossarystyle{treenoname}%
11741     \renewcommand*{\glossaryheader}{%
11742       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
11743       \nobreak\@afterheading\indexspace}%
11744     \renewcommand*{\glsgroupheading}[1]{%
11745       \par\noindent
11746       \glstreegroupheaderfmt
11747       {\glsnavigationtarget{##1}{\glsgrouptitle{##1}}}%
11748     \nopagebreak\indexspace\nobreak\@afterheading}%
11749   }
11750 }
11751 {}

```

The `almtree` style is redefined to make it easier to make minor adjustments.

```

11752 \ifdef{\@glsstyle@almtree}
11753 {%

```

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.

```

11754 \newcommand{\glxtralttreeSymbolDescLocation}[2]{%

```

```

11755   {%
11756       \let\par\glxstrAltTreePar
11757       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
11758       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
11759   }%
11760 }

```

trAltTreeIndent Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```

11761 \newlength\glxstrAltTreeIndent

```

lsxtrAltTreePar Multi-paragraph descriptions need to keep the hanging indent.

```

11762 \newcommand{\glxstrAltTreePar}{%
11763     @@par
11764     \glxstrAltTreeSetHangIndent
11765     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
11766 }

```

symbolDescLocation `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

11767 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
11768     \glxstralttreeSymbolDescLocation{#2}{#3}%
11769 }

```

trreetopindent The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```

11770 \newlength\glxstrtreetopindent

```

lxstralttreeInit User-level initialisation for the alttree style.

```

11771 \newcommand*{\glxstralttreeInit}{%
11772     \settowidth{\glxstrtreetopindent}{\glstreenamfmt{\glsgetwidestname\space}}%
11773     \glxstrAltTreeIndent=\parindent
11774 }

```

\glsglsetwidest The original `\glsglsetwidest` only uses `\def`. This uses `\gdef`.

```

11775 \newcommand*{\glsglsetwidest}[2][0]{%
11776     \csgdef{@glswidestname\romannumeral#1}{#2}%
11777 }

```

\eglssetwidest The original `\glsglsetwidest` only uses `\def`. This uses `\protected@csdef`.

```

11778 \newcommand*{\eglssetwidest}[2][0]{%
11779     \protected@csdef{@glswidestname\romannumeral#1}{#2}%
11780 }

```

`\xglsssetwidest` Like the above but uses `\protected@csxdef`.

```

11781 \newcommand*{\xglsssetwidest}[2][0]{%
11782   \protected@csxdef{\glswidestname\romannumeral#1}{#2}%
11783 }

```

`\glswidestname` Provide a user-level macro to obtain the widest top-level name.

```

11784 \newcommand*{\glswidestname}{\@glswidestname}

```

`\glswidestsubname` Provide a user-level macro to obtain the widest sub-entry name.

```

11785 \newcommand*{\glswidestsubname}[1]{%
11786   \ifcsundef{\@glswidestname\romannumeral#1}%
11787   {\@glswidestname}%
11788   {\csuse{\@glswidestname\romannumeral#1}}%
11789 }

```

`\glswidestTopLevelName` CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidesttoplevelname`.

```

11790 \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.

```

11791 \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
11792   \dimen@=0pt\relax
11793   \gls@tmplen=0pt\relax
11794   \forallglossaries[#1]{\@gls@type}%
11795   {%
11796     \forallglsentries[\@gls@type]{\@glo@label}%
11797     {%
11798       \ifglswidest{\@glo@label}%
11799       {%
11800         \ifglshasparent{\@glo@label}%
11801         {}%
11802         {%
11803           \settowidth{\dimen@}%
11804             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11805           \ifdim\dimen@>\gls@tmplen
11806             \gls@tmplen=\dimen@
11807             \eglssetwidest{\glsentryname{\@glo@label}}%
11808           \fi
11809         }%
11810       }%
11811     }%
11812   }%
11813 }%
11814 }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

11815 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
11816   \dimen@=0pt\relax
11817   \gls@tmplen=0pt\relax
11818   \forallglossaries[#1]{\@gls@type}%
11819   {%
11820     \forallgsentries[\@gls@type]{\@glo@label}%
11821     {%
11822       \ifglsused{\@glo@label}%
11823       {%
11824         \settowidth{\dimen@}%
11825           {\glstreenamefmt{\glseentryname{\@glo@label}}}%
11826         \ifdim\dimen@>\gls@tmplen
11827           \gls@tmplen=\dimen@
11828           \eglssetwidest{\glseentryname{\@glo@label}}%
11829         \fi
11830       }%
11831     }%
11832   }%
11833 }%
11834 }

```

`ndWidestAnyName` Like the above but doesn't check is the entry has been used.

```

11835 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
11836   \dimen@=0pt\relax
11837   \gls@tmplen=0pt\relax
11838   \forallglossaries[#1]{\@gls@type}%
11839   {%
11840     \forallgsentries[\@gls@type]{\@glo@label}%
11841     {%
11842       \settowidth{\dimen@}%
11843         {\glstreenamefmt{\glseentryname{\@glo@label}}}%
11844       \ifdim\dimen@>\gls@tmplen
11845         \gls@tmplen=\dimen@
11846         \eglssetwidest{\glseentryname{\@glo@label}}%
11847       \fi
11848     }%
11849   }%
11850 }

```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

11851 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
11852   \dimen@=0pt\relax
11853   \dimen@i=0pt\relax
11854   \dimen@ii=0pt\relax
11855   \forallglossaries[#1]{\@gls@type}%
11856   {%

```



```

11857 \forglsentries[\@gls@type]{\@glo@label}%
11858 {%
11859 \ifglsused{\@glo@label}%
11860 {%
11861 \ifglsashasparent{\@glo@label}%
11862 {%
11863 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
11864 \ifglsashasparent{\@glo@parent}%
11865 {%
11866 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
11867 \ifglsashasparent{\@glo@parent}%
11868 {}%
11869 {%
11870 \settowidth{\gls@tmplen}%
11871 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11872 \ifdim\gls@tmplen>\dimen@ii
11873 \dimen@ii=\gls@tmplen
11874 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
11875 \fi
11876 }%
11877 }%
11878 {%
11879 \settowidth{\gls@tmplen}%
11880 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11881 \ifdim\gls@tmplen>\dimen@i
11882 \dimen@i=\gls@tmplen
11883 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
11884 \fi
11885 }%
11886 }%
11887 {%
11888 \settowidth{\gls@tmplen}%
11889 {\glstreenamefmt{\glsentryname{\@glo@label}}}%
11890 \ifdim\gls@tmplen>\dimen@
11891 \dimen@=\gls@tmplen
11892 \eglssetwidest{\glsentryname{\@glo@label}}%
11893 \fi
11894 }%
11895 }%
11896 {}%
11897 }%
11898 }%
11899 }

```

dWidestLevelTwo This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

11900 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
11901 \dimen@=0pt\relax
11902 \dimen@i=0pt\relax
11903 \dimen@ii=0pt\relax

```

```

11904 \forall glossaries[#1]{\@gls@type}%
11905 {%
11906   \forall sentries[\@gls@type]{\@glo@label}%
11907   {%
11908     \ifglshasparent{\@glo@label}%
11909     {%
11910       \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@label}@parent}}%
11911       \ifglshasparent{\@glo@parent}%
11912       {%
11913         \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@parent}@parent}}%
11914         \ifglshasparent{\@glo@parent}%
11915         {}%
11916         {%
11917           \settowidth{\gls@tmplen}%
11918             {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11919           \ifdim\gls@tmplen>\dimen@ii
11920             \dimen@ii=\gls@tmplen
11921             \eglssetwidest[2]{\glsentryname{\@glo@label}}%
11922           \fi
11923         }%
11924       }%
11925     }%
11926     \settowidth{\gls@tmplen}%
11927       {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11928     \ifdim\gls@tmplen>\dimen@i
11929       \dimen@i=\gls@tmplen
11930       \eglssetwidest[1]{\glsentryname{\@glo@label}}%
11931     \fi
11932   }%
11933 }%
11934 {%
11935   \settowidth{\gls@tmplen}%
11936     {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11937   \ifdim\gls@tmplen>\dimen@
11938     \dimen@=\gls@tmplen
11939     \eglssetwidest{\glsentryname{\@glo@label}}%
11940   \fi
11941 }%
11942 }%
11943 }%
11944 }

```

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.

```

11945 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
11946   \dimen@=0pt\relax
11947   \gls@tmplen=0pt\relax
11948   #2=0pt\relax
11949   \forall glossaries[#1]{\@gls@type}%

```

```

11950  {%
11951    \forglsentries[\@gls@type]{\@glo@label}%
11952  {%
11953    \ifglsused{\@glo@label}%
11954  {%
11955    \settowidth{\dimen@}%
11956      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11957    \ifdim\dimen@>\gls@tmplen
11958      \gls@tmplen=\dimen@
11959      \eglssetwidest{\glsentryname{\@glo@label}}%
11960    \fi
11961    \settowidth{\dimen@}%
11962      {\glsentrysymbol{\@glo@label}}%
11963    \ifdim\dimen@>#2\relax
11964      #2=\dimen@
11965    \fi
11966  }%
11967  }%
11968 }%
11969 }%
11970 }

```

stAnyNameSymbol Like the above but doesn't check if the entry has been used.

```

11971  \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
11972    \dimen@=0pt\relax
11973    \gls@tmplen=0pt\relax
11974    #2=0pt\relax
11975    \forallglossaries[#1]{\@gls@type}%
11976  {%
11977    \forglsentries[\@gls@type]{\@glo@label}%
11978  {%
11979    \settowidth{\dimen@}%
11980      {\glstreenamfmt{\glsentryname{\@glo@label}}}%
11981    \ifdim\dimen@>\gls@tmplen
11982      \gls@tmplen=\dimen@
11983      \eglssetwidest{\glsentryname{\@glo@label}}%
11984    \fi
11985    \settowidth{\dimen@}%
11986      {\glsentrysymbol{\@glo@label}}%
11987    \ifdim\dimen@>#2\relax
11988      #2=\dimen@
11989    \fi
11990  }%
11991  }%
11992  }

```

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.

```

11993 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
11994   \dimen@=0pt\relax
11995   \gls@tmplen=0pt\relax
11996   #2=0pt\relax
11997   #3=0pt\relax
11998   \foralllglossaries[#1]{\@gls@type}%
11999   {%
12000     \forglsentries[\@gls@type]{\@glo@label}%
12001     {%
12002       \ifglsused{\@glo@label}%
12003       {%
12004         \settowidth{\dimen@}%
12005         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12006         \ifdim\dimen@>\gls@tmplen
12007           \gls@tmplen=\dimen@
12008           \eglssetwidest{\glsentryname{\@glo@label}}%
12009         \fi
12010         \settowidth{\dimen@}%
12011         {\glsentrysymbol{\@glo@label}}%
12012         \ifdim\dimen@>#2\relax
12013           #2=\dimen@
12014         \fi
12015         \settowidth{\dimen@}%
12016         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12017         \ifdim\dimen@>#3\relax
12018           #3=\dimen@
12019         \fi
12020       }%
12021     }%
12022   }%
12023 }%
12024 }
```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

12025 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
12026   \dimen@=0pt\relax
12027   \gls@tmplen=0pt\relax
12028   #2=0pt\relax
12029   #3=0pt\relax
12030   \foralllglossaries[#1]{\@gls@type}%
12031   {%
12032     \forglsentries[\@gls@type]{\@glo@label}%
12033     {%
12034       \settowidth{\dimen@}%
12035       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12036       \ifdim\dimen@>\gls@tmplen
12037         \gls@tmplen=\dimen@
12038       \eglssetwidest{\glsentryname{\@glo@label}}%
```

```

12039      \fi
12040      \settowidth{\dimen@}%
12041      {\glstrysymbol{\@glo@label}}}%
12042      \ifdim\dimen@>#2\relax
12043      #2=\dimen@
12044      \fi
12045      \settowidth{\dimen@}%
12046      {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12047      \ifdim\dimen@>#3\relax
12048      #3=\dimen@
12049      \fi
12050    }%
12051  }%
12052 }

```

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.

```

12053 \newrobustcmd*{\glstryFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
12054   \dimen@=0pt\relax
12055   \glstry@tmplen=0pt\relax
12056   #2=0pt\relax
12057   \forallglossaries[#1]{\@gls@type}%
12058   {%
12059     \forallglsentries[\@gls@type]{\@glo@label}%
12060     {%
12061       \ifglstryused{\@glo@label}%
12062       {%
12063         \settowidth{\dimen@}%
12064         {\glstrynamefmt{\glstryname{\@glo@label}}}%
12065         \ifdim\dimen@>\glstry@tmplen
12066         \glstry@tmplen=\dimen@
12067         \glstrysetwidest{\glstryname{\@glo@label}}%
12068         \fi
12069         \settowidth{\dimen@}%
12070         {\GlsXtrFormatLocationList{\glstrynumberlist{\@glo@label}}}%
12071         \ifdim\dimen@>#2\relax
12072         #2=\dimen@
12073         \fi
12074       }%
12075     }%
12076   }%
12077 }%
12078 }

```

AnyNameLocation Like the `\glstryFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

12079 \newrobustcmd*{\glstryFindWidestAnyNameLocation}[2][\@glo@types]{%
12080   \dimen@=0pt\relax
12081   \glstry@tmplen=0pt\relax

```

```

12082     #2=Opt\relax
12083     \forallglossaries[#1]{\@gls@type}%
12084     {%
12085         \forglsentries[\@gls@type]{\@glo@label}%
12086         {%
12087             \settowidth{\dimen@}%
12088             {\glstreenamefmt{\glsentryname{\@glo@label}}}%
12089             \ifdim\dimen@>\gls@tmplen
12090                 \gls@tmplen=\dimen@
12091                 \eglssetwidest{\glsentryname{\@glo@label}}%
12092             \fi
12093             \settowidth{\dimen@}%
12094             {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
12095             \ifdim\dimen@>#2\relax
12096                 #2=\dimen@
12097             \fi
12098         }%
12099     }%
12100 }

```

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`.

```

12101 \newcommand*{\glsxtrComputeTreeIndent}[1]{%
12102     \glstreeindent=\glsxtrtreetopindent\relax
12103 }

```

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.

```

12104 \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
12105     \ifcsundef{@glswidestname\romannumeral#1}%
12106     {%
12107         \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
12108     }%
12109     {%
12110         \settowidth{#3}{\glstreenamefmt{%
12111             \csname @glswidestname\romannumeral#1\endcsname\space}}%
12112     }%
12113 }

```

treeSetHangIndent Set `\hangindent` for top-level entries:

```

12114 \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
12115 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
12116 \renewglossarystyle{alttree}{%
12117   \renewenvironment{theglossary}%
12118     {%
12119       \glxtralttreeInit
12120       \def\@gls@prevlevel{-1}%
12121       \mbox{}\par}%
12122     {\par}%
12123   \renewcommand*{\glossaryheader}{}%
12124   \renewcommand*{\glsgroupheading}[1]{}%
12125   \renewcommand{\glossentry}[2]{%
12126     \ifnum\@gls@prevlevel=0\relax
12127     \else
12128       \glxtrComputeTreeIndent{##1}%
12129     \fi
12130     \parindent\glstreeindent
12131     \glxtrAltTreeSetHangIndent
12132     \makebox[0pt][r]%
12133     {%
12134       \glstreenamebox{\glstreeindent}%
12135       {%
12136         \glssentryitem{##1}%
12137         \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
12138       }%
12139     }%
12140     \glxtralttreeSymbolDescLocation{##1}{##2}%
12141     \def\@gls@prevlevel{0}%
12142   }
12143   \renewcommand{\subglossentry}[3]{%
12144     \ifnum##1=1\relax
12145       \glssubentryitem{##2}%
12146     \fi
12147     \ifnum\@gls@prevlevel=##1\relax
12148     \else
12149       \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
12150       \ifnum\@gls@prevlevel<##1\relax
12151         \setlength\glstreeindent\gls@tmplen
12152         \addtolength\glstreeindent\parindent
12153         \parindent\glstreeindent
12154       \else
12155         \ifnum\@gls@prevlevel=0\relax
12156           \glxtrComputeTreeIndent{##2}%
12157         \else
12158           \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
12159         \fi
12160         \addtolength\parindent{-\glstreeindent}%

```

```

12161         \setlength\glstreeindent\parindent
12162     \fi
12163 \fi
12164 \glstrAltTreeSetSubHangIndent{##1}%
12165 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
12166     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
12167 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
12168 \def\@gls@prevlevel{##1}%
12169 }%
12170 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12171 }
12172 }%
12173 {%
12174 }

```

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.

```

12175 \ifdef{\@glsstyle@alttreegroup}
12176 {%
12177     \renewglossarystyle{alttreegroup}{%
12178         \setglossarystyle{alttree}%
12179         \renewcommand{\glsgroupheading}[1]{\par
12180             \def\@gls@prevlevel{-1}%
12181             \hangindentOpt\relax
12182             \parindentOpt\relax
12183             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}%
12184             \nopagebreak\indexspace\nopagebreak
12185         }%
12186     }%
12187 }%
12188 {%
12189 }

```

Similarly for `alttreehypergroup`.

```

12190 \ifdef{\@glsstyle@alttreehypergroup}
12191 {%
12192     \renewglossarystyle{alttreehypergroup}{%
12193         \setglossarystyle{alttree}%
12194         \renewcommand*\{\glossaryheader}{%
12195             \par
12196             \def\@gls@prevlevel{-1}%
12197             \hangindentOpt\relax
12198             \parindentOpt\relax
12199             \glstreenavigationfmt{\glsnavigation}\par\indexspace
12200         }%
12201         \renewcommand*\{\glsgroupheading}[1]{%
12202             \par
12203             \def\@gls@prevlevel{-1}%
12204             \hangindentOpt\relax
12205             \parindentOpt\relax

```



```

12206      \glstreegroupheaderfmt
12207      {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
12208      \nopagebreak\indexspace\nopagebreak
12209    }%
12210  }
12211 }%
12212 {%
12213 }

```

2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

12214 \ifdef{\@glsstyle@mcolindexgroup}
12215 {%
12216   \renewglossarystyle{mcolindexgroup}{%
12217     \setglossarystyle{mcolindex}%
12218     \renewcommand*\{glsgroupheading}[1]{%
12219       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
12220       \nopagebreak\indexspace\nobreak\@afterheading
12221     }%
12222   }
12223 }%
12224 {%
12225 }

```

Similarly for mcolindexhypergroup.

```

12226 \ifdef{\@glsstyle@mcolindexhypergroup}
12227 {%
12228   \renewglossarystyle{mcolindexhypergroup}{%
12229     \setglossarystyle{mcolindex}%
12230     \renewcommand*\{glossaryheader}{%
12231       \item\glstreenavigationfmt{\glsnavigation}%
12232       \indexspace
12233     }%
12234     \renewcommand*\{glsgroupheading}[1]{%
12235       \item\glstreegroupheaderfmt
12236       {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12237       \nopagebreak\indexspace\nobreak\@afterheading
12238     }%
12239   }
12240 }%
12241 {%
12242 }

```

Similarly for mcolindexspannav.

```

12243 \ifdef{\@glsstyle@mcolindexspannav}
12244 {%
12245   \renewglossarystyle{mcolindexspannav}{%
12246     \setglossarystyle{index}%

```

```

12247 \renewenvironment{theglossary}%
12248 {%
12249 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
12250 \setlength{\parindent}{0pt}%
12251 \setlength{\parskip}{0pt plus 0.3pt}%
12252 \let\item\glstreeitem}%
12253 {\end{multicols}}}%
12254 \renewcommand*{\glsgroupheading}[1]{%
12255 \item\glstreegroupheaderfmt
12256 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12257 \nopagebreak\indexspace\nobreak\@afterheading
12258 }%
12259 }
12260 }%
12261 {%
12262 }

```

Similarly for mcoltreegroup.

```

12263 \ifdef{\@glsstyle@mcoltreegroup}
12264 {%
12265 \renewglossarystyle{mcoltreegroup}{%
12266 \setglossarystyle{mcoltree}%
12267 \renewcommand{\glsgroupheading}[1]{\par
12268 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12269 \nopagebreak\indexspace\nobreak\@afterheading
12270 }%
12271 }
12272 }%
12273 {%
12274 }

```

Similarly for mcoltreehypergroup.

```

12275 \ifdef{\@glsstyle@mcoltreehypergroup}
12276 {%
12277 \renewglossarystyle{mcoltreehypergroup}{%
12278 \setglossarystyle{mcoltree}%
12279 \renewcommand*{\glossaryheader}{%
12280 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
12281 }%
12282 \renewcommand*{\glsgroupheading}[1]{%
12283 \par\noindent
12284 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12285 \nopagebreak\indexspace\nobreak\@afterheading
12286 }%
12287 }
12288 }%
12289 {%
12290 }

```

Similarly for mcoltreespannav.

```

12291 \ifdef{\@glsstyle@mcoltreespannav}

```

```

12292 {%
12293   \renewglossarystyle{mcoltreespannav}{%
12294     \setglossarystyle{tree}%
12295     \renewenvironment{theglossary}%
12296     {%
12297       \begin{multicols}{\glsmcols}%
12298       [\noindent\glstreenavigationfmt{\glsnavigation}]%
12299       \setlength{\parindent}{0pt}%
12300       \setlength{\parskip}{0pt plus 0.3pt}%
12301     }%
12302   {\end{multicols}}}%
12303   \renewcommand*{\glsgroupheading}[1]{%
12304     \par\noindent
12305     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12306     \nopagebreak\indexspace\nobreak\@afterheading
12307   }%
12308 }
12309 }%
12310 {%
12311 }

```

Similarly for mcoltreenonamegroup.

```

12312 \ifdef{\@glsstyle@mcoltreenonamegroup}
12313 {%
12314   \renewglossarystyle{mcoltreenonamegroup}{%
12315     \setglossarystyle{mcoltreenoname}%
12316     \renewcommand*{\glsgroupheading}[1]{\par
12317       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
12318     \nopagebreak\indexspace\nobreak\@afterheading
12319   }%
12320 }
12321 }%
12322 {%
12323 }

