

# `glossaries-extra.sty v1.28: documented code`

Nicola L.C. Talbot

Dickimaw Books

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

2018-03-06

## Abstract

This is the documented code for the glossaries-extra package. See [glossaries-extra-manual.pdf](#) for the user manual.

This package is experimental and not stable. It's provided for testing purposes only. Future versions may not be compatible with this version. Once it has stabilised I'll add it to CTAN, at which point compatibility with the first stable version will be maintained.

# Contents

<b>1</b>	<b>Main Package Code (glossaries-extra.sty)</b>	<b>5</b>
1.1	Package Initialisation and Options . . . . .	5
1.2	Extra Utilities . . . . .	25
1.3	Modifications to Commands Provided by glossaries . . . . .	33
1.3.1	Existence Checks . . . . .	37
1.3.2	Document Definitions . . . . .	45
1.3.3	Existing Glossary Style Modifications . . . . .	50
1.3.4	Entry Formatting, Hyperlinks and Indexing . . . . .	54
1.3.5	Entry Counting . . . . .	90
1.3.6	Acronym Modifications . . . . .	103
1.3.7	Indexing and Displaying Glossaries . . . . .	106
1.4	Link Counting . . . . .	141
1.5	Integration with glossaries-accsupp . . . . .	143
1.6	Categories . . . . .	153
1.7	Abbreviations . . . . .	179
1.7.1	Abbreviation Styles Setup . . . . .	198
1.7.2	Predefined Styles (Default Font) . . . . .	201
1.7.3	Predefined Styles (Small Capitals) . . . . .	218
1.7.4	Predefined Styles (Fake Small Capitals) . . . . .	232
1.7.5	Predefined Styles (Emphasized) . . . . .	246
1.7.6	Predefined Styles (User Parentheses Hook) . . . . .	268
1.7.7	Predefined Styles (Hyphen) . . . . .	277
1.7.8	Predefined Styles (No Short on First Use) . . . . .	291
1.8	Using Entries in Headings . . . . .	293
1.9	Multi-Lingual Support . . . . .	313
1.10	glossaries-extra-bib2gls.sty . . . . .	314
<b>2</b>	<b>Style Adjustments (glossaries-extra-stylemods.sty)</b>	<b>347</b>
2.1	Package Initialisation . . . . .	347
2.2	List-Like Styles . . . . .	348
2.3	Longtable Styles . . . . .	351
2.4	Long Ragged Styles . . . . .	353
2.5	Supertabular Styles . . . . .	355
2.6	Super Ragged Styles . . . . .	357
2.7	Inline Style . . . . .	359
2.8	Tree Styles . . . . .	359
2.9	Multicolumn Styles . . . . .	376

<b>3 bookindex style (glossary-bookindex.sty)</b>	<b>382</b>
3.1 Package Initialisation and Options . . . . .	382
<b>Glossary</b>	<b>388</b>
<b>Change History</b>	<b>389</b>
<b>Index</b>	<b>406</b>

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

## 1.1 Package Initialisation and Options

```
1 \NeedsTeXFormat{LaTeX2e}
2 \ProvidesPackage{glossaries-extra}[2018/03/06 v1.28 (NLCT)]
```

Requires xkeyval to define package options.

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

Requires etoolbox package.

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

Has glossaries already been loaded?

```
5 \@ifpackageloaded{glossaries}
6 {%
```

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

```
7   \newcommand{\glstr@dooption}[1]{\setupglossaries{#1}}%
8   \let\@glstr@declareoption\@gls@declareoption
9 }
10 {%
```

Not already loaded, so pass options to glossaries.

```
11   \newcommand{\glstr@dooption}[1]{%
12     \PassOptionsToPackage{#1}{glossaries}%
13   }%
```

Set the defaults.

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

Declare package options.

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

```

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

```

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

```

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

```

`\f@for@gl@sentries`

```

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

```

`\f@for@gl@sentries`

```

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

```

```

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

63\define@choicekey{glossaries-extra.sty}{undefaction}%
64  [\glstr@undefaction@val\glstr@undefaction@nr]%
65  {warn,error}%
66  {%
67    \ifcase\glstr@undefaction@nr\relax
68    \let\glstrundefaction\@glstr@warn@undefaction
69    \let\glstr@warnonexistssordo\@glstr@warn@onexistssordo
70    \let\@glstr@redef@forglsentries\@glstr@do@redef@forglsentries
71    \or
72    \let\glstrundefaction\@glstr@err@undefaction
73    \let\glstr@warnonexistssordo\@gobble
74    \let\@glstr@redef@forglsentries\relax
75    \fi
76  }

```

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.

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

`glstr@recordsee` Does nothing by default.

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

`ultnumberformat`

```
79\newcommand*{\@glstr@defaultnumberformat}{glsnnumberformat}%
```

`ultNumberFormat`

```

80\newcommand*{\GlsXtrSetDefaultNumberFormat}[1]{%
81  \renewcommand*{\@glstr@defaultnumberformat}{#1}%
82}%

```

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.

`record@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.

```

83 \newcommand*{\@glsxtr@do@record@wrglossary}[1]{%
84   \begingroup
85     \ifKV@glslink@noindex
86     \else
87       \edef\@gls@label{\glsdetoklabel{#1}}%
88       \let\glslabel\@gls@label
89       \glswriteentry{#1}%
90       {%
91         \ifdefempty{\@glsxtr@thevalue}%
92         {%
93           \ifx\@glsxtr@org@theHvalue\@glsxtr@theHvalue
94           \else
95             \let\theHglentrycounter\@glsxtr@theHvalue
96             \fi
97             \glsxtr@saveentrycounter
98             \let\@@do@wrglossary\@glsxtr@dorecord
99           }%
100          {%
101            \let\theHglentrycounter\@glsxtr@thevalue
102            \let\theHglentrycounter\@glsxtr@theHvalue
103            \let\@@do@wrglossary\@glsxtr@dorecordnodefer
104          }%
105          \ifx\@glsxtr@record@setting\@glsxtr@record@setting@alsoindex
106            \glsxtr@@do@wrglossary{#1}%
107          \else
108            \@glsxtrwrglossmark
109            \@do@wrglossary
110          \fi
111        }%
112      \fi
113 \endgroup
114 }
```

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

```

115 \newcommand*{\glsxtr@do@alsoindex@wrglossary}[1]{%
116   \glsxtr@@do@wrglossary{#1}%
117   \@glsxtr@do@record@wrglossary{#1}%
118 }
```

`\@glsxtr@record` The `record=only` option sets `\@glsxtr@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`).

```

119 \newcommand*{\@glsxtr@record}[3]{%
```



```

120 \ifglentryexists{#2}{}%
121 {%
122   \@@glxtrwrglossmark
123   \begingroup

```

Save the label in case it's needed.

```

124   \edef\@gls@label{\glstoklabel{#2}}%
125   \let\glslabel\@gls@label
126   \let\@glsnumberformat\@glxtr@defaultnumberformat
127   \def\@glxtr@thevalue{%
128     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
129     \let\@glxtr@org@theHvalue\@glxtr@theHvalue

```

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

```

130   \def\@gls@counter{page}%

```

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

```

131   \@gls@setdefault@glslink@opts
132   \setkeys{#3}{#1}%
133   \ifKV@glslink@noindex
134   \else
135     \glswriteentry{#2}%
136     {%

```

Check if thevalue has been set.

```

137       \ifdefempty{\@glxtr@thevalue}%
138       {%

```

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

```

139         \ifx\@glxtr@org@theHvalue\@glxtr@theHvalue
140         \else
141           \let\theHglentrycounter\@glxtr@theHvalue
142         \fi

```

Save the entry counter.

```

143         \glxtr@saveentrycounter

```

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

```

144         \let\@@do@wrglossary\glxtr@dorecord
145        }%
146         {%

```

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.)

```

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

```

No need to escape special characters.

```

154         \@@do@@wrglossary
155         \fi
156     }%
157     \fi
158 \endgroup
159 }%
160 }

```

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

```

161 \newcommand*\@glsxtr@dorecord{%
162     \global\let\@glsrecordlocref\theglsentrycounter
163     \let\@glsxtr@orgprefix\@glo@counterprefix
164     \ifx\theglsentrycounter\theHglentrycounter
165         \def\@glo@counterprefix{%
166     \else
167         \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
168             {\theglsentrycounter}{\theHglentrycounter}%
169         }%
170         \@do@gls@getcounterprefix
171     \fi
172     \protected@write\@auxout{\let\@glsrecordlocref\relax}{\string\glsxtr@record
173         {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
174         {\@glsrecordlocref}}%
175     \@glsxtr@counterrecordhook
176     \let\@glo@counterprefix\@glsxtr@orgprefix
177 }

```

`dorecordnodefer` As above, but don't defer expansion of location. This uses `\theglsentrycounter` directly for the location rather than `\@glslocref` since there's no need to guard against premature expansion of the page counter.

```

178 \newcommand*\@glsxtr@dorecordnodefer{%
179     \ifx\theglsentrycounter\theHglentrycounter
180         \protected@write\@auxout{{\string\glsxtr@record
181             {\@gls@label}{\@gls@counter}{\@glsnumberformat}%
182             {\theglsentrycounter}}%
183     \else
184         \edef\@do@gls@getcounterprefix{\noexpand\@gls@getcounterprefix
185             {\theglsentrycounter}{\theHglentrycounter}%
186         }%
187         \@do@gls@getcounterprefix
188         \protected@write\@auxout{{\string\glsxtr@record
189             {\@gls@label}{\@glo@counterprefix}{\@gls@counter}{\@glsnumberformat}%
190             {\theglsentrycounter}}%
191     \fi
192     \@glsxtr@counterrecordhook
193 }

```

r@recordcounter

```
194 \newcommand*{\@glsxtr@recordcounter}{%
195   \@glsxtr@noop@recordcounter
196 }
```

p@recordcounter

```
197 \newcommand*{\@glsxtr@noop@recordcounter}[1]{%
198   \PackageError{glossaries-extra}{\string\GlsXtrRecordCounter\space
199     requires record=only or record=alsoindex package option}{}%
200 }
```

p@recordcounter

```
201 \newcommand*{\@glsxtr@op@recordcounter}[1]{%
202   \eappto\@glsxtr@counterrecordhook{\noexpand\@glsxtr@docounterrecord{#1}}%
203 }
```

lsxtr@recordsee Deal with \glssee in record mode.

```
204 \newcommand*{\@glsxtr@recordsee}[2]{%
205   \@glsxtrwrglossmark
206   \def\@gls@xref{#2}%
207   \@onelevel@sanitize\@gls@xref
208   \protected@write\@auxout{}\{\string\glsxtr@recordsee{#1}\{\@gls@xref}\}%
209 }
```

srtglossaryunit

```
210 \newcommand{\printunsrtglossaryunit}{%
211   \print@noop@unsrtglossaryunit
212 }
```

tr@setup@record Initialise.

```
213 \newcommand*{\glsxtr@setup@record}{\let\@do@wrglossary\glsxtr@do@wrglossary}
```

aveentrycounter Only store the entry counter information if the indexing is on.

```
214 \newcommand*{\glsxtr@indexonly@saveentrycounter}{%
215   \ifKV@glslink@noindex
216   \else
217     \glsxtr@saveentrycounter
218   \fi
219 }
```

addloclistfield

```
220 \newcommand*{\glsxtr@addloclistfield}{%
221   \key@ifundefined{glossentry}{loclist}{%
222     {%
223       \define@key{glossentry}{loclist}{\def\@glo@loclist{##1}}%
224       \appto\@gls@keymap{,{loclist}{loclist}}%
225       \appto\@newglossaryentryprehook{\def\@glo@loclist{}}%
226       \appto\@newglossaryentryposthook{%
```

```

227 \gls@assign@field{}{\@glo@label}{loclist}{\@glo@loclist}%
228 }%
229 \glssetnoexpandfield{loclist}%
230 }%
231 {}%

```

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

```

232 \key@ifundefined{glossentry}{location}%
233 {%
234 \define@key{glossentry}{location}{\def\@glo@location{##1}}%
235 \appto\@gls@keymap{,{location}{location}}%
236 \appto\@newglossaryentryprehook{\def\@glo@location{}}%
237 \appto\@newglossaryentryposthook{%
238 \gls@assign@field{}{\@glo@label}{location}{\@glo@location}%
239 }%
240 \glssetnoexpandfield{location}%
241 }%
242 {}%

```

Add a key to store the group heading.

```

243 \key@ifundefined{glossentry}{group}%
244 {%
245 \define@key{glossentry}{group}{\def\@glo@group{##1}}%
246 \appto\@gls@keymap{,{group}{group}}%
247 \appto\@newglossaryentryprehook{\def\@glo@group{}}%
248 \appto\@newglossaryentryposthook{%
249 \gls@assign@field{}{\@glo@label}{group}{\@glo@group}%
250 }%
251 \glssetnoexpandfield{group}%
252 }%
253 {}%
254 }

```

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

```

255 \newcommand*{\@glsxtr@record@setting}{off}

```

etting@alsoindex

```

256 \newcommand*{\@glsxtr@record@setting@alsoindex}{alsoindex}

```

rd@setting@only

```

257 \newcommand*{\@glsxtr@record@setting@only}{only}

```

ord@setting@off

```

258 \newcommand*{\@glsxtr@record@setting@off}{off}

```

Now define the record package option.

```

259 \define@choicekey{glossaries-extra.sty}{record}
260 [\@glsxtr@record@setting\glsxtr@record@nr]%
261 {off,only,alsoindex}%

```

```

262 [only]%
263 {%
264 \ifcase\glxtr@record@nr\relax

```

Don't record.

```

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

```

Only record (don't index).

```

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

```

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

```

289 \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.

```

290 \ifdef\@glxtr@setupsort@none{\@glxtr@setupsort@none}{}%

```

Load glossaries-extra-bib2gls:

```

291 \RequirePackage{glossaries-extra-bib2gls}%
292 }%
293 \or

```

Record and index. This option doesn't load glossaries-extra-bib2gls as the sorting is performed by xindy or makeindex.

```

294 \def\glxtr@setup@record{%
295 \renewcommand*{\@do@seeglossary}{\@glxtr@dosee@alsoindex@glossary}%
296 \let\@glxtr@record\@glxtr@record
297 \let\@do@wrglossary\glxtr@do@alsoindex@wrglossary

```

```

298 \let\@gls@saveentrycounter\glsxtr@indexonly@saveentrycounter
299 \let\glsxtrundefaction\@glsxtr@warn@undefaction
300 \let\glsxtr@warnonexistssordo\@glsxtr@warn@onexistssordo
301 \glsxtr@addloclistfield
302 \let\@glsxtr@recordcounter\@glsxtr@op@recordcounter
303 \def\printunsrtglossaryunit{\print@op@unsrtglossaryunit}%
304 \undef\glsxtrsetaliasnoindex
305 }%
306 \fi
307 }

```

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.

```

\glsxtr@docdefval The docdef value is stored as an integer: 0 (false), 1 (true) and 2 (restricted).
308 \newcommand*{\@glsxtr@docdefval}{0}

```

Need to provide conditional commands that are backward compatible:

```

\if@glsxtrdocdef
309 \newcommand*{\if@glsxtrdocdef}{\ifnum\@glsxtr@docdefval>0 }

\glsxtrdocdeftrue
310 \newcommand*{\@glsxtrdocdeftrue}{\def\@glsxtr@docdefval{1}}

\glsxtrdocdeffalse
311 \newcommand*{\@glsxtrdocdeffalse}{\def\@glsxtr@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.

```

312 \define@choicekey{glossaries-extra.sty}{docdef}
313 [\@glsxtr@docdefsetting\@glsxtr@docdefval]%
314 {false,true,restricted}[true]%
315 {%
316 \ifnum\@glsxtr@docdefval=2\relax
317 \renewcommand*{\@glsdoifexistssorwarn}{\glsdoifexists}%
318 \fi
319 }

```

```

\docdefrestricted
320 \newcommand*{\if@glsxtrdocdefrestricted}{\ifnum\@glsxtr@docdefval=2 }

```

```

\ifexistssorwarn 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).

```

```

321 \newcommand*{\@glsdoifexistssorwarn}{\glsdoifexistssorwarn}

```

**indexcrossrefs** Automatically index cross references at the end of the document

```

322 \define@boolkey{glossaries-extra.sty}[@glxtr]{indexcrossrefs}[true]{%
323   \if@glxtrindexcrossrefs
324   \else
325     \renewcommand*{\@glxtr@autoindexcrossrefs}{}%
326   \fi
327 }

```

Switch off since this can increase the build time.

```

328 \@glxtrindexcrossrefsfalse

```

But allow see key to switch it on automatically.

**oindexcrossrefs**

```

329 \newcommand*{\@glxtr@autoindexcrossrefs}{\@glxtrindexcrossrefstrue}

```

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

```

330 \define@boolkey{glossaries-extra.sty}[@glxtr@]{autoseeindex}[true]{%
331 }
332 \@glxtr@autoseeindextrue

```

**iesExtraWarning** Allow users to suppress warnings.

```

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

```

**raWarningNoLine** Allow users to suppress warnings.

```

334 \newcommand*{\GlossariesExtraWarningNoLine}[1]{%
335   \PackageWarningNoLine{glossaries-extra}{#1}}

336 \@glxtr@declareoption{nowarn}{%
337   \let\GlossariesExtraWarning\@gobble
338   \let\GlossariesExtraWarningNoLine\@gobble
339   \glxtr@doption{nowarn}%
340 }

```

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

```

341 \newcommand*{\@glxtr@defpostpunc}{}

```

**postdot** Shortcut for nopostdot=false

```

342 \@glxtr@declareoption{postdot}{%
343   \glxtr@doption{nopostdot=false}%
344   \renewcommand*{\@glxtr@defpostpunc}{%
345     \renewcommand*{\glspostdescription}{%
346       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
347   }%
348 }

```

nopostdot Needs to redefine \@glsxtr@defpostpunc

```
349 \define@choicekey{glossaries-extra.sty}{nopostdot}{true,false}[true]{%
350   \glsxtr@doption{nopostdot=#1}%
351   \renewcommand*{\@glsxtr@defpostpunc}{%
352     \renewcommand*{\glspostdescription}{%
353       \ifglsnopostdot\else.\spacefactor\sfcode'\. \fi}%
354   }%
355 }
```

postpunc Set the post-description punctuation. This also sets the \ifglsnopostdot conditional, which now indicates if the post-description punctuation has been suppressed.

```
356 \define@key{glossaries-extra.sty}{postpunc}{%
357   \glsxtr@doption{nopostdot=false}%
358   \ifstrequal{#1}{dot}%
359   {%
360     \renewcommand*{\@glsxtr@defpostpunc}{%
361       \renewcommand*{\glspostdescription}{.\spacefactor\sfcode'\. }%
362     }%
363   }%
364   {%
365     \ifstrequal{#1}{comma}%
366     {%
367       \renewcommand*{\@glsxtr@defpostpunc}{%
368         \renewcommand*{\glspostdescription}{,}%
369       }%
370     }%
371     {%
372       \ifstrequal{#1}{none}%
373       {%
374         \glsxtr@doption{nopostdot=true}%
375         \renewcommand*{\@glsxtr@defpostpunc}{%
376           \renewcommand*{\glspostdescription}{}%
377         }%
378       }%
379       {%
380         \renewcommand*{\@glsxtr@defpostpunc}{%
381           \renewcommand*{\glspostdescription}{#1}%
382         }%
383       }%
384     }%
385   }%
386 }
```

glsxtrabbrvtype Glossary type for abbreviations.

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

abbreviationsdef Set by abbreviations option.

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



bbreviationsdef

```

389 \newcommand*{\@glxtr@doabbreviationsdef}{%
390   \ifpackageloaded{babel}%
391     {\providecommand{\abbreviationsname}{\acronymname}}%
392     {\providecommand{\abbreviationsname}{Abbreviations}}%
393   \newglossary[glg-abr]{abbreviations}{glg-abr}{glo-abr}{\abbreviationsname}%
394   \renewcommand*{\glxtrabbrvtype}{abbreviations}%
395   \newcommand*{\printabbreviations}[1][1]{%
396     \printglossary[type=\glxtrabbrvtype,##1]%
397   }%
398   \disable@keys{glossaries-extra.sty}{abbreviations}%

   If the acronym option hasn't been used, change \acronymtype to \glxtrabbrvtype.
399   \ifglssacronym
400   \else
401     \renewcommand*{\acronymtype}{\glxtrabbrvtype}%
402   \fi
403 }%
```

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

```

404 \@glxtr@declareoption{abbreviations}{%
405   \let\@glxtr@abbreviationsdef\@glxtr@doabbreviationsdef
406 }
```

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

```

407 \newcommand*{\GlsXtrDefineAbbreviationShortcuts}{%
408   \newcommand*{\ab}{\cglg}%
409   \newcommand*{\abp}{\cglgpl}%
410   \newcommand*{\as}{\glxtrshort}%
411   \newcommand*{\asp}{\glxtrshortpl}%
412   \newcommand*{\al}{\glxtrlong}%
413   \newcommand*{\alp}{\glxtrlongpl}%
414   \newcommand*{\af}{\glxtrfull}%
415   \newcommand*{\afp}{\glxtrfullpl}%
416   \newcommand*{\Ab}{\cGls}%
417   \newcommand*{\Abp}{\cGlspl}%
418   \newcommand*{\As}{\Glsxtrshort}%
419   \newcommand*{\Asp}{\Glsxtrshortpl}%
420   \newcommand*{\Al}{\Glsxtrlong}%
421   \newcommand*{\Alp}{\Glsxtrlongpl}%
422   \newcommand*{\Af}{\Glsxtrfull}%
423   \newcommand*{\Afp}{\Glsxtrfullpl}%
424   \newcommand*{\AB}{\cGLS}%
425   \newcommand*{\ABP}{\cGLSpl}%
426   \newcommand*{\AS}{\GLSxtrshort}%
427   \newcommand*{\ASP}{\GLSxtrshortpl}%
428   \newcommand*{\AL}{\GLSxtrlong}%

```

```

429 \newcommand*{\ALP}{\GLSxtrlongpl}%
430 \newcommand*{\AF}{\GLSxtrfull}%
431 \newcommand*{\AFP}{\GLSxtrfullpl}%

432 \providecommand*{\newabbr}{\newabbreviation}%

  Disable this command after it's been used.
433 \let\GlsXtrDefineAbbreviationShortcuts\relax
434 }

```

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

```

435 \newcommand*{\GlsXtrDefineAcShortcuts}{%
436 \newcommand*{\ac}{\cgl}%
437 \newcommand*{\acp}{\cglsp}%
438 \newcommand*{\acs}{\glxtrshort}%
439 \newcommand*{\acsp}{\glxtrshortpl}%
440 \newcommand*{\acl}{\glxtrlong}%
441 \newcommand*{\aclp}{\glxtrlongpl}%
442 \newcommand*{\acf}{\glxtrfull}%
443 \newcommand*{\acfp}{\glxtrfullpl}%
444 \newcommand*{\Ac}{\cGls}%
445 \newcommand*{\Acp}{\cGlspl}%
446 \newcommand*{\Acs}{\Glsxtrshort}%
447 \newcommand*{\Acsp}{\Glsxtrshortpl}%
448 \newcommand*{\Acl}{\Glsxtrlong}%
449 \newcommand*{\Aclp}{\Glsxtrlongpl}%
450 \newcommand*{\Acf}{\Glsxtrfull}%
451 \newcommand*{\Acfp}{\Glsxtrfullpl}%
452 \newcommand*{\AC}{\cGLS}%
453 \newcommand*{\ACP}{\cGLSpl}%
454 \newcommand*{\ACS}{\GLSxtrshort}%
455 \newcommand*{\ACSP}{\GLSxtrshortpl}%
456 \newcommand*{\ACL}{\GLSxtrlong}%
457 \newcommand*{\ACLP}{\GLSxtrlongpl}%
458 \newcommand*{\ACF}{\GLSxtrfull}%
459 \newcommand*{\ACFP}{\GLSxtrfullpl}%

460 \providecommand*{\newabbr}{\newabbreviation}%

  Disable this command after it's been used.
461 \let\GlsXtrDefineAcShortcuts\relax
462 }

```

**OtherShortcuts** Similarly provide shortcut versions for the commands provided by the symbols and numbers options.

```

463 \newcommand*{\GlsXtrDefineOtherShortcuts}{%
464 \newcommand*{\newentry}{\newglossaryentry}%
465 \ifdef\printsymbols
466 {%

```

```

467 \newcommand*{\newsym}{\glxtrnewsymbol}%
468 }{}%
469 \ifdef\printnumbers
470 {%
471 \newcommand*{\newnum}{\glxtrnewnumber}%
472 }{}%
473 \let\GlsXtrDefineOtherShortcuts\relax
474 }

```

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.

```
475 \newcommand*{\@glxtr@setupshortcuts}{}%
```

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

```
476 \newcommand*{\@glxtr@shortcutsval}{\ifglssacrshortcuts 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).