```

Similarly for mcoltreenonamehypergroup.

```

12324 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
12325 {%
12326   \renewglossarystyle{mcoltreenonamehypergroup}{%
12327     \setglossarystyle{mcoltreenoname}%
12328     \renewcommand*{\glossaryheader}{%
12329       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
12330     \renewcommand*{\glsgroupheading}[1]{%
12331       \par\noindent
12332       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
12333       \nopagebreak\indexspace\nobreak\@afterheading}%
12334   }
12335 }%
12336 {%
12337 }

```

Similarly for mcoltreenonamespannav.

```

12338 \ifdef{\@glsstyle@mcoltreenonamespannav}
12339 {%
12340   \renewglossarystyle{mcoltreenonamespannav}{%
12341     \setglossarystyle{treenoname}%
12342     \renewenvironment{theglossary}%
12343     {%
12344       \begin{multicols}{\glsmcols}%
12345       [\noindent\glstreenavigationfmt{\glsnavigation}}%
12346       \setlength{\parindent}{0pt}%
12347       \setlength{\parskip}{0pt plus 0.3pt}%
12348     }%
12349     {\end{multicols}}}%
12350   \renewcommand*{\glsgroupheading}[1]{%
12351     \par\noindent
12352     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12353     \nopagebreak\indexspace\nobreak\@afterheading}%
12354   }
12355 }%
12356 {%
12357 }

```

mcolalmtree needs adjusting so that it uses \glxtralmtreeInit This doesn't use \mbox{\par which would unbalance the top of the columns.

```

12358 \ifdef{\@glsstyle@mcolalmtree}
12359 {%
12360   \renewglossarystyle{mcolalmtree}{%
12361     \setglossarystyle{almtree}%
12362     \renewenvironment{theglossary}%
12363     {%
12364       \glxtralmtreeInit
12365       \def\@gls@prevlevel{-1}%
12366       \begin{multicols}{\glsmcols}%
12367     }%
12368     {\par\end{multicols}}}%
12369   }
12370 }%
12371 {%
12372 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

12373 \ifdef{\@glsstyle@mcolalmtreegroup}
12374 {%
12375   \renewglossarystyle{mcolalmtreegroup}{%
12376     \setglossarystyle{mcolalmtree}%
12377     \renewcommand{\glsgroupheading}[1]{\par
12378       \def\@gls@prevlevel{-1}%
12379       \hangindent0pt\relax
12380       \parindent0pt\relax
12381       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

12382     \nopagebreak\indexspace\nopagebreak
12383   }%
12384 }
12385 }%
12386 {%
12387 }

```

Similarly for mcolalttreehypergroup.

```

12388 \ifdef{\@glsstyle@mcolalttreehypergroup}
12389 {%
12390   \renewglossarystyle{mcolalttreehypergroup}{%
12391     \setglossarystyle{mcolalttree}%
12392     \renewcommand*{\glossaryheader}{%
12393       \par
12394       \def\@gls@prevlevel{-1}%
12395       \hangindent0pt\relax
12396       \parindent0pt\relax
12397       \glstreenavigationfmt{\glsnavigation}%
12398       \par\indexspace
12399     }%
12400     \renewcommand*{\glsgroupheading}[1]{%
12401       \par
12402       \def\@gls@prevlevel{-1}%
12403       \hangindent0pt\relax
12404       \parindent0pt\relax
12405       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12406       \nopagebreak\indexspace\nopagebreak
12407     }%
12408   }
12409 }%
12410 {%
12411 }

```

Similarly for mcolalttreespannav.

```

12412 \ifdef{\@glsstyle@mcolalttreespannav}
12413 {%
12414   \renewglossarystyle{mcolalttreespannav}{%
12415     \setglossarystyle{alttree}%
12416     \renewenvironment{theglossary}%
12417     {%
12418       \glsxtralttreeInit
12419       \def\@gls@prevlevel{-1}%
12420       \begin{multicols}{\glsncols}%
12421         [\noindent\glstreenavigationfmt{\glsnavigation}]%
12422       }%
12423     {\par\end{multicols}}%
12424     \renewcommand*{\glsgroupheading}[1]{%
12425       \par
12426       \def\@gls@prevlevel{-1}%
12427       \hangindent0pt\relax

```

```

12428      \parindent0pt\relax
12429      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
12430      \noprogrambreak\indexspace\noprogrambreak
12431    }%
12432  }
12433 }%
12434 {%
12435 }

Reset the default style
12436 \ifx\@glossary@default@style\relax
12437 \else
12438   \setglossarystyle{\@glstr@current@style}
12439 \fi

```

3 bookindex style (glossary-bookindex.sty)

3.1 Package Initialisation and Options

```
12440 \NeedsTeXFormat{LaTeX2e}
12441 \ProvidesPackage{glossary-bookindex}[2017/11/08 v1.22 (NLCT)]
```

Load required packages.

```
12442 \RequirePackage{multicol}
12443 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
12444 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
12445 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`ookindexsubname` Format used for sub entries.

```
12446 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`strprelocation` Provide in case glossaries-stylemods isn't loaded.

```
12447 \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.

```
12448 \newcommand*{\glstrbookindexprelocation}[1]{%
12449   \glstrifhasfield{location}{#1}%
12450   {\,\glstrprelocation}%
12451   {\glstrprelocation}%
12452 }
```

`xsubprelocation` Separator used before location list for sub-entries.

```
12453 \newcommand*{\glstrbookindexsubprelocation}[1]{%
12454   \glstrbookindexprelocation{#1}%
12455 }
```

`xparentchildsep` Separator used between top-level parent and child entry.

```
12456 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`rentsubchildsep` Separator used between sub-level parent and child entry.

```
12457 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
12458 \newcommand{\glstrbookindexbetween}[2] {}
```

`indexsubbetween` Between two level 1 entries identified by the labels in the arguments.

```
12459 \newcommand{\glstrbookindexsubbetween}[2] {}
```

`exsubsubbetween` Between two level 2 entries identified by the labels in the arguments.

```
12460 \newcommand{\glstrbookindexsubsubbetween}[2] {}
```

`indexatendgroup` At the end of a letter group. The argument is the index of the last top-level entry.

```
12461 \newcommand{\glstrbookindexatendgroup}[1] {}
```

`exsubatendgroup` At the end of a letter group. The argument is the index of the last level 1 entry.

```
12462 \newcommand{\glstrbookindexsubatendgroup}[1] {}
```

`subsubatendgroup` At the end of a letter group. The argument is the index of the last level 2 entry.

```
12463 \newcommand{\glstrbookindexsubsubatendgroup}[1] {}
```

`bookindexgroupskip` Group separator.

```
12464 \newcommand{\glstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

`indexformatheader` Group separator.

```
12465 \newcommand*{\glstrbookindexformatheader}[1] {%
12466   \par{\centering\glstreegroupheaderfmt{#1}\par}%
12467 }
```

`bookindexbookmark` Book mark group heading if supported.

```
12468 \ifdef\pdfbookmark
12469 {%
12470   \newcommand*{\glstrbookindexbookmark}[2] {%
12471     \ifdefstring{\@glossarysec}{chapter}%
12472     {\pdfbookmark[1]{#1}{#2}}%
12473     {\pdfbookmark[2]{#1}{#2}}%
12474   }
12475 }
12476 {%
12477   \newcommand*{\glstrbookindexbookmark}[2] {}
12478 }
```

`bookindexcolspread`

```
12479 \newcommand*{\glstrbookindexcolspread} {}
```

Define the style.

```
12480 \newglossarystyle{bookindex} {%
12481   \setglossarystyle{index}%
12482   \renewenvironment{theglossary}%

```



```

12483 {%
12484   \ifdefempty\glxstrbookindexcolspread
12485   {\begin{multicols}{\glxstrbookindexcols}}%
12486   {\begin{multicols}{\glxstrbookindexcols}[\glxstrbookindexcolspread]}%
12487   \setlength{\parindent}{0pt}%
12488   \setlength{\parskip}{0pt plus 0.3pt}%
12489   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12490   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12491   \let\@glxstr@bookindex@between\@gobble
12492   \let\@glxstr@bookindex@subbetween\@gobble
12493   \let\@glxstr@bookindex@subsubbetween\@gobble
12494   \let\@glxstr@bookindex@atendgroup\relax
12495   \let\@glxstr@bookindex@subatendgroup\relax
12496   \let\@glxstr@bookindex@subsubatendgroup\relax
12497   \let\@glxstr@bookindexgroupskip\relax
12498 }%
12499 {%

```

Do end group hooks.

```

12500   \@glxstr@bookindex@subsubatendgroup
12501   \@glxstr@bookindex@subatendgroup
12502   \@glxstr@bookindex@atendgroup

```

End multicols environment.

```

12503   \end{multicols}%
12504 }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

12505   \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

12506   \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

12507   \@glxstr@bookindex@between{##1}%

```

Update separators.

```

12508   \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
12509   \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
12510   \let\@glxstr@bookindex@subbetween\@gobble
12511   \let\@glxstr@bookindex@subsubbetween\@gobble
12512   \edef\@glxstr@bookindex@between{%
12513     \noexpand\glxstrbookindexbetween{##1}%
12514   }%
12515   \edef\@glxstr@bookindex@atendgroup{%
12516     \noexpand\glxstrbookindexatendgroup{##1}%
12517   }%
12518   \let\@glxstr@bookindex@subatendgroup\relax
12519   \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

12520   \glstreeitem

```

```

12521      \glsentryitem{##1}%
12522      \glstarget{##1}{\glxtrbookindexname{##1}}%
12523      \glxtrbookindexprelocation{##1}##2%
12524  }%
12525  \renewcommand{\subglossentry}[3]{%
12526    \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

12527      \glstreeitem
12528      \or

```

Level 1.

```

12529      \@glxtr@bookindex@sep
12530      \@glxtr@bookindex@subbetween{##2}%
12531      \let\@glxtr@bookindex@sep\relax

```

Update separators.

```

12532      \let\@glxtr@bookindex@subsubbetween\@gobble
12533      \let\@glxtr@bookindex@subsep\glxtrbookindexparentschildsep
12534      \edef\@glxtr@bookindex@subbetween{%
12535        \noexpand\glxtrbookindexsubbetween{##2}%
12536      }%
12537      \edef\@glxtr@bookindex@atsubendgroup{%
12538        \noexpand\glxtrbookindexatsubendgroup{##1}%
12539      }%

```

Start sub-item.

```

12540      \glstreesubitem
12541      \glssubentryitem{##2}%
12542      \else

```

All other levels.

```

12543      \@glxtr@bookindex@subsep
12544      \@glxtr@bookindex@subsubbetween{##2}%

```

Update separators.

```

12545      \let\@glxtr@bookindex@subsep\relax
12546      \edef\@glxtr@bookindex@subsubbetween{%
12547        \noexpand\glxtrbookindexsubsubbetween{##2}%
12548      }%
12549      \edef\@glxtr@bookindex@atsubsubendgroup{%
12550        \noexpand\glxtrbookindexatsubsubendgroup{##1}%
12551      }%

```

Start sub-sub-item.

```

12552      \glstreesubsubitem
12553      \fi

```

Format entry.

```

12554      \glstarget{##2}{\glxtrbookindexsubname{##2}}%
12555      \glxtrbookindexsubprelocation{##2}##3%
12556  }%

```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
12557 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
12558 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
12559 \@glstr@bookindex@subsubatendgroup
```

```
12560 \@glstr@bookindex@subatendgroup
```

```
12561 \@glstr@bookindex@atendgroup
```

```
12562 \@glstr@bookindexgroupskip
```

Update separators.

```
12563 \let\@glstr@bookindexgroupskip\glstrbookindexgroupskip
```

```
12564 \let\@glstr@bookindex@between\@gobble
```

```
12565 \let\@glstr@bookindex@atendgroup\relax
```

```
12566 \let\@glstr@bookindex@subatendgroup\relax
```

```
12567 \let\@glstr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
12568 \glstrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
12569 \glstrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
12570 \glstrbookindexformatheader{\thisgrptitle}%
```

```
12571 \nopagebreak\indexspace\nopagebreak\@afterheading
```

```
12572 }%
```

```
12573 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glstrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`bookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
12574 \newcommand{\glstrbookindexthepage}{%
```

```
12575 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
```

```
12576 }
```

`bookindexmarkentry` Writes entry information to the `.aux` file. The argument is the entry label.

```
12577 \newcommand*{\glstrbookindexmarkentry}[1]{%
```

```
12578 \protected@write\@auxout
```

```
12579 {\let\glstrbookindexthepage\relax}%
```

```
12580 {\string\glstr@setbookindexmark{\glstrbookindexthepage}{#1}}%
```

```
12581 }
```

etbookindexmark

```
12582 \newcommand*{\glxtr@setbookindexmark}[2]{%
12583   \ifcsundef{glxtr@idxfirstmark@#1}%
12584   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
12585   {}%
12586   \csgdef{glxtr@idxlastmark@#1}{#2}%
12587 }
```

dexfirstmarkfmt

```
12588 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
12589   \glentryname{#1}%
12590 }
```

kindexfirstmark

```
12591 \newcommand*{\glxtrbookindexfirstmark}{%
12592   \letcs{glxtr@label}{glxtr@idxfirstmark@glxtrbookindexthepage}%
12593   \ifdefglxtr@label
12594   {\glxtrbookindexfirstmarkfmt{glxtr@label}}%
12595   {}%
12596 }
```

ndexlastmarkfmt

```
12597 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
12598   \glentryname{#1}%
12599 }
```

okindexlastmark

```
12600 \newcommand*{\glxtrbookindexlastmark}{%
12601   \letcs{glxtr@label}{glxtr@idxlastmark@glxtrbookindexthepage}%
12602   \ifdefglxtr@label
12603   {\glxtrbookindexlastmarkfmt{glxtr@label}}%
12604   {}%
12605 }
```

Glossary

First use The first time a glossary entry is used (from the start of the document or after a reset) with one of the following commands: `\gls`, `\Gls`, `\GLS`, `\glspl`, `\Glspl`, `\GLSpl` or `\glsdisp`. *see* **First use flag** & **First use text**

First use flag A conditional that determines whether or not the entry has been used according to the rules of **first use**.

First use text The text that is displayed on **first use**, which is governed by the first and first-plural keys of `\newglossaryentry`. (May be overridden by `\glsdisp`.)

`makeindex` An indexing application.

`xindy` An flexible indexing application with multilingual support written in Perl.

Change History

0.1 (2015-11-22)		\@Glsymbol@: added redefinition	64
General: Initial experimental release	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition	64
\Glsfmtshort: new	301	\@Glstext@: added redefinition	59
\glsfmtshort: new	301	\@Glsuseri@: added redefinition	64
\Glsfmtshortpl: new	302	\@Glsuserii@: added redefinition	65
\glsfmtshortpl: new	301	\@Glsuseriii@: added redefinition	65
short: switched inline full form to short		\@Glsuseriv@: added redefinition	65
(long)	205	\@Glsuserv@: added redefinition	66
0.3 (2015-12-02)		\@Glsuservi@: added redefinition	66
\@ACRlong: added redefinition	69	\@acrlong: added redefinition	68
\@ACRlongpl: added redefinition	70	\@acrlongpl: added redefinition	69
\@ACRshort: added redefinition	67	\@acrshort: added redefinition	66
\@ACRshortpl: added redefinition	68	\@acrshortpl: added redefinition	67
\@Acrlong: added redefinition	69	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition	70	argument	53
\@Acrshort: added redefinition	67	\@glsdescplural@: added redefinition	63
\@Acrshortpl: added redefinition	68	\@glsfirst@: added redefinition	60
\@GLSdesc@: added redefinition	63	\@glsfirstplural@: added redefinition	61
\@GLSdescplural@: added redefinition	63	\@glsplural@: added redefinition	61
\@GLSfirst@: added redefinition	60	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition	62	redefinition	64
\@GLSname@: added redefinition	62	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition	61	new	119
\@GLSSymbol@: added redefinition	64	\@glsxtr@field@linkdefs: new	59
\@GLSSymbolplural@: added		\@glsxtr@insertdots: new	175
redefinition	64	\@print@glossary: added redefinition	116
\@GLStext@: added redefinition	59	\glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition	65	\abbrvdefaultfont	180
\@GLSuserii@: added redefinition	65	\glsaccessdesc: new	142
\@GLSuseriii@: added redefinition	65	\glsaccessdescplural: new	143
\@GLSuseriv@: added redefinition	66	\glsaccessfirst: new	140
\@GLSuserv@: added redefinition	66	\glsaccessfirstplural: new	141
\@GLSuservi@: added redefinition	66	\Glsaccesslong: new	145
\@Glsdesc@: added redefinition	63	\glsaccesslong: new	145
\@Glsdescplural@: added redefinition	63	\glsaccessname: new	138
\@Glsfirst@: added redefinition	60	\glsaccessplural: new	139
\@Glsfirstplural@: added redefinition	61	\Glsaccessshort: new	144
\@Glsname@: added redefinition	62	\glsaccessshort: new	144
\@Glsplural@: added redefinition	61	\Glsaccessshortpl: new	144

\glsaccesssshortpl: new	144	\cGLSpl: new	94
\glsaccessssymbol: new	141	\cGLSpl@: new	94
\glsaccessssymbolplural: new	142	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new	139	new	89
\glsentryfmt: added check for short ..	53	\cGLS: new	93
\glslongpltok: new	175	\cGLSformat: new	94
\glsshortpltok: new	174	\cGLSpl: new	94
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new	94
name in \setkeys	176	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new	15
for plural	171	\glsenableentrycount: new	89
\GLSxtrlongpl: new	189	\glsfirstabbrvdefaultfont: new ..	180
\Glsxtrlongpl: new	189	\glsfirstlongdefaultfont: new ...	180
\glsxtrlongpl: new	188	\Glsfmtfirst: new	304
\glsxtrNoGlossaryWarning: new	20	\glsfmtfirst: new	304
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new	304
new	171	\glsfmtfirstpl: new	304
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new	303
new	171	\glsfmtplural: new	303
\glsxtrpostlinkendsentence: new ..	171	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new	188	\Glsxtrtitleshort	301
\Glsxtrshortpl: new	187	renamed from \Glsentryfmtshort ..	301
\glsxtrshortpl: new	187	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort	301
\glslabeltok	200	renamed from \glsentryfmtshort ..	301
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok	198	\Glsxtrtitleshortpl	302
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glsentryfmtshortpl	302
redefinition of \acronymtype	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl	301
\Glsxtrshort	301	renamed from	
\glsfmtshort: changed to use		\glsentryfmtshortpl	301
\glsxtrshort	301	\Glsfmttext: new	303
\Glsfmtshortpl: changed to use		\glsfmttext: new	302
\glsxtrshortpl	302	\glshasattribute: new	150
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	150
\glsxtrshortpl	301	\glsxtremsuffix: new	241
\glsxtrifemptyglossary: new	25	\GlsXtrEnableEntryCounting: new ..	88
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new	91
argument	154	\glsxtrscfont: new	212
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new	212
argument	153	\glsxtrsmfont: new	227
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new	227
default type to \acronymtype	102	short-em: new	248
\newterm: fixed name argument	153	short-em-desc: new	249
0.5 (2015-12-07)		short-em-footnote: new	259
\@cGLS: new	93	short-em-long: new	244
\@cGLS@: new	94	short-em-long-desc: new	246

short-em-postfootnote: new	260	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new	223	attribute	292
short-sc-postfootnote: new	225	\Glsxtrheadtext: now uses headuc	
short-sm: new	230	attribute	294
short-sm-desc: new	232	\glxtrheadtext: now uses headuc	
short-sm-footnote: new	237	attribute	294
short-sm-long: new	229	short-em-footnote: switch off regular	
short-sm-long-desc: new	230	attribute if set	259
short-sm-postfootnote: new	239	short-long: switch off regular attribute	
long-noshort-em: new	251	if set	199
long-noshort-em-desc: new	255	short-long-desc: switch off regular	
long-noshort-sm: new	234	attribute if set	200
long-noshort-sm-desc: new	235	short-sc-footnote: switch off regular	
long-short-em: new	241	attribute if set	223
long-short-em-desc: new	242	short-sm-footnote: switch off regular	
long-short-sm: new	227	attribute if set	237
long-short-sm-desc: new	228	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set	197
\Glsaccessstext: new	139	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set	198
\@glxtr@doaccsupp: new	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	291	attribute if set	214
\Glsaccessdesc: new	143	footnote: switch off regular attribute if	
\Glsaccessdescplural: new	143	set	201
\Glsaccessfirst: new	140	postfootnote: switch off regular	
\Glsaccessfirstplural: new	141	attribute if set	203
\Glsaccessname: new	139	0.5.2 (2015-12-08)	
\Glsaccessplural: new	140	\@GLSdesc@: added accessibility support	63
\Glsaccesssymbol: new	141	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new	142	support	63
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute	296	support	60
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute	296	support	62
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	62
headuc attribute	297	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support	61
headuc attribute	296	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support	64
attribute	295	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support	64
attribute	295	\@GLStext@: added accessibility support	59
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	63
attribute	292	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support	63
attribute	292	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support	60
attribute	293	\@GLSfirstplural@: added accessibility	
		support	61

\@Glsname@: add accessibility support ..	62	\GLSaccesssymbolplural: new ..	142, 147
\@Glsplural@: added accessibility support	61	\GLSaccessstext: new	139, 146
\@Glsymbol@: added accessibility support	64	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support	64	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	59	\glsxtrabbrvfmt to here	53
\@glsdesc@: added accessibility support	62	\GlsXtrEnableInitialTagging: new	168
\@glsdescplural@: added accessibility support	63	\glsxtrfieldtitlecase: new	154
\@glsfirst@: added accessibility support	60	\GlsXtrFormatLocationList: new ...	50
\@glsfirstplural@: added accessibility support	61	\glsxtrnewabbrevpresetkeyhook:	
\@Glsname@: added accessibility support	62	new	178
\@Glsplural@: added accessibility support	61	\glsxtrtagfont: new	169
\@Glsymbol@: added accessibility support	63	\KV@printgloss@nonumberlist: added	52
\@Glsymbolplural@: added accessibility support	64	\mfu@checkword@do: added	168
\@Glstext@: added accessibility support	59	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch	193
new	169	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	169	\@glsxtr@autoindex@at: new	165
\@glsxtr@tag: new	169	\@glsxtr@autoindex@encap: new ...	165
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new	166
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	166
\@ifpackageloaded	138	\@glsxtr@autoindex@setname: new ..	164
removed \glsxtrabbrvfmt	190	\@glsxtr@doabbreviationsdef: new ..	16
\glossaryentrynumbers: added	50	General: removed	
\Glossentrydesc: added	167	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added	159	118
\Glossentrysymbol: added	167	\glsdescwidth: added	49
\glossentrysymbol: added	167	\glspagelistwidth: added	50
\GLSaccessdesc: new	143, 148	\glsxtrdoautoindexname: new	163
\GLSaccessdescplural: new ...	143, 148	\glsxtrpostnamehook: new	160
\GLSaccessfirst: new	140, 147	\if@glsxtr@format@override: new ..	162
\GLSaccessfirstplural: new ..	141, 147	\ProvidesGlossariesExtraLang: new	307
\GLSaccesslong: new	145, 148	\RequireGlossariesExtraLang: new	307
\GLSaccesslongpl: new	145, 149	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new	145	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new	145	new	95
\GLSaccessname: new	139, 146	\@GLSxtr@p@acrlong@: new	82
\GLSaccessplural: new	140, 146	\@GLSxtr@p@acrlongpl@: new	82
\GLSaccesssshort: new	144, 148	\@GLSxtr@p@acrshort@: new	82
\GLSaccesssshortpl: new	145, 148	\@GLSxtr@p@acrshortpl@: new	82
\GLSaccesssymbol: new	142, 147	\@GLSxtr@p@long@: new	81
		\@GLSxtr@p@longpl@: new	82
		\@GLSxtr@p@plural@: new	80
		\@GLSxtr@p@short@: new	81
		\@GLSxtr@p@shortpl@: new	81
		\@GLSxtr@p@text@: new	80
		\@GlsXtrEnableOnTheFly: new	46
		\@Glsxtr: new	47
		\@Glsxtr@p@acrlong@: new	82
		\@Glsxtr@p@acrlongpl@: new	82

\@Glsxtr@p@acrshort@: new	82	\glsenableentryunitcount: new	97
\@Glsxtr@p@acrshortpl@: new	82	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	81	entry's existence	150
\@Glsxtr@p@longpl@: new	81	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	80	entry's existence	151
\@Glsxtr@p@short@: new	80	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	81	check	170
\@Glsxtr@p@text@: new	80	\Glsxtr: new	46
\@Glsxtrpl: new	47	\glxtr: new	46
\@alt@glshyp@opt: new	76	\glxtrcat: new	46
\@glscalt@hyp@opt: new	76	\glxtrdowrglossaryhook: new	76
\@glscalt@hyp@opt@char: new	76	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	76	new	100
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	45
new	96	\Glsxtrpl: new	47
\@glsc@local@increment@currunitcount:		\glxtrpl: new	47
new	96	\glxtrpostlocalreset: new	88
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	88
new	73	\glxtrpostreset: new	88
\@glsxtr: new	46	\glxtrpostunset: new	88
\@glsxtr@addunitcounter: new	95	\glxtrprotectlinks: new	79
\@glsxtr@currunitcount: new	97	\GlsXtrSetAltModifier: new	76
\@glsxtr@ifunitcounter: new	95	\GlsXtrSetDefaultGlsOpts: new	75
\@glsxtr@p@acrlong@: new	82	\glxtrstarflywarn: new	46
\@glsxtr@p@acrlongpl@: new	82	\GlsXtrWarning: new	48
\@glsxtr@p@acrshort@: new	82	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	82	disables \setacronymstyle	102
\@glsxtr@p@long@: new	81	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	81	\@glsxtr@autoindexcrossrefs: new	15
\@glsxtr@p@plural@: new	80	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	80	new	110
\@glsxtr@p@shortpl@: new	81	\@glsxtr@idx@entrynumberlist: new	112
\@glsxtr@p@text@: new	80	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	97	new	110
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	100	new	111
\@glsxtr@unitcountlist: new	95	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	47	new	111
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	103
empty see value if not set and added		\makeglossaries: new	104
'see' to field key map	37	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	45	\glxtrdiscardperiod: added check	
\cGlsformat: added	94	for first use	171
\cGlsformat: added	94	short-desc: fixed typo in	
\cGlsplformat: added	95	\glxtrinlinefullformat and	
\cGlsplformat: added	94	added missing second argument	206
\glstdisablehyper: added	78	1.02 (2016-04-25)	
\glstdohyperlink: added	77	\@glsxtr@current@style: new	48
\glstdonohyperlink: added	79	\Glsfmtfull: new	306

\glsfmtfull: new	306	\@GLSdescplural@: set abbreviation and regular format	63
\Glsfmtfullpl: new	307	\@GLSfirst@: set abbreviation format ..	60
\glsfmtfullpl: new	306	\@GLSfirstplural@: set abbreviation and regular format	62
\Glsfmtlong: new	305	\@GLSname@: set abbreviation and regular format	62
\glsfmtlong: new	305	\@GLSplural@: set abbreviation and regular format	61
\Glsfmtlongpl: new	306	\@GLSsymbol@: set regular format	64
\glsfmtlongpl: new	305	\@GLSsymbolplural@: set regular format	64
\Glsxtrheadfull: new	300	\@GLStext@: set abbreviation and regular format	59
\glsxtrheadfull: new	299	\@GLSuseri@: set regular format	65
\Glsxtrheadfullpl: new	300	\@GLSuserii@: set regular format	65
\glsxtrheadfullpl: new	299	\@GLSuseriii@: set regular format	65
\Glsxtrheadlong: new	298	\@GLSuseriv@: set regular format	66
\glsxtrheadlong: new	297	\@GLSuservi@: set regular format	66
\Glsxtrheadlongpl: new	299	\@GLSdesc@: set abbreviation and regular format	63
\glsxtrheadlongpl: new	298	\@GLSdescplural@: set abbreviation and regular format	63
\Glsxtrtitlefull: new	300	\@GLSfirst@: set abbreviation and regular format	60
\glsxtrtitlefull: new	299	\@GLSfirstplural@: set abbreviation and regular format	61
\Glsxtrtitlefullpl: new	301	\@GLSname@: set abbreviation and regular format	62
\glsxtrtitlefullpl: new	300	\@GLSplural@: set abbreviation and regular format	61
\Glsxtrtitlelong: new	298	\@GLsymbol@: set regular format	64
\glsxtrtitlelong: new	298	\@GLsymbolplural@: set regular format	64
\Glsxtrtitlelongpl: new	299	\@GLstext@: set abbreviation and regular format	59
\glsxtrtitlelongpl: new	298	\@GLSuseri@: set regular format	64
\ifglsxtrinsertinside: new	196	\@GLSuserii@: set regular format	65
postfootnote: added redef of		\@GLSuseriii@: set regular format	65
\glsxtrsetupfulldefs	203	\@GLSuseriv@: set regular format	65
stylemods: new	21	\@GLSuservi@: set regular format	66
1.03 (2016-04-27)		\@GLSdesc@: set abbreviation and regular format	63
\@GLSfirstplural@: bug fix: misspelt cs name	62	\@GLSdescplural@: set abbreviation and regular format	61
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	61	\@GLsymbol@: set regular format	64
\@GLSfirstplural@: bug fix: misspelt cs name	61	\@GLsymbolplural@: set regular format	64
\@GLsplural@: fixed bug \@GLsplural@ should be redefined not \@GLsplural	61	\@GLstext@: set abbreviation and regular format	59
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	61	\@GLSuseri@: set regular format	64
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	298	\@GLSuserii@: set regular format	65
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	292	\@GLSuseriii@: set regular format	65
1.04 (2015-04-30)		\@GLSuseriv@: set regular format	65
short-em-footnote: renamed from “footnote-em”	259	\@GLSuservi@: set regular format	66
1.04 (2016-05-02)		\@GLSdesc@: set abbreviation and regular format	62
\@glsxtrpostloctag: new	52	\@GLSdescplural@: set abbreviation and regular format	63
\@GLSdesc@: set abbreviation and regular format	63	\@GLSfirst@: set abbreviation and regular format	60

\@glsfirstplural@: set abbreviation and regular format	61	\glsxtruserparen: new	262
\@glsname@: set abbreviation and regular format	62	\glsxtrusersuffix: new	263
\@glsplural@: set abbreviation and regular format	61	\GlsXtrWarnDeprecatedAbbrStyle: new	195
\@glsymbol@: set regular format	63	short-em-long-em: new	246
\@glsymbolplural@: set regular format	64	short-em-long-em-desc: new	247
\@glstext@: set abbreviation and regular format	59	short-em-nolong: new	249
\@glsxtr@deprecated@abbrstyle: new	195	short-em-nolong-desc: new	251
\@glsxtr@do@style: new	21	short-em-postfootnote: renamed from “postfootnote-em”	260
\@glsxtr@doloctag: new	52	short-footnote: new	202
\@glsxtr@idx@entrynumberlist: switched from \let to \newcommand	112	short-long-user: new	269
\@glsxtr@pagetag: new	52	short-long-user-desc: new	271
\@glsxtr@pagetag: new	52	short-nolong: new	206
\@glsxtr@preloctag: new	52	short-nolong-desc: new	208
\@glsxtr@preloctag: new	51	short-postfootnote: new	204
\glossentrydesc: added glossdescfont attribute check	155	short-sc-footnote: renamed from “footnote-sc”	223
\Glossentryname: added glossnamefont attribute check	159	short-sc-nolong: new	217
\glossentryname: added glossnamefont attribute check	157	short-sc-nolong-desc: new	219
moved post name hook inside condition	159	short-sc-postfootnote: renamed from “postfootnote-sc”	225
\glsabbrvmfont: new	241	short-sm-footnote: renamed from “footnote-sm”	237
\glsabbrvuserfont: new	263	short-sm-nolong: new	232
\glsfirstabbrvmfont: new	241	short-sm-nolong-desc: new	233
\glsfirstabbrvuserfont: new	263	short-sm-postfootnote: renamed from “postfootnote-sm”	239
\glsfirstlongemfont: new	241	\letabbreviationstyle: new	195
\glsfirstlonguserfont: new	263	\newabbreviationstyle: bug fix: corrected test for existence	194
\glsifnotregularcategory: new ...	151	long-em-noshort-em: new	253
\glslongdefaultfont: new	180	long-em-noshort-em-desc: new	257
\glslongemfont: new	241	long-em-short-em: new	243
\glslongfont: new	180	long-em-short-em-desc: new	244
\glslonguserfont: new	263	long-noshort: new	212
\glsxtrassignfieldfont: new	59	long-noshort-desc: new	211
\GlsXtrEnablePreLocationTag: new .	51	long-noshort-em: renamed from “long-em”	251
\glsxtrfirstscfont: new	212	long-noshort-em-desc: renamed from “long-desc-em”	255
\glsxtrfirstsmfont: new	227	long-noshort-sc: renamed from “long-sc”	220
\glsxtrlongshortdescsort: new ...	197	long-noshort-sc-desc: renamed from “long-desc-sc”	221
\glsxtrpostnamehook: added category check	160	long-noshort-sm: renamed from “long-sm”	234
\glsxtrregularfont: new	53	long-noshort-sm-desc: renamed from \long-desc-sm	235
\glsxtruserfield: new	262		

long-short-user: new	263	docdef option changed to choice	14
long-short-user-desc: new	269	\glxtr@usesee: new	38
\renewabbreviationstyle: new	194	\glxtrusesee: new	38
style: new	21	\glxtruseseeformat: new	38
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	14
\eglssetwidest: new	326	1.07 (2016-08-15)	
\glsFindWidestAnyName: new	328	\@@glxtrp: new	83
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new	333	nohyperfirst attribute	60
\glsFindWidestAnyNameSymbol: new	331	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute	62
new	332	\@GLSxtrp: new	84
\glsFindWidestLevelTwo: new	329	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new ..	327	nohyperfirst attribute	60
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new	333	nohyperfirst attribute	61
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new	83
new	330	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts	170
new	331	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new .	328	nohyperfirst attribute	60
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new	327	nohyperfirst attribute	61
\glsfirstlongfootnotefont: new ..	200	\@glxtrinmark: new	289
\glsgetwidestname: new	327	\@glxtrnotinmark: new	289
\glsgetwidestsubname: new	327	\@glxtrp: new	83
\glslongfootnotefont: new	200	\@glxtrp@opt: new	82
\glxtrAltTreeIndent: new	326	\glossxtrsetpopts: new	83
\glxtralttreeInit: new	326	\glsps: new	85
\glxtrAltTreePar: new	326	\glspt: new	85
\glxtrAltTreeSetHangIndent: new	334	\glxtr@entry@p: new	84
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new	201
new	335	\glxtrchecknohyperfirst: new	60
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new	155
new	326	\glxtrifinmark: new	289
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new	86
new	325	\Glsxtrp: new	85
\glxtrComputeTreeIndent: new ...	334	\glxtrp: new	84
\glxtrComputeTreeSubIndent: new	334	\glxtrsetpopts: new	83
\glxtrtreetopindent: new	326	short-long-desc: added text key	200
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form	245	plural key	200
\xglsetwidest: new	327	long-short-desc: added missing text	
1.06 (2016-06-18)		key	198
\@glsdoifexistsorwarn: new	14	fixed misspelling of \glsabbrvfont .	198
\@glxtr@docdefval: new	14	footnote: changed first forms to use	
\@glxtr@usesee: new	38	\glsfirstlongfootnotefont ...	201
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document	25	from first keys	202

switched from \glsfirstlongfont to	1.10 (2016-12-17)
\glsfirstlongfootnotefont ... 204	\@GLSp1@: fixed bug caused by typo in
\RestoreAcronyms: modified	command name 54
\@gls@link@checkfirsthyper to	1.11 (2017-01-19)
set \glsxtrifwasfirstuse 103	\@glsxtr@do@redef@for@gl@sentries:
1.08 (2016-12-13)	new 6
\@glsxtr@record: new 8	\@glsxtr@noidx@do: new 129
\@GLS@: added \@glsxtr@record 54	\@glsxtr@redef@for@gl@sentries: new . 6
\@GLSp1@: added \@glsxtr@record ... 54	\@glsxtr@shortcutsval: new 19
\@Gls@: added \@glsxtr@record 54	\@glsxtr@unsrt@getgroup@title: new 128
\@GLspl@: added \@glsxtr@record ... 54	\@print@noidx@glossary: added
\@gls@: added \@glsxtr@record 54	redefinition 113
\@gls@@link@: added	\glsxtr@addloc@listfield: added
\@glsxtr@record 55	group key 12
\@gls@field@link: added	added location key 12
\@glsxtr@record 53	\glsxtr@fields: new 121
\@gls@saveentrycounter: new 25	\glsxtr@linkprefix: new 122
\@glsdisp: added \@glsxtr@record .. 55	\glsxtr@org@new@ignored@glossary:
\@glspl@: added \@glsxtr@record ... 54	new 33
\@glsxtr@dorecord: new 10	\glsxtr@s@new@ignored@glossary: new 33
\@glsxtr@err@undefaction: new 6	\glsxtr@shortcutsval: new 121
\@glsxtr@record: new 7	\glsxtr@texencoding: new 121
\@glsxtr@warn@onexists@do: new ... 6	\glsxtr@writefields: new 122
\@glsxtr@warn@undefaction: new 6	\GlsXtrLoadResources: new 121
\@print@unsrt@glossary: new 126	\glsxtr@resourcefile: changed
General: added record package option ... 12	extension to .glstex 120
\glsadd: added \@glsxtr@record 58	\new@ignored@glossary: added starred
\glsdoifexists: now defines	version 33
\glslabel 36	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 170
new 11	\@glsxtr@counter@recordhook: new . 123
\glsxtr@record: new 123	\@glsxtr@display@loc: new 114
\glsxtr@resource: new 121	\@glsxtr@docounter@record: new ... 123
\glsxtr@saveentrycounter: new 25	\@glsxtr@longnew@glossary@entry:
\glsxtr@setup@record: new 11	new 32
\glsxtr@assignfieldfont: added check	\@glsxtr@noop@recordcounter: new . 11
for existence 59	\@glsxtr@op@recordcounter: new ... 11
\glsxtr@resourcefile: new 120	\@glsxtr@provide@storagekey: new . 26
\printunsrt@glossaries: new 126	\@glsxtr@s@longnew@glossary@entry:
\printunsrt@glossary: new 125	new 32
1.09 (2016-12-16)	\@glsxtre@entryfmt: new 27
\@glsxtr@gettype: new 110	\@glsxtr@index@aliased: new 74
\@glsxtr@mixed@assign@sortkey:	\@glsxtr@set@alias@noindex: new 74
new 110	\@new@glossary@entry@posthook: added
\@print@glossary: redefined to save	check for alias key 42
options 109	\@no@glsxtr@index@aliased: new 74
\glsxtr@make@glossaries: new 110	\@printunsrt@glossary: new 125

General: added target key to printgloss family	109	\GlsXtrLoadResources: removed restriction on only one per document	121
\apptoglossarypreamble: new	31	\glxtrlocrangefmt: new	115
\csGlsXtrLetField: new	30	\glxtrpostlongdescription: new ..	33
\eGlsXtrSetField: new	30	\glxtrprovidestoragekey: new	26
\gGlsXtrSetField: new	30	\GlsXtrRecordCounter: new	123
\glsdohyperlink: added check for alias field	77	\glxtrresourcecount: new	121
\glsnoidxdisplayloc: added redefinition	114	\glxtrresourcefile: added catcode change for @	121
\glissettoctitle: added patch	34	\glxtrsetaliasnoindex: new	74
\glxtr@counterrecord: new	123	\GlsXtrSetField: new	30
\glxtr@langtag: new	121	\glxtrsetfieldifexists: new	29
\glxtr@newabbreviation: new	176	\glxtrunsrtdo: new	129
\glxtr@org@newignoredglossary: Added check for existence	33	\Glsxtrusefield: new	29
\glxtr@pluralsuffixes: new	121	\glxtrusefield: new	29
\glxtr@provideignoredglossary: new	34	short-postlong-user: new	266
\glxtr@s@newignoredglossary: Added check for existence	33	short-postlong-user-desc: new ...	268
\glxtr@s@provideignoredglossary: new	35	\longnewglossaryentry: added starred version	32
\glxtrabbrvpluralsuffix: new ...	180	long-postshort-user: new	264
\glxtralias: new	42	long-postshort-user-desc: new ...	266
\glxtrcopytoglossary: new	36	postdot: new	15
\glxtrdeffield: new	29	\pretoglossarypreamble: new	31
\glxtrdisplayendloc: new	115	\print@noop@unsrtglossaryunit: new	128
\glxtrdisplayendlochook: new ...	115	\print@op@unsrtglossaryunit: new	128
\glxtrdisplaysingleloc: new	114	\printunsrtglossary: added starred form	125
\glxtrdisplaystartloc: new	114	\printunsrtglossaryhandler: new .	127
\glxtrdeffield: new	29	\printunsrtglossaryunit: new	11
\glxtrentryfmt: new	27	\printunsrtglossaryunitsetup: new	128
\glxtrfielddolistloop: new	28	\provideignoredglossary: new	34
\glxtrfieldforlistloop: new	28	\s@glxtr@provide@storagekey: new	26
\glxtrfieldifinlist: new	28	\s@printunsrtglossary: new	126
\glxtrfieldlistadd: new	28	\xGlsXtrSetField: new	30
\glxtrfieldlistead: new	28	1.13 (2017-02-07)	
\glxtrfieldlistgadd: new	28	\@glsdisp: removed	
\glxtrfieldlistxadd: new	28	\@glxtr@org@glsdisp	55
\glxtrfieldxifinlist: new	29	\glxtrsetaliasnoindex: switched to \providecommand	74
\glxtrfmt: new	27	1.14 (2017-04-18)	
\GlsXtrFmtDefaultOptions: new	27	\@gls@link: added redefinition	56
\GlsXtrFmtField: new	27	\@gls@noidx@getgrouptitle: new ..	112
\glxtrifkeydefined: new	26	\@gls@removespaces: new	115
\glxtrindexaliased: new	74	\@glxtr@do@automake@err: new ...	123
\GlsXtrLetField: new	30	\@glxtr@org@gloautosee: new	23
\GlsXtrLetFieldToField: new	30	\@glxtr@record: added third arg	7
		\@glxtr@recordsee: new	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue	58	\@glo@autosee: added redefinition	24
added \glsadd option thevalue	58	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	78	bug	112
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@	90	check for seealso field	43
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@	98	instead of \csname	129
\glsnavigation: new	113	\@glsxtr@dorecordnodefer: new	10
\glsxtr@org@getgrouptitle: new ..	112	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new	7	misspelt command	126
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake	123	new	127
\glsxtrdisplayendloc: added check		General: added check for	
for empty format	115	\@gls@setupsort@none	13
\glsxtrgetgrouptitle: new	112	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new	55	redefinition	24
\glsxtrlocationhyperlink: new ...	115	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new	113	\providecommand lines	122
\glsxtrsupphypernumber: new	116	\@glsxtrautoindex: new	163
\ifglsxtrwrglossbefore: new	55	\@glsxtrautoindexentry: new	164
1.15 (2017-05-10)		\@glsxtrautoindexsort: new	164
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new	39
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new	42
short-em-long-em: fixed spelling of		\@glsxtrseelist: new	39
\glsabbrvfont	246	\@glsxtruseseealso: new	38
short-long: fixed spelling of		\@glsxtruseseealsoformat: new	39
\glsabbrvfont	199	\@seealso: new	39
short-long-user: fixed spelling of		autoseeindex: new	15
\glsabbrvfont	270	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new	175
\glsabbrvfont	267	\@glsxtr@markwordseps: new	175
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont	268	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag	110
\glsabbrvfont	243	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont	264	\glsxtrundeftag	111
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont	266	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag	111
\glsabbrvfont	197	\@glsxtrifhyphenstart: new	272
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont	263	in the abbreviation styles	196
footnote: fixed spelling of		\glsabbrvhyphenfont: new	272
\glsabbrvfont	201	\glsabbrvonlyfont: new	285
postfootnote: fixed spelling of		\glsabbrvscfont: new	212
\glsabbrvfont	203	\glsabbrvsmfont: new	227