```

477 \define@choicekey{glossaries-extra.sty}{shortcuts}%
478 [\@glxtr@shortcutsval\@glxtr@shortcutsnr]%
479 {acronyms,acro,abbreviations,abbr,other,all,true,ac,none,false}[true]{%
480 \ifcase\@glxtr@shortcutsnr\relax % acronyms
481 \renewcommand*{\@glxtr@setupshortcuts}{%
482 \glssacrshortcutstrue
483 \DefineAcronymSynonyms
484 }%
485 \or % acro
486 \renewcommand*{\@glxtr@setupshortcuts}{%
487 \glssacrshortcutstrue
488 \DefineAcronymSynonyms
489 }%
490 \or % abbreviations
491 \renewcommand*{\@glxtr@setupshortcuts}{%
492 \GlsXtrDefineAbbreviationShortcuts
493 }%
494 \or % abbr
495 \renewcommand*{\@glxtr@setupshortcuts}{%
496 \GlsXtrDefineAbbreviationShortcuts
497 }%
498 \or % other
499 \renewcommand*{\@glxtr@setupshortcuts}{%
500 \GlsXtrDefineOtherShortcuts
501 }%

```

```

502 \or % all
503   \renewcommand*{\@glxtr@setupshortcuts}{%
504     \glsacrshortcutstrue
505     \GlsXtrDefineAcShortcuts
506     \GlsXtrDefineAbbreviationShortcuts
507     \GlsXtrDefineOtherShortcuts
508   }%
509 \or % true
510   \renewcommand*{\@glxtr@setupshortcuts}{%
511     \glsacrshortcutstrue
512     \GlsXtrDefineAcShortcuts
513     \GlsXtrDefineAbbreviationShortcuts
514     \GlsXtrDefineOtherShortcuts
515   }%
516 \or % ac
517   \renewcommand*{\@glxtr@setupshortcuts}{%
518     \glsacrshortcutstrue
519     \GlsXtrDefineAcShortcuts
520   }%

```

Leave none and false as last option.

```

521 \else % none, false
522   \renewcommand*{\@glxtr@setupshortcuts}{}%
523 \fi
524 }

```

**lsxtr@doaccsupp**

```

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

```

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

```

526 \@glxtr@declareoption{accsupp}{%
527 \renewcommand*{\@glxtr@doaccsupp}{\RequirePackage{glossaries-accsupp}}}

```

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

```

528 \newcommand{\glxtrNoGlossaryWarning}[1]{%
529 \@glxtr@defaultnoglossarywarning{#1}%
530 }

```

**omissingglstext** If true, suppress the text produced if the external glossary file is missing.

```

531 \define@choicekey{glossaries-extra.sty}{omissingglstext}
532 [ \@glxtr@nomissingglstextval \@glxtr@nomissingglstextnr ]%
533 {true,false}[true]{%
534   \ifcase \@glxtr@nomissingglstextnr \relax % true
535   \renewcommand{\glxtrNoGlossaryWarning}[1]{\null}%
536   \else % false

```

```

537     \renewcommand{\glxtrNoGlossaryWarning}[1]{%
538         \@glxtr@defaultnoglossarywarning{#1}%
539     }%
540     \fi
541 }

```

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

xtr@redefstyles

```

542 \newcommand*{\@glxtr@redefstyles}{%

```

stylemods

```

543 \define@key{glossaries-extra.sty}{stylemods}[default]{%
544     \ifstrequal{#1}{default}%
545     {%
546         \renewcommand*{\@glxtr@redefstyles}{%
547             \RequirePackage{glossaries-extra-stylemods}}%
548     }%
549     {%
550         \ifstrequal{#1}{all}%
551         {%
552             \renewcommand*{\@glxtr@redefstyles}{%
553                 \PassOptionsToPackage{all}{glossaries-extra-stylemods}%
554                 \RequirePackage{glossaries-extra-stylemods}%
555             }%
556         }%
557         {%
558             \renewcommand*{\@glxtr@redefstyles}{}%
559             \@for\@glxtr@tmp:=#1\do{%
560                 \IfFileExists{glossary-\@glxtr@tmp.sty}%
561                 {%
562                     \eappto\@glxtr@redefstyles{%
563                         \noexpand\RequirePackage{glossary-\@glxtr@tmp}}%
564                 }%
565                 {%
566                     \PackageError{glossaries-extra}%
567                     {Glossaries style package ‘glossary-\@glxtr@tmp.sty’
568                     doesn’t exist (did you mean to use the ‘style’ key?)}%
569                     {The list of values (#{1}) in the ‘stylemods’ key should
570                     match the glossary-xxx.sty files provided with
571                     glossaries.sty}%
572                 }%
573             }%
574             \appto\@glxtr@redefstyles{\RequirePackage{glossaries-extra-stylemods}}%
575         }
576     }%
577 }

```

glxtr@do@style

```

578 \newcommand*{\@glxtr@do@style}{%

```

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

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

Defer actual style change:

```
580 \renewcommand*{\@glxstr@do@style}{%
```

Set this as the default style:

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

Set this style:

```
582 \setglossarystyle{#1}%
```

```
583 }%
```

```
584 }
```

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

```
585 \newcommand*{\@glxstrwrglossmark}{}
```

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

```
586 \newcommand*{\@@glxstrwrglossmark}{}
```

```
587 \AtBeginDocument{\renewcommand*{\@glxstrwrglossmark}{\@glxstrwrglossmark}}
```

**sxtrwrglossmark** Does nothing by default.

```
588 \newcommand*{\glxstrwrglossmark}{\ensuremath{\cdot}}
```

**debug** Provide extra debug options.

```
589 \define@choicekey{glossaries-extra.sty}{debug}
```

```
590 [\@glxstr@debugval\@glxstr@debugnr]%
```

```
591 {true,false,showtargets,showwrgloss,all}[true]{%
```

```
592 \ifcase\@glxstr@debugnr\relax % true
```

```
593 \glxstr@doooption{debug=true}%
```

```
594 \renewcommand*{\@glxstrwrglossmark}{}%
```

```
595 \or % false
```

```
596 \glxstr@doooption{debug=false}%
```

```
597 \renewcommand*{\@glxstrwrglossmark}{}%
```

```
598 \or % showtargets
```

```
599 \glxstr@doooption{debug=showtargets}%
```

```
600 \or % showwrgloss
```

```
601 \glxstr@doooption{debug=true}%
```

```
602 \renewcommand*{\@glxstrwrglossmark}{\glxstrwrglossmark}%
```

```
603 \or % all
```

```
604 \glxstr@doooption{debug=showtargets}%
```

```
605 \renewcommand*{\@glxstrwrglossmark}{\glxstrwrglossmark}%
```

```
606 \fi
```

```
607 }
```

Pass all other options to `glossaries`.

```
608 \DeclareOptionX*{%
```

```
609 \expandafter\glxstr@doooption\expandafter{\CurrentOption}}
```

Process options.

```
610 \ProcessOptionsX
```

Load glossaries if not already loaded.

```
611 \RequirePackage{glossaries}
```

Load the glossaries-accsupp package if required.

```
612 \@glsxtr@doaccsupp
```

Redefine \glspostdescription if required.

```
613 \@glsxtr@defpostpunc
```

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

```
614 \def\glsshowtarget#1{%
```

```
615 \glsxtrtitleorpdforheading
```

```
616 {%
```

```
617 \ifmmode
```

```
618 \texttt{\small [#1]}%
```

```
619 \else
```

```
620 \ifinner
```

```
621 \texttt{\small [#1]}%
```

```
622 \else
```

```
623 \marginpar{\texttt{\small #1}}%
```

```
624 \fi
```

```
625 \fi
```

```
626 }%
```

```
627 {[#1]}%
```

```
628 {\texttt{\small [#1]}}%
```

```
629 }
```

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

```
630 \let\@glsxtr@org@doseeglossary\@do@seeglossary
```

`r@doseeglossary`

```
631 \newcommand*{\@glsxtr@doseeglossary}[2]{%
```

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

```
633 {%
```

```
634 \@glsxtrwrglossmark
```

```
635 \@glsxtr@org@doseeglossary{#1}{#2}%
```

```
636 }%
```

```
637 }
```

`oindex@glossary`

```
638 \newcommand*{\@glsxtr@dosee@alsoindex@glossary}[2]{%
```

```
639 \@glsxtr@recordsee{#1}{#2}%
```

```
640 \@glsxtr@doseeglossary{#1}{#2}%
```

```
641 }
```

@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.)

```
642 \let\@glxtr@org@gloautosee\@glo@autosee
```

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

```
643 \if@glxtr@autoseeindex
644 \else
645   \ifdef\@glxtr@org@gloautosee
646   {}%
647   {\PackageError{glossaries-extra}{‘autoseeindex=false’ package
648     option requires at least v4.30 of glossaries.sty}%
649     {You need to update the glossaries.sty package}%
650   }
651 \fi
```

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

```
652 \ifdef\@glo@autosee
653 {%
654   \renewcommand*{\@glo@autosee}{%
655     \if@glxtr@autoseeindex\@glxtr@org@gloautosee\fi}%
656 }%
657 {}
```

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

```
658 \renewcommand*{\gls@checkseeallowed}{%
659   \if@glxtr@autoseeindex\@gls@see@noindex\fi
660 }
```

Define abbreviations glossaries if required.

```
661 \@glxtr@abbreviationsdef
662 \let\@glxtr@abbreviationsdef\relax
```

Setup shortcuts if required.

```
663 \@glxtr@setupshortcuts
```

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

```
664 \@glxtr@redef@for@gl@sentries
```

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

```
665 \renewcommand{\glxtr@doption}[1]{\setupglossaries{#1}}%
```

Now define the user command:

```
666 \newcommand*{\glossariesextrasetup}[1]{%
667   \let\glxtr@setup@record\relax
668   \let\@glxtr@setupshortcuts\relax
```



```

669 \let\@glxtr@redef@forglsentries\relax
670 \setkeys{glossaries-extra.sty}{#1}%
671 \@glxtr@abbreviationsdef
672 \let\@glxtr@abbreviationsdef\relax
673 \@glxtr@setupshortcuts
674 \glxtr@setup@record
675 \@glxtr@redef@forglsentries
676 }

@@do@wrglossary Save original definition of \@@do@wrglossary.
677 \let\glxtr@org@@do@wrglossary\@@do@wrglossary

@@do@wrglossary The new version adds code that can show a marker for debugging.
678 \newcommand*{\glxtr@@do@wrglossary}[1]{%
679 \@@glxtrwrglossmark
680 \glxtr@org@@do@wrglossary{#1}%
681 }

saveentrycounter Save original definition of \@gls@saveentrycounter.
682 \let\glxtr@saveentrycounter\@gls@saveentrycounter

saveentrycounter Change \@gls@saveentrycounter so that it only stores the entry counter information if the
indexing is on.
683 \let\@gls@saveentrycounter\glxtr@indexonly@saveentrycounter

Provide script dialect hook (does nothing unless redefined by glossaries-extra-bib2gls).

sxdialecthook
684 \newcommand*{\@glxtrdialecthook}{}

Set up record option if required.
685 \glxtr@setup@record

Disable preamble-only options and switch on the undefined tag at the start of the docu-
ment.
686 \AtBeginDocument{%
687 \disable@keys{glossaries-extra.sty}{abbreviations,docdef,record}%
688 \def\@glxtrundeftag{\glxtrundeftag}%
689 }

```

## 1.2 Extra Utilities

```

rifemptyglossary \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 `\glo@list@{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.)

```

690 \newcommand{\glxtrifemptyglossary}[3]{%
691   \ifcsdef{glo@list@#1}%
692   {%
693     \ifcsstring{glo@list@#1}{,}{#2}{#3}%
694   }%
695   {%
696     \glxtrundefaction{Glossary type '#1' doesn't exist}{}%
697     #2%
698   }%
699 }

```

`\glxtrifkeydefined` Tests if the key given in the first argument has been defined.

```

700 \newcommand*{\glxtrifkeydefined}[3]{%
701   \key@ifundefined{glossentry}{#1}{#3}{#2}%
702 }

```

`\glxtrprovidestoragekey` Like `\glxaddstoragekey` but does nothing if the key has already been defined.

```

703 \newcommand*{\glxtrprovidestoragekey}{%
704   \@ifstar\@sglxtr@provide@storagekey\@glxtr@provide@storagekey
705 }

```

`\glxtrprovide@storagekey` Unstarred version.

```

706 \newcommand*{\@glxtr@provide@storagekey}[3]{%
707   \key@ifundefined{glossentry}{#1}%
708   {%
709     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
710     \appto\@gls@keymap{, {#1}{#1}}%
711     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
712     \appto\@newglossaryentryposthook{%
713       \letcs{@glo@tmp}{@glo@#1}%
714       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
715     }%

```

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

```

716   \ifblank{#3}
717   {}%
718   {%
719     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
720   }%
721 }%
722 {%

```

Provide the no-link command if not already defined.

```

723 \ifblank{#3}
724 {}%
725 {%
726 \providecommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
727 }%
728 }%
729 }

```

vide@storagekey Starred version.

```

730 \newcommand*{\s@glstr@provide@storagekey}[1]{%
731 \key@ifundefined{glossentry}{#1}%
732 {%
733 \expandafter\newcommand\expandafter*\expandafter
734 {\csname gls@assign@#1@field\endcsname}[2]{%
735 \@gls@expand@field{##1}{#1}{##2}%
736 }%
737 }%
738 {}%
739 \@glstr@provide@addstoragekey{#1}%
740 }

```

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

```

741 \newcommand{\GlsXtrFmtField}{useri}

```

tDefaultOptions

```

742 \newcommand{\GlsXtrFmtDefaultOptions}{noindex}

```

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

```

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

```

`\@glstrfmt` Unstarred form.

```

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

```

`\s@glstrfmt` Starred form.

```

745 \newcommand*{\s@glstrfmt}[3][\@glstrfmt{#1}{#2}{#3}}%
746 \new@ifnextchar[\s@glstrfmt{#1}{#2}{#3}}%
747 {\@glstrfmt{#1}{#2}{#3}{}%
748 }

```

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

```

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

```

`\@glsxtrfmt` Actual inner working.

```
750 \newcommand*{\@glsxtrfmt}[4]{%
```

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

```
751 \begingroup
752   \def\glslabel{#2}%
753   \glsdoifexistsordo{#2}%
754   {%
755     \ifglshasfield{\GlsXtrFmtField}{#2}%
756     {%
757       \let\do@glsl@link@checkfirsthyper\relax
758       \expandafter\@glsl@link\expandafter[\GlsXtrFmtDefaultOptions,#1]{#2}%
759       {\glsxtrfmtdisplay{\glscurrentfieldvalue}{#3}{#4}}%
760     }%
761     {\glsxtrfmtdisplay{@firstofone}{#3}{#4}}%
762   }%
763 }
```

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

```
764   \begingroup
765     \@glsl@setdefault@glsl@link@opts
766     \setkeys{glsl@link}{\GlsXtrFmtDefaultOptions,#1}%
767     \ifKV@glsl@link@noindex\else\glsadd{#2}\fi
768   \endgroup
769   \glsxtrfmtdisplay{@firstofone}{#3}{#4}%
770 }%
771 \endgroup
772 }
```

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

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

`\glsxtrententryfmt` No link or indexing.

```
774 \ifdef\teorpdfstring
775 {
776   \newcommand*{\glsxtrententryfmt}[2]{%
777     \teorpdfstring{\@glsxtrententryfmt{#1}{#2}}{#2}%
778   }
779 }
780 {
781   \newcommand*{\glsxtrententryfmt}{\@glsxtrententryfmt}
782 }
```

`@glsxtrententryfmt`

```
783 \newrobustcmd*{\@glsxtrententryfmt}[2]{%
```

```

784 \glsdoifexistsordo{#1}%
785 {%
786   \ifglshasfield{\GlsXtrFmtField}{#1}%
787   {%
788     \csuse{\glscurrentfieldvalue}{#2}%
789   }%
790   {#2}%
791 }%
792 {#2}%
793 }

```

**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.

```

794 \newcommand*{\glxtrfieldlistadd}[3]{%
795   \listcsadd{glo@\glsdetoklabel{#1}@#2}{#3}%
796 }

```

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

```

797 \newcommand*{\glxtrfieldlistgadd}[3]{%
798   \listcsgadd{glo@\glsdetoklabel{#1}@#2}{#3}%
799 }

```

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

```

800 \newcommand*{\glxtrfieldlistead}[3]{%
801   \listcseadd{glo@\glsdetoklabel{#1}@#2}{#3}%
802 }

```

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

```

803 \newcommand*{\glxtrfieldlistxadd}[3]{%
804   \listcsxadd{glo@\glsdetoklabel{#1}@#2}{#3}%
805 }

```

Now provide commands to iterate over these lists.

**fieldddolistloop**

```

806 \newcommand*{\glxtrfieldddolistloop}[2]{%
807   \dolistcsloop{glo@\glsdetoklabel{#1}@#2}%
808 }

```

**fieldforlistloop**

```

809 \newcommand*{\glxtrfieldforlistloop}[3]{%
810   \forlistcsloop{glo@\glsdetoklabel{#1}@#2}{#3}%
811 }

```

List element tests:

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

```
812 \newcommand*\glstrfieldifinlist[5]{%
813   \ifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
814 }
```

rfieldxifinlist Expands item.

```
815 \newcommand*\glstrfieldxifinlist[5]{%
816   \xifinlistcs{#3}{glo@\glsetoklabel{#1}@#2}{#4}{#5}%
817 }
```

lsxtrforcsvfield `\glstrforcsvfield{<label>}{<field>}{<cs handler>}`

```
818 \newcommand*\glstrforcsvfield[3]{%
819   \@glstrifhasfield{#2}{#1}%
820   {%
821     \let\glxtrendfor\endfortrue
822     \@for\@glstr@label:=\glscurrentfieldvalue\do
823     {\expandafter#3\expandafter{\@glstr@label}}}%
824   {%
825 }
```

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.

```
826 \newrobustcmd{\glstrifhasfield}{%
827   \@ifstar{\s@glstrifhasfield}{\@glstrifhasfield}%
828 }
```

lsxtrifhasfield Unstarred version adds grouping.

```
829 \newcommand{\@glstrifhasfield}[4]{%
830   {\s@glstrifhasfield{#1}{#2}{#3}{#4}}%
831 }
```

lsxtrifhasfield Starred version omits grouping.

```
832 \newcommand{\s@glstrifhasfield}[4]{%
833   \letcs{\glscurrentfieldvalue}{glo@\glsetoklabel{#2}@#1}%
834   \ifundef\glscurrentfieldvalue
835     {#4}%
836     {%
837       \ifdefempty\glscurrentfieldvalue{#4}{#3}%
838     }%
839 }
```

sXtrIfFieldUndef

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

Just uses \ifcsundef.

```
840 \newcommand{\GlsXtrIfFieldUndef}[2]{%
841   \ifcsundef{glo@\glsdetoklabel{#2}@#1}%
842 }
```

\glstrusefield

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

```
843 \newcommand*{\glstrusefield}[2]{%
844   \@gls@entry@field{#1}{#2}%
845 }
```

\Glsxtrusefield

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

```
846 \newcommand*{\Glsxtrusefield}[2]{%
847   \@Gls@entry@field{#1}{#2}%
848 }
```

\glstrdeffield

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

```
849 \newcommand*{\glstrdeffield}[2]{\csdef{glo@\glsdetoklabel{#1}@#2}}
```

glxtredeffield

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

```
850 \newcommand*{\glxtredeffield}[2]{\protected@csedef{glo@\glsdetoklabel{#1}@#2}}
```

etfieldifexists

```
851 \newcommand*{\glstrsetfieldifexists}[3]{\glsdoifexists{#1}{#3}}
```

\GlsXtrSetField

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

```
852 \newrobustcmd*{\GlsXtrSetField}[3]{%
853   \glstrsetfieldifexists{#1}{#2}%
854   {\csdef{glo@\glsdetoklabel{#1}@#2}{#3}}%
855 }
```

\GlsXtrLetField

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

```
856 \newrobustcmd*{\GlsXtrLetField}[3]{%
857   \glstrsetfieldifexists{#1}{#2}%
858   {\cslet{glo@\glsdetoklabel{#1}@#2}{#3}}%
859 }
```

sGlsXtrLetField

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

```
860 \newrobustcmd*{\csGlsXtrLetField}[3]{%
861   \glstrsetfieldifexists{#1}{#2}%
862   {\csletcs{glo@\glsdetoklabel{#1}@#2}{#3}}%
863 }
```

**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.

```
864 \newrobustcmd*{\GlsXtrLetFieldToField}[4]{%
865   \glstrsetfieldifexists{#1}{#2}%
866   {\csletcs{glo@\glsetoklabel{#1}@#2}{glo@\glsetoklabel{#3}@#4}}%
867 }
```

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

```
868 \newrobustcmd*{\gGlsXtrSetField}[3]{%
869   \glstrsetfieldifexists{#1}{#2}%
870   {\csgdef{glo@\glsetoklabel{#1}@#2}{#3}}%
871 }
```

**xGlsXtrSetField**

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

**eGlsXtrSetField**

```
876 \newrobustcmd*{\eGlsXtrSetField}[3]{%
877   \glstrsetfieldifexists{#1}{#2}%
878   {\protected@csedef{glo@\glsetoklabel{#1}@#2}{#3}}%
879 }
```

**XtrIfFieldEqStr**

```
880 \newrobustcmd*{\GlsXtrIfFieldEqStr}[5]{%
881   \glstrifhasfield{#1}{#2}%
882   {%
883     \ifdefstring{\glscurrentfieldvalue}{#3}{#4}{#5}%
884   }%
885   {#5}%
886 }
```

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

```
887 \ifglentrycounter
888   \newcommand*{\glstrpageref}[1]{\pageref{glentry-\glsetoklabel{#1}}}
889 \else
890   \ifglssubentrycounter
891     \newcommand*{\glstrpageref}[1]{\pageref{glentry-\glsetoklabel{#1}}}
892   \else
893     \newcommand*{\glstrpageref}[1]{\gls{#1}}
894   \fi
895 \fi
```

**lossarypreamble**

```
896 \newcommand{\apptoglossarypreamble}[2][\glsdefaulttype]{%
```



```

897 \ifcsdef{glolist@#1}%
898 {%
899 \ifcsundef{@glossarypreamble@#1}%
900 {\csdef{@glossarypreamble@#1}{}}%
901 {}%
902 \csappto{@glossarypreamble@#1}{#2}%
903 }%
904 {%
905 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
906 }%
907 }

```

lossarypreamble

```

908 \newcommand{\preglossarypreamble}[2][\glsdefaultttype]{%
909 \ifcsdef{glolist@#1}%
910 {%
911 \ifcsundef{@glossarypreamble@#1}%
912 {\csdef{@glossarypreamble@#1}{}}%
913 {}%
914 \cspreto{@glossarypreamble@#1}{#2}%
915 }%
916 {%
917 \GlossariesExtraWarning{Glossary ‘#1’ is not defined}%
918 }%
919 }

```

## 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 `\glxtrpostlongdescription` instead.

ewglossaryentry

```

920 \renewcommand*{\longnewglossaryentry}{%
921 \@ifstar{\glxtr@s@longnewglossaryentry\glxtr@longnewglossaryentry
922 }

```

ewglossaryentry Starred version.

```

923 \newcommand{\@glxtr@s@longnewglossaryentry}[3]{%
924 \glsdoifnoexists{#1}%
925 {%
926 \bgroup
927 \let\@org@newglossaryentryprehook\@newglossaryentryprehook
928 \long\def\@newglossaryentryprehook{%
929 \long\def\@glo@desc{#3}%

```

```

930      \@org@newglossaryentryprehook
931    }%
932    \renewcommand*{\gls@assign@desc}[1]{%
933      \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%
934      \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
935    }
936    \gls@defglossaryentry{#1}{#2}%
937  \egroup
938 }%
939 }

```

`\newglossaryentry` Unstarred version.

```

940 \newcommand{\@glsxtr@longnewglossaryentry}[3]{%
941   \glsdoifnoexists{#1}%
942   {%
943     \bgroup
944     \let\@org@newglossaryentryprehook\@newglossaryentryprehook
945     \long\def\@newglossaryentryprehook{%
946       \long\def\@glo@desc{#3\glsxtrpostlongdescription}%
947       \@org@newglossaryentryprehook
948     }%
949     \renewcommand*{\gls@assign@desc}[1]{%
950       \global\cslet{glo@\glsdetoklabel{#1}@desc}{\@glo@desc}%

```

The following is different from the base glossaries.sty:

```

951       \global\cslet{glo@\glsdetoklabel{#1}@descplural}{\@glo@descplural}%
952     }
953     \gls@defglossaryentry{#1}{#2}%
954   \egroup
955 }%
956 }

```

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

```

957 \newcommand*{\glsxtrpostlongdescription}{\leavevmode\unskip\nopostdesc}

```

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

`\ignoredglossary` Redefine to check for star.

```

958 \renewcommand{\newignoredglossary}{%
959   \@ifstar\glsxtr@s@newignoredglossary\glsxtr@org@newignoredglossary
960 }

```

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

```

961 \newcommand*{\glsxtr@org@newignoredglossary}[1]{%
962   \ifcsdef{glolist@#1}
963   {%
964     \glsxtrundefaction{Glossary type ‘#1’ already exists}{}%
965   }%

```

```

966 {%
967   \ifdefempty\@ignored@glossaries
968   {%
969     \edef\@ignored@glossaries{#1}%
970   }%
971   {%
972     \eappto\@ignored@glossaries{,#1}%
973   }%
974   \csgdef{glolist@#1}{,}%
975   \ifcsundef{gls@#1@entryfmt}%
976   {%
977     \defglsentryfmt[#1]{\glsentryfmt}%
978   }%
979   {}%
980   \ifdefempty\@gls@nohyperlist
981   {%
982     \renewcommand*{\@gls@nohyperlist}{#1}%
983   }%
984   {%
985     \eappto\@gls@nohyperlist{,#1}%
986   }%
987 }%
988 }

```

`ignoredglossary` Starred form.

```

989 \newcommand*{\glxtr@s@newignoredglossary}[1]{%
990   \ifcsdef{glolist@#1}
991   {%
992     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
993   }%
994   {%
995     \ifdefempty\@ignored@glossaries
996     {%
997       \edef\@ignored@glossaries{#1}%
998     }%
999     {%
1000       \eappto\@ignored@glossaries{,#1}%
1001     }%
1002     \csgdef{glolist@#1}{,}%
1003     \ifcsundef{gls@#1@entryfmt}%
1004     {%
1005       \defglsentryfmt[#1]{\glsentryfmt}%
1006     }%
1007     {}%
1008   }%
1009 }

```

`\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.

```

1010 \glsifusetranslator
1011 {%
1012   \renewcommand*{\glssettocitle}[1]{%
1013     \ifcsdef{gls@tr@set@#1@tocitle}%
1014     {%
1015       \csuse{gls@tr@set@#1@tocitle}%
1016     }%
1017     {%
1018       \ifcsdef{@glotype@#1@title}%
1019       {\def\glossarytocitle{\csname @glotype@#1@title\endcsname}}%
1020       {\def\glossarytocitle{\glossarytitle}}%
1021     }%
1022   }%
1023 }
1024 {
1025   \renewcommand*{\glssettocitle}[1]{%
1026     \ifcsdef{@glotype@#1@title}%
1027     {\def\glossarytocitle{\csname @glotype@#1@title\endcsname}}%
1028     {\def\glossarytocitle{\glossarytitle}}%
1029   }
1030 }

```

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

```

1031 \newcommand{\provideignoredglossary}{%
1032   \@ifstar\glsxtr@s@provideignoredglossary\glsxtr@provideignoredglossary
1033 }

```

`ignoredglossary` Unstarred version.

```

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

```

```

1055     {%
1056     \eappto\@gls@nohyperlist{,#1}%
1057     }%
1058     }%
1059 }

```

`ignoredglossary` Starred form.

```

1060 \newcommand*{\glstr@s@provideignoredglossary}[1]{%
1061   \ifcsdef{glolist@#1}
1062   {%
1063   {%
1064     \ifdefempty\@ignored@glossaries
1065     {%
1066       \edef\@ignored@glossaries{#1}%
1067     }%
1068     {%
1069       \eappto\@ignored@glossaries{,#1}%
1070     }%
1071     \csgdef{glolist@#1}{,}%
1072     \ifcsundef{gls@#1@entryfmt}%
1073     {%
1074       \defglsentryfmt[#1]{\glsentryfmt}%
1075     }%
1076     {}%
1077   }%
1078 }

```

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

```

1079 \newcommand*{\glstrcopytoglossary}[2]{%
1080   \glsdoidexists{#1}%
1081   {%
1082     \ifcsdef{glolist@#2}
1083     {%
1084       \cseappto{glolist@#2}{#1,}%
1085     }%
1086     {%
1087       \glstrundefaction{Glossary type ‘#2’ doesn’t exist}{}%
1088     }%
1089   }%
1090 }

```

### 1.3.1 Existence Checks

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

```

1091 \renewcommand{\glsdoidexists}[2]{%
1092   \ifglsentryexists{#1}{#2}%
1093   {%

```

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

```

1094 \edef\glslabel{\glsdetoklabel{#1}}%
1095 \glstrundefaction{Glossary entry '\glslabel'
1096 has not been defined}{You need to define a glossary entry before
1097 you can reference it.}%
1098 }%
1099 }

```

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

```

1100 \renewcommand{\glsdoifnoexists}[2]{%
1101 \ifglentryexists{#1}{%
1102 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1103 has already been defined}{}}{#2}%
1104 }

```

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

```

1105 \ifdef\glsdoifexistsordo
1106 {%
1107 \renewcommand{\glsdoifexistsordo}[3]{%
1108 \ifglentryexists{#1}{#2}%
1109 {%
1110 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1111 has not been defined}{You need to define a glossary entry
1112 before you can use it.}%
1113 #3%
1114 }%
1115 }%
1116 }
1117 {%
1118 \glstr@warnonexistsordo\glsdoifexistsordo
1119 \newcommand{\glsdoifexistsordo}[3]{%
1120 \ifglentryexists{#1}{#2}%
1121 {%
1122 \glstrundefaction{Glossary entry '\glsdetoklabel{#1}'
1123 has not been defined}{You need to define a glossary entry
1124 before you can use it.}%
1125 #3%
1126 }%
1127 }%
1128 }

```

`glsdoifexistsordo` Similarly for `\doifglossarynoexistsordo`.

```

1129 \ifdef\doifglossarynoexistsordo
1130 {%
1131 \renewcommand{\doifglossarynoexistsordo}[3]{%
1132 \ifglossaryexists{#1}%

```

```

1133   {%
1134     \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
1135     #3%
1136   }%
1137   {#2}%
1138 }%
1139 }
1140 {%
1141   \glxtr@warnonexistsordo\doifglossarynoexistsordo
1142   \newcommand{\doifglossarynoexistsordo}[3]{%
1143     \ifglossaryexists{#1}%
1144     {%
1145       \glxtrundefaction{Glossary type ‘#1’ already exists}{}%
1146       #3%
1147     }%
1148     {#2}%
1149   }%
1150 }
1151

```

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.

```

1152 \appto\@newglossaryentryposthook{%
1153   \ifdefvoid\@glo@see
1154   {\csxdef{glo@\@glo@label @see}{}}%
1155   {%
1156     \csxdef{glo@\@glo@label @see}{\@glo@see}%
1157     \if@glxtr@autoseeindex
1158       \@glxtr@autoindexcrossrefs
1159     \fi
1160   }%
1161 }
1162 \appto\@gls@keymap{,{see}{see}}

```

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

```

1163 \newcommand*{\glxtrusesee}[1]{%
1164   \glsdoifexists{#1}%
1165   {%
1166     \letcs{\@glo@see}{glo\@glsdetoklabel{#1}@see}%
1167     \ifdefempty\@glo@see
1168     {}%
1169     {%
1170       \expandafter\glxtr@usesee\@glo@see\@end@glxtr@usesee
1171     }%
1172   }%
1173 }

```

```

\glxtr@usesee
1174 \newcommand*{\glxtr@usesee}[1][\seename]{%
1175   \@glxtr@usesee[#1]%
1176 }

\@glxtr@usesee
1177 \def\@glxtr@usesee[#1]#2\@endglxtr@usesee{%
1178   \glxtruseseeformat{#1}{#2}%
1179 }

xtruseseeformat  The format used by \glxtrusesee. The first argument is the tag (such as \seename). The
                  second argument is the comma-separated list of cross-referenced labels.
1180 \newcommand*{\glxtruseseeformat}[2]{%
1181   \glssformat{#1}{#2}{}%
1182 }

lsseeitemformat  glossaries originally defined \glssseeitemformat to use \glstryname but in v3.0 this was
                  switched to use \glstrytext due to problems occurring with the name field being sani-
                  tized. 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 \glssaccesstext
                  for abbreviations.
1183 \renewcommand*{\glssseeitemformat}[1]{%
1184   \ifglshashshort{\glslabel}{\glssaccesstext{#1}}{\glssaccessname{#1}}%
1185 }

lsxtruseseealso  Apply \glssformat to the seealso key if not empty. There's no optional tag to worry about
                  here.
1186 \newcommand*{\glxtruseseealso}[1]{%
1187   \glsdexists{#1}%
1188   {%
1189     \letcs{\@glo@see}{glo@glsdetoklabel{#1}@seealso}%
1190     \ifdefempty\@glo@see
1191     {%
1192       {%
1193         \expandafter\glxtruseseealsoformat\expandafter{\@glo@see}%
1194       }%
1195     }%
1196 }

sesealsoformat  The format used by \glxtruseseealso. The argument is the comma-separated list of
                  cross-referenced labels.
1197 \newcommand*{\glxtruseseealsoformat}[1]{%
1198   \glssformat[\seealso]{#1}{}%
1199 }

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

```



```

1200 \newrobustcmd{\glxtrseelist}[1]{%
1201   \edef\@glo@tmp{\noexpand\glseelist{#1}}\@glo@tmp
1202 }

```

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

```

1203 \providecommand{\seealso}{see also}

```

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

```

1204 \ifdef\@xdycrossrefhook
1205 {

```

Add the cross-reference class definition to the hook.

```

1206   \appto\@xdycrossrefhook{%
1207     \write\glswrite{(define-crossref-class \string"seealso\string"
1208       :unverified )}%
1209     \write\glswrite{(markup-crossref-list
1210       :class \string"seealso\string"^^J\space\space\space
1211       :open \string"\string\glxtruseealsoformat\glsoopenbrace\string"
1212       :close \string"\glsclosebrace\string")}%
1213   }

```

Append to class list.

```

1214   \appto\@xdylocationclassorder{\space\string"seealso\string"}

```

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

```

1215   \newrobustcmd*{\glxtrindexseealso}[2]{%
1216     \ifx\@glxtr@record@setting\@glxtr@record@setting@alsoindex
1217       \@glxtr@recordsee{#1}{#2}%
1218     \fi
1219     \glsoifexists{#1}%
1220     {%
1221       \@@glxtrwrglossmark
1222       \def\@gls@xref{#2}%
1223       \@onelevel@sanitize\@gls@xref
1224       \@gls@checkmkidxchars\@gls@xref
1225       \gls@glossary{\csname glo@#1@type\endcsname}{%
1226         (indexentry
1227           :tkey (\csname glo@#1@index\endcsname)
1228           :xref (\string"\@gls@xref\string")
1229           :attr \string"seealso\string"
1230         )
1231       }%
1232     }%
1233   }
1234 }
1235 {

```

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

```

1236 \newrobustcmd*{\glxtrindexseealso}{\glssee[\seealsoname]}
1237 }

```

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

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

```

1238 \ifdef\gls@set@xr@key
1239 {

```

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

```

1240 \define@key{glossentry}{alias}{%
1241   \gls@set@xr@key{alias}{\@glo@alias}{#1}%
1242 }
1243 \define@key{glossentry}{seealso}{%
1244   \gls@set@xr@key{seealso}{\@glo@seealso}{#1}%
1245 }

```

Add to the key mappings.

```

1246 \appto\@gls@keymap{,{alias}{alias},{seealso}{seealso}}

```

Set the default value.

```

1247 \appto\@newglossaryentryprehook{\def\@glo@alias{}\def\@glo@seealso{}}%

```

Assign the field values.

```

1248 \appto\@newglossaryentryposthook{%
1249   \ifdefvoid\@glo@seealso
1250     {\csxdef{glo@\@glo@label @seealso}{}}%
1251     {%
1252       \csxdef{glo@\@glo@label @seealso}{\@glo@seealso}%
1253       \ifglxtr@autoseeindex
1254         \@glxtr@autoindexcrossrefs
1255       \fi
1256     }%

```

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

```

1257   \ifdefvoid\@glo@alias
1258     {\csxdef{glo@\@glo@label @alias}{}}%
1259     {%
1260       \csxdef{glo@\@glo@label @alias}{\@glo@alias}%
1261     }%
1262 }

```

Provide user-level commands to access the values.

`\glxtralias`

```

1263 \newcommand*{\glxtralias}[1]{\@gls@entry@field{#1}{alias}}

```

trseealsolabels

```
1264 \newcommand*{\glxtrseealsolabels}[1]{\@gls@entry@field{#1}{seealso}}
```

Add to the \@glo@autosee hook.

```
1265 \appto\@glo@autoseehook{%
1266   \ifdefvoid\@glo@alias
1267   {%
1268     \ifdefvoid\@glo@seealso
1269     }%
1270   {%
1271     \edef\@do@glsssee{\noexpand\glxtrindexseealso
1272       {\@glo@label}{\@glo@seealso}}%
1273     \@do@glsssee
1274   }%
1275 }%
1276 {%
```

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

```
1277   \ifdefvoid\@glo@see
1278   {%
1279     \edef\@do@glsssee{\noexpand\glsssee{\@glo@label}{\@glo@alias}}%
1280     \@do@glsssee
1281   }%
1282   }%
1283 }%
1284 }%
1285 }
1286 {
```

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

\glxtralias

```
1287 \glsaddstoragekey*{alias}{}{\glxtralias}
```

trseealsolabels

```
1288 \glsaddstoragekey*{seealso}{}{\glxtrseealsolabels}
```

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

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

```
1289 \appto\@newglossaryentryposthook{%
1290   \ifcsvoid{glo@\@glo@label @alias}%
1291   {%
1292     \ifcsvoid{glo@\@glo@label @seealso}%
1293     }%
1294   {%
1295     \edef\@do@glsssee{\noexpand\glxtrindexseealso
1296       {\@glo@label}{\csuse{glo@\@glo@label @seealso}}}%
1297     \@do@glsssee
```

```

1298     }%
1299     }%
1300     {%

```

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

```

1301     \ifdefvoid\@glo@see
1302     {%
1303         \edef\@do@glsssee{\noexpand\glsssee
1304             {\@glo@label}{\csuse{glo@\@glo@label @alias}}}%
1305         \@do@glsssee
1306     }%
1307     }%
1308     }%
1309 }
1310 }

```

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

```

1311 \AtEndDocument{\if@glstrindexcrossrefs\glstraddallcrossrefs\fi}

```

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

```

1312 \newcommand*\@glstraddallcrossrefs{%
1313     \forallglossaries{\@glo@type}%
1314     {%
1315         \forglsentries[\@glo@type]{\@glo@label}%
1316         {%
1317             \ifglssused{\@glo@label}%
1318             {\expandafter\@glstr@addunusedxrefs\expandafter{\@glo@label}}}%
1319         }%
1320     }%
1321 }

```

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

```

1322 \newcommand*\@glstr@addunusedxrefs[1]{%
1323     \letcs{\@glo@see}{glo@\glstetoklabel{#1}@see}%
1324     \ifdefvoid\@glo@see
1325     {%
1326     {%
1327         \expandafter\glstr@addunused\@glo@see\@end@glstr@addunused
1328     }%
1329     \letcs{\@glo@see}{glo@\glstetoklabel{#1}@seealso}%
1330     \ifdefvoid\@glo@see
1331     {%
1332     {%
1333         \expandafter\glstr@addunused\@glo@see\@end@glstr@addunused
1334     }%
1335 }

```

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

```
1336 \newcommand*{\glxtr@addunused}[1] [] {%
1337   \@glxtr@addunused
1338 }
```

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

```
1339 \def\@glxtr@addunused#1\@end@glxtr@addunused{%
1340   \@for\@glxtr@label:=#1\do
1341   {%
1342     \ifglxsused{\@glxtr@label}{}%
1343     {%
1344       \glxadd[format=glxtrunusedformat]{\@glxtr@label}%
1345       \glxunset{\@glxtr@label}%
1346       \expandafter\@glxtr@addunusedxrefs\expandafter{\@glxtr@label}%
1347     }%
1348   }%
1349 }
```

xtrunusedformat

```
1350 \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.

```
1351 \let\glxtr@orgmakenoidxglossaries\makenoidxglossaries
1352 \renewcommand{\makenoidxglossaries}{%
1353   \ifdefequal\@glxtr@record@setting\@glxtr@record@setting@off
1354   {%
1355     \glxtr@orgmakenoidxglossaries
```

Add marker to \@do@seeglossary

```
1356   \renewcommand{\@do@seeglossary}[2]{%
1357     \@glxtrwrglossmark
1358     \edef\@gls@label{\glxdetoklabel{##1}}%
1359     \protected@write\@auxout{}{%
1360       \string\@gls@reference
1361       {\csname glo@\@gls@label @type\endcsname}%
1362       {\@gls@label}%
1363       {%
1364         \string\glxseeformat##2{}%
1365       }%
1366     }%
1367   }%
```

Check for docdefs=restricted:

```
1368   \if@glxtrdocdefrestricted
```

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

```

1369 \renewcommand*{\@gls@reference}[3]{%
1370 \ifcsundef{\glsref@##1}{\csgdef{\glsref@##1}{}}{}%
1371 \ifinlistcs{##2}{\glsref@##1}%
1372 {}%
1373 {\listcsgadd{\glsref@##1}{##2}}%
1374 \ifcsundef{glo@\glsdetoklabel{##2}@loclist}%
1375 {\csgdef{glo@\glsdetoklabel{##2}@loclist}{}}%
1376 {}%
1377 \listcsgadd{glo@\glsdetoklabel{##2}@loclist}{##3}%
1378 }%
1379 \else

```

Disable document definitions.

```

1380 \@glsxtrdocdeffalse
1381 \fi
1382 \disable@keys{glossaries-extra}{docdef}%
1383 }%
1384 {%
1385 \PackageError{glossaries-extra}{\string\makenoidxglossaries\space
1386 not permitted\MessageBreak
1387 with record=\@glsxtr@record@setting\space package option}%
1388 {You may only use \string\makenoidxglossaries\space with the
1389 record=off option}%
1390 }%
1391 }

```

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

```

1392 \renewcommand*{\gls@defdocnewglossaryentry}{%
1393 \ifcase\@glsxtr@docdefval
1394 docdef=false:
1395 \renewcommand*{\newglossaryentry}[2]{%
1396 \PackageError{glossaries-extra}{Glossary entries must
1397 be \MessageBreak defined in the preamble with \MessageBreak
1398 package option 'docdef=false'\MessageBreak(consider using
1399 'docdef=restricted')}{Move your glossary definitions to
1400 the preamble. You can also put them in a \MessageBreak separate file
1401 and load them with \string\loadglsentries.}%
1402 }%
1403 \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.

```

1403 \let\gls@checkseeallowed\relax
1404 \let\newglossaryentry\newglossaryentry
1405 \or

```

Restricted mode just needs to allow the `see` value.

```

1406 \let\gls@checkseeallowed\relax
1407 \fi
1408 }%

```

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`

```

1409 \newcommand*{\GlsXtrEnableOnTheFly}{%
1410 \ifstar\@sGlsXtrEnableOnTheFly\@GlsXtrEnableOnTheFly
1411 }

```

`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.)

```

1412 \newcommand*{\@sGlsXtrEnableOnTheFly}{%
1413 \renewcommand*{\glsdetoklabel}[1]{%
1414 \expandafter\@glsxtr@ifcsstart\string##1 \@glsxtr@end@
1415 {%
1416 \expandafter\detokenize\expandafter{##1}%
1417 }%
1418 {\detokenize{##1}}}%
1419 }%
1420 \@GlsXtrEnableOnTheFly
1421 }
1422 \def\@glsxtr@ifcsstart#1#2\@glsxtr@end@#3#4{%
1423 \expandafter\if\glsbackslash#1%
1424 #3%
1425 \else
1426 #4%
1427 \fi
1428 }

```

`sxtrstarflywarn`

```

1429 \newcommand*{\glsxtrstarflywarn}{%
1430 \GlossariesExtraWarning{Experimental starred version of
1431 \string\GlsXtrEnableOnTheFly\space in use (please ensure you have
1432 read the warnings in the glossaries-extra user manual)}}%
1433 }

```

`rEnableOnTheFly`

```

1434 \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:

```

\glstrcat
1435 \newcommand*\glstrcat{general}

\glstr
1436 \newcommand*\glstr[1] [] {%
1437 \def\glstr@keylist{##1}%
1438 \@glstr
1439 }

\@glstr
1440 \newcommand*\@glstr[2] [] {%
1441 \ifglentryexists{##2}%
1442 {%
1443 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1444 }%
1445 {%
1446 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1447 description={\nopostdesc},##1}%
1448 }%
1449 \expandafter\gls\expandafter[\glstr@keylist]{##2}%
1450 }

\Glsxtr
1451 \newcommand*\Glsxtr[1] [] {%
1452 \def\glstr@keylist{##1}%
1453 \@Glsxtr
1454 }

\@Glsxtr
1455 \newcommand*\@Glsxtr[2] [] {%
1456 \ifglentryexists{##2}%
1457 {%
1458 \ifblank{##1}{-}{\GlsXtrWarning{##1}{##2}}%
1459 }%
1460 {%
1461 \gls@defglossaryentry{##2}{name={##2},category=\glstrcat,
1462 description={\nopostdesc},##1}%
1463 }%
1464 \expandafter\Gls\expandafter[\glstr@keylist]{##2}%
1465 }

\glstrpl
1466 \newcommand*\glstrpl[1] [] {%
1467 \def\glstr@keylist{##1}%
1468 \@glstrpl
1469 }

\@glstrpl

```



```

1470 \newcommand*{\@glxstrpl}[2][ ]{%
1471   \ifglentryexists{##2}%
1472   {%
1473     \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
1474   }%
1475   {%
1476     \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1477       description={\nopostdesc},##1}%
1478   }%
1479   \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
1480 }

```

\Glsxtrpl

```

1481 \newcommand*{\Glsxtrpl}[1][ ]{%
1482   \def\glxtr@keylist{##1}%
1483   \@Glsxtrpl
1484 }

```

\@Glsxtrpl

```

1485 \newcommand*{\@Glsxtrpl}[2][ ]{%
1486   \ifglentryexists{##2}
1487   {%
1488     \ifblank{##1}{ }\GlsXtrWarning{##1}{##2}}%
1489   }%
1490   {%
1491     \gls@defglossaryentry{##2}{name={##2},category=\glxtrcat,
1492       description={\nopostdesc},##1}%
1493   }%
1494   \expandafter\glsp\expandafter[\glxtr@keylist]{##2}%
1495 }

```

\GlsXtrWarning