\glsabbrvuserfont: initialised to default font	263	short-long-user-desc: corrected first forms	271
\glsfirstabbrvhyphenfont: new	272	short-nolong-desc-noreg: new	208
\glsfirstabbrvonlyfont: new	285	short-nolong-noreg: new	206
\glsfirstabbrvscfont: new	212	long-em-noshort-em-desc-noreg: new	258
\glsfirstabbrvsmfont: new	227	long-em-noshort-em-noreg: new	255
\glsfirstlonghyphenfont: new	272	long-hyphen-noshort-desc-noreg: new	274
\glsfirstlongonlyfont: new	285	long-hyphen-postshort-hyphen: new	278
\glslonghyphenfont: new	272	long-hyphen-postshort-hyphen-desc: new	279
\glslongonlyfont: new	285	long-hyphen-short-hyphen: new	272
\glslonguserfont: initialised to default font	263	long-hyphen-short-hyphen-desc: new	273
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong	176	long-noshort-desc-noreg: new	211
\GlsXtrDefineAcShortcuts: new	18	long-noshort-noreg: new	212
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside	190	long-only-short-only: new	286
\glsxtrhyphensuffix: new	272	long-only-short-only-desc: new	287
\glsxtrifhyphenstart: new	271	long-short-user-desc: corrected first forms	269
\glsxtrlonghyphen: new	277	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new	274	stylemods: changed default value to "default"	21
\glsxtrlonghyphenshort: new	272	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new	198	\@glsxtr@defaultnumberformat: new	7
\glsxtronlydescname: new	287	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref	10
\glsxtronlydescsort: new	287	\@glsxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glslocref	10
\glsxtronlysuffix: new	286	\@glsxtr@record@setting: new	12
\glsxtrparen: new	178	\@glsxtr@record@setting@alsoindex: new	12
\glsxtrposthyphenlong: new	282	\@glsxtrifhasfield: new	29
\glsxtrposthyphenshort: new	277	General: added \glslink option theHvalue	56
\glsxtrposthyphensubsequent: new	277	added \glslink option thevalue	56
\glsxtrshortdescname: new	206	\glsxtr@writefields: removed double-quotes around \jobname	123
\glsxtrshorthyphen: new	282	\glsxtrdoautoindexname: changed format test	163
\glsxtrshorthyphenlong: new	280	\glsxtrhyperlink: new	78
\glsxtrshortlongdescname: new	200	\glsxtrifhasfield: new	29
\glsxtrshortlongdescsort: new	199	\GlsXtrSetDefaultNumberFormat: new	7
\Glsxtrsubsequentfmt: new	193	\s@glsxtrifhasfield: new	29
\glsxtrsubsequentfmt: new	192		
\Glsxtrsubsequentplfmt: new	193		
\glsxtrsubsequentplfmt: new	192		
\glsxtrword: new	175		
\glsxtrwordsep: new	175		
short-hyphen-long-hyphen: new	280		
short-hyphen-long-hyphen-desc: new	281		
short-hyphen-postlong-hyphen: new	283		
short-hyphen-postlong-hyphen-desc: new	285		

headings	339	\glxtrbookindexbetween: new	344
redefined		\glxtrbookindexbookmark: new ...	344
mcoltreenonamehypergroup to		\glxtrbookindexcols: new	343
discourage breaks after group		\glxtrbookindexcolspread: new ..	344
headings	339	\glxtrbookindexfirstmark: new ..	348
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	348
discourage breaks after group		\glxtrbookindexformatheader: new	344
headings	340	\glxtrbookindexgroupskip: new ..	344
redefined mcoltreepannav to		\glxtrbookindexlastmark: new ...	348
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	348
headings	338	\glxtrbookindexmarkentry: new ..	347
redefined treegroup to discourage		\glxtrbookindexname: new	343
breaks after group headings	323	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new	343
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings	324	new	343
redefined treenonamegroup to		\glxtrbookindexprelocation: new	343
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings	325	new	344
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	344
discourage breaks after group		\glxtrbookindexsubname: new	343
headings	325	\glxtrbookindexsubprelocation:	
debug: new	22	new	343
\gglssetwidest: new	326	\glxtrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new	344
existence	78	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	78	new	344
\glsenablehyper: changed to use \def		\glxtrbookindexthepage: new	347
rather than \let	78	\glxtrdetoklocation: new	133
\Glsfmtname: new	302	\glxtrenablerecordcount: new ...	133
\glsfmtname: new	302	\glxtrglossentry: new	124
\glshex: new	131	\glxtrgroupfield: new	129
\glslstchildpostlocation: new ..	311	\Glsxtrheadname: new	294
\glslstchildprelocation: new ...	311	\glxtrheadname: new	293
\glslstprelocation: new	311	\GlsXtrIfFieldEqStr: new	30
\glsnahyperlink: patched	76	\glxtriflabelinlist: new	128
\glssseeitemformat: new	38	\glxtrifrecordtrigger: new	134
\glsshowsorttarget: new	23	\glxtrindexseealso: added check	
\glstreechildprelocation: new ...	321	that the entry exists	39
\glstreeprelocation: new	321	\glxtrtrinityhyperoutside: new	56
\glstriggerrecordformat: new	135	\GlsXtrLocationRecordCount: new ..	133
\glseuseabbrvfont: new	190	\glxtrnewgls: new	131, 132
\glseuselongfont: new	190	\glxtrnewGLSlike: new	132
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new	132
new	8	\glxtrnewrgls: new	132
\glxtr@org@@do@wrglossary: new ..	25	\glxtrnewrGLSlike: new	133
\glxtr@org@dohyperlink: new	76	\glxtrnewrglslike: new	132
\glxtr@setbookindexmark: new ...	348	\glxtrprelocation: new	310, 343
\glxtrbookindexatendgroup: new ..	344	\GlsXtrRecordCount: new	133

\glxtrrecordtriggervalue: new ..	133	\rgls: new	135
\glxtrresourcefile: now disables		\rGLSformat: new	138
record key	120	\rGlsformat: new	138
\glxtrresourceinit: new	131	\rglsformat: new	137
\GlsXtrSetRecordCountAttribute:		\rGLSpl: new	137
new	134	\rGlspl: new	136
\glxtrtrtitlename: new	294	\rglspl: new	135
\glxtrtrtitleorpdforheading: new .	289	\rGLSplformat: new	138
\GlsXtrTotalRecordCount: new	133	\rGlsplformat: new	138
\glxtrwrglossmark: new	22	\rglsplformat: new	137
short-em: new	249	\s@glxtrifhasfield: switched from	
short-sc: corrected first letter		\ifdef to \ifundef	29
uppercasing	217	1.22 (2017-11-08)	
short-sm: corrected first letter		\@glxtr@nopostpunc: new	109
uppercasing	231	\@glxtr@orgprintglossary: changed	
\ifglxtr@hyperoutside: new	56	explicit \let for \nopostdesc to	
all: new	309	\glxtractivatenopost	108
nolong-short: new	208	\@glxtrglossentryother: new	125
nolong-short-em: new	251	\glossentrynameother: new	161
nolong-short-noreg: new	209	\glsseeitemformat: switched check	
nolong-short-sc: new	219	from regular to short	38
nolong-short-sm: new	233	\glxtr@setaccessdisplay: new ...	160
nopostdot: new	15	\glxtr@writefields: provide	
\printunstrtglossaryentryprocesshook:		\glxtr@record in aux file	122
new	127	\glxtractivatenopost: new	108
\printunstrtglossarypredoglossary:		\glxtrbookindexprelocation:	
new	127	removed check for no post dot	343
\rGLS: new	137	\glxtrglossentryother: new	124
\rGls: new	136	\glxtrnopostpunc: new	109

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, 171, 321
\@	121
\@cGLS@	90, 99
\@cGLSpl@	90, 99
\@cGLs@	90, 98
\@cGLspl@	90, 99
\@cgl@	90, 98
\@cglspl@	90, 98
\@do@wrglossary	8, 9, 105
\@do@wrglossary	11, 13, 25, 58, 74
\@glo@assign@sortkey	110
\@glo@list	6
\@glo@type	126
\@glossarysec	344
\@gls@expand@field	26
\@glslocalreset	88
\@glslocalunset	88
\@glsreset	88
\@glsunset	88
\@glsxtr@autoindex@escspch	165–167
\@glsxtr@checkspch	164, 165, 167
\@glsxtr@disabledflycommand	48
\@glsxtr@org@postdescription	109
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 123
\@glsxtrp	83, 84
\@glsxtrpostloctag	51
\@glsxtrpreloctag	51
\@glsxtrwrglossmark	8, 11, 23, 25, 40, 44, 105
\@newglossaryentry@defcounters	89
\@newglossaryentry@defunitcounters	97
\@par	326
\@ACRlong	80
\@ACRlongpl	80
\@ACRshort	79
\@ACRshortpl	80
\@Acrlong	80
\@Acrlongpl	80
\@Acrshort	79
\@Acrshortpl	80
\@GLS@	79, 93, 94, 137
\@GLSdesc@	63
\@GLSpl@	79, 93, 94, 137
\@GLSplural@	80
\@GLSSymbol@	64
\@GLStext@	80
\@GLSxtr@full	182
\@GLSxtr@fullpl	183
\@GLSxtrp@acrlong@	80
\@GLSxtrp@acrlongpl@	80
\@GLSxtrp@acrshort@	79
\@GLSxtrp@acrshortpl@	80
\@GLSxtrp@long@	79
\@GLSxtrp@longpl@	79
\@GLSxtrp@plural@	79
\@GLSxtrp@short@	79
\@GLSxtrp@shortpl@	79
\@GLSxtrp@text@	79
\@GLSxtrlong	79, 186
\@GLSxtrlongpl	79, 189, 190
\@GLSxtrp	87
\@GLSxtrshort	79, 185
\@GLSxtrshortpl	79, 188
\@Gls@	79, 92, 93, 136
\@Gls@acentryname	101
\@Gls@entry@field	71, 86, 162
\@Gls@entryname	101
\@GlsXtrEnableOnTheFly	45
\@Glspl@	79, 92, 93, 136
\@Glsplural@	80
\@Glstext@	80
\@Glsxtr	46, 48
\@Glsxtr@full	181

\@Glsxtr@fullpl	183	\@firstofthree	55,
\@Glsxtr@p@acrlong@	80	59, 67–70, 76, 181, 182, 184, 185, 187, 189	
\@Glsxtr@p@acrlongpl@	80	\@firstoftwo ..	60–64, 68, 70, 73, 76, 103,
\@Glsxtr@p@acrshort@	79	161, 172, 173, 181–183, 187–190, 289, 290	
\@Glsxtr@p@acrshortpl@	80	\@for	6, 21, 43, 89,
\@Glsxtr@p@long@	79	101, 104, 107, 113, 126, 134, 154, 161, 168	
\@Glsxtr@p@longpl@	79	\@glo@alias	40, 41
\@Glsxtr@p@plural@	79	\@glo@assign@sortkey	107
\@Glsxtr@p@short@	79	\@glo@autosee	23
\@Glsxtr@p@shortpl@	79	\@glo@autoseehook	41
\@Glsxtr@p@text@	79	\@glo@category	95
\@Glsxtrlong	79, 186	\@glo@check@sortallowed	107
\@Glsxtrlongpl	79, 189	\@glo@counterprefix	10, 115
\@Glsxtrp	86	\@glo@countunit	95
\@Glsxtrpl	47, 48	\@glo@default@sorttype	107
\@Glsxtrshort	79, 184	\@glo@desc	32
\@Glsxtrshortpl	79, 187	\@glo@descplural	32
\@acrlong	80	\@glo@group	12
\@acrlongpl	80	\@glo@label	
\@acrshort	79	. 11, 12, 26, 37, 38, 41–43, 71, 78, 327–334	
\@acrshortpl	80	\@glo@location	12
\@afterheading		\@glo@loclist	11
..... 312, 313, 322–325, 337–340, 347		\@glo@name	163, 164
\@alt@glshyp@opt	76	\@glo@no@assign@sortkey	110
\@auxout	10, 11, 44, 52,	\@glo@parent	329, 330
91, 99, 104, 105, 116, 117, 120, 122–124, 347		\@glo@see	37–39, 41–43
\@bibgl@restoreat	121	\@glo@seealso	40, 41
\@cGLS	93	\@glo@sort	164
\@cGLS@	90, 93, 99	\@glo@sorttype	107, 113
\@cGLSpl	94	\@glo@text	55
\@cGLSpl@	90, 94, 99	\@glo@thislettergrp	129
\@cGls@	90, 98	\@glo@thisvalue	263
\@cGlspl@	90, 99	\@glo@tmp	26, 39, 71
\@cgl@	90, 98	\@glo@type	42, 77, 101, 104, 107,
\@cglspl@	90, 98	108, 110, 113, 114, 116, 117, 119, 120, 126	
\@disable@onlypremakeg	105	\@glo@types	152, 153, 327–333
\@do@auxoutstuff	116, 117	\@glossary@default@style .	48, 49, 108, 342
\@do@gl@getcounterprefix	10	\@glossarystyle	108
\@do@gl@see	41, 42	\@gl@	79, 92, 93, 135
\@do@newglossaryentry	101, 102, 178	\@gl@@link	55
\@do@seeglossary	13, 23, 44, 105	\@gl@ReturnAfterFi	115
\@do@wrglossary	57, 58, 135	\@gl@actualchar	164
\@empty	59, 67–70, 164, 165, 181–190	\@gl@adjustmode	58
\@end@glxtr@addunused	43	\@gl@alt@hyp@opt	76
\@end@glxtr@gettype	107, 110	\@gl@alt@hyp@opt@char	76
\@end@glxtr@usesee	38	\@gl@alt@hyp@opt@keys	76
\@end@glxtrifhyphenstart	271, 272	\@gl@automake	107
\@endfortrue	161, 193	\@gl@between	113
\@firstofone	59, 126, 155, 156, 163, 168	\@gl@checkedmkidx	164, 165, 167

<code>\@gls@checkmkidxchars</code>	40, 164	<code>\@gls@noref@warn</code>	105, 114
<code>\@gls@codepage</code>	117	<code>\@gls@org@glsnoidxdisplayloc</code>	111
<code>\@gls@counter</code>	9, 10, 56, 58, 74, 134	<code>\@gls@org@glsseeformat</code>	111
<code>\@gls@currentlettergroup</code> ...	113, 126, 129	<code>\@gls@preglossaryhook</code>	108, 168
<code>\@gls@declareoption</code>	5	<code>\@gls@prevlevel</code>	335, 336, 340, 341
<code>\@gls@default@longpl</code>	176, 177	<code>\@gls@quotechar</code>	164
<code>\@gls@doautomake</code>	107, 123	<code>\@gls@reference</code>	44, 104, 105
<code>\@gls@doautomake@err</code>	123	<code>\@gls@saveentrycounter</code> .	13, 25, 57, 58, 134
<code>\@gls@encapchar</code>	165	<code>\@gls@see@noindex</code>	24, 120, 121
<code>\@gls@entry@count</code>	90, 91	<code>\@gls@setdefault@glslink@opts</code> .	9, 57, 75
<code>\@gls@entry@field</code>		<code>\@gls@setsort</code>	57
.....	26, 29, 41, 71, 84–87, 90, 125	<code>\@gls@setupsort@none</code>	13
<code>\@gls@entry@unitcount</code>	99	<code>\@gls@short</code>	177
<code>\@gls@field@font</code>	59–66	<code>\@gls@shortpl</code>	174, 177, 178
<code>\@gls@field@link</code>	59–66, 71, 72	<code>\@gls@sort</code>	129
<code>\@gls@getcounterprefix</code>	10	<code>\@gls@thisval</code>	161
<code>\@gls@getgrouptitle</code>	112, 126	<code>\@gls@tmp</code>	113
<code>\@gls@grptitle</code>	77, 113	<code>\@gls@tmpb</code>	167
<code>\@gls@hyp@opt</code>		<code>\@gls@type</code>	105, 107, 193, 327–334
. 71, 72, 76, 93, 94, 131, 135–137, 180–189		<code>\@gls@write@entrycounts</code>	90
<code>\@gls@hyp@opt@cs</code>	76	<code>\@gls@write@entryunitcounts</code>	99
<code>\@gls@ifinlist</code>	128	<code>\@gls@write@entryunitcounts@do</code> ...	100
<code>\@gls@increment@currcount</code>	90	<code>\@gls@xref</code>	11, 40
<code>\@gls@increment@currunitcount</code>	98	<code>\@gls@abbrv@current@abbreviation</code> 176, 190	
<code>\@gls@keymap</code> .. 11, 12, 26, 38, 40, 71, 122, 161		<code>\@gls@acronymlists</code>	101
<code>\@gls@label</code> ... 8–10, 44, 75, 76, 105, 124, 193		<code>\@gls@doifexistsorwarn</code>	14, 157–161
<code>\@gls@levelchar</code>	164	<code>\@gls@entry</code>	91, 99, 100
<code>\@gls@link</code>	27, 53, 55, 67–70, 181–190	<code>\@gls@link</code>	57, 77–79
<code>\@gls@link@checkfirsthyper</code>	55, 103	<code>\@gls@nextpages</code>	108
<code>\@gls@link@label</code>	56, 134	<code>\@gls@nonextpages</code>	108
<code>\@gls@link@nocheckfirsthyper</code>		<code>\@gls@numberformat</code>	
.....	53, 66–70, 181–190	9, 10, 56, 58, 74, 134, 160, 163
<code>\@gls@link@opts</code>	56	<code>\@gls@order</code>	104
<code>\@gls@list</code>	113	<code>\@gls@pl@</code>	79, 92, 93, 136
<code>\@gls@local@increment@currcount</code>	90	<code>\@gls@plural@</code>	80
<code>\@gls@local@increment@currunitcount</code> 98		<code>\@gls@punc@token</code>	173
<code>\@gls@location</code>	130	<code>\@gls@recordlocref</code>	10
<code>\@gls@loclist</code>	110, 111, 130	<code>\@gls@showtarget</code>	78
<code>\@gls@long</code>	176	<code>\@gls@style@altlist</code>	311
<code>\@gls@longpl</code>	174, 176–178	<code>\@gls@style@altlistgroup</code>	312
<code>\@gls@map</code>	161	<code>\@gls@style@altlisthypergroup</code>	313
<code>\@gls@nohyperlist</code>	33, 35	<code>\@gls@style@alttree</code>	325
<code>\@gls@noidx@do</code>	113	<code>\@gls@style@alttreegroup</code>	336
<code>\@gls@noidx@getgrouptitle</code>	126	<code>\@gls@style@alttreehypergroup</code>	336
<code>\@gls@noidx@nosanitizesort</code>	106	<code>\@gls@style@index</code>	321
<code>\@gls@noidx@sanitizesort</code>	106	<code>\@gls@style@indexgroup</code>	322
<code>\@gls@noidx@loclist@finalsep</code>	110	<code>\@gls@style@indexhypergroup</code>	322
<code>\@gls@noidx@loclist@prev</code>	110	<code>\@gls@style@inline</code>	321
<code>\@gls@noidx@loclist@sep</code>	110	<code>\@gls@style@list</code>	311

\@glsstyle@listdotted	310	\@glsxtr@bookindex@atsubendgroup ..	346
\@glsstyle@listgroup	312	\@glsxtr@bookindex@atsubsubendgroup	346
\@glsstyle@listhypergroup	312	\@glsxtr@bookindex@between	345, 347
\@glsstyle@mcolalttree	340	\@glsxtr@bookindex@sep	345, 346
\@glsstyle@mcolalttreegroup	340	\@glsxtr@bookindex@subatendgroup ..	
\@glsstyle@mcolalttreehypergroup ..	341	345, 347
\@glsstyle@mcolalttreespannav	341	\@glsxtr@bookindex@subbetween .	345, 346
\@glsstyle@mcolindexgroup	337	\@glsxtr@bookindex@subsep	345, 346
\@glsstyle@mcolindexhypergroup	337	\@glsxtr@bookindex@subsubatendgroup	
\@glsstyle@mcolindexspannav	337	345, 347
\@glsstyle@mcoltreegroup	338	\@glsxtr@bookindex@subsubbetween ..	
\@glsstyle@mcoltreehypergroup	338	345, 346
\@glsstyle@mcoltreenamegroup	339	\@glsxtr@bookindexgroupskip ...	345, 347
\@glsstyle@mcoltreenamehypergroup	339	\@glsxtr@cat	89, 101, 134, 168
\@glsstyle@mcoltreenamepannav ..	340	\@glsxtr@checkgroup	127
\@glsstyle@mcoltreespannav	338	\@glsxtr@counterrecordhook	10, 11
\@glsstyle@tree	323	\@glsxtr@csname	96, 98
\@glsstyle@treegroup	323	\@glsxtr@current@style	49, 342
\@glsstyle@treehypergroup	324	\@glsxtr@currentunitcount	96, 98
\@glsstyle@treenoname	324	\@glsxtr@currunitcount	97, 99
\@glsstyle@treenonamegroup	325	\@glsxtr@declareoption	5, 15, 17, 20
\@glsstyle@treenonamehypergroup ...	325	\@glsxtr@defaultnoglossarywarning ..	20
\@glstarget	78, 79, 109	\@glsxtr@defaultnumberformat	
\@glstext@	80	7, 9, 56, 58, 74, 160, 163
\@glswidestname	327, 334	\@glsxtr@defpostpunc	15, 16, 23
\@glsxtr	46, 48	\@glsxtr@deprecated@abbrstyle	
\@glsxtr@@do@wrglossary	105	221, 223, 225,
\@glsxtr@abbreviationsdef	17, 24	226, 235, 237, 239, 240, 253, 257, 260, 262	
\@glsxtr@accessdisplay	161, 162	\@glsxtr@disabledflycommand	48
\@glsxtr@activate@initialtagging ..		\@glsxtr@display@loc	114
.....	168, 169	\@glsxtr@do@@wrintex	75
\@glsxtr@addunitcounter	95	\@glsxtr@do@glstdisablehyperinlist ..	73
\@glsxtr@addunused	43	\@glsxtr@do@record@wrglossary	8, 13
\@glsxtr@addunusedxrefs	43	\@glsxtr@do@redef@forglsentries	7
\@glsxtr@attrval	57, 155–161, 163	\@glsxtr@do@style	21, 308
\@glsxtr@autoindex@at	164, 165	\@glsxtr@do@titlecaps@warn	
\@glsxtr@autoindex@doextra@esc	164	155–158, 162, 169
\@glsxtr@autoindex@encap	163, 165	\@glsxtr@doabbreviationsdef	17
\@glsxtr@autoindex@esc	164, 166, 167	\@glsxtr@doaccsupp	20, 22
\@glsxtr@autoindex@escat	164, 165	\@glsxtr@docdefval	14, 45
\@glsxtr@autoindex@escencap	165	\@glsxtr@docounterrecord	11
\@glsxtr@autoindex@esclevel ...	164, 166	\@glsxtr@doglossary	126, 127
\@glsxtr@autoindex@escquote ...	164, 166	\@glsxtr@doiflabelinlist	128
\@glsxtr@autoindex@level	164–166	\@glsxtr@doloctag	51
\@glsxtr@autoindex@setname	163	\@glsxtr@dorecord	8, 9
\@glsxtr@autoindexcrossrefs	13, 14, 38, 41	\@glsxtr@dorecordnodefer	8, 9
\@glsxtr@autoseeindexfalse	13	\@glsxtr@dosee@alsoindex@glossary ..	13
\@glsxtr@autoseeindextrue	15	\@glsxtr@doseeglossary	13, 23
\@glsxtr@bookindex@atendgroup .	345, 347	\@glsxtr@dostylewarn	193