```

1496 \newcommand*{\GlsXtrWarning}[2]{%
1497   \def\glxtr@optlist{##1}%
1498   \@onelevel@sanitize\glxtr@optlist
1499   \GlossariesExtraWarning{The options ‘\glxtr@optlist’ have
1500     been ignored for entry ‘##2’ as it has already been defined}%
1501 }

```

Disable commands after the glossary:

```

1502 \renewcommand\@printglossary[2]{%
1503   \def\glxtr@printglossopts{##1}%
1504   \@glxtr@orgprintglossary{##1}{##2}%
1505   \def\glxtr{\@glxtr@disabledflycommand\glxtr}%
1506   \def\glxtrpl{\@glxtr@disabledflycommand\glxtrpl}%
1507   \def\@Glsxtr{\@glxtr@disabledflycommand\Glsxtr}%
1508   \def\@Glsxtrpl{\@glxtr@disabledflycommand\Glsxtrpl}%
1509 }

```

abledflycommand

```
1510 \newcommand*{\@glxtr@disabledflycommand}[1]{%
1511   \PackageError{glossaries-extra}%
1512   {\string##1\space can't be used after any of the \MessageBreak
1513    glossaries have been displayed}%
1514   {The on-the-fly commands enabled by
1515    \string\GlsXtrEnableOnTheFly\space may only be used \MessageBreak
1516    before the glossaries. If you want to use any entries \MessageBreak
1517    after any of the glossaries, you must use the standard \MessageBreak
1518    method of first defining the entry and then using the \MessageBreak
1519    entry with commands like \string\gls}%
1520   \@glxtr@disabledflycommand
1521 }%
1522 \newcommand*{\@glxtr@disabledflycommand}[2][\{##2}
```

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

```
1523 \let\GlsXtrEnableOnTheFly\relax
1524 }
1525 \@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.

```
1526 \newcommand*{\@glxtr@current@style}{\@glossary@default@style}
```

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

etglossarystyle

```
1527 \renewcommand*{\setglossarystyle}[1]{%
1528   \ifcsundef{@glsstyle@#1}%
1529   {%
1530     \PackageError{glossaries-extra}{Glossary style ‘#1’ undefined}{}%
1531   }%
1532   {%
1533     \csname @glsstyle@#1\endcsname
```

Only set the current style if it exists.

```
1534   \protected@edef\@glxtr@current@style{#1}%
1535 }%
1536 \ifx\@glossary@default@style\relax
1537   \protected@edef\@glossary@default@style{#1}%
1538 \fi
1539 }
```

In case we have an old version of glossaries:

```
1540 \ifdef\@glossary@default@style
1541 {}
```

```

1542 {%
1543   \let\@glossary@default@style\relax
1544 }

```

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

```

1545 \ifdef\glslistdottedwidth
1546 {%
1547   \ifdim\glslistdottedwidth=.5\hsize
1548     \setlength{\glslistdottedwidth}{-\dimexpr\maxdimen-1sp\relax}
1549     \AtBeginDocument{%
1550       \ifdim\glslistdottedwidth=-\dimexpr\maxdimen-1sp\relax
1551         \setlength{\glslistdottedwidth}{.5\columnwidth}%
1552       \fi
1553     }%
1554   \fi
1555 }
1556 {}%

```

Similarly for \glsdescwidth:

\glsdescwidth

```

1557 \ifdef\glsdescwidth
1558 {%
1559   \ifdim\glsdescwidth=.6\hsize
1560     \setlength{\glsdescwidth}{-\dimexpr\maxdimen-1sp\relax}
1561     \AtBeginDocument{%
1562       \ifdim\glsdescwidth=-\dimexpr\maxdimen-1sp\relax
1563         \setlength{\glsdescwidth}{.6\columnwidth}%
1564       \fi
1565     }%
1566   \fi
1567 }
1568 {}%

```

and for \glspagelistwidth:

lspagelistwidth

```

1569 \ifdef\glspagelistwidth
1570 {%
1571   \ifdim\glspagelistwidth=.1\hsize
1572     \setlength{\glspagelistwidth}{-\dimexpr\maxdimen-1sp\relax}
1573     \AtBeginDocument{%
1574       \ifdim\glspagelistwidth=-\dimexpr\maxdimen-1sp\relax
1575         \setlength{\glspagelistwidth}{.1\columnwidth}%
1576       \fi
1577     }%
1578   \fi
1579 }
1580 {}%

```

aryentrynumbers Has the nonumberlist option been used?

```
1581 \def\org@glossaryentrynumbers#1{#1\gls@save@numberlist{#1}}%
1582 \ifx\org@glossaryentrynumbers\glossaryentrynumbers
1583   \glsnonumberlistfalse
1584   \renewcommand*{\glossaryentrynumbers}[1]{%
1585     \ifglsentryexists{\glscurrententrylabel}%
1586     {%
1587       \@glsxtrpreloctag
1588       \GlsXtrFormatLocationList{#1}%
1589       \@glsxtrpostloctag
1590       \gls@save@numberlist{#1}%
1591     }{}%
1592   }%
1593 \else
1594   \glsnonumberlisttrue
1595   \renewcommand*{\glossaryentrynumbers}[1]{%
1596     \ifglsentryexists{\glscurrententrylabel}%
1597     {%
1598       \gls@save@numberlist{#1}%
1599     }{}%
1600   }%
1601 \fi
```

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

```
1602 \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

```
1603 \newcommand*{\GlsXtrEnablePreLocationTag}[2]{%
1604   \let\@glsxtrpreloctag\@glsxtrpreloctag
1605   \let\@glsxtrpostloctag\@glsxtrpostloctag
1606   \renewcommand*{\@glsxtr@pagetag}{#1}%
1607   \renewcommand*{\@glsxtr@pagetag}{#2}%
1608   \renewcommand*{\@glsxtr@savepreloctag}[2]{%
1609     \csgdef{\@glsxtr@preloctag@##1}{##2}%
1610   }%
1611   \renewcommand*{\@glsxtr@doloctag}{%
1612     \ifcsundef{\@glsxtr@preloctag\@glscurrententrylabel}%
1613     {%
1614       \GlossariesWarning{Missing pre-location tag for ‘\glscurrententrylabel’.
1615         Rerun required}%
1616     }%
1617   }%
```

```

1618      \csuse{@glxstr@preloctag@\glscurrententrylabel}%
1619    }%
1620  }%
1621 }
1622 \@onlypreamble\GlsXtrEnablePreLocationTag

```

glxstrpreloctag

```

1623 \newcommand*{\@@glxstrpreloctag}{%
1624   \let\@glxstr@org@delimN\delimN
1625   \let\@glxstr@org@delimR\delimR
1626   \let\@glxstr@org@glsgignore\glsgignore
1627   \gdef is required as the delimiters may occur inside a scope.
1628   \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1629   \renewcommand*{\delimN}{%
1630     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1631     \@glxstr@org@delimN}%
1632   \renewcommand*{\delimR}{%
1633     \gdef\@glxstr@thisloctag{\@glxstr@pagetag}%
1634     \@glxstr@org@delimR}%
1635   \renewcommand*{\glsgignore}[1]{%
1636     \gdef\@glxstr@thisloctag{\relax}%
1637     \@glxstr@org@glsgignore{##1}}%
1638   \@glxstr@doloctag

```

glxstrpreloctag

```

1639 \newcommand*{\@glxstrpreloctag}{%

```

@glxstr@pagetag

```

1640 \newcommand*{\@glxstr@pagetag}{%

```

glxstr@pagetag

```

1641 \newcommand*{\@glxstr@pagetag}{%

```

lsxtrpostloctag

```

1642 \newcommand*{\@@glxtrpostloctag}{%
1643   \let\delimN\@glxtr@org@delimN
1644   \let\delimR\@glxtr@org@delimR
1645   \let\glsgignore\@glxtr@org@glsgignore
1646   \protected@write\@auxout{%
1647     {\string\@glxtr@savepreloctag{\glscurrententrylabel}{\@glxtr@thisloctag}}%
1648   }

```

lsxtrpostloctag

```

1649 \newcommand*{\@glxtrpostloctag}{%

```

lsxtr@preloctag

```
1650 \newcommand*{\@glxtr@savepreloctag}[2]{%
1651 \protected@write\@auxout{}{%
1652   \string\providecommand\string\@glxtr@savepreloctag[2]{}}
```

glxtr@doloctag

```
1653 \newcommand*{\@glxtr@doloctag}{}
```

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

```
1654 \renewcommand*{\KV@printgloss@nonumberlist}[1]{%
1655   \XKV@plfalse
1656   \XKV@sttrue
1657   \XKV@checkchoice[\XKV@resa]{#1}{true,false}%
1658   {%
1659     \csname glsnonumberlist\XKV@resa\endcsname
1660     \ifglsnonumberlist
1661       \def\glossaryentrynumbers##1{\gls@save@numberlist{##1}}%
1662     \else
1663       \def\glossaryentrynumbers##1{%
1664         \@glxtr@preloctag
1665         \GlsXtrFormatLocationList{##1}%
1666         \@glxtr@postloctag
1667         \gls@save@numberlist{##1}}%
1668     \fi
1669   }%
1670 }
```

### 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.

```
1671 \renewcommand*{\glsentryfmt}{%
1672   \ifglshasshort{\glslabel}{\glssetabbrvfmt{\glscategory{\glslabel}}{}}%
1673   \glsifregular{\glslabel}%
1674   {\glxtrregularfont{\glsgenentryfmt}}%
1675   {%
1676     \ifglshasshort{\glslabel}%
1677     {\glxtrgenabbrvfmt}%
1678     {\glxtrregularfont{\glsgenentryfmt}}%
1679   }%
1680 }
```

sxtrregularfont Font used for regular entries.

```
1681 \newcommand*{\glxsxtrregularfont}[1]{#1}
```

Commands like `\glisifplural` 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 `\glstryfmt`.

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

```
1682 \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).

```
1683 \glxsxtr@record{#2}{#3}{glslink}%
1684 \glsdoidexists{#3}%
1685 {%
```

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).

```
1686 \let\glxsxtrorg@ifKV@glslink@hyper\ifKV@glslink@hyper
1687 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
1688 \def\glscustomtext{#4}%
1689 \@glxsxtr@field@linkdefs
1690 #1%
1691 \@gls@link[#2]{#3}{#4}%
1692 \let\ifKV@glslink@hyper\glxsxtrorg@ifKV@glslink@hyper
1693 }%
1694 \glspostlinkhook
1695 }
```

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

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

```
1696 \let\@glxsxtr@org@gls@\@gls@
1697 \def\@gls@#1#2{%
1698 \@glxsxtr@record{#1}{#2}{glslink}%
1699 \@glxsxtr@org@gls@{#1}{#2}%
1700 }%
```

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

```
1701 \let\@glxsxtr@org@glspl@\@glspl@
1702 \def\@glspl@#1#2{%
1703 \@glxsxtr@record{#1}{#2}{glslink}%
1704 \@glxsxtr@org@glspl@{#1}{#2}%
1705 }%
```

\@Gls@ Save the original definition and redefine.

```
1706 \let\@glxtr@org@Gls@\@Gls@
1707 \def\@Gls@#1#2{%
1708   \@glxtr@record{#1}{#2}{glslink}%
1709   \@glxtr@org@Gls@{#1}{#2}%
1710 }%
```

\@Glspl@ Save the original definition and redefine.

```
1711 \let\@glxtr@org@Glspl@\@Glspl@
1712 \def\@Glspl@#1#2{%
1713   \@glxtr@record{#1}{#2}{glslink}%
1714   \@glxtr@org@Glspl@{#1}{#2}%
1715 }%
```

\@GLS@ Save the original definition and redefine.

```
1716 \let\@glxtr@org@GLS@\@GLS@
1717 \def\@GLS@#1#2{%
1718   \@glxtr@record{#1}{#2}{glslink}%
1719   \@glxtr@org@GLS@{#1}{#2}%
1720 }%
```

\@GLSpl@ Save the original definition and redefine.

```
1721 \let\@glxtr@org@GLSpl@\@GLSpl@
1722 \def\@GLSpl@#1#2{%
1723   \@glxtr@record{#1}{#2}{glslink}%
1724   \@glxtr@org@GLSpl@{#1}{#2}%
1725 }%
```

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

```
1726 \renewcommand*{\@glsdisp}[3][{}]{%
1727   \@glxtr@record{#1}{#2}{glslink}%
1728   \glsdoifexists{#2}{%
1729     \let\do@gls@link@checkfirsthyper\@gls@link@checkfirsthyper
1730     \let\glsifplural\@secondoftwo
1731     \let\glsapscase\@firstofthree
1732     \def\glscustomtext{#3}%
1733     \def\glsinsert{}%
1734     \def\@glo@text{\csname gls@\glstype @entryfmt\endcsname}%
1735     \@gls@link[#1]{#2}{\@glo@text}%
1736     \ifKV@glslink@local
1737       \glslocalunset{#2}%
1738     \else
1739       \glsunset{#2}%
1740     \fi
1741   }%
1742   \glspostlinkhook
1743 }
```



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

```
1744 \renewcommand*{\@gls@@link}[3][\{%
1745   \@glsxtr@record{#1}{#2}{glslink}%
1746   \glsdoifexistsordo{#2}%
1747   {%
1748     \let\do@gls@link@checkfirsthyper\relax
1749     \@gls@link[#1]{#2}{#3}%
1750   }%
1751   {%
1752     \glstextformat{#3}%
1753   }%
1754   \glspostlinkhook
1755 }
```

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

```
1756 \newcommand*{\gsxtrinitwrgloss}{%
1757   \glsifattribute{\glslabel}{wrgloss}{after}%
1758   {%
1759     \gsxtrinitwrglossbeforefalse
1760   }%
1761   {%
1762     \gsxtrinitwrglossbeforetrue
1763   }%
1764 }
```

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

```
1765 \newif\ifgsxtrinitwrglossbefore
1766 \gsxtrinitwrglossbeforetrue
```

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

```
1767 \define@choicekey{glslink}{wrgloss}%
1768 [\@glsxtr@wrglossval\@glsxtr@wrglossnr]%
1769 {before,after}%
1770 {%
1771   \ifcase\@glsxtr@wrglossnr\relax
1772     \gsxtrinitwrglossbeforetrue
1773   \or
1774     \gsxtrinitwrglossbeforefalse
1775   \fi
1776 }
```

```
1777 \define@key{glslink}{thevalue}{\def\@glsxtr@thevalue{#1}}
```

```
1778 \define@key{glslink}{theHvalue}{\def\@glsxtr@theHvalue{#1}}
```

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

```
1779 \define@boolkey{glslink}[glsxtr@]{hyperoutside}[true]{}
1780 \glsxtr@hyperoutsidettrue
```

nithyperoutside Set the default if the hyperoutside is omitted.

```
1781 \newcommand*{\glxtrnithyperoutside}{%
1782   \gl@ifattribute{\glslabel}{hyperoutside}{false}%
1783   {%
1784     \glxtr@hyperoutsidefalse
1785   }%
1786   {%
1787     \glxtr@hyperoutsidetrue
1788   }%
1789 }
```

r@inc@linkcount Does nothing by default.

```
1790 \newcommand*{\glxtr@inc@linkcount}{}
```

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

```
1791 \newcommand*{\glslinkpresetkeys}{}
```

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

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

Initialise thevalue and theHvalue (v1.19).

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

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

```
1803 \glxtr@initwrgloss
```

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

```
1804 \glxtrnithyperoutside
```

Note that the default link options may override \glxtr@initwrgloss.

```
1805 \@gls@setdefault@glslink@opts
```

Increment link counter if enabled (new to v1.26).

```
1806 \glxtr@inc@linkcount
```

As the original definition.

```
1807 \do@gl:disablehyperinlist
```

```
1808 \do@gls@link@checkfirsthyper
```

User hook before options are set (new to v1.26):

```
1809 \glslinkpresetkeys
```

Set options.

```
1810 \setkeys{glslink}{#1}%
```

User hook after options are set:

```
1811 \glslinkpostsetkeys
```

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

```
1812 \ifdefempty{\@glsxtr@thevalue}%
1813 {%
1814   \@gls@saveentrycounter
1815 }%
1816 {%
1817   \let\theglsentrycounter\@glsxtr@thevalue
1818   \def\theHglentrycounter{\@glsxtr@theHvalue}%
1819 }%
1820 \@gls@setsort{\glslabel}%
```

Check textformat attribute (new to v1.21).

```
1821 \gls@hasattribute{\glslabel}{textformat}%
1822 {%
1823   \edef\@glsxtr@attrval{\gls@getattribute{\glslabel}{textformat}}%
1824   \ifcsdef{\@glsxtr@attrval}%
1825   {%
1826     \letcs{\@glsxtr@textformat}{\@glsxtr@attrval}%
1827   }%
1828   {%
1829     \GlossariesExtraWarning{Unknown control sequence name
1830       '\@glsxtr@attrval' supplied in textformat attribute
1831       for entry '\glslabel'. Reverting to default \string\glstextformat}%
1832     \let\@glsxtr@textformat\glstextformat
1833   }%
1834 }%
1835 {%
1836   \let\@glsxtr@textformat\glstextformat
1837 }%
```

Do write if it should occur before the link text:

```
1838 \ifglsxtrinitwrglossbefore
1839   \@do@wrglossary{#2}%
1840 \fi
```

Do the link text:

```
1841 \ifKV@glslink@hyper
1842   \ifglsxtr@hyperoutside
1843     \glslink{\glolinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
1844   \else
1845     \@glsxtr@textformat{\@glslink{\glolinkprefix\glslabel}{#3}}%
1846   \fi
1847 \else
1848   \ifglsxtr@hyperoutside
1849     \glsdonohyperlink{\glolinkprefix\glslabel}{\@glsxtr@textformat{#3}}%
```

```

1850 \else
1851 \glxtr@textformat{\glsdonohyperlink{\glolinkprefix\glslabel}{#3}}%
1852 \fi
1853 \fi

```

Do write if it should occur after the link text:

```

1854 \ifglxtrinitwrglossbefore
1855 \else
1856 \do@wrglossary{#2}%
1857 \fi

```

As the original definition:

```

1858 \let\ifKV@glslink@hyper\org@ifKV@glslink@hyper
1859 }

```

```

1860 \define@key{glossadd}{thevalue}{\def\glxtr@thevalue{#1}}

```

```

1861 \define@key{glossadd}{theHvalue}{\def\glxtr@theHvalue{#1}}

```

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

```

1862 \renewrobustcmd*{\glsadd}[2][\]{%
1863 \glxtr@trifinmark
1864 }%
1865 {%
1866 \gl@adjustmode
1867 \glxtr@record{#1}{#2}{glossadd}%
1868 \gl@doifexists{#2}%
1869 {%
1870 \let\gl@numberformat\glxtr@defaultnumberformat
1871 \edef\gl@counter{\csname glo@\gl@detoklabel{#2}@counter\endcsname}%
1872 \def\glxtr@thevalue{}%
1873 \def\glxtr@theHvalue{\glxtr@thevalue}%
1874 \setkeys{glossadd}{#1}%
1875 \ifdefempty{\glxtr@thevalue}%
1876 {%
1877 \gl@saveentrycounter
1878 }%
1879 {%
1880 \let\thegl@entrycounter\glxtr@thevalue
1881 \def\theHgl@entrycounter{\glxtr@theHvalue}%
1882 }%

```

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

```

1883 \gl@setsort{#2}%
1884 \do@wrglossary{#2}%
1885 }%
1886 }%
1887 }

```

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

```

1888 \newcommand*{\@glxtr@field@linkdefs}{%
1889   \let\glxtrifwasfirstuse\@secondoftwo
1890   \let\glxtrifplural\@secondoftwo
1891   \let\glscapscase\@firstofthree
1892   \let\glxtrinsert\@empty
1893 }

```

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

assignfieldfont

```

1894 \newcommand*{\glxtrassignfieldfont}[1]{%
1895   \ifglstryexists{#1}%
1896   {%
1897     \ifglshasshort{#1}%
1898     {%
1899       \glsssetabbrvfmt{\glscategory{#1}}%
1900       \glxtrifregular{#1}%
1901       {\let\@glxtr@field@font\glxtrregularfont}%
1902       {\let\@glxtr@field@font\@firstofone}%
1903     }%
1904     {%
1905       \glxtrifnotregular{#1}%
1906       {\let\@glxtr@field@font\@firstofone}%
1907       {\let\@glxtr@field@font\glxtrregularfont}%
1908     }%
1909   }%
1910   {%
1911     \let\@glxtr@field@font\@gobble
1912   }%
1913 }

```

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

```

1914 \def\@glstext@#1#2[#3]{%
1915   \glxtrassignfieldfont{#2}%
1916   \@glxtr@field@link{#1}{#2}{\@glxtr@field@font{\glssaccesstext{#2}{#3}}}%
1917 }

```

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

```

1918 \def\@GLStext@#1#2[#3]{%
1919   \glxtrassignfieldfont{#2}%
1920   \@glxtr@field@link[\let\glscapscase\@thirdofthree]{#1}{#2}%
1921   {\@glxtr@field@font{\@GLSsaccesstext{#2}\mfirstucMakeUppercase{#3}}}%
1922 }

```

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

```

1923 \def\@GlStext@#1#2[#3]{%
1924   \glxtrassignfieldfont{#2}%
1925   \@glxtr@field@link[\let\glscapscase\@secondofthree]{#1}{#2}%

```

```

1926   {\@gls@field@font{\Glsaccessstext{#2}#3}}}%
1927 }

```

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`

```

1928 \newcommand*{\glstrchecknohyperfirst}[1]{%
1929   \glsifattribute{#1}{nohyperfirst}{true}{\KV@glslink@hyperfalse}{}%
1930 }

```

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

```

1931 \def\@glsfirst@#1#2[#3]{%
1932   \glstrassignfieldfont{#2}%
   Ensure that \glsfirst honours the nohyperfirst attribute.
1933   \@gls@field@link
1934   [\let\glstrifwasfirstuse\@firstoftwo
1935     \glstrchecknohyperfirst{#2}%
1936   ]{#1}{#2}%
1937   {\@gls@field@font{\glsaccessfirst{#2}#3}}}%
1938 }

```

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

```

1939 \def\@Glsfirst@#1#2[#3]{%
1940   \glstrassignfieldfont{#2}%
   Ensure that \Glsfirst honours the nohyperfirst attribute.
1941   \@gls@field@link
1942   [\let\glstrifwasfirstuse\@firstoftwo
1943     \let\glscapscase\@secondofthree
1944     \glstrchecknohyperfirst{#2}%
1945   ]%
1946   {#1}{#2}{\@gls@field@font{\Glsaccessfirst{#2}#3}}}%
1947 }

```

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

```

1948 \def\@GLSfirst@#1#2[#3]{%
1949   \glstrassignfieldfont{#2}%
   Ensure that \GLSfirst honours the nohyperfirst attribute.
1950   \@gls@field@link
1951   [\let\glstrifwasfirstuse\@firstoftwo
1952     \let\glscapscase\@thirdofthree
1953     \glstrchecknohyperfirst{#2}%
1954   ]%
1955   {#1}{#2}{\@gls@field@font{\GLSaccessfirst{#2}\mfirstucMakeUppercase{#3}}}%
1956 }

```

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

```
1957 \def\@glsplural@#1#2[#3]{%
1958   \glsxtrassignfieldfont{#2}%
1959   \@gls@field@link[\let\glsifplural\@firstoftwo]{#1}{#2}%
1960   {\@gls@field@font{\glsaccessplural{#2}#3}}%
1961 }
```

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

```
1962 \def\@Glsplural@#1#2[#3]{%
1963   \glsxtrassignfieldfont{#2}%
1964   \@gls@field@link
1965   [\let\glsifplural\@firstoftwo
1966   \let\glscapscase\@secondofthree
1967   ]%
1968   {#1}{#2}{\@gls@field@font{\Glsaccessplural{#2}#3}}%
1969 }
```

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

```
1970 \def\@GLSplural@#1#2[#3]{%
1971   \glsxtrassignfieldfont{#2}%
1972   \@gls@field@link
1973   [\let\glsifplural\@firstoftwo
1974   \let\glscapscase\@thirdofthree
1975   ]%
1976   {#1}{#2}{\@gls@field@font{\GLSaccessplural{#2}\mfirstucMakeUppercase{#3}}}%
1977 }
```

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

```
1978 \def\glsfirstplural@#1#2[#3]{%
1979   \glsxtrassignfieldfont{#2}%
1980   \@gls@field@link
1981   [\let\glsxtrifwasfirstuse\@firstoftwo
1982   \let\glsifplural\@firstoftwo
1983   \glsxtrchecknohyperfirst{#2}%
1984   ]%
1985   {#1}{#2}{\@gls@field@font{\glsaccessfirstplural{#2}#3}}%
1986 }
```

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

```
1987 \def\Glsfirstplural@#1#2[#3]{%
1988   \glsxtrassignfieldfont{#2}%
1989   \@gls@field@link
1990   [\let\glsxtrifwasfirstuse\@firstoftwo
1991   \let\glsifplural\@firstoftwo
1992   \let\glscapscase\@secondofthree
```

```

1993 \glstrchecknohyperfirst{#2}%
1994 ]%
1995 {#1}{#2}{\@gls@field@font{\Glsaccessfirstplural{#2}#3}}%
1996 }

```

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

```

1997 \def\@GLSfirstplural@#1#2[#3]{%
1998 \glstrassignfieldfont{#2}%
    Ensure that \glsfirstplural honours the nohyperfirst attribute.
1999 \@gls@field@link
2000 [\let\glstrifwasfirstuse\@firstoftwo
2001 \let\glsifplural\@firstoftwo
2002 \let\glscapscase\@thirdofthree
2003 \glstrchecknohyperfirst{#2}%
2004 ]%
2005 {#1}{#2}%
2006 {\@gls@field@font{\GLSaccessfirstplural{#2}\mfirstucMakeUppercase{#3}}}%
2007 }

```

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

```

2008 \def\@glsname@#1#2[#3]{%
2009 \glstrassignfieldfont{#2}%
2010 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessname{#2}#3}}%
2011 }

```

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

```

2012 \def\@Glsname@#1#2[#3]{%
2013 \glstrassignfieldfont{#2}%
2014 \@gls@field@link
2015 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2016 {\@gls@field@font{\Glsaccessname{#2}#3}}%
2017 }

```

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

```

2018 \def\@GLSname@#1#2[#3]{%
2019 \glstrassignfieldfont{#2}%
2020 \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2021 {#1}{#2}%
2022 {\@gls@field@font{\GLSaccessname{#2}\mfirstucMakeUppercase{#3}}}%
2023 }

```

**\@glsdesc@**

```

2024 \def\@glsdesc@#1#2[#3]{%
2025 \glstrassignfieldfont{#2}%
2026 \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccessdesc{#2}#3}}%
2027 }

```



```

\@Glsdesc@    First letter uppercase version.
2028 \def\@Glsdesc@#1#2[#3]{%
2029   \glstrassignfieldfont{#2}%
2030   \@gls@field@link
2031   [\let\glscapscase\@secondoftwo]{#1}{#2}%
2032   {\@gls@field@font{\Glsaccessdesc{#2}#3}}%
2033 }

\@GLSdesc@    All uppercase version.
2034 \def\@GLSdesc@#1#2[#3]{%
2035   \glstrassignfieldfont{#2}%
2036   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2037   {#1}{#2}{\@gls@field@font{\GLSaccessdesc{#2}\mfirstucMakeUppercase{#3}}}%
2038 }

@glsdscplural@    No case-changing version.
2039 \def\@glsdscplural@#1#2[#3]{%
2040   \glstrassignfieldfont{#2}%
2041   \@gls@field@link
2042   [\let\glscapscase\@secondoftwo
2043    \let\glsifplural\@firstoftwo
2044   ]{#1}{#2}{\@gls@field@font{\Glsaccessdscplural{#2}#3}}%
2045 }

@Glsdscplural@    First letter uppercase version.
2046 \def\@Glsdscplural@#1#2[#3]{%
2047   \glstrassignfieldfont{#2}%
2048   \@gls@field@link
2049   [\let\glscapscase\@secondoftwo
2050    \let\glsifplural\@firstoftwo
2051   ]{#1}{#2}{\@gls@field@font{\Glsaccessdscplural{#2}#3}}%
2052 }

@GLSdscplural@    All uppercase version.
2053 \def\@GLSdsc@#1#2[#3]{%
2054   \glstrassignfieldfont{#2}%
2055   \@gls@field@link
2056   [\let\glscapscase\@thirdoftwo
2057    \let\glsifplural\@firstoftwo
2058   ]%
2059   {#1}{#2}%
2060   {\@gls@field@font{\GLSaccessdscplural{#2}\mfirstucMakeUppercase{#3}}}%
2061 }

\@glssymbol@
2062 \def\@glssymbol@#1#2[#3]{%
2063   \glstrassignfieldfont{#2}%
2064   \@gls@field@link{#1}{#2}{\@gls@field@font{\glsaccesssymbol{#2}#3}}%
2065 }

```

\@Glssymbol@ First letter uppercase version.

```
2066 \def\@Glssymbol@#1#2[#3]{%
2067   \glstrassignfieldfont{#2}%
2068   \@gls@field@link
2069   [\let\glscapscase\@secondoftwo]%
2070   {#1}{#2}{\@gls@field@font{\Glsaccesssymbol{#2}#3}}%
2071 }
```

\@GLSsymbol@ All uppercase version.

```
2072 \def\@GLSsymbol@#1#2[#3]{%
2073   \glstrassignfieldfont{#2}%
2074   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2075   {#1}{#2}{\@gls@field@font{\GLSaccesssymbol{#2}\mfirstucMakeUppercase{#3}}}%
2076 }
```

lssymbolplural@ No case-changing version.

```
2077 \def\@lssymbolplural@#1#2[#3]{%
2078   \glstrassignfieldfont{#2}%
2079   \@gls@field@link
2080   [\let\glscapscase\@secondoftwo
2081   \let\glsifplural\@firstoftwo
2082   ]{#1}{#2}{\@gls@field@font{\glsaccesssymbolplural{#2}#3}}%
2083 }
```

lssymbolplural@ First letter uppercase version.

```
2084 \def\@lssymbolplural@#1#2[#3]{%
2085   \glstrassignfieldfont{#2}%
2086   \@gls@field@link
2087   [\let\glscapscase\@secondoftwo
2088   \let\glsifplural\@firstoftwo
2089   ]{#1}{#2}{\@gls@field@font{\Glsaccesssymbolplural{#2}#3}}%
2090 }
```

LSsymbolplural@ All uppercase version.

```
2091 \def\@LSsymbol@#1#2[#3]{%
2092   \glstrassignfieldfont{#2}%
2093   \@gls@field@link
2094   [\let\glscapscase\@thirdoftwo
2095   \let\glsifplural\@firstoftwo
2096   ]%
2097   {#1}{#2}%
2098   {\@gls@field@font{\GLSaccesssymbolplural{#2}\mfirstucMakeUppercase{#3}}}%
2099 }
```

\@Glsuseri@ First letter uppercase version.

```
2100 \def\@Glsuseri@#1#2[#3]{%
2101   \glstrassignfieldfont{#2}%
2102   \@gls@field@link
```

```

2103 [\let\glscapscase\@secondoftwo]{#1}{#2}%
2104 {\@gls@field@font{\Glsentryuseri{#2}#3}}%
2105 }

```

\@GLSuseri@ All uppercase version.

```

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

```

\@Glsuserii@ First letter uppercase version.

```

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

```

\@GLSuserii@ All uppercase version.

```

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

```

\@Glsuseriii@ First letter uppercase version.

```

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

```

\@GLSuseriii@ All uppercase version.

```

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

```

\@Glsuseriv@ First letter uppercase version.

```

2133 \def\@Glsuseriv@#1#2[#3]{%
2134   \glstrassignfieldfont{#2}%
2135   \@gls@field@link
2136   [\let\glscapscase\@secondoftwo]%
2137   {#1}{#2}{\@gls@field@font{\Glsentryuseriv{#2}#3}}%
2138 }

```

\@GLSuseriv@ All uppercase version.

```
2139 \def\@GLSuseriv@#1#2[#3]{%
2140   \glstrassignfieldfont{#2}%
2141   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2142   {#1}{#2}%
2143   {\@gls@field@font{\mfirstucMakeUppercase{\glstentryuseriv{#2}#3}}}%
2144 }
```

\@Glsuserv@ First letter uppercase version.

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

\@GLSuserv@ All uppercase version.

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

\@Glsuservi@ First letter uppercase version.

```
2156 \def\@Glsuservi@#1#2[#3]{%
2157   \glstrassignfieldfont{#2}%
2158   \@gls@field@link
2159   [\let\glscapscase\@secondoftwo]%
2160   {#1}{#2}{\@gls@field@font{\Glsentryuservi{#2}#3}}}%
2161 }
```

\@GLSuservi@ All uppercase version.

```
2162 \def\@GLSuservi@#1#2[#3]{%
2163   \glstrassignfieldfont{#2}%
2164   \@gls@field@link[\let\glscapscase\@thirdoftwo]%
2165   {#1}{#2}{\@gls@field@font{\mfirstucMakeUppercase{\glstentryuservi{#2}#3}}}%
2166 }
```

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

\@acrshort No case change.

```
2167 \def\@acrshort#1#2[#3]{%
2168   \glsdoifexists{#2}%
2169   {%
2170     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2171     \let\glstrifwasfirstuse\@secondoftwo
2172     \let\glsifplural\@secondoftwo
```

```

2173 \let\glscapscase\@firstofthree
2174 \let\glsinsert\@empty
2175 \def\glscustomtext{%
2176   \acronymfont{\glsaccesssshort{#2}}#3%
2177 }%
2178 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2179 }%
2180 \glspostlinkhook
2181 }

```

\@Acrshort First letter uppercase.

```

2182 \def\@Acrshort#1#2[#3]{%
2183   \glsdoifexists{#2}%
2184   {%
2185     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2186     \let\glstrifwasfirstuse\@secondoftwo
2187     \let\glsifplural\@secondoftwo
2188     \let\glscapscase\@secondofthree
2189     \let\glsinsert\@empty
2190     \def\glscustomtext{%
2191       \acronymfont{\Glsaccesssshort{#2}}#3%
2192     }%
2193     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2194   }%
2195   \glspostlinkhook
2196 }

```

\@ACRshort All uppercase.

```

2197 \def\@ACRshort#1#2[#3]{%
2198   \glsdoifexists{#2}%
2199   {%
2200     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2201     \let\glstrifwasfirstuse\@secondoftwo
2202     \let\glsifplural\@secondoftwo
2203     \let\glscapscase\@thirdofthree
2204     \let\glsinsert\@empty
2205     \def\glscustomtext{%
2206       \mfirstucMakeUppercase{\acronymfont{\glsaccesssshort{#2}}#3}%
2207     }%
2208     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2209   }%
2210   \glspostlinkhook
2211 }

```

\@acrshortpl No case change.

```

2212 \def\@acrshortpl#1#2[#3]{%
2213   \glsdoifexists{#2}%
2214   {%
2215     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

2216 \let\glxtrifwasfirstuse\@secondoftwo
2217 \let\glsifplural\@firstoftwo
2218 \let\glscapscase\@firstofthree
2219 \let\glsinsert\@empty
2220 \def\glscustomtext{%
2221     \acronymfont{\glsaccessshortpl{#2}}#3%
2222 }%
2223 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2224 }%
2225 \glspostlinkhook
2226 }

```

\@Acrshortpl First letter uppercase.

```

2227 \def\@Acrshortpl#1#2[#3]{%
2228     \glsdoifexists{#2}%
2229     {%
2230         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2231         \let\glxtrifwasfirstuse\@secondoftwo
2232         \let\glsifplural\@firstoftwo
2233         \let\glscapscase\@secondofthree
2234         \let\glsinsert\@empty
2235         \def\glscustomtext{%
2236             \acronymfont{\Glsaccessshortpl{#2}}#3%
2237         }%
2238         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2239     }%
2240     \glspostlinkhook
2241 }

```

\@ACRshortpl All uppercase.

```

2242 \def\@ACRshortpl#1#2[#3]{%
2243     \glsdoifexists{#2}%
2244     {%
2245         \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
2246         \let\glxtrifwasfirstuse\@secondoftwo
2247         \let\glsifplural\@firstoftwo
2248         \let\glscapscase\@thirdofthree
2249         \let\glsinsert\@empty
2250         \def\glscustomtext{%
2251             \mfirstucMakeUppercase{\acronymfont{\glsaccessshortpl{#2}}#3}%
2252         }%
2253         \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2254     }%
2255     \glspostlinkhook
2256 }

```

\@acrlong No case change.

```

2257 \def\@acrlong#1#2[#3]{%
2258     \glsdoifexists{#2}%

```

```

2259 {%
2260   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2261   \let\glxtrifwasfirstuse\@secondoftwo
2262   \let\gl@ifplural\@secondoftwo
2263   \let\gl@scapscase\@firstofthree
2264   \let\gl@insert\@empty
2265   \def\gl@customtext{%
2266     \acronymfont{\gl@saccesslong{#2}}#3%
2267   }%
2268   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2269 }%
2270 \glspostlinkhook
2271 }

```

\@Acrlong First letter uppercase.

```

2272 \def\@Acrlong#1#2[#3]{%
2273   \gl@ifexists{#2}%
2274   {%
2275     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2276     \let\glxtrifwasfirstuse\@secondoftwo
2277     \let\gl@ifplural\@secondoftwo
2278     \let\gl@scapscase\@secondofthree
2279     \let\gl@insert\@empty
2280     \def\gl@customtext{%
2281       \acronymfont{\gl@saccesslong{#2}}#3%
2282     }%
2283     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2284   }%
2285   \glspostlinkhook
2286 }

```

\@ACRlong All uppercase.

```

2287 \def\@ACRlong#1#2[#3]{%
2288   \gl@ifexists{#2}%
2289   {%
2290     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2291     \let\glxtrifwasfirstuse\@secondoftwo
2292     \let\gl@ifplural\@secondoftwo
2293     \let\gl@scapscase\@thirdofthree
2294     \let\gl@insert\@empty
2295     \def\gl@customtext{%
2296       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslong{#2}}#3}%
2297     }%
2298     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2299   }%
2300   \glspostlinkhook
2301 }

```

\@acrlongpl No case change.

```

2302 \def\@acrlongpl#1#2[#3]{%
2303   \glsdoifexists{#2}%
2304   {%
2305     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2306     \let\glxtrifwasfirstuse\@secondoftwo
2307     \let\gl@sifplural\@firstoftwo
2308     \let\glscapscase\@firstofthree
2309     \let\gl$insert\@empty
2310     \def\glscustomtext{%
2311       \acronymfont{\gl@saccesslongpl{#2}}#3%
2312     }%
2313     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2314   }%
2315   \glspostlinkhook
2316 }

```

\@Acrlongpl First letter uppercase.

```

2317 \def\@Acrlongpl#1#2[#3]{%
2318   \glsdoifexists{#2}%
2319   {%
2320     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2321     \let\glxtrifwasfirstuse\@secondoftwo
2322     \let\gl@sifplural\@firstoftwo
2323     \let\glscapscase\@secondofthree
2324     \let\gl$insert\@empty
2325     \def\glscustomtext{%
2326       \acronymfont{\Glsaccesslongpl{#2}}#3%
2327     }%
2328     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2329   }%
2330   \glspostlinkhook
2331 }

```

\@ACRlongpl All uppercase.

```

2332 \def\@ACRlongpl#1#2[#3]{%
2333   \glsdoifexists{#2}%
2334   {%
2335     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
2336     \let\glxtrifwasfirstuse\@secondoftwo
2337     \let\gl@sifplural\@firstoftwo
2338     \let\glscapscase\@thirdofthree
2339     \let\gl$insert\@empty
2340     \def\glscustomtext{%
2341       \mfirstucMakeUppercase{\acronymfont{\gl@saccesslongpl{#2}}#3}%
2342     }%
2343     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
2344   }%
2345   \glspostlinkhook
2346 }

```



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

\@glsaddkey

```

2347 \renewcommand*{\@glsaddkey}[7]{%
2348   \key@ifundefined{glossentry}{#1}%
2349   {%
2350     \define@key{glossentry}{#1}{\csdef{@glo@#1}{##1}}%
2351     \appto\@gls@keymap{,{#1}{#1}}%
2352     \appto\@newglossaryentryprehook{\csdef{@glo@#1}{#2}}%
2353     \appto\@newglossaryentryposthook{%
2354       \letcs{\@glo@tmp}{@glo@#1}%
2355       \gls@assign@field{#2}{\@glo@label}{#1}{\@glo@tmp}%
2356     }%
2357     \newcommand*{#3}[1]{\@gls@entry@field{##1}{#1}}%
2358     \newcommand*{#4}[1]{\@Gls@entry@field{##1}{#1}}%

```

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

```

2359   \ifcsdef{@gls@user@#1@}%
2360   {%
2361     \PackageError{glossaries}%
2362     {Can't define '\string#5' as helper command
2363     '\expandafter\string\csname @gls@user@#1@endcsname' already
2364     exists}%
2365     {}%
2366   }%
2367   {%
2368     \expandafter\newcommand\expandafter*\expandafter
2369     {\csname @gls@user@#1@endcsname}[2][ ]{%
2370       \new@ifnextchar[%
2371         {\csuse{@gls@user@#1@}{##1}{##2}}%
2372         {\csuse{@gls@user@#1@}{##1}{##2}[ ]}}%
2373     \csdef{@gls@user@#1@}##1##2[##3]{%
2374       \@gls@field@link{##1}{##2}{#3{##2}##3}%
2375     }%
2376     \newrobustcmd*{#5}{%
2377       \expandafter\@gls@hyp@opt\csname @gls@user@#1@endcsname}%
2378   }%

```

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

```

2379   \ifcsdef{@Gls@user@#1@}%
2380   {%
2381     \PackageError{glossaries}%
2382     {Can't define '\string#6' as helper command
2383     '\expandafter\string\csname @Gls@user@#1@endcsname' already
2384     exists}%
2385     {}%
2386   }%
2387   {%
2388     \expandafter\newcommand\expandafter*\expandafter

```

```

2389     {\csname @Gls@user@#1\endcsname}[2] [] {%
2390         \new@ifnextchar[%
2391             {\csuse{@Gls@user@#1@}{##1}{##2}}}%
2392             {\csuse{@Gls@user@#1@}{##1}{##2} [] }}}%
2393     \csdef{@Gls@user@#1@}##1##2[##3]{%
2394         \@gls@field@link[\let\glscapscase\secondofthree]%
2395         {##1}{##2}{#4{##2}##3}%
2396     }%
2397     \newrobustcmd*{#6}{%
2398         \expandafter\@gls@hyp@opt\csname @Gls@user@#1\endcsname}%
2399     }%

```

Finally the all caps version (modified):

```

2400     \ifcsdef{@GLS@user@#1@}%
2401     {%
2402         \PackageError{glossaries}%
2403         {Can't define '\string#7' as helper command
2404         '\expandafter\string\csname @GLS@user@#1\endcsname' already
2405         exists}%
2406     }%
2407 }%
2408 {%
2409     \expandafter\newcommand\expandafter*\expandafter
2410     {\csname @GLS@user@#1\endcsname}[2] [] {%
2411         \new@ifnextchar[%
2412             {\csuse{@GLS@user@#1@}{##1}{##2}}}%
2413             {\csuse{@GLS@user@#1@}{##1}{##2} [] }}}%
2414     \csdef{@GLS@user@#1@}##1##2[##3]{%
2415         \@gls@field@link[\let\glscapscase\thirdofthree]%
2416         {##1}{##2}{\mfirstucMakeUppercase{#3{##2}##3}}%
2417     }%
2418     \newrobustcmd*{#7}{%
2419         \expandafter\@gls@hyp@opt\csname @GLS@user@#1\endcsname}%
2420     }%
2421 }%
2422 {%
2423     \PackageError{glossaries-extra}{Key '#1' already exists}{}%
2424 }%
2425 }

```

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

```

2426 \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.

```

2427 \let\@glsxtr@org@checkfirsthyper\@gls@link@checkfirsthyper
2428 \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.

```

2429 \ifglssused{\glslabel}%
2430 {\let\glstrifwasfirstuse\@secondoftwo}
2431 {\let\glstrifwasfirstuse\@firstoftwo}%

Store the category label for convenience.

2432 \edef\glscategorylabel{\glscategory{\glslabel}}%
2433 \ifglssused{\glslabel}%
2434 {%
2435   \glssifcategoryattribute{\glscategorylabel}{nohypernext}{true}%
2436   {\KV@glsslink@hyperfalse}{}}%
2437 }%
2438 {%
2439   \glssifcategoryattribute{\glscategorylabel}{nohyperfirst}{true}%
2440   {\KV@glsslink@hyperfalse}{}}%
2441 }%
2442 \glsslinkcheckfirsthyperhook
2443 }

```

`\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.

```

2444 \ifdefined\do@glssdisablehyperinlist
2445 {%
2446   \let\@glstr@do@glssdisablehyperinlist\do@glssdisablehyperinlist
2447   \renewcommand*{\do@glssdisablehyperinlist}{%
2448     \@glstr@do@glssdisablehyperinlist
2449     \glssifattribute{\glslabel}{nohyper}{true}{\KV@glsslink@hyperfalse}{}}%
2450   }
2451 }
2452 {}

```

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

```

2453 \define@boolkey{glsslink}{noindex}[true]{}
2454 \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`

```

2455 \ifdefined\@glss@setdefault@glsslink@opts
2456 {
2457   \renewcommand*{\@glss@setdefault@glsslink@opts}{%
2458     \KV@glsslink@noindexfalse
2459     \@glstrsetaliasnoindex
2460   }
2461 }
2462 {}

```

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

```

2463 \newcommand*{\@glsglslSetDefault@glslLink@opts}{%
2464   \KV@glslLink@noindexfalse
2465   \@glslxtrsetaliasnoindex
2466 }
2467 \preto\do@glsglslDisableHyperinlist{\@glsglslSetDefault@glslLink@opts}
2468 }

```

`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.)

```

2469 \providecommand*{\glslxtrsetaliasnoindex}{%
2470   \KV@glslLink@noindextrue
2471 }

```

`setaliasnoindex`

```

2472 \newcommand*{\@glslxtrsetaliasnoindex}{%
2473   \glslxtrifhasfield{alias}{\glslabel}%
2474   {%
2475     \let\glslxtrindexaliased\@glslxtrindexaliased
2476     \glslxtrsetaliasnoindex
2477     \let\glslxtrindexaliased\@no@glslxtrindexaliased
2478   }%
2479   {%}%
2480 }

```

`xtrindexaliased`

```

2481 \newcommand{\@glslxtrindexaliased}{%
2482   \ifKV@glslLink@noindex
2483   \else
2484     \begingroup
2485     \let\@glslnumberformat\@glslxtr@defaultnumberformat
2486     \edef\@glsglsl@counter{\csname glo@glsglslDetokLabel{\glslabel}@counter\endcsname}%
2487     \glslxtr@saveentrycounter
2488     \@@do@wrglossary{\glslxtralias{\glslabel}}%
2489   \endgroup
2490   \fi
2491 }

```

`xtrindexaliased`

```

2492 \newcommand{\@no@glslxtrindexaliased}{%
2493   \PackageError{glossaries-extra}{\string\glslxtrindexaliased\space
2494     not permitted outside definition of \string\glslxtrsetaliasnoindex}%
2495   {%}%
2496 }

```

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

```

2497 \let\glslxtrindexaliased\@no@glslxtrindexaliased

```

DefaultGlsOpts Set the default options for \glslink etc.

```
2498 \newcommand*{\GlsXtrSetDefaultGlsOpts}[1]{%
2499   \renewcommand*{\@gls@setdefault@glslink@opts}{%
2500     \setkeys{glslink}{#1}%
2501     \@glsxtrsetaliasnoindex
2502   }%
2503 }
```

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

```
2504 \newcommand*{\glsxtrifindexing}[2]{%
2505   \ifKV@glslink@noindex #2\else #1\fi
2506 }
```

\glswriteentry Redefine to test for indexonlyfirst category attribute.