<code>\@glstr@enabletagging</code>	168	<code>\@glstr@org@starttoc</code>	289
<code>\@glstr@end@</code>	45	<code>\@glstr@org@GLS@</code>	54
<code>\@glstr@endescspch</code>	164–167	<code>\@glstr@org@GLSpl@</code>	54
<code>\@glstr@entrycount@org@localreset</code>	90	<code>\@glstr@org@Gls@</code>	54
<code>\@glstr@entrycount@org@localunset</code>	90	<code>\@glstr@org@Glspl@</code>	54
<code>\@glstr@entrycount@org@reset</code>	90	<code>\@glstr@org@Glsxtrtitlefirst</code>	290, 291
<code>\@glstr@entrycount@org@unset</code>	90	<code>\@glstr@org@Glsxtrtitlefirstplural</code>	
<code>\@glstr@entryunitcount@org@localreset</code>			290, 291
<code>\@glstr@entryunitcount@org@localunset</code>	98	<code>\@glstr@org@Glsxtrtitlefull</code>	290, 291
<code>\@glstr@entryunitcount@org@reset</code>	98	<code>\@glstr@org@Glsxtrtitlefullpl</code>	290, 291
<code>\@glstr@entryunitcount@org@unset</code>	98	<code>\@glstr@org@Glsxtrtitlelong</code>	290, 291
<code>\@glstr@err@undefaction</code>	7, 13	<code>\@glstr@org@Glsxtrtitlelongpl</code>	290, 291
<code>\@glstr@field@linkdefs</code>	53	<code>\@glstr@org@Glsxtrtitlename</code>	290, 291
<code>\@glstr@format@overridefalse</code>	162	<code>\@glstr@org@Glsxtrtitleplural</code>	290, 291
<code>\@glstr@format@overridetrue</code>	163	<code>\@glstr@org@Glsxtrtitleshort</code>	290, 291
<code>\@glstr@foundinlist</code>	173	<code>\@glstr@org@Glsxtrtitleshortpl</code>	290, 291
<code>\@glstr@full</code>	180	<code>\@glstr@org@Glsxtrtitletext</code>	290, 291
<code>\@glstr@fullpl</code>	182	<code>\@glstr@org@MakeUppercase</code>	290, 291
<code>\@glstr@gettype</code>	107	<code>\@glstr@org@checkfirsthyper</code>	72, 103
<code>\@glstr@glossdescfont</code>	155, 156	<code>\@glstr@org@delimN</code>	51, 52
<code>\@glstr@glossnamefont</code>	157–162	<code>\@glstr@org@delimR</code>	51, 52
<code>\@glstr@gobbleto@endescspch</code>	167	<code>\@glstr@org@doseeglossary</code>	23, 105
<code>\@glstr@groupheading</code>	127, 129	<code>\@glstr@org@gloautosee</code>	24
<code>\@glstr@idx@displaynumberlist</code>	106	<code>\@glstr@org@gls@</code>	54
<code>\@glstr@idx@entrynumberlist</code>	106	<code>\@glstr@org@glsdohypertarget</code>	109
<code>\@glstr@ifcsstart</code>	45	<code>\@glstr@org@glsignore</code>	51, 52
<code>\@glstr@ifpunctoken</code>	173	<code>\@glstr@org@glspl@</code>	54
<code>\@glstr@ifunitcounter</code>	95	<code>\@glstr@org@glsxtrtitlefirst</code>	290, 291
<code>\@glstr@insert@dots</code>	175	<code>\@glstr@org@glsxtrtitlefirstplural</code>	
<code>\@glstr@insert@dots@next</code>	175		290, 291
<code>\@glstr@insert@dots</code>	177	<code>\@glstr@org@glsxtrtitlefull</code>	290, 291
<code>\@glstr@label</code>	43, 154	<code>\@glstr@org@glsxtrtitlefullpl</code>	290, 291
<code>\@glstr@loadstyles</code>	309, 310	<code>\@glstr@org@glsxtrtitlelong</code>	290, 291
<code>\@glstr@longnewglossaryentry</code>	32	<code>\@glstr@org@glsxtrtitlelongpl</code>	290, 291
<code>\@glstr@mark@wordseps</code>	175	<code>\@glstr@org@glsxtrtitlename</code>	290, 291
<code>\@glstr@mark@wordseps@next</code>	176	<code>\@glstr@org@glsxtrtitleorpdforheading</code>	
<code>\@glstr@markwordseps</code>	176–178		290, 291
<code>\@glstr@mixed@assign@sortkey</code>	107	<code>\@glstr@org@glsxtrtitleplural</code>	290, 291
<code>\@glstr@noidx@displaynumberlist</code>	106	<code>\@glstr@org@glsxtrtitleshort</code>	290, 291
<code>\@glstr@noidx@do</code>	129	<code>\@glstr@org@glsxtrtitleshortpl</code>	290, 291
<code>\@glstr@noidx@entrynumberlist</code>	106	<code>\@glstr@org@glsxtrtitletext</code>	290, 291
<code>\@glstr@noidx@numberlistloop</code>	106	<code>\@glstr@org@makeglossaries</code>	104
<code>\@glstr@noop@recordcounter</code>	10, 13	<code>\@glstr@org@markboth</code>	289
<code>\@glstr@nopostpunc</code>	108	<code>\@glstr@org@markright</code>	288, 289
<code>\@glstr@notfoundinlist</code>	173	<code>\@glstr@org@newacronymstyle</code>	102, 103
<code>\@glstr@op@recordcounter</code>	13	<code>\@glstr@org@postdescription</code>	170
<code>\@glstr@optlist</code>	48	<code>\@glstr@org@see@noindex</code>	120, 121
		<code>\@glstr@org@setacronymstyle</code>	102, 103
		<code>\@glstr@org@theHvalue</code>	8, 9

\@glxstr@orgprefix	10	\@glxstr@unsrt@getgrouptitle	126
\@glxstr@orgprintglossary	48, 109	\@glxstr@usesee	38
\@glxstr@orgwarndep	174	\@glxstr@warn@onexistsordo	7, 13
\@glxstr@p@acrlong@	80	\@glxstr@warn@undefaction	7, 13
\@glxstr@p@acrlongpl@	80	\@glxstrdocdeffalse	44
\@glxstr@p@acrshort@	79	\@glxstrentryfmt	27
\@glxstr@p@acrshortpl@	80	\@glxstrglossentry	124
\@glxstr@p@long@	79	\@glxstrglossentryother	125
\@glxstr@p@longpl@	79	\@glxstrhypernameprefix	109, 128
\@glxstr@p@plural@	79	\@glxstrifhasfield	29
\@glxstr@p@short@	79	\@glxstrifhyphenstart	271
\@glxstr@p@shortpl@	79	\@glxstrindexaliased	74
\@glxstr@p@text@	79	\@glxstrindexcrossrefsfalse	14
\@glxstr@pagetag	51	\@glxstrindexcrossrefstrue	15
\@glxstr@pagetag	51	\@glxstrinmark	288, 289
\@glxstr@prevunitcount	97	\@glxstrlong	79, 185
\@glxstr@printglossopts	48, 107, 109	\@glxstrlongpl	79, 188, 189
\@glxstr@printunsrtglossaryskipentry	126, 127	\@glxstrnewgls	132, 133
\@glxstr@provide@addstoragekey	27	\@glxstrnewgls@inner	131
\@glxstr@provide@storagekey	26	\@glxstrnewgls@innercsname	131, 132
\@glxstr@record	13, 53–55, 58	\@glxstrnotinmark	288, 289
\@glxstr@record@setting 8, 9, 12, 39, 44, 104		\@glxstrp	85
\@glxstr@record@setting@alsoindex	8, 9, 39, 104	\@glxstrp@opt	83
\@glxstr@record@setting@off	44	\@glxstrpl	47, 48
\@glxstr@record@setting@only	104	\@glxstrpostloctag	50–52
\@glxstr@recordsee	13, 23, 40	\@glxstrpreloctag	50–52
\@glxstr@redef@forglsentries	7, 24	\@glxstrsetaliasnoindex	73–75
\@glxstr@redefstyles	21, 308	\@glxstrshort	79, 184
\@glxstr@reg@glosslist	104–107, 110	\@glxstrshortpl	79, 187
\@glxstr@rglstrigger@record ...	135–137	\@glxstrundeftag	6, 25
\@glxstr@s@longnewglossaryentry	32	\@glxstrwrglossmark	22
\@glxstr@savepreloctag	51, 52	\@gobble .. 7, 13, 15, 59, 127, 128, 175, 345–347	
\@glxstr@setentrycountunsetattr	89	\@gobbletwo	174
\@glxstr@setentryunitcountunsetattr	100	\@ifnextchar	76
\@glxstr@setupshortcuts	19, 20, 24	\@ifpackageloaded	5, 16, 122, 138, 155, 156, 159, 160, 162, 307
\@glxstr@shortcutsval	19, 122	\@ifstar	26, 29, 32–34, 45, 76, 125, 168
\@glxstr@swaptwo	174	\@ifundefined	307
\@glxstr@tag	168	\@ignored@glossaries	33–35
\@glxstr@taggingcs	168	\@input	121
\@glxstr@textformat	57, 58	\@input@	116
\@glxstr@theHvalue ...	8, 9, 56–58, 134, 135	\@istfilename	104
\@glxstr@thevalue	8, 9, 56–58, 134, 135	\@makeglossary	104
\@glxstr@thisloctag	51, 52	\@mfu@domakefirsttuc	168, 169
\@glxstr@titlelabel	112, 113, 128	\@mfu@nocaplist	169
\@glxstr@tmp	21, 115	\@one	91, 100, 131
\@glxstr@type	154	\@newglossaryentry@defcounters ..	89, 97
\@glxstr@unitcountlist	95	\@newglossaryentryposthook	11, 12, 26, 40, 71

\@newglossaryentryprehook		A	
..... 11, 12, 26, 32, 40, 71	\AB		17
\@nil	\Ab		17
..... 115, 129	\ab		17
\@nnil	abbreviation styles:		
..... 164, 165, 167, 173, 175, 176	long-hyphen-postshort-hyphen ...	277, 279	
\@no@glstrindexaliased	long-hyphen-short-hyphen	273, 278	
..... 74	long-postshort-user	266	
\@no@makeglossaries	long-short-user	264	
..... 120	nolong-short	209	
\@nopostdesc	short	206	
..... 108	short-hyphen-long-hyphen	281, 283	
\@onelevel@sanitize 11, 40, 48, 112, 113, 128	short-hyphen-postlong-hyphen ...	282, 285	
\@onlypreamble	short-long-user	267	
... 48, 51, 99, 121, 123, 163, 165, 166, 168	short-nolong	206, 208	
\@org@glossaryentrynumbers	short-nolong-desc	208	
..... 108	short-postlong-user	268	
\@org@newglossaryentryprehook	\abbreviationsname	16	
..... 32	\abbrvpluralsuffix		
\@print@unsrt@glossary 122, 177, 197, 199, 201, 203,		
..... 125, 126	205, 207, 210, 213, 215, 216, 218, 220,		
\@printgloss@setsort	221, 223, 225, 227, 229, 231, 232, 234,		
..... 107, 108	236, 238, 240, 241, 243, 245, 247, 248,		
\@printglossary	250, 252, 253, 255, 257, 259, 261, 264,		
..... 48, 125, 126	265, 267, 270, 273, 275, 278, 281, 283, 286		
\@printunsrt@glossary@handler	\ABP	17	
..... 127	\Abp	17	
\@printunsrtglossary	\abp	17	
..... 125	\AC	18	
\@rGLS	\Ac	18	
..... 137	\ac	18	
\@rGLS@	\ACF	18	
..... 137	\Acf	18	
\@rGLSpl	\acf	18	
..... 137	\ACFP	18	
\@rGLSpl@	\Acfp	18	
..... 137	\acfp	18	
\@rGls	\ACL	18	
..... 136	\Acl	18	
\@rGls@	\acl	18	
..... 136	\ACLP	18	
\@rGlspl	\Aclp	18	
..... 136	\aclp	18	
\@rGlspl@	\ACP	18	
..... 136	\Acp	18	
\@rgls	\acp	18	
..... 135	\ACRfullfmt	102	
\@rgls@	\Acrfullfmt	102	
..... 135	\acrfullfmt	102	
\@rglspl			
..... 135			
\@rglspl@			
..... 135			
\@sGlsXtrEnableOnTheFly			
..... 45			
\@secondofthree			
..... 60,			
61, 67–70, 72, 181, 183, 184, 186, 188, 189			
\@secondoftwo			
..... 55, 59,			
62–70, 73, 78, 103, 109, 161, 173, 181,			
182, 184–190, 203, 225, 239, 261, 289, 291			
\@sglsxtr@provide@storagekey			
..... 26			
\@starttoc			
..... 289			
\@thirdofthree			
..... 59–62,			
67–70, 72, 182, 183, 185, 186, 188, 190, 290			
\@thirdoftwo			
..... 62–66			
\@this@key			
..... 161			
\@warn@nomakeglossaries			
..... 117			
\@xdy@main@language			
..... 116			
\@xdycrossrefhook			
..... 39			
\@xdy@language			
..... 116			
\@xdy@locationclassorder			
..... 39			
\\			
..... 115			
_			
..... 44, 118, 119			

<code>\ACRfullplfmt</code>	102	<code>bib2gls</code>	3, 129, 131, 134, 135
<code>\Acrfullplfmt</code>	102		
<code>\acrfullplfmt</code>	102		
<code>\acronymentry</code>	101		
<code>\acronymfont</code>	67–70, 82, 102, 103		
<code>\acronymname</code>	16		
<code>\acronymsort</code>	101		
<code>\acronymtype</code>	17, 101, 102		
<code>\acrpluralsuffix</code>	101, 122		
<code>\ACS</code>	18		
<code>\Acs</code>	18		
<code>\acs</code>	18		
<code>\ACSP</code>	18		
<code>\Acsp</code>	18		
<code>\acsp</code>	18		
<code>\actualchar</code>	166		
<code>\addtolength</code>	335		
<code>\advance</code>	91, 100, 121, 131		
<code>\AF</code>	17		
<code>\Af</code>	17		
<code>\af</code>	17		
<code>\AFP</code>	17		
<code>\Afp</code>	17		
<code>\afp</code>	17		
<code>\AL</code>	17		
<code>\Al</code>	17		
<code>\al</code>	17		
<code>\ALP</code>	17		
<code>\Alp</code>	17		
<code>\alp</code>	17		
<code>\AnyTrackedLanguages</code>	307		
<code>\appto</code>	11, 12, 21, 26, 37–42, 71, 75, 89, 97, 127, 128, 163, 172, 175, 176, 309		
<code>\arabic</code>	347		
<code>\AS</code>	17		
<code>\As</code>	17		
<code>\as</code>	17		
<code>\ASP</code>	17		
<code>\Asp</code>	17		
<code>\asp</code>	17		
<code>\AtBeginDocument</code>	22, 25, 49, 50, 122		
<code>\AtEndDocument</code>	42, 90, 99, 116, 117		
	B		
<code>babel package</code>	163, 165, 172		
<code>\begin</code>	15, 16, 113, 118, 126, 311, 313–320, 338–341, 345		
<code>\begingroup</code>	8, 9, 74, 124–126		
<code>\bgroup</code>	32, 108		
		C	
		<code>\catcode</code>	121
		category attributes:	
		<code>apospplural</code>	177
		<code>discardperiod</code>	171
		<code>entrycount</code>	88, 89, 91, 100
		<code>firstuc</code>	159
		<code>glossdesc</code>	155
		<code>glossdescfont</code>	155
		<code>glossname</code>	156
		<code>glossnamefont</code>	157, 159
		<code>headuc</code>	291
		<code>indexname</code>	164
		<code>indexonlyfirst</code>	75
		<code>insertdots</code>	177
		<code>markshortwords</code>	177
		<code>markwords</code>	176, 177, 271, 272, 280
		<code>nohyper</code>	73
		<code>nohyperfirst</code>	60–62
		<code>noshortplural</code>	177
		<code>regular</code>	53, 94, 196– 201, 203, 205–212, 214–218, 221–223, 225, 228–231, 233, 235, 237, 239, 242– 246, 248–250, 253–256, 258, 259, 261, 263, 269–271, 273–276, 281, 282, 286, 287
		<code>textformat</code>	57
		<code>\cdot</code>	22
		<code>\centering</code>	344
		<code>\cGLS</code>	17, 18, 89, 100
		<code>\cGls</code>	17, 18, 89, 100
		<code>\cglS</code>	17, 18, 89, 100
		<code>\cGLSformat</code>	93
		<code>\cGlsformat</code>	92
		<code>\cglSformat</code>	92, 94
		<code>\cGLSpl</code>	17, 18, 89, 100
		<code>\cGlspl</code>	17, 18, 89, 100
		<code>\cglSpl</code>	17, 18, 89, 100
		<code>\cGLSplformat</code>	93
		<code>\cGlsplformat</code>	92
		<code>\cglSplformat</code>	92, 94
		<code>\changes</code>	16, 294
		<code>\char</code>	112
		<code>\columnwidth</code>	49, 50
		<code>\count@</code>	91, 99, 100
		<code>\cs</code>	16
		<code>\csappto</code>	31

<code>\csdef</code>	26, 29–31, 71, 72, 90, 95, 96, 98, 149, 193–195, 203, 225, 239, 261, 264, 266–268, 278, 280, 283, 285, 310
<code>\cseappto</code>	36
<code>\csedef</code>	29, 96
<code>\csgdef</code> 30, 33–35, 44, 51, 90, 96, 98, 99, 326, 348	
<code>\cslet</code>	30, 32, 108
<code>\csletcs</code>	30, 195
<code>\csname</code> 6, 26, 34, 40, 44, 49, 52, 55, 56, 58, 67– 72, 74, 83, 96, 105, 113, 116, 119, 120, 126, 131–134, 155, 174, 181–190, 196, 334	
<code>\cspreto</code>	31
<code>\csuse</code>	27, 28, 34, 42, 51, 71, 72, 84–86, 95–99, 107, 112–114, 124, 125, 128–130, 149, 160, 170, 171, 194, 195, 327, 329, 330
<code>\csxdef</code>	37, 38, 41, 96, 99, 113
<code>\currentglossary</code>	108, 124, 125, 347
<code>\CurrentOption</code>	22, 309, 310
<code>\CurrentTrackedLanguageTag</code>	122
<code>\CurrentTrackedTag</code>	308
<code>\CustomAbbreviationFields</code>	178, 196, 198, 200, 201, 203, 204, 206, 209, 211, 213–216, 218, 220, 223, 225, 227–230, 232, 234, 237, 239, 241–244, 246–249, 251, 253, 257, 259, 260, 263, 264, 266–269, 271, 273, 274, 276, 278, 279, 281, 283, 285–287
D	
<code>\DeclareAcronymList</code>	101
<code>\DeclareOption</code>	5, 309
<code>\DeclareOptionX</code>	5, 22
<code>\def</code> 9–13, 23, 25, 32, 34, 38, 40, 43, 45–48, 50, 52–70, 76, 78–82, 92–94, 101, 105, 107– 110, 112–116, 126, 129, 131, 134–137, 164–169, 173–177, 181–190, 193, 272, 274, 277, 280, 282, 283, 335, 336, 340, 341	
<code>\defglentryfmt</code>	33–35
<code>\define@boolkey</code>	14, 15, 56, 73
<code>\define@choicekey</code>	7, 12, 14, 15, 19, 20, 22, 56, 109
<code>\define@key</code>	11, 12, 16, 21, 26, 40, 56, 58, 71, 109, 174
<code>\DefineAcronymSynonyms</code>	19
<code>\delimN</code>	51, 52
<code>\delimR</code>	51, 52
<code>\detokenize</code>	45
<code>\dimen@</code>	103, 327–334
<code>\dimen@i</code>	328–330
<code>\dimen@ii</code>	328–330
<code>\dimexpr</code>	49, 50, 326
<code>\disable@keys</code>	17, 25, 44, 120
<code>\do</code>	6, 21, 43, 89, 101, 104, 107, 113, 126, 134, 154, 161, 168
<code>\do@glx@link@checkfirsthyper</code>	27, 53, 55, 57, 66–70, 181–190
<code>\do@glx@disablehyperinlist</code>	57, 74
<code>doc package</code>	166
<code>\dolistcsloop</code>	28
<code>\DTLifinlist</code>	105, 106, 110
<code>\DTLifint</code>	112
E	
<code>\eappto</code>	11, 21, 33–35, 127, 129, 163, 309
<code>\edef</code> .	6, 8–10, 33–36, 39, 41, 42, 44, 56–58, 73, 74, 77, 78, 95, 96, 98, 104, 105, 110, 112, 114–117, 121, 124, 125, 134, 155– 161, 164, 165, 167, 174, 329, 330, 345, 346
<code>\eglssetwidest</code>	327–334
<code>\egroup</code>	32, 33, 108
<code>\else</code>	8–11, 14, 15, 17, 19, 20, 23, 31, 44, 46, 50, 52, 55, 57, 58, 74, 75, 91, 103, 104, 106, 108, 109, 112, 114, 115, 118, 120, 121, 134, 135, 163–165, 167, 173, 175– 177, 184–190, 192, 193, 197, 199, 201– 211, 213–215, 217–224, 226, 228–240, 242–245, 247–262, 264–268, 270–272, 274, 277, 279, 280, 283, 284, 286, 287, 311, 313–323, 325, 335, 336, 342, 344, 346
<code>\emph</code>	241
<code>\empty</code>	114, 115
<code>\encapchar</code>	166
<code>\end</code>	113, 118, 119, 127, 311, 313–320, 338–341, 345
<code>\end@glx@tr@display@loc</code>	114
<code>\endcsname</code>	6, 26, 34, 40, 44, 49, 52, 55, 56, 58, 67– 72, 74, 83, 96, 105, 113, 116, 119, 120, 126, 131–134, 155, 174, 181–190, 196, 334
<code>\endgroup</code>	8, 10, 74, 124–126
<code>\ensuremath</code>	22
entry categories:	
abbreviation	190
general	149, 151
index	153
<code>\epreto</code>	164
<code>\equal</code>	119, 120
<code>etoolbox package</code>	5