```
2507 \renewcommand*{\glswriteentry}[2]{%
2508   \glsxtrifindexing
2509   {%
2510     \ifglsindexonlyfirst
2511       \ifglsused{#1}
2512       {\glsxtrdoautoindexname{#1}{dualindex}}}%
2513     {#2}%
2514   \else
2515     \glsifattribute{#1}{indexonlyfirst}{true}%
2516     {\ifglsused{#1}
2517      {\glsxtrdoautoindexname{#1}{dualindex}}}%
2518     {#2}}%
2519   {#2}%
2520   \fi
2521 }%
2522 {}%
2523 }
```

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

```
2524 \appto\@do@@wrglossary{\@glsxtr@do@@wrindex
2525   \glsxtrdowrglossaryhook{\@gls@label}%
2526 }
```

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

Similarly for the “noidx” version:

s@noidxglossary

```
2527 \appto\gls@noidxglossary{\@glsxtr@do@@wrindex
2528   \glsxtrdowrglossaryhook{\@gls@label}%
2529 }
```

xtr@do@@wrindex

```
2530 \newcommand*{\@glsxtr@do@@wrindex}{%
```

```

2531 \glsxtrdoautoindexname{\@gls@label}{dualindex}%
2532 }

\newcommand*{\glsxtrdowrglossaryhook}[1]{%

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

2534 \newcommand*{\@gls@alt@hyp@opt}[1]{%
2535 \let\glslinkvar\@firstofthree
2536 \let\@gls@hyp@opt@cs#1\relax
2537 \@ifstar{\s@gls@hyp@opt}%
2538 {\@ifnextchar+%
2539 {\@firstoftwo{\p@gls@hyp@opt}}%
2540 {%
2541 \expandafter\@ifnextchar\@gls@alt@hyp@opt@char
2542 {\@firstoftwo{\@alt@gls@hyp@opt}}%
2543 {#1}%
2544 }%
2545 }%
2546 }

\alt@gls@hyp@opt User version

2547 \newcommand*{\@alt@gls@hyp@opt}[1][ ]{%
2548 \let\glslinkvar\@firstofthree
2549 \expandafter\@gls@hyp@opt@cs\expandafter[\@gls@alt@hyp@opt@keys,#1]}

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

2550 \newcommand*{\@gls@alt@hyp@opt@char}{%

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

2551 \newcommand*{\@gls@alt@hyp@opt@keys}{%

\SetAltModifier

2552 \newcommand*{\GlsXtrSetAltModifier}[2]{%
2553 \let\@gls@hyp@opt\@gls@alt@hyp@opt
2554 \def\@gls@alt@hyp@opt@char{#1}%
2555 \def\@gls@alt@hyp@opt@keys{#2}%
2556 }

\org@dohyperlink

2557 \let\glsxtr@org@dohyperlink\glsdohyperlink

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

```

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

```
2558 \ifdef\glsnavhyperlink
2559 {
2560   \renewcommand*{\glsnavhyperlink}[3][\@glo@type]{%
2561     \edef\gls@grplabel{#2}\protected@edef\@gls@grptitle{#3}%
```

Scope:

```
2562   {%
2563     \let\glsdohyperlink\glsxtr@org@dohyperlink
2564     \@glslink{\glsnavhyperlinkname{#1}{#2}}{#3}%
2565   }%
2566 }%
2567 }
2568 {}
```

`\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.

```
2569 \renewcommand*{\glsdohyperlink}[2]{%
2570   \glsasattribute{\glslabel}{targeturl}%
2571   {%
2572     \glsasattribute{\glslabel}{targetname}%
2573     {%
2574       \glsasattribute{\glslabel}{targetcategory}%
2575       {%
2576         \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2577           {\glsgetattribute{\glslabel}{targetcategory}}%
2578           {\glsgetattribute{\glslabel}{targetname}}%
2579           {\glsxtrprotectlinks#2}}%
2580       }%
2581     }%
2582     \hyperref{\glsgetattribute{\glslabel}{targeturl}}{%
2583       }%
2584       {\glsgetattribute{\glslabel}{targetname}}%
2585       {\glsxtrprotectlinks#2}}%
2586   }%
2587 }%
2588 {%
2589   \href{\glsgetattribute{\glslabel}{targeturl}}{%
2590     {\glsxtrprotectlinks#2}}%
2591 }%
2592 }%
2593 {}
```

Check for alias.

```

2594 \glsfieldfetch{\glslabel}{alias}{\gloaliaslabel}%
2595 \ifdefvoid\gloaliaslabel
2596 {%
2597   \glsxtrhyperlink{#1}{\glsxtrprotectlinks#2}}%
2598 }%
2599 {%

```

Redirect link to the alias target.

```

2600 \glsxtrhyperlink
2601 {\glolinkprefix\glsdetoklabel{\gloaliaslabel}}%
2602 {\glsxtrprotectlinks#2}}%
2603 }%
2604 }%
2605 }

```

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

```

2606 \ifdef\@glsshowtarget
2607 {
2608   \newcommand{\glsxtrhyperlink}[2]{%
2609     \@glsshowtarget{#1}%
2610     \hyperlink{#1}{#2}%
2611   }%
2612 }
2613 {
2614   \newcommand{\glsxtrhyperlink}[2]{\hyperlink{#1}{#2}}%
2615 }

```

**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.

```

2616 \renewrobustcmd*{\glsdohyperlink}[2][\glentrytext{\@glo@label}]{%
2617   \glsdoifexists{#2}%
2618   {%
2619     \def\@glo@label{#2}%
2620     {\edef\glslabel{#2}%
2621       \@glslink{\glolinkprefix\glslabel}{#1}}%
2622   }%
2623 }

```

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

```

2624 \renewcommand{\glsdisablehyper}{%
2625   \KV@glslink@hyperfalse
2626   \def\@glslink{\glsdohyperlink}%
2627   \let\@glstarget\@secondoftwo
2628 }

```



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

```
2629 \renewcommand{\glsenablehyper}{%
2630   \KV@glslink@hypertrue
2631   \def\@glslink{\glsdohyperlink}%
2632   \def\@glstarget{\glsdohypertarget}%
2633 }
```

`\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.

```
2634 \def\glsdonohyperlink#1#2{{\glstrprotectlinks #2}}
```

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

```
2635 \ifcsundef{hyperlink}%
2636 {%
2637   \def\@glslink{\glsdonohyperlink}
2638 }%
2639 {%
2640   \def\@glslink{\glsdohyperlink}
2641 }
```

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

```
2642 \newcommand*{\glstrprotectlinks}{%
2643   \KV@glslink@hyperfalse
2644   \KV@glslink@noindextrue
2645   \let\@gls@\@glstr@p@text@
2646   \let\@Gls@\@Glsxtr@p@text@
2647   \let\@GLS@\@GLSxtr@p@text@
2648   \let\@glspl@\@glstr@p@plural@
2649   \let\@Glspl@\@Glsxtr@p@plural@
2650   \let\@GLSpl@\@GLSxtr@p@plural@
2651   \let\@glstrshort@\@glstr@p@short@
2652   \let\@Glsxtrshort@\@Glsxtr@p@short@
2653   \let\@GLSxtrshort@\@GLSxtr@p@short@
2654   \let\@glstrlong@\@glstr@p@long@
2655   \let\@Glsxtrlong@\@Glsxtr@p@long@
2656   \let\@GLSxtrlong@\@GLSxtr@p@long@
2657   \let\@glstrshortpl@\@glstr@p@shortpl@
2658   \let\@Glsxtrshortpl@\@Glsxtr@p@shortpl@
2659   \let\@GLSxtrshortpl@\@GLSxtr@p@shortpl@
2660   \let\@glstrlongpl@\@glstr@p@longpl@
2661   \let\@Glsxtrlongpl@\@Glsxtr@p@longpl@
2662   \let\@GLSxtrlongpl@\@GLSxtr@p@longpl@
2663   \let\@acrshort@\@glstr@p@acrshort@
2664   \let\@Acrshort@\@Glsxtr@p@acrshort@
2665   \let\@ACRshort@\@GLSxtr@p@acrshort@}
```

```

2666 \let\@acrshortpl\@glsxtrp@acrshortpl@
2667 \let\@Acrshortpl\@Glsxtrp@acrshortpl@
2668 \let\@ACRshortpl\@GLSxtrp@acrshortpl@
2669 \let\@acrlong\@glsxtrp@acrlong@
2670 \let\@Acrlong\@Glsxtrp@acrlong@
2671 \let\@ACRlong\@GLSxtrp@acrlong@
2672 \let\@acrlongpl\@glsxtrp@acrlongpl@
2673 \let\@Acrlongpl\@Glsxtrp@acrlongpl@
2674 \let\@ACRlongpl\@GLSxtrp@acrlongpl@
2675 }

```

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

@glsxtrp@text@

```
2676 \def\@glsxtrp@text@#1#2[#3]{\@glstext@{#1}{#2}[#3]}
```

@Glsxtrp@text@

```
2677 \def\@Glsxtrp@text@#1#2[#3]{\@Glstext@{#1}{#2}[#3]}
```

@GLSxtrp@text@

```
2678 \def\@GLSxtrp@text@#1#2[#3]{\@GLStext@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2679 \def\@lsxtrp@plural@#1#2[#3]{\@glsplural@{#1}{#2}[#3]}
```

lsxtrp@plural@

```
2680 \def\@lsxtrp@plural@#1#2[#3]{\@Glsplural@{#1}{#2}[#3]}
```

LSxtrp@plural@

```
2681 \def\@LSxtrp@plural@#1#2[#3]{\@GLSplural@{#1}{#2}[#3]}
```

glsxtrp@short@

```

2682 \def\@glsxtrp@short@#1#2[#3]{%
2683   {%
2684     \glssetabbrvfmt{\glscategory{#2}}%
2685     \glsabbrvfont{\glsentryshort{#2}}#3%
2686   }%
2687 }

```

Glsxtrp@short@

```

2688 \def\@Glsxtrp@short@#1#2[#3]{%
2689   {%
2690     \glssetabbrvfmt{\glscategory{#2}}%
2691     \glsabbrvfont{\Glsentryshort{#2}}#3%
2692   }%
2693 }

```

GLSxtr@p@short@

```
2694 \def\@GLSxtr@p@short@#1#2[#3]{%
2695   {%
2696     \glsetabbrvfmt{\glscategory{#2}}%
2697     \mfirstucMakeUppercase{\glabbrvfont{\glentryshort{#2}}#3}%
2698   }%
2699 }
```

sxtr@p@shortpl@

```
2700 \def\@glxtr@p@shortpl@#1#2[#3]{%
2701   {%
2702     \glsetabbrvfmt{\glscategory{#2}}%
2703     \glabbrvfont{\glentryshortpl{#2}}#3%
2704   }%
2705 }
```

Sxtr@p@shortpl@

```
2706 \def\@Glsxtr@p@shortpl@#1#2[#3]{%
2707   {%
2708     \glsetabbrvfmt{\glscategory{#2}}%
2709     \glabbrvfont{\Glsentryshortpl{#2}}#3%
2710   }%
2711 }
```

Sxtr@p@shortpl@

```
2712 \def\@GLSxtr@p@shortpl@#1#2[#3]{%
2713   {%
2714     \glsetabbrvfmt{\glscategory{#2}}%
2715     \mfirstucMakeUppercase{\glabbrvfont{\glentryshortpl{#2}}#3}%
2716   }%
2717 }
```

@glxtr@p@long@

```
2718 \def\@glxtr@p@long@#1#2[#3]{\{\glentrylong{#2}#3}}
```

@Glsxtr@p@long@

```
2719 \def\@Glsxtr@p@long@#1#2[#3]{\{\Glsentrylong{#2}#3}}
```

@GLSxtr@p@long@

```
2720 \def\@GLSxtr@p@long@#1#2[#3]{%
2721   {\mfirstucMakeUppercase{\glslongfont{\glentrylong{#2}}#3}}}
```

lsxtr@p@longpl@

```
2722 \def\@glxtr@p@longpl@#1#2[#3]{\{\glentrylongpl{#2}#3}}
```

lsxtr@p@longpl@

```
2723 \def\@Glsxtr@p@longpl@#1#2[#3]{\{\glslongfont{\Glsentrylongpl{#2}}#3}}
```

```

LSxtr@p@longpl@
2724 \def\@GLSxtr@p@longpl@#1#2[#3]{%
2725   {\mfirstucMakeUppercase{\glslongfont{\glsentrylongpl{#2}}#3}}}

xtr@p@acrshort@
2726 \def\@glsxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\glsentryshort{#2}}#3}}

xtr@p@acrshort@
2727 \def\@GLSxtr@p@acrshort@#1#2[#3]{\{\acronymfont{\Glsentryshort{#2}}#3}}

xtr@p@acrshort@
2728 \def\@GLSxtr@p@acrshort@#1#2[#3]{%
2729   {\mfirstucMakeUppercase{\acronymfont{\glsentryshort{#2}}#3}}}

r@p@acrshortpl@
2730 \def\@glsxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2731 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{\{\acronymfont{\Glsentryshortpl{#2}}#3}}

r@p@acrshortpl@
2732 \def\@GLSxtr@p@acrshortpl@#1#2[#3]{%
2733   {\mfirstucMakeUppercase{\acronymfont{\glsentryshortpl{#2}}#3}}}

sxtr@p@acrlong@
2734 \def\@glsxtr@p@acrlong@#1#2[#3]{\{\glsentrylong{#2}}#3}}

sxtr@p@acrlong@
2735 \def\@GLSxtr@p@acrlong@#1#2[#3]{\{\Glsentrylong{#2}}#3}}

Sxtr@p@acrlong@
2736 \def\@GLSxtr@p@acrlong@#1#2[#3]{%
2737   {\mfirstucMakeUppercase{\glsentrylong{#2}}#3}}}

tr@p@acrlongpl@
2738 \def\@glsxtr@p@acrlongpl@#1#2[#3]{\{\glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2739 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{\{\Glsentrylongpl{#2}}#3}}

tr@p@acrlongpl@
2740 \def\@GLSxtr@p@acrlongpl@#1#2[#3]{%
2741   {\mfirstucMakeUppercase{\glsentrylongpl{#2}}#3}}}

Commands to minimise conflict.

\@glsxtrp@opt
2742 \newcommand*{\@glsxtrp@opt}{hyper=false,noindex}

```

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

```
2743 \newcommand*{\glsxtrsetpopts}[1]{%
2744   \renewcommand*{\@glsxtrp@opt}{#1}%
2745 }
```

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

```
2746 \newcommand*{\glossxtrsetpopts}{%
2747   \glsxtrsetpopts{noindex}%
2748 }
```

`\@glsxtrp`

```
2749 \newrobustcmd*{\@glsxtrp}[2]{%
```

Add scope.

```
2750   {%
2751     \let\glspostlinkhook\relax
2752     \csname#1\expandafter\endcsname\expandafter[\@glsxtrp@opt]{#2}[]%
2753   }%
2754 }
```

`\@glsxtrp`

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

`\@Glsxtrp`

```
2771 \newrobustcmd*{\@Glsxtrp}[2]{%
2772   \ifcsdef{Gls#1}%
2773     {%
2774       \@glsxtrp{Gls#1}{#2}%
2775     }%
2776     {%
2777       \ifcsdef{Glsxtr#1}%
2778         {%
2779           \@glsxtrp{Glsxtr#1}{#2}%
2780         }%
2781       }%
2782 }
```

```

2781   {%
2782     \PackageError{glossaries-extra}{‘#1’ not recognised by
2783       \string\Glsxtrp}{}%
2784   }%
2785 }%
2786 }

```

\@GLSxtrp

```

2787 \newrobustcmd*{\@GLSxtrp}[2]{%
2788   \ifcsdef{GLS#1}%
2789   {%
2790     \@@glxtrp{GLS#1}{#2}%
2791   }%
2792   {%
2793     \ifcsdef{GLSxtr#1}%
2794     {%
2795       \@@glxtrp{GLSxtr#1}{#2}%
2796     }%
2797     {%
2798       \PackageError{glossaries-extra}{‘#1’ not recognised by
2799         \string\GLSxtrp}{}%
2800     }%
2801   }%
2802 }

```

\glxtr@entry@p

```

2803 \newrobustcmd*{\glxtr@headentry@p}[2]{%
2804   \gl@ifattribute{#1}{headuc}{true}%
2805   {%
2806     \mfirstucMakeUppercase{\@gl@entry@field{#1}{#2}}%
2807   }%
2808   {%
2809     \@gl@entry@field{#1}{#2}%
2810   }%
2811 }

```

\glxtrp Not robust as it needs to expand somewhat.

```

2812 \ifdef\texorpdfstring
2813 {
2814   \newcommand{\glxtrp}[2]{%
2815     \protect\NoCaseChange
2816     {%
2817       \protect\texorpdfstring
2818       {%
2819         \protect\glxtrifinmark
2820         {%
2821           \ifcsdef{glxtrhead#1}%
2822           {%
2823             {\protect\csuse{glxtrhead#1}{#2}}%

```

```

2824         }%
2825         {%
2826         \glxtr@headentry@p{#2}{#1}%
2827         }%
2828     }%
2829     {%
2830     \@glxtrp{#1}{#2}%
2831     }%
2832 }%
2833 {%
2834 \protect\@gls@entry@field{#2}{#1}%
2835 }%
2836 }%
2837 }
2838 }
2839 {
2840 \newcommand{\glxtrp}[2]{%
2841 \protect\NoCaseChange
2842 {%
2843 \protect\glxtrifinmark
2844 {%
2845 \ifcsdef{glxtrhead#1}%
2846 {%
2847 {\protect\csuse{glxtrhead#1}}%
2848 }%
2849 {%
2850 \glxtr@headentry@p{#2}{#1}%
2851 }%
2852 }%
2853 {%
2854 \@glxtrp{#1}{#2}%
2855 }%
2856 }%
2857 }
2858 }

```

Provide short synonyms for the most common option.

`\glsp`

```
2859 \newcommand*{\glsp}{\glxtrp{short}}
```

`\glsp`

```
2860 \newcommand*{\glsp}{\glxtrp{text}}
```

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

```
2861 \ifdef\textorpdfstring
```

```
2862 {
```

```
2863 \newcommand{\Glsxtrp}[2]{%
```

```

2864 \protect\NoCaseChange
2865 {%
2866 \protect\texorpdfstring
2867 {%
2868 \protect\glxtrifinmark
2869 {%
2870 \ifcsdef{Glsxtrhead#1}%
2871 {%
2872 {\protect\csuse{Glsxtrhead#1}{#2}}%
2873 }%
2874 {%
2875 \protect\@Gls@entry@field{#2}{#1}%
2876 }%
2877 }%
2878 {%
2879 \@Glsxtrp{#1}{#2}%
2880 }%
2881 }%
2882 {%
2883 \protect\@gls@entry@field{#2}{#1}%
2884 }%
2885 }%
2886 }
2887 }
2888 {
2889 \newcommand{\Glsxtrp}[2]{%
2890 \protect\NoCaseChange
2891 {%
2892 \protect\glxtrifinmark
2893 {%
2894 \ifcsdef{Glsxtrhead#1}%
2895 {%
2896 {\protect\csuse{Glsxtrhead#1}}%
2897 }%
2898 {%
2899 \protect\@Gls@entry@field{#2}{#1}%
2900 }%
2901 }%
2902 {%
2903 \@Glsxtrp{#1}{#2}%
2904 }%
2905 }%
2906 }
2907 }

```

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

```

2908 \ifdef\texorpdfstring
2909 {
2910 \newcommand{\GlsXtrp}[2]{%

```



```

2911 \protect\NoCaseChange
2912 {%
2913 \protect\texorpdfstring
2914 {%
2915 \protect\glstrifinmark
2916 {%
2917 \ifcsdef{GLSxtr#1}%
2918 {%
2919 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2920 }%
2921 {%
2922 \protect\mfirstucMakeUppercase
2923 {%
2924 \protect\@gls@entry@field{#2}{#1}%
2925 }%
2926 }%
2927 }%
2928 {%
2929 \@GLSxtrp{#1}{#2}%
2930 }%
2931 }%
2932 {%
2933 \protect\@gls@entry@field{#2}{#1}%
2934 }%
2935 }%
2936 }
2937 }
2938 {
2939 \newcommand{\GLSxtrp}[2]{%
2940 \protect\NoCaseChange
2941 {%
2942 \protect\glstrifinmark
2943 {%
2944 \ifcsdef{GLSxtr#1}%
2945 {%
2946 {\protect\GLSxtrshort[noindex,hyper=false]{#1}[]}%
2947 }%
2948 {%
2949 \protect\mfirstucMakeUppercase
2950 {%
2951 \protect\@gls@entry@field{#2}{#1}%
2952 }%
2953 }%
2954 }%
2955 {%
2956 \@GLSxtrp{#1}{#2}%
2957 }%
2958 }%
2959 }

```

2960 }

### 1.3.5 Entry Counting

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

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

\@glsunset Global unset.

```
2961 \renewcommand*{\@glsunset}[1]{%
2962   \@glsunset{#1}%
2963   \glstrpostunset{#1}%
2964 }%
```

glstrpostunset

```
2965 \newcommand*{\glstrpostunset}[1]{}
```

\@glslocalunset Local unset.

```
2966 \renewcommand*{\@glslocalunset}[1]{%
2967   \@glslocalunset{#1}%
2968   \glstrpostlocalunset{#1}%
2969 }%
```

rpostlocalunset

```
2970 \newcommand*{\glstrpostlocalunset}[1]{}
```

\@glsreset Global reset.

```
2971 \renewcommand*{\@glsreset}[1]{%
2972   \@glsreset{#1}%
2973   \glstrpostreset{#1}%
2974 }%
```

glstrpostreset

```
2975 \newcommand*{\glstrpostreset}[1]{}
```

\@glslocalreset Local reset.

```
2976 \renewcommand*{\@glslocalreset}[1]{%
2977   \@glslocalreset{#1}%
2978   \glstrpostlocalreset{#1}%
2979 }%
```

rpostlocalreset

```
2980 \newcommand*{\glstrpostlocalreset}[1]{}
```

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

```
2981 \newcommand*{\GlsXtrEnableEntryCounting}[2]{%
```

Enable entry counting:

```
2982 \glsenableentrycount
```

Redefine \gls etc:

```
2983 \renewcommand*{\gls}{\cglsl}%
2984 \renewcommand*{\Gls}{\cGls}%
2985 \renewcommand*{\glspl}{\cglspl}%
2986 \renewcommand*{\Glspl}{\cGlspl}%
2987 \renewcommand*{\GLS}{\cGLS}%
2988 \renewcommand*{\GLSpl}{\cGLSpl}%
```

Set the entrycount attribute:

```
2989 \@glsxtr@setentrycountunsetattr{#1}{#2}%
```

In case this command is used again:

```
2990 \let\GlsXtrEnableEntryCounting\@glsxtr@setentrycountunsetattr
2991 \renewcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%
2992 \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryUnitCounting\space
2993 can't be used with \string\GlsXtrEnableEntryCounting}%
2994 {Use one or other but not both commands}}%
2995 }
```

countunsetattr

```
2996 \newcommand*{\@glsxtr@setentrycountunsetattr}[2]{%
2997 \@for\@glsxtr@cat:=#1\do
2998 {%
2999 \ifdefempty{\@glsxtr@cat}{}%
3000 {%
3001 \glssetcategoryattribute{\@glsxtr@cat}{entrycount}{#2}%
3002 }%
3003 }%
3004 }
```

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

nableentrycount

```
3005 \renewcommand*{\glsenableentrycount}{%
```

Enable new fields:

```
3006 \appto\@newglossaryentry@defcounters{\@@newglossaryentry@defcounters}%
```

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

```
3007 \renewcommand*{\gls@defdocnewglossaryentry}{%
3008 \renewcommand*{\newglossaryentry}[2]{%
3009 \PackageError{glossaries}{\string\newglossaryentry\space
3010 may only be used in the preamble when entry counting has
3011 been activated}{If you use \string\glsenableentrycount\space
```

```

3012     you must place all entry definitions in the preamble not in
3013     the document environment}%
3014 }%
3015 }%

```

New commands to access new fields:

```

3016 \newcommand*{\glstentrycurrcount}[1]{%
3017   \ifcsundef{glo@\glstetoklabel{##1}@currcount}%
3018   {0}{\@gls@entry@field{##1}{currcount}}}%
3019 }%
3020 \newcommand*{\glstentryprevcount}[1]{%
3021   \ifcsundef{glo@\glstetoklabel{##1}@prevcount}%
3022   {0}{\@gls@entry@field{##1}{prevcount}}}%
3023 }%

```

Adjust post unset and reset:

```

3024 \let\@glstxtr@entrycount@org@unset\glstxtrpostunset
3025 \renewcommand*{\glstxtrpostunset}[1]{%
3026   \@glstxtr@entrycount@org@unset{##1}%
3027   \@gls@increment@currcount{##1}%
3028 }%
3029 \let\@glstxtr@entrycount@org@localunset\glstxtrpostlocalunset
3030 \renewcommand*{\glstxtrpostlocalunset}[1]{%
3031   \@glstxtr@entrycount@org@localunset{##1}%
3032   \@gls@local@increment@currcount{##1}%
3033 }%
3034 \let\@glstxtr@entrycount@org@reset\glstxtrpostreset
3035 \renewcommand*{\glstxtrpostreset}[1]{%
3036   \@glstxtr@entrycount@org@reset{##1}%
3037   \csgdef{glo@\glstetoklabel{##1}@currcount}{0}%
3038 }%
3039 \let\@glstxtr@entrycount@org@localreset\glstxtrpostlocalreset
3040 \renewcommand*{\glstxtrpostlocalreset}[1]{%
3041   \@glstxtr@entrycount@org@localreset{##1}%
3042   \csdef{glo@\glstetoklabel{##1}@currcount}{0}%
3043 }%

```

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

```

3044 \let\@cgl@%\@cgl@
3045 \let\@cgl@pl%\@cgl@pl@

3046 \let\@cGls%\@cGls@
3047 \let\@cGlspl%\@cGlspl@
3048 \let\@cGLS%\@cGLS@
3049 \let\@cGLSpl%\@cGLSpl@

```

The rest is as the original definition.