<code>\expandafter</code>	22, 26, 27, 38, 39, 43, 45–47, 71, 72, 76, 83, 94, 95, 105–107, 110, 113–115, 126, 127, 129, 131, 132, 138, 155, 158, 159, 161–163, 166, 167, 173, 176–178, 271
<code>\expandonce</code>	101, 129, 164, 165, 198, 275
F	
<code>\fi</code>	7–11, 13–15, 17, 19, 20, 22–24, 31, 38, 40–42, 44–46, 49, 50, 52, 55–58, 74, 75, 91, 99, 100, 103, 106–109, 112, 114, 115, 117–121, 123, 134, 135, 163–165, 167, 173, 175, 176, 178, 184–190, 192, 193, 197, 199, 201–211, 213–215, 217–224, 226, 228–240, 242–245, 247–262, 264–268, 270–272, 274, 277, 279, 280, 283, 284, 286, 287, 311, 313–325, 327–336, 342, 344, 346
first use	349
flag	349
text	349
<code>\firstacronymfont</code>	102, 103
fontspec package	122
<code>\footnote</code>	201
<code>\forall glossaries</code>	42, 126, 152–154, 327, 328, 330–334
<code>\forall glossentries</code>	91, 100
<code>\ForEachTrackedDialect</code>	308
<code>\forall glossentries</code>	6, 42, 152–154, 327–334
<code>\forall listcsloop</code>	28, 100, 113
<code>\forall listloop</code>	110, 111, 169
<code>\futurelet</code>	173
G	
<code>\gdef</code>	51, 165, 166
<code>\Genacrfullformat</code>	102
<code>\genacrfullformat</code>	102
<code>\GenericAcronymFields</code>	102
<code>\Genplacrfullformat</code>	102
<code>\genplacrfullformat</code>	102
<code>\glo@grabfirst</code>	129
<code>\glo@name</code>	157–159, 162
<code>\gloaliaslabel</code>	78
<code>\global</code>	10, 32, 108, 130
<code>\glo linkprefix</code>	57, 58, 78, 123
glossaries package	13, 23, 24, 37, 39, 40, 42, 107, 310
glossaries-accsupp package	20, 22, 138
glossaries-extra package	2
glossaries-extra-stylemods package	20, 170, 308
glossaries-stylemods package	343
glossaries.sty package	32
<code>\GlossariesExtraWarning</code>	6, 15, 31, 46, 48, 57, 103, 105, 115, 118, 121, 126, 155–161, 168, 169, 195
<code>\GlossariesExtraWarningNoLine</code>	15, 91, 100
<code>\GlossariesWarning</code> ..	51, 106, 108, 111, 193
<code>\GlossariesWarningNoLine</code>	105, 117
glossary styles:	
altlist	311
altlistgroup	312
altlisthypergroup	313
alttree	325, 326, 335
alttreegroup	336
alttreehypergroup	336
index	321
indexgroup	322
indexhypergroup	322
inline	321
list	311
listdotted	310
listdottedstyle	311
listgroup	312
listhypergroup	312
mcolalttree	340
mcolalttreegroup	340
mcolalttreehypergroup	341
mcolalttreespannav	341
mcolindexgroup	337
mcolindexhypergroup	337
mcolindexspannav	337
mcoltreegroup	338
mcoltreehypergroup	338
mcoltreenonamegroup	339
mcoltreenonamehypergroup	339
mcoltreenonamespannav	340
mcoltreespannav	338
sublistdotted	311
tree	323
treegroup	323
treehypergroup	324
treenoname	324
treenonamegroup	325
treenonamehypergroup	325
glossary-bookindex package	310
glossary-hypernav package	77
glossary-long package	315
glossary-longbooktabs package	315
<code>\glossaryentrynumbers</code>	52, 108, 130

<code>\glossaryheader</code>	113,	<code>\glsabbrvonlyfont</code>	285–287
126, 311–320, 322–325, 335–339, 341, 345		<code>\glsabbrvscfont</code>	
<code>\glossaryname</code>	107	212–216, 218, 220, 221, 223, 225
<code>\glossarypostamble</code>	113, 127, 128	<code>\glsabbrvsmfont</code>	227–232, 234, 236–240
<code>\glossarypreamble</code>	113, 126	<code>\glsabbrvuserfont</code>	263–268, 270
<code>\glossarysection</code> ...	113, 114, 119, 126, 128	<code>\GLSaccessdesc</code>	63
<code>\glossarytitle</code> 34, 107, 108, 113, 114, 119, 126		<code>\Glsaccessdesc</code>	63, 155, 167
<code>\glossarytoctitle</code> 34, 107, 113, 114, 119, 126		<code>\glsaccessdesc</code>	62, 156, 171
<code>\glossentry</code>	108,	<code>\GLSaccessdescplural</code>	63
130, 310, 311, 313–320, 322–324, 335, 345		<code>\Glsaccessdescplural</code>	63
<code>\glossentrydesc</code>	310–320, 322–326	<code>\glsaccessdescplural</code>	63
<code>\glossentryname</code>		<code>\GLSaccessfirst</code>	60
....	124, 310–320, 322–324, 335, 336, 343	<code>\Glsaccessfirst</code>	60
<code>\glossentrynameother</code>	125	<code>\glsaccessfirst</code>	60
<code>\glossentrysymbol</code> 314–318, 320, 322–324, 326		<code>\GLSaccessfirstplural</code>	62
<code>\glossxtrsetpopts</code>	170	<code>\Glsaccessfirstplural</code>	62
<code>\GLS</code>	89, 100, 133	<code>\glsaccessfirstplural</code>	61
<code>\Gls</code>	47, 89, 100, 133	<code>\Glsaccesslong</code>	69, 179, 186,
<code>\gls</code>	31, 46, 48, 89, 100, 106, 118, 133	197, 205, 210, 213, 219–222, 228, 234–	
<code>\gls@assign@desc</code>	32	236, 242, 244, 251, 252, 254, 256–258,	
<code>\gls@assign@field</code>	11, 12, 26, 71	264–266, 273, 275, 276, 279, 281, 286, 287	
<code>\gls@checkseeallowed</code>	45, 105	<code>\glsaccesslong</code>	69, 179, 185, 186,
<code>\gls@codepage</code>	117	197, 199, 201, 202, 204, 205, 207–211,	
<code>\gls@defdocnewglossaryentry</code>	89, 97	213, 215, 217–224, 226, 228–238, 240,	
<code>\gls@defglossaryentry</code>	32, 46, 47	242, 243, 245, 247–260, 262, 264, 265,	
<code>\gls@dotocitle</code>	108	268, 270, 273, 275, 276, 279, 281, 286, 287	
<code>\gls@glossary</code>	40	<code>\Glsaccesslongpl</code>	70, 179,
<code>\gls@grplabel</code>	77	189, 197, 205, 210, 214, 219–222, 228,	
<code>\gls@level</code>	130	234–236, 242, 244, 251–254, 256–258,	
<code>\gls@noidxglossary</code>	105	264–266, 273, 275, 276, 279, 281, 286, 287	
<code>\gls@org@glossaryentryfield</code>	108	<code>\glsaccesslongpl</code>	70, 179, 189,
<code>\gls@org@glossarysubentryfield</code>	108	190, 197, 199, 202, 204, 205, 207–211,	
<code>\gls@save@numberlist</code>	50, 52	213, 215, 217–224, 226, 228–240, 242,	
<code>\gls@set@xr@key</code>	40	244, 245, 247, 249–260, 262, 264, 265,	
<code>\gls@tmplen</code>	327–336	268, 270, 273, 275, 276, 279, 281, 286, 287	
<code>\gls@type</code>	105	<code>\GLSaccessname</code>	62
<code>\glsabbrvdefaultfont</code> ...	180, 197, 199,	<code>\Glsaccessname</code>	62
201, 203, 205, 207, 210, 263, 272, 275, 285		<code>\glsaccessname</code>	38, 62
<code>\glsabbrvmfont</code> .	241–253, 255, 257, 259–261	<code>\GLSaccessplural</code>	61
<code>\glsabbrvfont</code>		<code>\Glsaccessplural</code>	61
..	80, 81, 102, 180, 184, 185, 187, 188,	<code>\glsaccessplural</code>	61
190, 192, 193, 196–201, 203–207, 210,		<code>\Glsaccessshort</code>	67, 184, 193, 199,
211, 213, 215, 216, 218, 220, 221, 223,		202, 204, 207, 209, 215, 217, 218, 224,	
225, 227, 229, 231, 232, 234, 236, 238,		226, 229, 231, 232, 238, 240, 245, 247,	
240, 241, 243, 245, 247, 248, 250, 252,		249, 250, 260–262, 267, 268, 270, 278, 284	
253, 255, 257, 259, 261, 264, 265, 267,		<code>\glsaccessshort</code> 67, 179, 184, 185, 192, 197,	
269–271, 273, 275–279, 281, 283, 284, 286		199, 201–208, 210, 213, 215, 217–222,	
<code>\glsabbrvrhyphenfont</code>		224, 226, 228, 229, 231–236, 238, 240,	
.....	272–274, 278, 279, 281–283, 285		

242–245, 247–252, 254, 256–262, 264–268, 270, 273, 275, 276, 278, 281, 284, 287	\glsdisplaynumberlist	106, 110
\Glsaccessshortpl	\glsdohyperlink	76, 77, 79
202, 204, 207, 209, 215, 217, 218, 224, 226, 230, 231, 233, 238–240, 245, 247, 249, 250, 260–262, 268, 270, 279, 284, 287	\glsdohypertarget	79, 109
\glsaccessshortpl	\glsdoifexists	14, 23, 29, 36, 38–40, 53, 55, 58, 66–70, 78, 105, 110, 111, 124, 125, 181–190
68, 179, 187, 188, 192, 197, 199, 202, 204–208, 210, 213–215, 217–219, 221, 222, 224, 226, 228, 229, 231–238, 240, 242, 244, 245, 247–254, 256, 258–262, 264, 266–268, 270, 273, 275, 276, 278, 281, 284, 287	\glsdoifexistsordo	27, 55
\GLSaccesssymbol	\glsdoifexistsorwarn	14, 155, 156, 167
64, 167	\glsdoifnoexists	32
\glsaccesssymbol	\glsdonohyperlink	58, 78, 79
63, 167, 171	\glsdosanitizesort	106
\GLSaccesssymbolplural	\glsenableentrycount	89, 91, 99
64	\glsenableentryunitcount	90, 100
\Glsaccesssymbolplural	\glsentrycounter	115
64	\glsentrycurrcount	90, 91, 97
\glsaccesssymbolplural	\Glsentrydesc	143, 147, 156
64	\glsentrydesc	142, 143, 147, 148, 156
\GLSaccessstext	\Glsentrydescplural	143, 148
59	\glsentrydescplural	143, 148
\Glsaccessstext	\Glsentryfirst	94, 138, 140, 146
60	\glsentryfirst 94, 137, 140, 141, 146, 147, 304	
\glsaccessstext	\Glsentryfirstplural	95, 138, 141, 147
38, 59	\glsentryfirstplural	94, 138, 141, 147, 304, 305
\glsacrshortcutstrue	\glsentryfmt	33–35
19, 20	\Glsentryfull	102
\glsacspacemax	\glsentryfull	102
103	\Glsentryfullpl	102
\glsadd	\glsentryfullpl	102
43, 118	\glsentryitem	124, 125, 310, 311, 313–320, 322–324, 335, 346
\glsadd options	\Glsentrylong	81, 82, 94, 138, 145, 148
theHvalue	\glsentrylong	81, 82, 94, 137, 145, 148, 203, 225, 239, 261, 267, 268, 283, 305
thevalue	\Glsentrylongpl	81, 82, 95, 145, 149
9	\glsentrylongpl 81, 82, 94, 145, 149, 305, 306	
\glsaddstoragekey	\Glsentrylongplural	138
42, 149	\glsentrylongplural	138
\glsbackslash	\Glsentryname	139, 146, 157–160
46	\glsentryname	124, 138, 139, 146, 164, 302, 327–334, 348
\glsapscase	\glsentrynumberlist	106, 112, 332–334
55, 59–70, 72, 181–192	\Glsentryplural	140, 146
\glscategory	\glsentryplural	140, 146, 303
53, 59, 73, 80, 81, 150–152, 155–161, 167, 170, 171, 181–185, 187, 188	\glsentryprevcount	90, 91, 97
\glscategorylabel	\glsentryprevmaxcount	98
73, 174, 176, 177, 203, 225, 239, 261, 264, 266–268, 278, 280, 283, 285	\glsentryprevtotalcount	97
\glsclsebrace	\Glsentryshort	80, 82, 144, 148
39, 119, 120		
\glscurrententrylabel		
50–52, 108, 116, 124–127, 169, 170		
\glscurrentfieldvalue		
27–29, 31, 262		
\glscustomtext		
53, 55, 67–70, 181–190, 192		
\glsdefaulttype		
6, 16, 31, 107, 117, 125, 126, 128		
\glsdescriptionaccessdisplay 142, 143, 155		
\glsdescriptionpluralaccessdisplay 143		
\glsdescwidth		
313–320		
\glsdetoklabel		
8, 9, 28–32, 36–39, 43–45, 56, 58, 74, 78, 90, 95–100, 105, 108, 110, 111, 124, 125, 129, 130, 133, 134, 155, 157–159, 162, 329, 330		

<code>\glsentryshort</code> .. 80–82, 103, 144, 148, 265, 266, 277, 301	<code>\glsfirstlongemfont</code> 243, 244, 246–248, 253–255, 257, 258
<code>\Glsentryshortpl</code> 81, 82, 144, 148	<code>\glsfirstlongfont</code> ... 179, 196–201, 203,
<code>\glsentryshortpl</code> 81, 82, 144, 145, 148, 301, 302	205, 207–211, 213, 215, 216, 218, 220,
<code>\Glsentrysymbol</code> 142, 147	222, 223, 225, 227, 229, 231, 232, 234,
<code>\glsentrysymbol</code> 141, 142, 147, 331–333	236, 238, 240, 242, 243, 245, 247, 248,
<code>\Glsentrysymbolplural</code> 142, 147	250, 252, 253, 255, 257, 259, 261, 264,
<code>\glsentrysymbolplural</code> 142, 147	265, 267, 270, 273, 275, 278, 281, 283, 286
<code>\Glsentrytext</code> 139, 146	<code>\glsfirstlongfootnotefont</code> 201–204, 223–226, 237–240, 259–262
<code>\glsentrytext</code> 78, 139, 146, 303	<code>\glsfirstlonghyphenfont</code> 272–283
<code>\glsentrytype</code> 124, 125	<code>\glsfirstlongonlyfont</code> 286, 287
<code>\Glsentryuseri</code> 65	<code>\glsfirstlonguserfont</code> 263–271
<code>\glsentryuseri</code> 65	<code>\GLSfirstplural</code> 297
<code>\Glsentryuserii</code> 65	<code>\Glsfirstplural</code> 297
<code>\glsentryuserii</code> 65	<code>\glsfirstplural</code> 297
<code>\Glsentryuseriii</code> 65	<code>\glsfirstpluralaccessdisplay</code> 141
<code>\glsentryuseriii</code> 65	<code>\glsforeachincategory</code> 193
<code>\Glsentryuseriv</code> 65	<code>\glsgenentryfmt</code> 53
<code>\glsentryuseriv</code> 66	<code>\glsgetattribute</code> 57, 77, 91, 95–97, 116, 134, 155–161, 163
<code>\Glsentryuserv</code> 66	<code>\glsgetcategoryattribute</code> 150
<code>\glsentryuserv</code> 66	<code>\glsgetgrouptitle</code> 312, 313, 322–325, 336–342
<code>\Glsentryuservi</code> 66	<code>\glsgetwidestname</code> 326
<code>\glsentryuservi</code> 66	<code>\glsgroupheading</code> 129, 311–320, 322–325, 335–341, 347
<code>\glsfieldfetch</code> 78	<code>\glsgroupskip</code> 129, 311, 313–323, 325, 336, 347
<code>\glsfieldxdef</code> 154	<code>\glshasattribute</code> 57, 77,
<code>\glsfindwidesttoplevelname</code> 327	91, 96, 98, 100, 116, 134, 155–161, 163,
<code>\GLSfirst</code> 296	197–201, 203, 206, 208, 209, 211–216,
<code>\Glsfirst</code> 296	223, 225, 227–230, 237, 239, 241–248,
<code>\glsfirst</code> 296	255, 258, 259, 261, 263, 265–267, 269–
<code>\glsfirstabbrvdefaultfont</code> 180, 197, 199, 201, 203, 205, 207, 210, 275	271, 273–276, 278, 280–283, 285, 286, 288
<code>\glsfirstabbrvfont</code> 102, 179,	<code>\glshascategoryattribute</code> 150
196–210, 213, 215, 216, 218, 220, 221,	<code>\glshyperlink</code> 78
223, 225, 227, 229, 231, 232, 234, 236,	<code>\glshypernavsep</code> 113
238, 240, 241, 243, 245, 247, 248, 250,	<code>\glshypernumber</code> 116, 163
252, 253, 255, 257, 259, 261, 264, 265,	<code>\glsifattribute</code> 55, 56, 60, 73, 75,
267, 270, 273, 275, 276, 278, 281, 283, 286	84, 153, 155–158, 162, 169, 172, 292–300
<code>\glsfirstabbrvhyphenfont</code> 272–274, 277, 278, 280–285	<code>\glsifcategory</code> 152
<code>\glsfirstabbrvonlyfont</code> 286, 287	<code>\glsifcategoryattribute</code> . 73, 151, 176, 177
<code>\glsfirstabbrvscfont</code> 213–226	<code>\glsifnotregular</code> 59
<code>\glsfirstabbrvsmfont</code> 227–240	<code>\glsifnotregularcategory</code> 152
<code>\glsfirstabbrvuserfont</code> 263–271	<code>\glsifplural</code> 55, 59, 61–64, 66–70, 172, 181–191
<code>\glsfirstaccessdisplay</code> 140, 141	<code>\glsifregular</code> 53, 59, 94, 95, 137, 138
<code>\glsfirstlongdefaultfont</code> 197, 199, 205, 207, 210, 213–223, 227–	<code>\glsifregularcategory</code> 152
237, 241, 242, 245, 246, 248–253, 255, 256	<code>\glsifusetranslator</code> 34
	<code>\glsignore</code> 51, 52
	<code>\glsinlinedescformat</code> 321

<code>\glsinlinesubdescformat</code>	321	<code>\glslonghyphenfont</code>	
<code>\glsinsert</code>	55,	272, 273, 275, 276, 278, 281, 283
59, 67–70, 181–192, 271, 278, 280, 283, 285		<code>\glslongonlyfont</code>	285, 286
<code>\glskeylisttok</code>	101, 102, 176, 178	<code>\glslongpltok</code>	
<code>\glslabel</code> ...	8, 9, 36, 38, 53, 55–58, 73, 74,	. 178, 196, 198–201, 209, 211, 213–216,	
77, 78, 103, 134, 170, 171, 190–192, 203,		220, 223, 227–230, 234, 237, 241–246,	
225, 239, 261, 265–268, 278, 280, 283, 285		248, 252, 253, 257, 259, 263, 264, 266–	
<code>\glslabeltok</code>	101, 176, 178, 196–201,	271, 273–276, 278, 279, 281, 282, 286, 287	
203, 205–209, 211–216, 218, 220, 223,		<code>\glslongpluralaccessdisplay</code>	145
225, 227–232, 234, 237, 239, 241–248,		<code>\glslongtok</code>	
250, 252, 253, 255, 257–259, 261, 263–		. 101, 176, 178, 196–201, 203, 205, 207,	
271, 273–276, 278, 280–283, 285, 286, 288		209, 211, 213, 214, 216, 218, 220, 223,	
<code>\glsletentryfield</code>	164	225, 227–232, 234, 237, 239, 241–248,	
<code>\glslink</code>	102	250–253, 257, 259, 260, 263, 264, 266–	
<code>\glslink options</code>		271, 273–276, 278, 279, 281–283, 286, 287	
counter	9	<code>\glslonguserfont</code>	263–268, 270
format	162	<code>\glsmcols</code>	338–341
hyper	288	<code>\GLSname</code>	293, 294
hyperoutside	56	<code>\Glsname</code>	294
noindex	8, 73, 288	<code>\glsname</code>	293, 294
theHvalue	57	<code>\glsnameaccessdisplay</code> ..	138, 139, 157–159
thevalue	57, 131	<code>\glsnamefont</code>	157–160, 162
wrgloss	8, 55, 56	<code>\glsnavhyperlink</code>	113
<code>\glslinkcheckfirsthyperhook</code>	73	<code>\glsnavhyperlinkname</code>	77
<code>\glslinkpostsetkeys</code>	57, 134	<code>\glsnavhypertarget</code>	
<code>\glslinkvar</code>	76	312, 313, 323–325, 337–342
<code>\glslistchildpostlocation</code>	311	<code>\glsnavigation</code> ..	312, 313, 323–325, 336–341
<code>\glslistchildprelocation</code>	311, 312	<code>\glsnextpages</code>	108
<code>\glslistdottedwidth</code>	310	<code>\glsnoidxdisplayloc</code>	111
<code>\glslistgroupheaderfmt</code>	312, 313	<code>\glsnoidxdisplaylocclishandler</code>	110
<code>\glslistnavigationitem</code>	312, 313	<code>\glsnoidxloclist</code>	111, 130
<code>\glslistprelocation</code>	311, 312	<code>\glsnoidxnumberlistloophandler</code>	111
<code>\glslocalunset</code>	55, 135	<code>\glsnonextpages</code>	108
<code>\glslongaccessdisplay</code>	145	<code>\glsnonumberlistfalse</code>	50
<code>\glslongdefaultfont</code> .	180, 197, 199, 200,	<code>\glsnonumberlisttrue</code>	50
205, 207, 210, 213, 215, 216, 218, 220,		<code>\glsnopostdotfalse</code>	109
222, 227, 229, 231, 232, 234–236, 242,		<code>\glsnopostdottrue</code>	109
245, 248, 250, 252, 255, 256, 263, 272, 285		<code>\glsnumberlistloop</code>	106
<code>\glslongemfont</code> 241, 243, 246, 247, 253, 254, 257		<code>\glsnumlistlastsep</code>	110
<code>\glslongfont</code>	81, 82, 180, 185,	<code>\glsnumlistsep</code>	110
186, 189, 190, 197–201, 203, 205, 207,		<code>\glsopenbrace</code>	39, 119, 120
209–211, 213, 215, 216, 218, 220, 222,		<code>\glsorder</code>	104
223, 225, 227, 229, 231, 232, 234, 236,		<code>\glspagelistwidth</code>	314, 316, 318, 320
238, 240, 242, 243, 245, 247, 248, 250,		<code>\glspar</code>	128
252, 253, 255, 257, 259, 261, 264, 265,		<code>\GLSpl</code>	89, 100, 133
267, 270, 273, 275, 278, 281, 283, 286, 287		<code>\Glspl</code>	47, 89, 100, 133
<code>\glslongfootnotefont</code>		<code>\glspl</code>	47, 89, 100, 133
200, 201, 203, 223, 225, 238, 240, 259, 261		<code>\GLSplural</code>	295
		<code>\Glsplural</code>	295, 296

<code>\glsplural</code>	295	<code>\glstreegroupheaderfmt</code>	
<code>\glspluralaccessdisplay</code>	140	322–325, 336–342, 344
<code>\glspluralsuffix</code>	122, 176, 180	<code>\glstreeindent</code>	323, 324, 334–336
<code>\glspostdescription</code>		<code>\glstreeitem</code>	321, 338, 345, 346
.....	15, 16, 109, 170, 310–320, 322–326	<code>\glstreenamebox</code>	335, 336
<code>\glspostinline</code>	321	<code>\glstreenamefmt</code>	322–324, 326–336
<code>\glspostlinkhook</code> .	54, 55, 67–70, 83, 181–190	<code>\glstreenavigationfmt</code> ..	323–325, 336–341
<code>\glsprestandardsort</code>	106	<code>\glstreepredesc</code>	322–324
<code>\glsresetentrylist</code>	113, 126	<code>\glstreeprelocation</code>	321–324, 326
<code>\glssee</code>	40–42	<code>\glstreesubitem</code>	321, 346
<code>\glsseeformat</code>	38, 39, 44, 105, 111	<code>\glstreesubsubitem</code>	322, 346
<code>\glsseelist</code>	39	<code>\GlsrLetField</code>	30
<code>\glssetabbrvfmt</code>	53, 59, 80,	<code>\glstype</code>	55, 56, 67–70, 134, 181–190
81, 155–161, 167, 181–185, 187, 188, 190		<code>\glset</code>	43, 55, 92, 93, 135
<code>\glssetAttribute</code> ...	197–201, 203, 205–	<code>\glswrite</code>	39, 104
209, 211–216, 218, 220, 223, 225, 227–		<code>\glswriteentry</code>	8, 9
232, 234, 237, 239, 241–248, 250, 252,		<code>\Glsxtr</code>	48
253, 255, 257–259, 261, 264–267, 269–		<code>\glxtr</code>	48
271, 273–276, 278, 280–283, 285, 286, 288		<code>\glxtr@@do@wrglossary</code>	8, 9, 11, 13
<code>\glssetcategoryattribute</code>		<code>\glxtr@addloclistfield</code>	13
. 89, 101, 103, 134, 150, 151, 153, 154, 168		<code>\glxtr@addunused</code>	43
<code>\glssetnoexpandfield</code>	11, 12	<code>\glxtr@applyabbrvfmt</code>	190
<code>\glssettoctitle</code>	108	<code>\glxtr@applyabbrvstyle</code>	174, 176, 193
<code>\glsshortaccessdisplay</code>	144	<code>\glxtr@counterrecord</code>	124
<code>\glsshortpltok</code> 178, 196–201, 203, 205–207,		<code>\glxtr@do@alsoindex@wrglossary</code>	13
213–216, 218, 223, 225, 227–232, 237,		<code>\glxtr@doption</code>	5, 15, 16, 22, 24
239, 241–250, 259, 260, 263, 264, 266–		<code>\glxtr@fields</code>	122
271, 273, 274, 278, 279, 281–283, 285–287		<code>\glxtr@headentry@p</code>	84, 85
<code>\glsshortpluralaccessdisplay</code> ..	144, 145	<code>\glxtr@hyperoutsidefalse</code>	56
<code>\glsshorttok</code>	101,	<code>\glxtr@hyperoutsidettrue</code>	56
176–178, 196, 198, 200, 201, 203–207,		<code>\glxtr@ifnextpunc</code>	173
211, 213, 214, 216, 218, 220, 223, 225,		<code>\glxtr@ifpunctoken</code>	173
227–230, 232, 234, 237, 239, 241–249,		<code>\glxtr@indexonly@saveentrycounter</code>	
251, 253, 259, 260, 263, 264, 266–271,		13, 25
273, 274, 276, 278, 279, 281–283, 285–287		<code>\glxtr@keylist</code>	46, 47
<code>\glssubentryitem</code>		<code>\glxtr@label</code>	348
.....	124, 125, 310–320, 322–324, 335, 346	<code>\glxtr@langtag</code>	122
<code>\glssymbolaccessdisplay</code>	141, 142	<code>\glxtr@linkprefix</code>	122, 123
<code>\glssymbolpluralaccessdisplay</code>	142	<code>\glxtr@makeglossaries</code>	104
<code>\glstarget</code>		<code>\glxtr@newabbreviation</code>	102, 176
124, 125, 310–320, 322–324, 335, 336, 346		<code>\glxtr@next</code>	173
<code>\GLStext</code>	294, 295	<code>\glxtr@org@@do@wrglossary</code>	25
<code>\GlsText</code>	295	<code>\glxtr@org@dohyperlink</code>	77
<code>\glstext</code>	294	<code>\glxtr@org@getgrouptitle</code>	112
<code>\glstextaccessdisplay</code>	139	<code>\glxtr@org@newignoredglossary</code>	33
<code>\glstextformat</code>	55, 57	<code>\glxtr@orgmakenoidxglossaries</code> ..	43, 44
<code>\glstextup</code>	212	<code>\glxtr@pluralsuffixes</code>	122
<code>\glstreechildpredesc</code>	322, 323	<code>\glxtr@process</code>	126, 127
<code>\glstreechildprelocation</code> ...	322, 323, 325	<code>\glxtr@provideignoredglossary</code>	34

<code>\glxtr@punctlist</code>	172, 173	<code>\glxtrbookindexparentsubchildsep</code> .	345, 346
<code>\glxtr@record</code>	10, 122	<code>\glxtrbookindexprelocation</code> ...	343, 346
<code>\glxtr@recordsee</code>	11	<code>\glxtrbookindexsubbetween</code>	346
<code>\glxtr@resource</code>	120, 122	<code>\glxtrbookindexsubname</code>	346
<code>\glxtr@s@newignoredglossary</code>	33	<code>\glxtrbookindexsubprelocation</code> ...	346
<code>\glxtr@s@provideignoredglossary</code> ...	34	<code>\glxtrbookindexsubsubbetween</code>	346
<code>\glxtr@saveentrycounter</code>	8, 9, 11, 74	<code>\glxtrbookindexthepage</code>	347, 348
<code>\glxtr@setaccessdisplay</code>	161	<code>\glxtrcat</code>	46, 47
<code>\glxtr@setbookindexmark</code>	347	<code>\glxtrchecknohyperfirst</code>	60–62
<code>\glxtr@setup@record</code>	12, 13, 24, 25	<code>\glxtrComputeTreeIndent</code>	335
<code>\glxtr@shortcutsval</code>	122	<code>\glxtrComputeTreeSubIndent</code>	335
<code>\glxtr@texencoding</code>	122	<code>\Glsxtrdefaultsubsequentfmt</code> ...	193, 194
<code>\glxtr@usesee</code>	38	<code>\glxtrdefaultsubsequentfmt</code> ...	192, 194
<code>\glxtr@warnonexistsordo</code>	7, 13, 37	<code>\Glsxtrdefaultsubsequentplfmt</code> .	193, 194
<code>\glxtr@writefields</code>	120	<code>\glxtrdefaultsubsequentplfmt</code> .	193, 194
<code>\glxtrabbrvfootnote</code>	201–203, 223–225, 237–239, 259–261	<code>\GlsXtrDefineAbbreviationShortcuts</code>	19, 20
<code>\glxtrabbrvpluralsuffix</code>	122, 180, 197, 199, 201, 203, 205, 207, 210, 212, 227, 241, 263, 272, 275, 278, 283, 286	<code>\GlsXtrDefineAcShortcuts</code>	19, 20
<code>\glxtrabbrvtype</code>	16, 17, 178	<code>\GlsXtrDefineOtherShortcuts</code>	19, 20
<code>\glxtractivateno</code>	108	<code>\glxtrdetoklocation</code>	133
<code>\glxtraddallcrossrefs</code>	42	<code>\glxtrdiscardperiod</code>	171
<code>\glxtralias</code>	74	<code>\glxtrdisplayendloc</code>	114
<code>\glxtrAltTreeIndent</code>	326	<code>\glxtrdisplayendlochook</code>	115
<code>\glxtraltrtreeInit</code>	335, 340, 341	<code>\glxtrdisplaysingleloc</code>	114, 115
<code>\glxtrAltTreePar</code>	326	<code>\glxtrdisplaystartloc</code>	114
<code>\glxtrAltTreeSetHangIndent</code> ...	326, 335	<code>\glxtrdoautoindexname</code>	75, 76, 160
<code>\glxtrAltTreeSetSubHangIndent</code> ...	336	<code>\glxtrdopostpunc</code>	203, 225, 239, 261
<code>\glxtraltrtreeSubSymbolDescLocation</code>	336	<code>\glxtrdowrglossaryhook</code>	75
<code>\glxtraltrtreeSymbolDescLocation</code> ..	326, 335	<code>\glxtrremsuffix</code>	241, 243, 245, 247, 248, 250, 252, 253, 255, 257, 259, 261
<code>\glxtrassignfieldfont</code>	59–66	<code>\GlsXtrEnableEntryCounting</code>	100
<code>\glxtrautoindex</code>	163	<code>\GlsXtrEnableEntryUnitCounting</code>	89
<code>\glxtrautoindexassignsort</code>	164	<code>\GlsXtrEnableOnTheFly</code>	46, 48
<code>\glxtrautoindexentry</code>	164	<code>\glxtrfieldlistgadd</code>	123
<code>\glxtrbookindexatendgroup</code>	345	<code>\glxtrfieldtitlecase</code>	155–158, 162
<code>\glxtrbookindexatsubendgroup</code>	346	<code>\glxtrfieldtitlecasescs</code>	155
<code>\glxtrbookindexatsubsubendgroup</code> ..	346	<code>\glxtrfieldxifinlist</code>	128
<code>\glxtrbookindexbetween</code>	345	<code>\glxtrfirstscfont</code>	212
<code>\glxtrbookindexbookmark</code>	347	<code>\glxtrfirstsmfont</code>	227
<code>\glxtrbookindexcols</code>	345	<code>\GlsXtrFmtDefaultOptions</code>	27
<code>\glxtrbookindexcolspread</code>	345	<code>\GlsXtrFmtField</code>	27
<code>\glxtrbookindexfirstmarkfmt</code>	348	<code>\GlsXtrFormatLocationList</code> 50, 52, 332–334	
<code>\glxtrbookindexformatheader</code>	347	<code>\GLSxtrfull</code>	17, 18, 299, 300
<code>\glxtrbookindexgroupskip</code>	347	<code>\Glsxtrfull</code>	17, 18, 300
<code>\glxtrbookindexlastmarkfmt</code>	348	<code>\glxtrfull</code>	17, 18, 299
<code>\glxtrbookindexname</code>	343, 346	<code>\Glsxtrfullformat</code>	179, 192, 194, 195, 197, 199, 202, 204, 206, 207, 211, 213, 215, 217, 219, 221,
<code>\glxtrbookindexparentchildsep</code>	343, 345		

223, 224, 226, 228, 229, 231, 233, 235, 237, 238, 240, 242, 244, 245, 247, 249, 250, 253, 255, 256, 258, 259, 261, 264, 265, 267, 270, 273, 276, 279, 281, 284, 286	\Glsxtrheadplural 290
\Glsxtrfullformat 179, 192, 194, 195, 197, 199, 201, 203, 205, 207, 210, 213, 215, 217, 218, 221, 222, 224, 226, 228, 229, 231, 233, 235, 237, 238, 240, 242, 243, 245, 247, 249, 250, 253, 254, 256, 258, 259, 261, 264, 265, 267, 270, 273, 276, 279, 281, 284, 286	\glsxtrheadplural 290
\GLSxtrfullpl 17, 18, 300	\Glsxtrheadshort 290
\Glsxtrfullpl 17, 18, 300, 301	\glsxtrheadshort 290
\glsxtrfullpl 17, 18, 300	\Glsxtrheadshortpl 290
\Glsxtrfullplformat 179, 192, 194, 195, 197, 199, 202, 204, 206, 208, 211, 214, 215, 217, 219, 221, 223, 224, 226, 228, 230, 232, 233, 235, 237, 238, 240, 242, 244, 245, 247, 249, 250, 253, 255, 256, 258, 260, 261, 264, 265, 268, 270, 273, 276, 279, 281, 284, 286	\Glsxtrheadshortpl 290
\glsxtrfullplformat 191, 192, 194, 195, 197, 199, 202, 204, 205, 207, 211, 213, 215, 217, 218, 221, 222, 224, 226, 228, 229, 231, 233, 235, 237, 238, 240, 242, 243, 245, 247, 249, 250, 253, 254, 256, 258, 259, 261, 264, 265, 267, 270, 273, 276, 279, 281, 284, 286	\Glsxtrheadtext 290
\glsxtrfullsep 179, 196–200, 202, 204, 205, 207–210, 213–222, 224, 226–236, 238–254, 256– 258, 260, 262, 263, 272–277, 280–283, 287	\glsxtrheadtext 290
\glsxtrgenabbrvfmt 53	\glsxtrhyperlink 78, 115
\glsxtrgetgroupitle 113, 347	\glsxtrhyphensuffix 273, 281
\glsxtrgroupfield 129	\glsxtrifcounttrigger 92, 93
\Glsxtrheadfirst 291	\glsxtrifemptyglossary 114, 119, 126
\glsxtrheadfirst 291	\glsxtrifhasfield 31, 74, 343
\Glsxtrheadfirstplural 291	\glsxtrifhyphenstart 272, 274, 277, 280, 282, 283
\Glsxtrheadfull 291	\glsxtrifindexing 75
\glsxtrheadfull 291	\glsxtrifinmark 58, 84–87, 289–291
\Glsxtrheadfullpl 291	\glsxtrifnextpunc 173, 174
\glsxtrheadfullpl 291	\glsxtrifperiod 172
\Glsxtrheadlong 291	\glsxtrifrecordtrigger 135–137
\glsxtrheadlong 291	\glsxtrifwasfirstuse 59–62, 66– 70, 73, 103, 171, 172, 181, 184–190, 203, 225, 239, 261, 265–268, 278, 280, 283, 285
\Glsxtrheadlongpl 291	\glsxtrindexaliased 74
\glsxtrheadlongpl 291	\glsxtrindexseealso 41, 42
\Glsxtrheadname 290	\glsxtrinithyperoutside 57
\glsxtrheadname 124, 290	\glsxtrinitwrgloss 56, 134
	\glsxtrinitwrglossbeforefalse ... 55, 56
	\glsxtrinitwrglossbeforetrue ... 55, 56
	\Glsxtrinlinefullformat 179, 181, 194, 195, 202, 204, 205, 207, 208, 210, 217– 219, 221, 222, 224, 226, 231–233, 235, 236, 238, 240, 249–252, 254, 256, 258, 260, 262, 266, 268, 276, 279, 284, 287, 306
	\glsxtrinlinefullformat 179, 181, 182, 194, 195, 202, 204, 205, 207, 208, 210, 217– 220, 222, 224, 226, 231–233, 235, 236, 238, 240, 248, 250–252, 254, 256, 257, 260, 262, 265, 268, 275, 279, 284, 287, 306
	\Glsxtrinlinefullplformat 180, 183, 194, 195, 202, 204, 205, 207, 209, 210, 217– 219, 221, 222, 224, 226, 231, 232, 234– 236, 239, 240, 249–252, 254, 256, 258, 260, 262, 266, 268, 276, 279, 284, 287, 307
	\glsxtrinlinefullplformat 179, 180, 182, 183, 194, 195, 202, 204, 205, 207, 208, 210, 217– 220, 222, 224, 226, 231–233, 235, 236,

238, 240, 248, 250–252, 254, 256, 257, 260, 262, 265, 268, 275, 279, 284, 287, 307	\glsxtrposthyphenlong	283, 285
\glsxtrininsertinsidefalse	\glsxtrposthyphenshort	278, 280
\glsxtrlocationhyperlink	\glsxtrposthyphensubsequent	278, 280, 283, 285
\glsxtrlocrangefmt	\glsxtrpostlink	171
\GLSxtrlong	\glsxtrpostlinkendsentence	171
\Glsxtrlong	\glsxtrpostlinkhook	170
\glsxtrlong	\glsxtrpostlocalreset	88, 90, 98
\glsxtrlonghyphen	\glsxtrpostlocalunset	88, 90, 98
\glsxtrlonghyphennoshort	\glsxtrpostlongdescription	32
\glsxtrlonghyphenshort	\glsxtrpostnamehook	158–160, 162
\GLSxtrlongpl	\GlsXtrPostNewAbbreviation	178, 194, 195, 197–201, 203, 205–209, 211–216, 218, 220, 223, 225, 227–232, 234, 237, 239, 241–248, 250, 252, 253, 255, 257–259, 261, 263, 264, 266–271, 273–276, 278, 280–283, 285–287
\Glsxtrlongpl	\glsxtrpostreset	88, 90, 98
\glsxtrlongpl	\glsxtrpostunset	88, 90, 98
\glsxtrlongshortdescname	\glsxtrprelocation	311, 313, 315, 317, 319, 321, 343
.... 198, 214, 228, 242, 244, 269, 273, 279	\glsxtrprotectlinks	77–79
\glsxtrlongshortdescsort	\GlsXtrRecordCounter	11
.... 198, 214, 228, 242, 244, 269, 274, 279	\glsxtrrecordtriggervalue	134
\glsxtrmarkhook	\glsxtrregularfont	53, 59
\glsxtrnewabbrevpresetkeyhook	\glsxtrresourcecount	121
\glsxtrnewnumber	\glsxtrresourcefile	121
\glsxtrnewsymbol	\glsxtrresourceinit	120
\glsxtrNoGlossaryWarning	\glsxtrrestoremarkhook	288, 289
\GlsXtrNoGlsWarningAutoMake	\glsxtrscfont	212
\GlsXtrNoGlsWarningBuildInfo	\glsxtrscsuffix	213, 215, 216, 218, 220, 221, 223, 225
\GlsXtrNoGlsWarningCheckFile	\GlsXtrSetActualChar	166
\GlsXtrNoGlsWarningEmptyMain	\glsxtrsetaliasnoindex	13, 74
\GlsXtrNoGlsWarningEmptyNotMain	\GlsXtrSetEncapChar	166
\GlsXtrNoGlsWarningEmptyStart	\GlsXtrSetEscChar	166
\GlsXtrNoGlsWarningHead	\glsxtrsetfieldifexists	30
\GlsXtrNoGlsWarningMisMatch	\GlsXtrSetLevelChar	166
\GlsXtrNoGlsWarningNoOut	\glsxtrsetpopts	83
\GlsXtrNoGlsWarningTail	\glsxtrsetupfulldefs	181–183, 203, 225, 239, 261
\glsxtrnopostpunc	\GLSxtrshort	17, 18, 87, 292
\glsxtronlydescname	\Glsxtrshort	17, 18, 293
\glsxtronlydescsort	\glsxtrshort	17, 18, 292
\glsxtronlysuffix	\glsxtrshortdescname	206, 218, 232, 249
\glsxtrorg@ifKV@glslink@hyper	\glsxtrshorthyphen	284
\glsxtrorglong	\glsxtrshorthyphenlong	281
\glsxtrorgshort	\glsxtrshortlongdescname	200, 216, 230, 246, 247, 271, 282, 285
\GLSxtrp		
\Glsxtrp		
\glsxtrp		
\glsxtrparen		
179, 196–200, 202, 204, 205, 207–210, 213–222, 224, 226–254, 256– 258, 260, 262, 263, 272–277, 280–283, 287		
\Glsxtrpl		
\glsxtrpl		
\glsxtrpostdescription		