```

3050 \AtEndDocument{\@gls@write@entrycounts}%
3051 \renewcommand*{\@gls@entry@count}[2]{%
3052   \csgdef{glo@\glstetoklabel{##1}@prevcount}{##2}%

```

```

3053 }%
3054 \let\glsenableentrycount\relax
3055 \renewcommand*{\glsenableentryunitcount}{%
3056   \PackageError{glossaries-extra}{\string\glsenableentryunitcount\space
3057     can't be used with \string\glsenableentrycount}%
3058   {Use one or other but not both commands}}%
3059 }%
3060 }

```

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

```

3061 \renewcommand*{\@gls@write@entrycounts}{%
3062   \immediate\write\@auxout
3063     {\string\providecommand*{\string\@gls@entry@count}[2]{}}%
3064   \count@=0\relax
3065   \forallglsentries{\@glsentry}{%
3066     \glshasattribute{\@glsentry}{entrycount}%
3067     {%
3068       \ifglsused{\@glsentry}%
3069       {%
3070         \immediate\write\@auxout
3071           {\string\@gls@entry@count{\@glsentry}{\glsentrycurrcount{\@glsentry}}}%
3072       }%
3073     }%
3074     \advance\count@ by \@ne
3075   }%
3076 }%
3077 }%
3078 \ifnum\count@=0
3079   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3080     \MessageBreak with \string\glsenableentrycount\space but the
3081     \MessageBreak attribute 'entrycount' hasn't
3082     \MessageBreak been assigned to any of the defined
3083     \MessageBreak entries}%
3084 \fi
3085 }

```

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

```

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

```

```

3093 \fi
3094 }%
3095 {#3}%
3096 }

```

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

\@@cgl@

```

3097 \def\@@cgl@#1#2[#3]{%
3098 \glxtrifcounttrigger{#2}%
3099 {%
3100 \cglformat{#2}{#3}%
3101 \glunset{#2}%
3102 }%
3103 {%
3104 \@gls@{#1}{#2}[#3]%
3105 }%
3106 }%

```

\@@cglsp@

```

3107 \def\@@cglsp@#1#2[#3]{%
3108 \glxtrifcounttrigger{#2}%
3109 {%
3110 \cglspformat{#2}{#3}%
3111 \glunset{#2}%
3112 }%
3113 {%
3114 \@glsp@{#1}{#2}[#3]%
3115 }%
3116 }%

```

\@@cGls@

```

3117 \def\@@cGls@#1#2[#3]{%
3118 \glxtrifcounttrigger{#2}%
3119 {%
3120 \cGlsformat{#2}{#3}%
3121 \glunset{#2}%
3122 }%
3123 {%
3124 \@Gls@{#1}{#2}[#3]%
3125 }%
3126 }%

```

\@@cGlspl@

```

3127 \def\@@cGlspl@#1#2[#3]{%
3128 \glxtrifcounttrigger{#2}%
3129 {%
3130 \cGlsplformat{#2}{#3}%
3131 \glunset{#2}%

```

```

3132 }%
3133 {%
3134   \@Glspl@{#1}{#2}[#3]%
3135 }%
3136 }%

```

\@cGLS@

```

3137 \def\@cGLS@#1#2[#3]{%
3138   \glstrifcounttrigger{#2}%
3139   {%
3140     \cGLSformat{#2}{#3}%
3141     \glset{#2}%
3142   }%
3143   {%
3144     \@GLS@{#1}{#2}[#3]%
3145   }%
3146 }%

```

\@cGLSpl@

```

3147 \def\@cGLSpl@#1#2[#3]{%
3148   \glstrifcounttrigger{#2}%
3149   {%
3150     \cGLSplformat{#2}{#3}%
3151     \glset{#2}%
3152   }%
3153   {%
3154     \@GLSpl@{#1}{#2}[#3]%
3155   }%
3156 }%

```

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

\@cgl's@

```

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

```

\@cGls@

```

3158 \def\@cGls@#1#2[#3]{\@Gls@{#1}{#2}[#3]}

```

\@cgl'spl@

```

3159 \def\@cgl'spl@#1#2[#3]{\@gl'spl@{#1}{#2}[#3]}

```

\@cGlspl@

```

3160 \def\@cGlspl@#1#2[#3]{\@Glspl@{#1}{#2}[#3]}

```

Add all upper case versions not provided by glossaries.

\cGLS

```

3161 \newrobustcmd*{\cGLS}{\@gl's@hyp@opt\@cGLS}

```

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

```
3162 \newcommand*{\@cGLS}[2][\%  
3163   \new@ifnextchar[\@cGLS@{#1}{#2}]{\@cGLS@{#1}{#2}[]}%  
3164 }
```

`\@cGLS@`

```
3165 \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.

```
3166 \newcommand*{\cGLSformat}[2]{%  
3167   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglformat{#1}{#2}}%  
3168 }
```

`\cGLSpl`

```
3169 \newrobustcmd*{\cGLSpl}{\@gls@hyp@opt\@cGLSpl}
```

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

```
3170 \newcommand*{\@cGLSpl}[2][\%  
3171   \new@ifnextchar[\@cGLSpl@{#1}{#2}]{\@cGLSpl@{#1}{#2}[]}%  
3172 }
```

`\@cGLSpl@`

```
3173 \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.

```
3174 \newcommand*{\cGLSplformat}[2]{%  
3175   \expandafter\mfirstuc\MakeUppercase\expandafter{\cglsplformat{#1}{#2}}%  
3176 }
```

Modify the trigger formats to check for the regular attribute.

`\cglformat`

```
3177 \renewcommand*{\cglformat}[2]{%  
3178   \glsifregular{#1}  
3179   {\glsentryfirst{#1}}%  
3180   {\ifglshaslong{#1}{\glsentrylong{#1}}{\glsentryfirst{#1}}}%#2%  
3181 }
```

`\cGlsformat`

```
3182 \renewcommand*{\cGlsformat}[2]{%  
3183   \glsifregular{#1}  
3184   {\Glsentryfirst{#1}}%  
3185   {\ifglshaslong{#1}{\Glsentrylong{#1}}{\Glsentryfirst{#1}}}%#2%  
3186 }
```



\cglsplformat

```
3187 \renewcommand*{\cglsplformat}[2]{%
3188   \glsifregular{#1}
3189   {\glsentryfirstplural{#1}}%
3190   {\ifglshaslong{#1}{\glsentrylongpl{#1}}{\glsentryfirstplural{#1}}}\#2%
3191 }
```

\cGlsplformat

```
3192 \renewcommand*{\cGlsplformat}[2]{%
3193   \glsifregular{#1}
3194   {\Glsentryfirstplural{#1}}%
3195   {\ifglshaslong{#1}{\Glsentrylongpl{#1}}{\Glsentryfirstplural{#1}}}\#2%
3196 }
```

New code similar to above for unit counting.

defunitcounters

```
3197 \newcommand*{\@@newglossaryentry@defunitcounters}{%
3198   \edef\@glo@countunit{\csuse{@glxtr@categoryattr@@\@glo@category @unitcount}}%
3199   \ifdefvoid\@glo@countunit
3200   {}%
3201   {%
3202     \@glxtr@ifunitcounter{\@glo@countunit}%
3203     {}%
3204     {\expandafter\@glxtr@addunitcounter\expandafter{\@glo@countunit}}%
3205     }%
3206 }
```

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

```
3207 \newcommand*{\@glxtr@unitcountlist}{}
```

@addunitcounter

```
3208 \newcommand*{\@glxtr@addunitcounter}[1]{%
3209   \listadd{\@glxtr@unitcountlist}{#1}%
3210   \ifcsundef{glxtr@theunit@#1}
3211   {%
3212     \ifcsdef{theH#1}%
3213     {\csdef{glxtr@theunit@#1}{\csuse{theH#1}}}%
3214     {\csdef{glxtr@theunit@#1}{\csuse{the#1}}}%
3215   }%
3216   {}%
3217 }
```

r@ifunitcounter

```
3218 \newcommand*{\@glxtr@ifunitcounter}[3]{%
3219   \xifinlist{#1}{\@glxtr@unitcountlist}{#2}{#3}%
3220 }
```

urrentunitcount

```
3221 \newcommand*\@glxtr@currentunitcount[1]{%
3222   glo@\glstoklabel{#1}@currunit@\glsglobalattribute{#1}{unitcount}.%
3223   \csuse{glxtr@theunit@\glsglobalattribute{#1}{unitcount}}%
3224 }
```

previousunitcount

```
3225 \newcommand*\@glxtr@previousunitcount[1]{%
3226   glo@\glstoklabel{#1}@prevunit@\glsglobalattribute{#1}{unitcount}.%
3227   \csuse{glxtr@theunit@\glsglobalattribute{#1}{unitcount}}%
3228 }
```

t@currunitcount

```
3229 \newcommand*\@gl@increment@currunitcount[1]{%
3230   \glshasattribute{#1}{unitcount}%
3231   {%
3232     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
3233     \ifcsundef{\@glxtr@csname}%
3234     {%
3235       \csgdef{\@glxtr@csname}{1}%
3236       \listcsxadd
3237         {glo@\glstoklabel{#1}@unitlist}%
3238         {\glsglobalattribute{#1}{unitcount}.%
3239         \csuse{glxtr@theunit@\glsglobalattribute{#1}{unitcount}}}%
3240       }%
3241     }%
3242     {%
3243       \csxdef{\@glxtr@csname}%
3244       {\number\numexpr\csname\@glxtr@csname\endcsname+1}%
3245     }%
3246   }%
3247   {%
3248 }
```

t@currunitcount

```
3249 \newcommand*\@gl@local@increment@currunitcount[1]{%
3250   \glshasattribute{#1}{unitcount}%
3251   {%
3252     \edef\@glxtr@csname{\@glxtr@currentunitcount{#1}}%
3253     \ifcsundef{\@glxtr@csname}%
3254     {%
3255       \csdef{\@glxtr@csname}{1}%
3256       \listcseadd
3257         {glo@\glstoklabel{#1}@unitlist}%
3258         {\glsglobalattribute{#1}{unitcount}.%
3259         \csuse{glxtr@theunit@\glsglobalattribute{#1}{unitcount}}}%
3260       }%
3261     }%
3262     {%
```

```

3263      \csedef{\@glsxtr@csname}%
3264      {\number\numexpr\csname\@glsxtr@csname\endcsname+1}%
3265      }%
3266      }%
3267      {}%
3268  }

```

r@currunitcount

```

3269 \newcommand*{\@glsxtr@currunitcount}[2]{%
3270   \ifcsundef
3271   {glo@\glsdetoklabel{#1}@currunit@#2}%
3272   {0}%
3273   {\csuse{glo@\glsdetoklabel{#1}@currunit@#2}}%
3274 }%

```

r@prevunitcount

```

3275 \newcommand*{\@glsxtr@prevunitcount}[2]{%
3276   \ifcsundef
3277   {glo@\glsdetoklabel{#1}@prevunit@#2}%
3278   {0}%
3279   {\csuse{glo@\glsdetoklabel{#1}@prevunit@#2}}%
3280 }%

```

eentryunitcount

```

3281 \newcommand*{\glsenableentryunitcount}{%
  Enable new fields:
3282   \appto\@newglossaryentry@defcounters{\@newglossaryentry@defunitcounters}%
  Just in case the user has switched on the docdef option.
3283   \renewcommand*{\gls@defdocnewglossaryentry}{%
3284     \renewcommand*\newglossaryentry[2]{%
3285       \PackageError{glossaries}{\string\newglossaryentry\space
3286       may only be used in the preamble when entry counting has
3287       been activated}{If you use \string\glsenableentryunitcount\space
3288       you must place all entry definitions in the preamble not in
3289       the document environment}%
3290     }%
3291   }%

  New commands to access new fields:
3292   \newcommand*{\glsentrycurrcount}[1]{%
3293     \@glsxtr@currunitcount{##1}{\glsgetattribute{##1}{unitcount}.%
3294     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
3295   }%
3296   \newcommand*{\glsentryprevcount}[1]{%
3297     \@glsxtr@prevunitcount{##1}{\glsgetattribute{##1}{unitcount}.%
3298     \csuse{glsxtr@theunit@\glsgetattribute{##1}{unitcount}}}%
3299   }%

```

Access total count:

```

3300 \newcommand*{\glstentryprevtotalcount}[1]{%
3301   \ifcsundef{glo@\glstetoklabel{##1}@prevunittotal}%
3302   {0}%
3303   {%
3304     \number\csuse{glo@\glstetoklabel{##1}@prevunittotal}
3305   }%
3306 }%
```

Access max value:

```

3307 \newcommand*{\glstentryprevmaxcount}[1]{%
3308   \ifcsundef{glo@\glstetoklabel{##1}@prevunitmax}%
3309   {0}%
3310   {%
3311     \number\csuse{glo@\glstetoklabel{##1}@prevunitmax}
3312   }%
3313 }%
```

Adjust post unset and reset:

```

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

```

3345    {}%
3346  }%

```

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

```

3347  \let\@cgls@\@cgls@
3348  \let\@cglspl@\@cglspl@

3349  \let\@cGls@\@cGls@
3350  \let\@cGlspl@\@cGlspl@
3351  \let\@cGLS@\@cGLS@
3352  \let\@cGLSpl@\@cGLSpl@

```

Write information to the aux file.

```

3353  \AtEndDocument{\@gls@write@entryunitcounts}%
3354  \renewcommand*{\@gls@entry@unitcount}[3]{%
3355    \csgdef{glo@glstdetoklabel{##1}@prevunit@##3}{##2}%
3356    \ifcsundef{glo@glstdetoklabel{##1}@prevunittotal}%
3357    {\csgdef{glo@glstdetoklabel{##1}@prevunittotal}{##2}}%
3358    {%
3359      \csxdef{glo@glstdetoklabel{##1}@prevunittotal}{
3360        \number\numexpr\csuse{glo@glstdetoklabel{##1}@prevunittotal}+##2}%
3361      }%
3362      \ifcsundef{glo@glstdetoklabel{##1}@prevunitmax}%
3363      {\csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}}%
3364      {%
3365        \ifnum\csuse{glo@glstdetoklabel{##1}@prevunitmax}<##2
3366        \csgdef{glo@glstdetoklabel{##1}@prevunitmax}{##2}%
3367        \fi
3368      }%
3369    }%
3370  \let\glsenableentryunitcount\relax
3371  \renewcommand*{\glsenableentrycount}{%
3372    \PackageError{glossaries-extra}{\string\glsenableentrycount\space
3373      can't be used with \string\glsenableentryunitcount}%
3374    {Use one or other but not both commands}%
3375  }%
3376 }
3377 \@onlypreamble\glsenableentryunitcount

```

entry@unitcount

```

3378 \newcommand*{\@gls@entry@unitcount}[3]{}

```

ryunitcounts@do

```

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

```

```

3385     {#1}}%
3386 }

```

entryunitcounts

```

3387 \newcommand*{\@gls@write@entryunitcounts}{%
3388   \immediate\write\@auxout
3389     {\string\providecommand*{\string\@gls@entry@unitcount}[3]{}}%
3390   \count@=0\relax
3391   \forallglsentries{\@glsentry}{%
3392     \glshasattribute{\@glsentry}{unitcount}%
3393     {%
3394       \ifglsused{\@glsentry}%
3395       {%
3396         \forlistcsloop
3397           {\@gls@write@entryunitcounts@do}%
3398           {glo@\glsdetoklabel{\@glsentry}@unitlist}%
3399       }%
3400     }%
3401     \advance\count@ by \@ne
3402   }%
3403 }%
3404 }%
3405 \ifnum\count@=0
3406   \GlossariesExtraWarningNoLine{Entry counting has been enabled
3407     \MessageBreak with \string\glsenableentryunitcount\space but the
3408     \MessageBreak attribute ‘unitcount’ hasn’t
3409     \MessageBreak been assigned to any of the defined
3410     \MessageBreak entries}%
3411 \fi
3412 }

```

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.

```

3413 \newcommand*{\GlsXtrEnableEntryUnitCounting}[3]{%

```

Enable entry counting:

```

3414   \glsenableentryunitcount

```

Redefine \gls etc:

```

3415   \renewcommand*{\gls}{\cgl}%
3416   \renewcommand*{\Gls}{\cGls}%
3417   \renewcommand*{\glspl}{\cglspl}%
3418   \renewcommand*{\Glspl}{\cGlspl}%
3419   \renewcommand*{\GLS}{\cGLS}%
3420   \renewcommand*{\GLSpl}{\cGLSpl}%

```

Set the entrycount attribute:

```

3421   \@glsxtr@setentryunitcountunsetattr{#1}{#2}{#3}%

```

In case this command is used again:

```

3422 \let\GlsXtrEnableEntryUnitCounting\@glxtr@setentryunitcountunsetattr
3423 \renewcommand*{\GlsXtrEnableEntryCounting}[2]{%
3424   \PackageError{glossaries-extra}{\string\GlsXtrEnableEntryCounting\space
3425     can't be used with \string\GlsXtrEnableEntryUnitCounting}%
3426   {Use one or other but not both commands}}%
3427 }

```

tcountunsetattr

```

3428 \newcommand*{\@glxtr@setentryunitcountunsetattr}[3]{%
3429   \@for\@glxtr@cat:=#1\do
3430   {%
3431     \ifdefempty{\@glxtr@cat}{}%
3432     {%
3433       \glsssetcategoryattribute{\@glxtr@cat}{entrycount}{#2}%
3434       \glsssetcategoryattribute{\@glxtr@cat}{unitcount}{#3}%
3435     }%
3436   }%
3437 }

```

### 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.

nericNewAcronym

```

3438 \renewcommand*{\SetGenericNewAcronym}{%
3439   \let\@Gls@entryname\@Gls@acrenryname
3440   \renewcommand{\newacronym}[4][{}]{%
3441     \ifdefempty{\@glsacronymlists}%
3442     {%
3443       \def\@glo@type{\acronymtype}%
3444       \setkeys{glossentry}{##1}%
3445       \DeclareAcronymList{\@glo@type}%
3446     }%
3447   }%
3448   \glskeylisttok{##1}%
3449   \glslabeltok{##2}%
3450   \glssshorttok{##3}%
3451   \glslongtok{##4}%
3452   \newacronymhook
3453   \protected@edef\@do@newglossaryentry{%
3454     \noexpand\newglossaryentry{\the\glslabeltok}%
3455     {%
3456       type=\acronymtype,%

```

```

3457     name={\expandonce{\acronymentry{##2}}},%
3458     sort={\acronymsort{\the\glsshorttok}{\the\glslongtok}},%
3459     text={\the\glsshorttok},%
3460     short={\the\glsshorttok},%
3461     shortplural={\the\glsshorttok\noexpand\acrpluralsuffix},%
3462     long={\the\glslongtok},%
3463     longplural={\the\glslongtok\noexpand\acrpluralsuffix},%
3464     category=acronym,%
3465     \GenericAcronymFields,%
3466     \the\glskeylisttok
3467   }%
3468 }%
3469 \do@newglossaryentry
3470 }%
3471 \renewcommand*{\acrfullfmt}[3]{%
3472   \glslink[##1]{##2}{\genacrfullformat{##2}{##3}}}%
3473 \renewcommand*{\Acrfullfmt}[3]{%
3474   \glslink[##1]{##2}{\Genacrfullformat{##2}{##3}}}%
3475 \renewcommand*{\ACRfullfmt}[3]{%
3476   \glslink[##1]{##2}{%
3477     \mfirstucMakeUppercase{\genacrfullformat{##2}{##3}}}%
3478 \renewcommand*{\acrfullplfmt}[3]{%
3479   \glslink[##1]{##2}{\genplacrfullformat{##2}{##3}}}%
3480 \renewcommand*{\Acrfullplfmt}[3]{%
3481   \glslink[##1]{##2}{\Genplacrfullformat{##2}{##3}}}%
3482 \renewcommand*{\ACRfullplfmt}[3]{%
3483   \glslink[##1]{##2}{%
3484     \mfirstucMakeUppercase{\genplacrfullformat{##2}{##3}}}%
3485 \renewcommand*{\glsentryfull}[1]{\genacrfullformat{##1}{}}}%
3486 \renewcommand*{\Glsentryfull}[1]{\Genacrfullformat{##1}{}}}%
3487 \renewcommand*{\glsentryfullpl}[1]{\genplacrfullformat{##1}{}}}%
3488 \renewcommand*{\Glsentryfullpl}[1]{\Genplacrfullformat{##1}{}}}%
3489 }

```

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:

```

3490 \let\@glxtr@org@setacronymstyle\setacronymstyle
3491 \let\@glxtr@org@newacronymstyle\newacronymstyle

```

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

```

3492 \newcommand*{\MakeAcronymsAbbreviations}{%
3493   \renewcommand*{\newacronym}[4][{}]{%
3494     \glxtr@newabbreviation{type=\acronymtype,category=acronym,##1}{##2}{##3}{##4}%
3495   }%
3496   \renewcommand*{\firstacronymfont}[1]{\glsfirstabbrvfont{##1}}%
3497   \renewcommand*{\acronymfont}[1]{\glsabbrvfont{##1}}%

```



```

3498 \renewcommand*{\setacronymstyle}[1]{%
3499     \PackageError{glossaries-extra}{\string\setacronymstyle{##1}
3500     unavailable.
3501     Use \string\setabbreviationstyle\space instead.
3502     The original acronym interface can be restored with
3503     \string\RestoreAcronyms}{}%
3504 }%
3505 \renewcommand*{\newacronymstyle}[1]{%
3506     \GlossariesExtraWarning{New acronym style ‘##1’ won’t be
3507     available unless you restore the original acronym interface with
3508     \string\RestoreAcronyms}%
3509     \@glxtr@org@newacronymstyle{##1}%
3510 }%
3511 }

```

Switch acronyms to abbreviations:

```

3512 \MakeAcronymsAbbreviations

```

**RestoreAcronyms** Restore acronyms to glossaries interface.

```

3513 \newcommand*{\RestoreAcronyms}{%
3514     \SetGenericNewAcronym
3515     \renewcommand{\firstacronymfont}[1]{\acronymfont{##1}}%
3516     \renewcommand{\acronymfont}[1]{##1}%
3517     \let\setacronymstyle\@glxtr@org@setacronymstyle
3518     \let\newacronymstyle\@glxtr@org@newacronymstyle