<code>\glxtrshortlongdescsort</code>	<code>\glxtrundeftag</code>
.... 200, 216, 230, 246, 247, 271, 282, 285	25, 110, 111
<code>\GLSxtrshortpl</code>	<code>\glxtrunsrtdo</code>
17, 18, 292, 293	127, 128
<code>\Glsxtrshortpl</code>	<code>\GlsXtrUseAbbrStyleFmts</code>
17, 18, 293	198, 200, 206, 208, 209,
<code>\glxtrshortpl</code>	211, 212, 214, 216, 219, 229, 230, 233,
17, 18, 292	243, 244, 246, 248, 251, 255, 259, 266,
<code>\glxtrsmfont</code>	269, 271, 274, 275, 277, 280, 282, 285, 288
227	<code>\GlsXtrUseAbbrStyleSetup</code> 206, 208, 209,
<code>\glxtrsmsuffix</code>	211, 212, 219, 221, 233, 236, 251, 255, 258
.... 227, 229, 231, 232, 234, 236, 238, 240	<code>\glxtruserfield</code>
<code>\Glsxtrsubsequentfmt</code>	262, 263
191, 194, 210, 220, 222,	<code>\glxtruserparen</code>
234, 236, 252, 254, 256, 257, 275, 278, 284	263–271
<code>\glxtrsubsequentfmt</code>	<code>\glxtrusersuffix</code>
191, 194, 210, 220, 222,	264, 265, 267, 270
234, 236, 252, 254, 255, 257, 275, 278, 284	<code>\glxtruseseealsoformat</code>
<code>\Glsxtrsubsequentplfmt</code>	39
191, 194, 210, 220, 222,	<code>\glxtruseseeformat</code>
234, 236, 252, 254, 256, 257, 275, 278, 284	38
<code>\glxtrsubsequentplfmt</code>	<code>\GlsXtrWarnDeprecatedAbbrStyle</code> 174, 195
191, 194, 210, 220, 222,	<code>\GlsXtrWarning</code>
234, 236, 252, 254, 256, 257, 275, 278, 284	46, 47
<code>\glxtrsubsequentplfmt</code>	<code>\glxtrword</code>
191, 194, 210, 220, 222,	176
234, 236, 252, 254, 255, 257, 275, 278, 284	<code>\glxtrwordsep</code> 176, 272, 274, 277, 280, 282, 283
<code>\glxtrsupplocationurl</code>	<code>\glxtrwrglossmark</code>
115, 116	22
<code>\glxtrtagfont</code>	
169	H
<code>\Glsxtrtitlefirst</code>	<code>\hangindent</code> . 323, 324, 326, 334–336, 340, 341
290, 291, 304	<code>\hbox</code>
<code>\glxtrtitlefirst</code>	310
290, 291, 304	<code>\hfill</code>
<code>\Glsxtrtitlefirstplural</code>	310
290, 291, 305	<code>\href</code>
<code>\glxtrtitlefirstplural</code>	77
290, 291, 304	<code>\hsize</code>
<code>\Glsxtrtitlefull</code>	49, 50
290, 291, 306	<code>\hss</code>
<code>\glxtrtitlefull</code>	310
290, 291, 306	<code>\hyperlink</code>
<code>\Glsxtrtitlefullpl</code>	78
290, 291, 307	<code>\hyperpage</code>
<code>\glxtrtitlefullpl</code>	163
290, 291, 307	<code>\hyperref</code>
<code>\Glsxtrtitlelong</code>	77, 115
290, 291, 305	hyperref package
<code>\glxtrtitlelong</code>	79, 163, 288, 301
290, 291, 305	
<code>\Glsxtrtitlelongpl</code>	I
290, 291, 306	<code>\if</code>
<code>\glxtrtitlelongpl</code>	46
290, 291, 305, 306	<code>\if@glxtr@autoseeindex</code>
<code>\Glsxtrtitlename</code>	23, 24, 38, 41
290, 291, 302	<code>\if@glxtr@format@override</code>
<code>\glxtrtitlename</code>	163
290, 291, 302	<code>\if@glxtrdocdefrestricted</code>
<code>\glxtrtitleorpdforheading</code>	44
23, 124, 125, 290, 291	<code>\if@glxtrindexcrossrefs</code>
<code>\Glsxtrtitleplural</code>	14, 42
290, 291, 303, 304	<code>\ifblank</code>
<code>\glxtrtitleplural</code>	26, 46, 47, 104
290, 291, 303	<code>\ifcase</code> ..
<code>\Glsxtrtitleshort</code>	7, 12, 19, 20, 22, 45, 56, 109, 322, 346
290, 291, 301, 302	<code>\ifcsdef</code>
<code>\glxtrtitleshort</code>	25, 31, 33–36, 57, 71, 72,
290, 291, 301	83–87, 95, 107, 112, 113, 124, 128, 131,
<code>\Glsxtrtitleshortpl</code>	133, 155–161, 171, 174, 190, 194, 313–320
290, 291, 302	<code>\ifcsstring</code>
<code>\glxtrtitleshortpl</code>	25, 151, 193
290, 291, 301	<code>\ifcsundef</code>
<code>\Glsxtrtitletext</code>	31, 33–
290, 291, 303	35, 44, 49, 51, 79, 90, 95–99, 112, 113,
<code>\glxtrtitletext</code>	116, 128, 151, 193–196, 310, 327, 334, 348
290, 291, 303	<code>\ifcsvoid</code>
<code>\GlsXtrTotalRecordCount</code>	42, 150
134	<code>\ifdef</code> 13, 18, 24, 27, 36, 37, 39, 40, 49, 50, 73,
<code>\glxtrtreetopindent</code>	77, 78, 84–86, 107, 110, 111, 122, 131,
326, 334	
<code>\glxtrundefaction</code> ..	
7, 13, 25, 33, 34, 36, 37	

153, 154, 166, 169, 262, 289, 301–307, 310–313, 321–325, 336–341, 344, 347, 348	\ifstrequal 16, 21
\ifdefempty 7–9, 29, 33–35, 38, 39, 57, 58, 89, 101, 104, 107, 115, 126, 129, 134, 168, 175, 190, 345	\ifthenelse 119, 120
\ifdefequal 44, 120, 129, 161	\IfTrackedLanguageFileExists 308
\ifdefstring 6, 31, 163, 168, 344	\ifundef 29, 104, 168–170
\ifdefvoid ... 37, 40–43, 78, 95, 112, 115, 130	\ifx 8–10, 39, 49, 50, 104, 108, 114, 115, 123, 163–165, 167, 173, 175–177, 271, 272, 342
\ifdim 49, 50, 103, 327–334	\immediate 91, 99, 116, 117, 123
\IfFileExists 21, 116, 119, 121, 123, 309	\index 163
\ifglossaryexists 37	\indexspace . 311, 322–325, 336–342, 344, 347
\ifglssacronym 17, 119	\input 307
\ifglssacrshortcuts 19	\inputencodingname 122
\ifglssautomake 107, 120, 123	\istfilename 104
\ifglssentrycounter 31	\item 118, 119, 310–313, 321–323, 337, 338
\ifglssentryexists 8, 36, 37, 46, 47, 50, 59, 129, 150, 151, 170	J
\ifglssfieldeq 149	\jobname 116, 118–121, 123
\ifglsshasfield 27, 262, 263	K
\ifglsshaslong 94, 95, 137, 138	\key@ifundefined 11, 12, 26, 71, 126, 129
\ifglsshasparent 124, 125, 127, 130, 327, 329, 330	\KV@glslink@hyperfalse 60, 73, 78, 79
\ifglsshashshort 38, 53, 59	\KV@glslink@hypertrue 79
\ifglsshashsymbol 171, 322–324, 326	\KV@glslink@noindexfalse 73, 74
\ifglssindexonlyfirst 75	\KV@glslink@noindextrue 74, 79
\ifglssnogroupskip 311, 313–323, 325, 336, 344	L
\ifglssnonumberlist 52	\LaTeX 118, 119
\ifglssnopostdot 15, 109	\leaders 310
\ifglsssanitizesort 106	\leavevmode 33, 56
\ifglssubentrycounter 31	\let 5, 7–13, 15, 17–19, 23–25, 27, 32, 43, 45, 48, 49, 51–70, 72–74, 76–80, 83, 89, 90, 98–105, 107–113, 120, 121, 123, 126, 127, 129, 134, 135, 155–165, 168–170, 173–177, 181–190, 192–194, 203, 225, 239, 261, 288–291, 321, 322, 326, 327, 338, 345–347
\ifglssused 42, 43, 73, 75, 91, 100, 103, 190, 327–329, 331–333	\letabbreviationstyle .. 202, 204, 206, 208, 211, 212, 217, 219, 232, 233, 249, 251
\ifglssxindy 116, 118	\letcs 26, 29, 38, 39, 43, 57, 71, 110–112, 128–130, 155–162, 348
\ifglssxtr@hyperoutside 57, 58	\levelchar 166
\ifglssxtrinitwrglossbefore 55, 57, 58, 135	\listadd 95
\ifglssxtrininsertinside 184–190, 192, 193, 197, 199, 201–211, 213–215, 217–224, 226, 228– 240, 242–245, 247–262, 264–268, 270, 272, 274, 277, 279, 280, 283, 284, 286, 287	\listbreak 168
\ifHy@hyperindex 163	\listcsadd 28
\ifinlistcs 28, 44	\listcseadd 28, 96
\ifinner 23	\listcsgadd 28, 44
\ifKV@glslink@hyper 53, 56–58	\listcsxadd 28, 96
\ifKV@glslink@local 55, 135	\loadglsentries 45, 117
\ifKV@glslink@noindex 8, 9, 11, 74, 75	\long 32
\ifmmode 23	M
\ifnum 14, 91, 99, 100, 112, 121, 134, 323, 324, 335	\MakeAcronymsAbbreviations 103
\ifstrempty 124, 131	

<code>\makeatletter</code>	116, 121, 165	212, 227, 241, 262, 263, 272, 274, 277, 280, 282, 285–287, 289–307, 309, 311, 321, 325–327, 334, 335, 343, 344, 347, 348
<code>\makeatother</code>	165	
<code>\makebox</code>	310, 335, 336	
<code>\makefirstuc</code>	169	<code>\newcount</code>
<code>makeglossaries</code>	110	14, 121, 131
<code>\makeglossaries</code>	104, 117–120, 123	<code>\newentry</code>
<code>\makeglossary</code>	104	18
<code>makeindex</code>	349	<code>\newglossary</code>
<code>makeindex</code>	103	16, 104
<code>\makenoidxglossaries</code>	118	<code>\newglossaryentry</code>
<code>\MakeTextUppercase</code>	290	18, 45, 89, 97, 101, 153, 154, 178
<code>\MakeUppercase</code>	290, 291	<code>\newglossaryentry options</code>
<code>\marginpar</code>	23	alias
<code>\markboth</code>	289	15, 37, 40–43
<code>\markright</code>	289	desc
<code>\maxdimen</code>	49, 50	142, 143, 147, 148
<code>\mbox</code>	312, 313, 335	descplural
<code>\medskip</code>	119, 120, 128	143, 148
<code>\MessageBreak</code>		first 77, 140, 146, 147, 196, 296, 297, 304, 349
.....	44, 45, 48, 91, 100, 104, 107, 108, 193	firstplural ..
<code>mfistuc package</code>	168, 169	141, 147, 196, 296, 297, 304, 349
<code>\mfistucMakeUppercase</code>	59–70, 72, 81, 82, 84, 87, 94, 102, 138–149, 157, 159, 162, 182, 183, 185, 186, 188, 190–192	group
<code>\mfu@checkword@arg</code>	168, 169	129
<code>\mfu@checkword@do</code>	169	loclist
		28
		long
		145, 148, 305
		longplural
		145, 149, 305
		name ...
		38, 138, 139, 146, 163, 293, 294, 302
		plural
		139, 140, 146, 196, 295, 296, 303
		see
		15, 24, 37, 38, 40, 43, 45, 105
		seealso
		15, 37, 38, 40, 42, 43, 360
		short
		144, 148, 175
		shortplural
		145, 148, 175
		symbol
		141, 142, 147
		symbolplural
		142, 147
		text
		77, 139, 146, 196, 198, 294, 295, 302
		<code>\newglossarystyle</code>
		344
		<code>\newif</code>
		55, 162, 196
		<code>\newlength</code>
		326
		<code>\newnum</code>
		18
		<code>\newrobustcmd</code>
		27, 29, 30, 39, 40, 71, 72, 83, 84, 93, 94, 109, 112, 124, 125, 128, 131–133, 135–137, 161, 168, 169, 180–190, 271, 289, 292–301, 327–333
		<code>\newsym</code>
		18
		<code>\newterm</code>
		153
		<code>\newtoks</code>
		174, 175
		<code>\newwrite</code>
		104
		<code>\nobreak</code>
		312, 313, 322–325, 337–340
		<code>\NoCaseChange</code>
		84–87, 125, 292–300
		<code>\noexpand</code> 10, 11, 21, 39, 41, 42, 101, 116, 117, 121, 127–129, 164, 165, 178, 309, 345, 346
		<code>\nofiles</code>
		119
		<code>\noindent</code>
		119, 120, 324, 325, 338–341
		<code>\nopagebreak</code>
		322–325, 336–343, 347
		<code>\nopostdesc</code>
		33, 46, 47, 108, 153
		<code>\nr</code>
		7, 12, 14, 19, 20, 22, 56, 109
		<code>\ns@GLSxtrfull</code>
		182

N

<code>\NeedsTeXFormat</code>	5, 309, 343
<code>\new@glossaryentry</code>	45, 107
<code>\new@ifnextchar</code>	71, 72, 93, 94, 131, 132, 135–137, 172, 180–189
<code>\newabbr</code>	17, 18
<code>\newabbreviation</code>	17, 18
<code>\newabbreviationhook</code>	178
<code>\newabbreviationstyle</code>	196, 198, 200–202, 204, 206, 208, 209, 211– 216, 218–221, 223, 225, 227–230, 232– 234, 236, 237, 239, 241–244, 246–249, 251, 253, 255, 257–260, 263, 264, 266– 269, 271–274, 276, 278–281, 283, 285–287
<code>\newacronym</code>	101, 102
<code>\newacronymhook</code>	101
<code>\newacronymstyle</code>	102, 103
<code>\newcommand</code>	5–8, 10–12, 14– 29, 31–39, 41–43, 45–48, 50–53, 55, 56, 59, 60, 71, 72, 74–76, 78, 79, 82–91, 93– 103, 107–129, 131–155, 160, 161, 163– 176, 178–190, 192–198, 200, 201, 206,

<code>\ns@Glsxtrfull</code>	181	<code>record</code>	7, 12, 43, 53, 104, 120, 358
<code>\ns@glxtrfull</code>	180	<code>alsoindex</code>	8, 10
<code>\ns@GLSxtrfullpl</code>	183	<code>only</code>	7, 8
<code>\ns@Glsxtrfullpl</code>	182	<code>shortcuts</code>	19
<code>\ns@glxtrfullpl</code>	182	<code>ac</code>	19
<code>\ns@GLSxtrlong</code>	186	<code>all</code>	19
<code>\ns@Glsxtrlong</code>	186	<code>false</code>	19
<code>\ns@glxtrlong</code>	185	<code>none</code>	19
<code>\ns@GLSxtrlongpl</code>	189	<code>true</code>	19
<code>\ns@Glsxtrlongpl</code>	189	<code>style</code>	21
<code>\ns@glxtrlongpl</code>	188	<code>stylemods</code>	21
<code>\ns@GLSxtrshort</code>	185	<code>symbols</code>	18, 153
<code>\ns@Glsxtrshort</code>	184	<code>undefaction</code>	36
<code>\ns@glxtrshort</code>	183, 184	<code>error</code>	6
<code>\ns@GLSxtrshortpl</code>	188	<code>warn</code>	6
<code>\ns@Glsxtrshortpl</code>	187	<code>xindy</code>	39, 40
<code>\ns@glxtrshortpl</code>	187	<code>\PackageError</code>	6, 11, 21, 24, 44, 45, 48, 49, 71, 72, 74, 83, 84, 89, 90, 97, 99, 100, 102, 104, 106, 107, 113, 123, 127, 128, 131, 193–196, 309, 310
<code>\null</code>	20	<code>\PackageWarning</code>	15
<code>\number</code>	96–99, 121, 131	<code>\PackageWarningNoLine</code>	15
<code>\numexpr</code>	96, 99	<code>\pageref</code>	31
O			
<code>\or</code>	7, 13, 19, 20, 22, 45, 56, 322, 346	<code>\par</code>	119, 120, 312, 313, 322–326, 335–341, 344
<code>\org@glossaryentrynumbers</code>	50, 108	<code>\parindent</code>	321, 323, 324, 326, 335, 336, 338–342, 345
<code>\org@glossarytitle</code>	108	<code>\parskip</code>	321, 323, 324, 338–340, 345
<code>\org@ifKV@glslink@hyper</code>	56, 58	<code>\PassOptionsToPackage</code>	5, 21
P			
<code>\p@gl@hyp@opt</code>	76	<code>\pdfbookmark</code>	344
package options:		<code>\preglossarypreamble</code>	31
<code>abbreviations</code>	16, 17	<code>\preto</code>	74
<code>accsupp</code>	20, 138	<code>\print@noop@unsrtglossaryunit</code>	11, 13
<code>acronym</code>	17	<code>\print@op@unsrtglossaryunit</code>	13
<code>automake</code>	107, 118, 123	<code>\printabbreviations</code>	17
<code>true</code>	123	<code>\printglossaries</code>	105, 118
<code>autoseeindex</code>	24	<code>\printglossary</code>	17, 105, 118
<code>false</code>	23	<code>\printglossary options</code>	
<code>debug</code>		<code>nonumberlist</code>	52
<code>showtargets</code>	78	<code>type</code>	107
<code>docdef</code>	14, 43, 44, 89, 97	<code>\printnoidxglossaries</code>	118
<code>false</code>	45	<code>\printnoidxglossary</code>	105, 106, 118
<code>restricted</code>	14	<code>\printnumbers</code>	18, 154
<code>true</code>	45	<code>\printsymbols</code>	18, 153
<code>docdefs</code>		<code>\printunsrtglossary</code>	126
<code>restricted</code>	44	<code>\printunsrtglossaryentryprocesshook</code>	126
<code>nonumberlist</code>	50	<code>\printunsrtglossaryhandler</code>	127, 128
<code>nopostdot</code>	15	<code>\printunsrtglossarypredoglossary</code>	127
<code>false</code>	15	<code>\printunsrtglossaryskipentry</code>	126, 127
<code>numbers</code>	18	<code>\printunsrtglossaryunit</code>	13, 128
<code>postdot</code>	15		

\printunsrtglossaryunitsetup	128
\ProcessOptions	310
\ProcessOptionsX	22
\protect	84– 87, 146–149, 176, 179, 196–211, 213– 225, 227–261, 263, 264, 266–271, 273– 276, 278, 279, 281–283, 285–287, 292–300
\protected@csedef	30, 326
\protected@csexdef	30, 327
\protected@edef	49, 77, 101, 112, 113, 128, 129, 163, 164, 178
\protected@write	10, 11, 44, 52, 104, 105, 120, 122–124, 347
\providecommand	16, 26, 39, 52, 72, 74, 91, 99, 104, 116, 117, 122, 310, 343
\ProvidesFile	307
\ProvidesPackage	5, 309, 343
Q	
\quotechar	166
R	
\raggedright	315, 316, 319, 320, 345
\relax	7, 10, 12–14, 17–20, 22, 24, 27, 45, 48–51, 55, 56, 76, 83, 90, 91, 99, 104, 105, 108, 111, 112, 114, 120, 121, 123, 130, 134, 164, 165, 167, 169, 171, 175–177, 271, 272, 322–324, 327–336, 340–342, 345–347
relese package	226
\renewcommand	6, 7, 13–17, 19–22, 24, 32–38, 44, 45, 48–53, 55, 71– 73, 75, 77–79, 83, 88–91, 94, 95, 97–109, 112–114, 116, 117, 128, 133, 153, 155– 160, 167–170, 179, 180, 194–271, 273– 276, 278–289, 310–325, 335–341, 345–347
\renewenvironment	311, 313–321, 323, 324, 335, 338–341, 344
\renewglossarystyle	310–325, 335–341
\renewrobustcmd	58, 78
\RequireGlossariesExtraLang	308
\RequirePackage	5, 20–22, 309, 343
\reserved@a	173
\reserved@b	173
\reserved@d	173
\RestoreAcronyms	102, 103
\rGLS	133
\rGlS	133
\rgls	133
\rGLSformat	137
\rGLSformat	135, 138
\rGLSpl	133
\rGlSpl	133
\rglsppl	133
\rGLSplformat	137
\rGlSplformat	136
\rglspplformat	136, 138
\romannumeral	326, 327, 334
S	
\sgls@hyp@opt	76
\sglsxtr@enabletagging	168
\sglsxtrifhasfield	29
\s@printunsrtglossary	125, 128
\seealsoname	39, 40
\seename	38
\setabbreviationstyle	102, 197, 206
\setacronymstyle	102, 103
\setentrycounter	114
\SetGenericNewAcronym	103
\setglossarystyle	22, 108, 310–313, 322, 324, 325, 336–342, 344
\setkeys	9, 21, 24, 57, 58, 75, 101, 108, 134, 176, 177
\setlength	49, 50, 321, 323, 324, 326, 335, 336, 338–340, 345
\settowidth	103, 326–334
\setupglossaries	5, 24
\sfcode	15, 16, 171, 321
\small	23
\space	6, 11, 39, 44, 46, 48, 74, 89–91, 97, 99, 100, 102–106, 108, 117, 119, 123, 127, 128, 131, 171, 175, 179, 198, 310, 311, 321–324, 326, 334, 343
\spacefactor	15, 16, 171, 177, 321
\string	6, 10, 11, 39, 40, 44–46, 48, 52, 57, 71, 72, 74, 83, 84, 89– 91, 97, 99, 100, 102–108, 116–120, 122– 124, 127, 128, 131, 157–160, 162, 164, 347
\strut	310–320, 324
\subglosstentry	108, 130, 310–320, 322–324, 335, 346
\subitem	321, 322
\subsubitem	322
T	
\tablehead	317–320
\tabletail	317–320
\tabularnewline	313–321