```

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

```

3519 \renewcommand*\@gls@link@checkfirsthyper{%
3520     \ifglused{\glslabel}%
3521     {\let\glxtrifwasfirstuse\@secondoftwo}
3522     {\let\glxtrifwasfirstuse\@firstoftwo}%
3523     \@glxtr@org@checkfirsthyper
3524 }
3525 \glssetcategoryattribute{acronym}{regular}{false}%
3526 \setacronymstyle{long-short}%
3527 }

```

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

```

3528 \renewcommand*{\glsacspace}[1]{%
3529     \settowidth{\dimen@}{(\firstacronymfont{\glsentryshort{#1}})}%
3530     \ifdim\dimen@<\glsacspacemax~\else\space\fi
3531 }

```

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

```

3532 \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

```
3533 \newcommand*{\@glxtr@reg@glosslist}{}
```

Save the original definition of `\makeglossaries`:

```
3534 \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

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

Iterate through each supplied glossary type and activate it.

```
3561   \@for\@glo@type:=#1\do{%
3562     \ifdefempty{\@glo@type}{\@makeglossary{\@glo@type}}%
3563   }%
```

New glossaries must be created before `\makeglossaries`:

```
3564 \renewcommand*\newglossary[4] [] {%
3565 \PackageError{glossaries}{New glossaries
3566 must be created before \string\makeglossaries}{You need
3567 to move \string\makeglossaries\space after all your
3568 \string\newglossary\space commands}}%
```

Any subsequence instances of this command should have no effect

```
3569 \let\@makeglossary\relax
3570 \let\makeglossary\relax
3571 \renewcommand\makeglossaries[1] [] {}%
```

Disable all commands that have no effect after `\makeglossaries`

```
3572 \@disable@onlypremakeg
```

Allow see key:

```
3573 \let\gls@checkseeallowed\relax
```

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

```
3574 \renewcommand*\@do@seeglossary[2] {%
3575 \glsdoifexists{##1}%
3576 {%
3577 \edef\@gls@label{\glsdetoklabel{##1}}%
3578 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3579 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3580 {\@glsxtr@org@doseeglossary{##1}{##2}}%
3581 {%
3582 \@@glsxtrwrglossmark
3583 \protected@write\@auxout{}{%
3584 \string\@gls@reference
3585 {\@gls@type}{\@gls@label}{\string\glsseeformat##2{}}%
3586 }%
3587 }%
3588 }%
3589 }%
```

Adjust `\@@do@@wrglossary`

```
3590 \let\@glsxtr@@do@@wrglossary\@@do@@wrglossary
3591 \def\@@do@@wrglossary{%
3592 \edef\@gls@type{\csname glo@\@gls@label @type\endcsname}%
3593 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3594 {\@glsxtr@@do@@wrglossary}%
3595 {\@gls@noidxglossary}%
3596 }%
```

Suppress warning about no `\makeglossaries`

```
3597 \let\warn@nomakeglossaries\relax
3598 \def\warn@noprntglossary{%
3599 \GlossariesWarningNoLine{No \string\printglossary\space
3600 or \string\printglossaries\space
3601 found.^^J(Remove \string\makeglossaries\space if you don't want
3602 any glossaries.)^^JThis document will not have a glossary}%
```

```

3603 }%
    Only warn for glossaries not listed.
3604 \renewcommand{\@gls@noref@warn}[1]{%
3605 \edef\@gls@type{##1}%
3606 \expandafter\DTLifinlist\expandafter{\@gls@type}{\@glsxtr@reg@glosslist}%
3607 {%
3608 \GlossariesExtraWarning{Can't use
3609 \string\printnoidxglossary[type={\@gls@type}]
3610 when '\@gls@type' is listed in the optional argument of
3611 \string\makeglossaries}%
3612 }%
3613 {%
3614 \GlossariesWarning{Empty glossary for
3615 \string\printnoidxglossary[type={##1}].
3616 Rerun may be required (or you may have forgotten to use
3617 commands like \string\gls)}%
3618 }%
3619 }%

```

Adjust display number list to check for type:

```

3620 \renewcommand*\@glsdisplaynumberlist[1]{%
3621 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3622 {\@glsxtr@idx@displaynumberlist{##1}}%
3623 {\@glsxtr@noidx@displaynumberlist{##1}}%
3624 }%

```

Adjust entry list:

```

3625 \renewcommand*\@glsentrynumberlist[1]{%
3626 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3627 {\@glsxtr@idx@entrynumberlist{##1}}%
3628 {\@glsxtr@noidx@entrynumberlist{##1}}%
3629 }%

```

Adjust number list loop

```

3630 \renewcommand*\@glsnumberlistloop[2]{%
3631 \expandafter\DTLifinlist\expandafter{##1}{\@glsxtr@reg@glosslist}%
3632 {%
3633 \PackageError{glossaries-extra}{\string\glsnumberlistloop\space
3634 not available for glossary '##1'}{}}%
3635 }%
3636 {\@glsxtr@noidx@numberlistloop{##1}{##2}}%
3637 }%

```

Only sanitize sort for normal indexing glossaries.

```

3638 \renewcommand*\@glsprestandardsort[3]{%
3639 \expandafter\DTLifinlist\expandafter{##2}{\@glsxtr@reg@glosslist}%
3640 {%
3641 \glsdosanitizesort
3642 }%
3643 {%

```

```

3644         \ifglssanitizesort
3645         \@gls@noidx@sanitizesort
3646     \else
3647         \@gls@noidx@nosanitizesort
3648     \fi
3649 }%
3650 }%

```

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

```

3651     \renewcommand*\new@glossaryentry[2]{%
3652         \PackageError{glossaries-extra}{Glossary entries must be defined
3653             in the preamble\MessageBreak when you use the optional argument
3654             of \string\makeglossaries}{Either move your definitions to the
3655             preamble or don't use the optional argument of
3656             \string\makeglossaries}%
3657     }%

```

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).

```

3658     \let\@glo@assign@sortkey\@glsextr@mixed@assign@sortkey
3659     \renewcommand*\@printgloss@setsort{%

```

Need to extract just the type value.

```

3660         \expandafter\@glsextr@gettype\expandafter,\@glsextr@printglossopts,%
3661         type=\glsdefaulttype,\@end@glsextr@gettype
3662     \def\@glo@sorttype{\@glo@default@sorttype}%
3663 }%

```

Check automake setting:

```

3664     \ifglssautomake
3665     \renewcommand*\@gls@doautomake{%
3666         \@for\@gls@type:=\@glsextr@reg@glosslist\do{%
3667             \ifdefempty{\@gls@type}{\@gls@automake{\@gls@type}}%
3668         }%
3669     }%
3670 \fi

```

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

```

3671     \ifdef\@glo@check@sortallowed{\@glo@check@sortallowed\makeglossaries}{}%
3672     \fi
3673 }%
3674 \fi
3675 }

```

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.)

```

3676 \newcommand{\@glstr@orgprintglossary}[2]{%
3677   \def\@glo@type{\glstexttype}%
3678   \def\@glo@type{\glstexttype}%
3679   \ifcsdef{\@glo@type\@glo@type @title}%
3680     {\csuse{\@glo@type\@glo@type @title}}%
3681     {\glossaryname}}%
3682 \def\glossarytitle{\glossarytitle}%
3683 \let\org@glossarytitle\glossarytitle
3684 \def\@glossarystyle{%
3685   \ifx\@glossary@default@style\relax
3686     \GlossariesWarning{No default glossary style provided \MessageBreak
3687       for the glossary '\@glo@type'. \MessageBreak
3688       Using deprecated fallback. \MessageBreak
3689       To fix this set the style with \MessageBreak
3690       \string\setglossarystyle\space or use the \MessageBreak
3691       style key=value option}%
3692   \fi
3693 }%
3694 \def\glstexttitle{\glstexttitle{\@glo@type}}%
3695 \let\org@glossaryentrynumbers\glossaryentrynumbers
3696 \bgroup
3697   \@printgloss@setsort
3698   \setkeys{printgloss}{#1}%
3699   \ifx\glossarytitle\org@glossarytitle
3700   \else
3701     \cslet{\@glo@type\@glo@type @title}{\glossarytitle}%
3702   \fi
3703   \let\currentglossary\@glo@type
3704   \let\org@glossaryentrynumbers\glossaryentrynumbers
3705   \let\glstextpages\glstextpages
3706   \let\glstextpages\glstextpages
3707   \glstextactivatenopost
3708   \glstexttitle
3709   \@glossarystyle
3710   \let\glstextorg@glossaryentryfield\glossentry
3711   \let\glstextorg@glossarysubentryfield\subglossentry
3712   \renewcommand{\glossentry}[1]{%
3713     \xdef\glstextcurrententrylabel{\glstexttoklabel{##1}}%
3714     \glstextorg@glossaryentryfield{##1}%
3715   }%
3716   \renewcommand{\subglossentry}[2]{%
3717     \xdef\glstextcurrententrylabel{\glstexttoklabel{##2}}%
3718     \glstextorg@glossarysubentryfield{##1}{##2}%
3719   }%
3720   \@glstextpreglossaryhook

```

```

3721     #2%
3722   \egroup
3723   \global\let\glossaryentrynumbers\@org@glossaryentrynumbers
3724   \global\let\warn@noprintglossary\relax
3725 }

```

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

```

3726 \newcommand*{\glxtractivatenopost}{%
3727   \let\nopostdesc\@nopostdesc
3728   \let\glxtrnopostpunc\@glxtr@nopostpunc
3729 }

```

lsxtrnopostpunc

```

3730 \newrobustcmd*{\glxtrnopostpunc}{%

```

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

```

3731 \newcommand{\@glxtr@nopostpunc}{%
3732   \let\@glxtr@org@postdescription\glspostdescription
3733   \ifglsnopostdot
3734     \renewcommand{\glspostdescription}{%
3735       \glsnopostdottrue
3736       \let\glspostdescription\@glxtr@org@postdescription
3737       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3738       \glxtrpostdescription
3739       \@glxtr@nopostpunc@postdesc}%
3740   \else
3741     \renewcommand{\glspostdescription}{%
3742       \let\glspostdescription\@glxtr@org@postdescription
3743       \let\glxtrrestorepostpunc\@glxtr@restore@postpunc
3744       \glxtrpostdescription
3745       \@glxtr@nopostpunc@postdesc}%
3746   \fi
3747   \glsnopostdotfalse
3748 }

```

stpunc@postdesc

```

3749 \newcommand*{\@glxtr@nopostpunc@postdesc}{%

```

estore@postpunc

```

3750 \newcommand*{\@glxtr@restore@postpunc}{%
3751   \def\@glxtr@nopostpunc@postdesc{%
3752     \@glxtr@org@postdescription
3753     \let\@glxtr@nopostpunc@postdesc\@empty
3754     \let\glxtrrestorepostpunc\@empty
3755   }%
3756 }

```

restorepostpunc Does nothing outside of glossary.

```
3757 \newcommand*{\glxtrrestorepostpunc}{}

\@printglossary Redefine.
```

```
3758 \renewcommand{\@printglossary}[2]{%
3759   \def\@glxtr@printglossopts{#1}%
3760   \@glxtr@orgprintglossary{#1}{#2}%
3761 }
```

Add a key that switches off the entry targets:

```
3762 \define@choicekey{printgloss}{target}
3763 [\@glxtr@printglossval\@glxtr@printglossnr]%
3764 {true,false}[true]%
3765 {%
3766   \ifcase\@glxtr@printglossnr
3767     \let\@glstarget\glstdohypertarget
3768   \else
3769     \let\@glstarget\@secondoftwo
3770   \fi
3771 }
```

hypernameprefix

```
3772 \newcommand{\@glxtrhypernameprefix}{}

New to v1.20:
```

```
3773 \define@key{printgloss}{targetnameprefix}{%
3774   \renewcommand{\@glxtrhypernameprefix}{#1}%
3775 }
```

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

```
3776 \let\@glxtr@org@glstdohypertarget\glstdohypertarget
3777 \renewcommand{\glstdohypertarget}[2]{%
3778   \@glxtr@org@glstdohypertarget{\@glxtrhypernameprefix#1}{#2}%
3779 }
```

@makeglossaries For the benefit of makeglossaries

```
3780 \newcommand*{\glxtr@makeglossaries}[1]{}

\@glxtr@gettype Get just the type.
```

```
3781 \def\@glxtr@gettype#1,type=#2,#3\@end@glxtr@gettype{%
3782   \def\@glo@type{#2}%
3783 }
```

@assign@sortkey Assign the sort key.

```
3784 \newcommand\@glxtr@mixed@assign@sortkey[1]{%
3785   \edef\@glo@type{\@glo@type}%
3786   \expandafter\DTLifinlist\expandafter{\@glo@type}{\@glxtr@reg@glosslist}%
3787   {%
```



```

3788 \@glo@no@assign@sortkey{#1}%
3789 }%
3790 {%
3791 \@@glo@assign@sortkey{#1}%
3792 }%
3793 }%

```

Display number list for the regular version:

splaynumberlist

```

3794 \let\@glstr@idx@displaynumberlist\glstrdisplaynumberlist

```

Display number list for the “noidx” version:

splaynumberlist

```

3795 \newcommand*{\@glstr@noidx@displaynumberlist}[1]{%
3796 \letcs{\@glstr@loclist}{glo\@glstrdetoklabel{#1}@loclist}%
3797 \ifdef\@glstr@loclist
3798 {%
3799 \def\@glstr@noidx@loclist@sep{%
3800 \def\@glstr@noidx@loclist@sep{%
3801 \def\@glstr@noidx@loclist@sep{%
3802 \glstrnumlistsep
3803 }%
3804 \def\@glstr@noidx@loclist@finalsep{\glstrnumlistlastsep}%
3805 }%
3806 }%
3807 \def\@glstr@noidx@loclist@finalsep{}%
3808 \def\@glstr@noidx@loclist@prev{}%
3809 \forlistloop{\glstrnoidxdisplayloclisthandler}{\@glstr@loclist}%
3810 \@glstr@noidx@loclist@finalsep
3811 \@glstr@noidx@loclist@prev
3812 }%
3813 {%
3814 \glstrtrundeftag
3815 \glstrdoifexists{#1}%
3816 {%
3817 \GlossariesWarning{Missing location list for ‘#1’. Either
3818 a rerun is required or you haven’t referenced the entry.}%
3819 }%
3820 }%
3821 }%
3822

```

And for the number list loop:

@numberlistloop

```

3823 \newcommand*{\@glstr@noidx@numberlistloop}[3]{%
3824 \letcs{\@glstr@loclist}{glo\@glstrdetoklabel{#1}@loclist}%
3825 \let\@glstr@org@glstrnoidxdisplayloc\glstrnoidxdisplayloc

```

```

3826 \let\@gls@org@glsseeformat\glsseeformat
3827 \let\glsnoidxdisplayloc#2\relax
3828 \let\glsseeformat#3\relax
3829 \ifdef\@gls@loclist
3830 {%
3831   \forlistloop{\glsnoidxnumberlistloophandler}{\@gls@loclist}%
3832 }%
3833 {%

3834   \glsxtrundeftag
3835   \glsdoifexists{#1}%
3836   {%
3837     \GlossariesWarning{Missing location list for ‘##1’. Either
3838       a rerun is required or you haven’t referenced the entry.}%
3839   }%
3840 }%
3841 \let\glsnoidxdisplayloc\@gls@org@glsnoidxdisplayloc
3842 \let\glsseeformat\@gls@org@glsseeformat
3843 }%

```

Same for entry number list.

entrynumberlist

```

3844 \newcommand*{\@glsxtr@noidx@entrynumberlist}[1]{%
3845   \letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
3846   \ifdef\@gls@loclist
3847   {%
3848     \glsnoidxloclist{\@gls@loclist}%
3849   }%
3850   {%

3851     \glsxtrundeftag
3852     \glsdoifexists{#1}%
3853     {%
3854       \GlossariesWarning{Missing location list for ‘#1’. Either
3855         a rerun is required or you haven’t referenced the entry.}%
3856     }%
3857   }%
3858 }%

```

entrynumberlist

```

3859 \newcommand*{\@glsxtr@idx@entrynumberlist}[1]{\glsentrynumberlist{#1}}

```

x@getgrouptitle Patch.

```

3860 \renewcommand*{\@gls@noidx@getgrouptitle}[2]{%
3861   \protected@edef\@glsxtr@titlelabel{#1}%
3862   \ifdefvoid\@glsxtr@titlelabel
3863   {}%
3864   {%
3865     \protected@edef\@glsxtr@titlelabel{\csuse{\glsxtr@grouptitle@#1}}%

```

```

3866 }%
3867 \ifdefvoid{\@glsxtr@titlelabel}%
3868 {%
3869   \DTLifint{#1}%
3870   {%
3871     \ifnum#1<256\relax
3872       \edef#2{\char#1\relax}%
3873     \else
3874       \edef#2{#1}%
3875     \fi
3876   }%
3877   {%
3878     \ifcsundef{#1groupname}%
3879     {\def#2{#1}}%
3880     {\letcs#2{#1groupname}}%
3881   }%
3882 }%
3883 {%
3884   \let#2\@glsxtr@titlelabel
3885 }%
3886 }

```

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

```

3887 \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.

```

3888 \newrobustcmd{\glsxtrgetgrouptitle}[2]{%
3889   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3890   \@onelevel@sanitize\@glsxtr@titlelabel
3891   \ifcsdef{\@glsxtr@titlelabel}
3892   {\letcs{#2}{\@glsxtr@titlelabel}}%
3893   {\glsxtr@org@getgrouptitle{#1}{#2}}%
3894 }
3895 \let\@gls@getgrouptitle\glsxtrgetgrouptitle

```

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

```

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

```

**alsetgrouptitle** As above put only locally defines the title.

```

3901 \newcommand{\glsxtrlocalsetgrouptitle}[2]{%
3902   \protected@edef\@glsxtr@titlelabel{\glsxtr@grouptitle@#1}%
3903   \@onelevel@sanitize\@glsxtr@titlelabel
3904   \protected@csedef{\@glsxtr@titlelabel}{#2}%
3905 }

```

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

```

3906 \renewcommand*{\glsnavigation}{%
3907   \def\@gls@between{}%
3908   \ifcsundef\@gls@hypergroup\list\@glo@type}%
3909   {%
3910     \def\@gls@list{}%
3911   }%
3912   {%
3913     \expandafter\let\expandafter\@gls@list
3914       \csname @gls@hypergroup\list\@glo@type\endcsname
3915   }%
3916   \@for\@gls@tmp:=\@gls@list\do{%
3917     \@gls@between
3918     \glsxtrgetgrouptitle{\@gls@tmp}{\@gls@grptitle}%
3919     \glsnavhyperlink{\@gls@tmp}{\@gls@grptitle}%
3920     \let\@gls@between\glshypernavsep
3921   }%
3922 }
```

`@noidx@glossary`

```

3923 \renewcommand*{\@print@noidx@glossary}{%
3924   \ifcsdef\@glsref\@glo@type}%
3925   {%
3926     \ifcsdef\@glo@sortmacro\@glo@sorttype}%
3927     {%
3928       \csuse\@glo@sortmacro\@glo@sorttype{\@glo@type}%
3929     }%
3930     {%
3931       \PackageError{glossaries}{Unknown sort handler ‘\@glo@sorttype’}{}%
3932     }%
3933     \glossarysection[\glossarytoctitle]{\glossarytitle}%
3934     \glossarypreamble
```

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

```

3935   \def\@gls@currentlettergroup{}%
3936   \begin{theglossary}%
3937     \glossaryheader
3938     \glsresetentrylist
3939     \forlistcsloop{\@gls@noidx@do}{\@glsref\@glo@type}%
3940     \end{theglossary}%
3941     \glossarypostamble
3942   }%
3943   {%
```

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

```

3944     \glsxtrifemptyglossary{\@glo@type}%
3945     {}%
3946     {\glossarysection[\glossarytoctitle]{\glossarytitle}}%
```

```

3947 \@gls@noref@warn{\@glo@type}%
3948 }%
3949 }

```

noidxdisplayloc Patch to check for range formations.

```

3950 \renewcommand*{\glsnoidxdisplayloc}[4]{%
3951 \setentrycounter[#1]{#2}%
3952 \@glsxtr@display@loc#3\empty\end@glxtr@display@loc{#4}%
3953 }

```

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

```

3954 \def\@glsxtr@display@loc#1#2\end@glxtr@display@loc#3{%
3955 \ifx#1\relax
3956 \glsxtrdisplaystartloc{#2}{#3}%
3957 \else
3958 \ifx#1\relax
3959 \glsxtrdisplayendloc{#2}{#3}%
3960 \else
3961 \glsxtrdisplaysingleloc{#1#2}{#3}%
3962 \fi
3963 \fi
3964 }

```

isplaysingleloc Single location.

```

3965 \newcommand*{\glsxtrdisplaysingleloc}[2]{%
3966 \csuse{#1}{#2}%
3967 }

```

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 `\glsxtrlocrangefmt`.

displaystartloc Start of a location range.

```

3968 \newcommand*{\glsxtrdisplaystartloc}[2]{%
3969 \edef\glsxtrlocrangefmt{#1}%
3970 \ifx\glsxtrlocrangefmt\empty
3971 \def\glsxtrlocrangefmt{glsnumberformat}%
3972 \fi
3973 \expandafter\glsxtrdisplaysingleloc
3974 \expandafter{\glsxtrlocrangefmt}{#2}%
3975 }

```

trdisplayendloc End of a location range.

```

3976 \newcommand*{\glsxtrdisplayendloc}[2]{%
3977 \edef\@glsxtr@tmp{#1}%
3978 \ifdefempty{\@glsxtr@tmp}{\def\@glsxtr@tmp{glsnumberformat}}{}}%
3979 \ifx\glsxtrlocrangefmt\@glsxtr@tmp
3980 \else
3981 \GlossariesExtraWarning{Mismatched end location range
3982 (start=\glsxtrlocrangefmt, end=\@glsxtr@tmp)}%

```

```

3983 \fi
3984 \expandafter\glxtrdisplayendlochook\expandafter{\@glxtr@tmp}{#2}%
3985 \expandafter\glxtrdisplaysingleloc
3986 \expandafter{\glxtrlocrangefmt}{#2}%
3987 \def\glxtrlocrangefmt{}%
3988 }

```

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

```
3989 \newcommand*\glxtrdisplayendlochook}[2]{}

```

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

```
3990 \newcommand*\glxtrlocrangefmt{}

```

glx@removespaces Redefine to allow adjustments to location hyperlink.

```

3991 \def\@glx@removespaces#1 #2\@nil{%
3992 \toks@=\expandafter{\the\toks@#1}%
3993 \ifx\#2\%
3994 \edef\x{\the\toks@}%
3995 \ifx\x\empty
3996 \else
3997 \glxtrlocationhyperlink{\glxentrycounter}{\@glo@counterprefix}{\the\toks@}%
3998 \fi
3999 \else
4000 \@glx@ReturnAfterFi{%
4001 \@glx@removespaces#2\@nil
4002 }%
4003 \fi
4004 }

```

glxtrlocationhyperlink

```

4005 \newcommand*\glxtrlocationhyperlink}[3]{%
4006 \ifdefined\glxtrsupplocationurl
4007 {%
4008 \glxtrhyperlink{#1#2#3}{#3}%
4009 }%
4010 {%
4011 \hyperref{\glxtrsupplocationurl}{#1#2#3}{#3}%
4012 }%
4013 }

```

glxtrsupphypernumber

```

4014 \newcommand*\glxtrsupphypernumber}[1]{%
4015 {%
4016 \glshasattribute{\glxcurrententrylabel}{externallocation}%
4017 {%
4018 \def\glxtrsupplocationurl{%
4019 \glxgetattribute{\glxcurrententrylabel}{externallocation}}%
4020 }%
4021 {%

```

```

4022 \def\glstrsupplocationurl{%
4023 }%
4024 \glshypernumber{#1}%
4025 }%
4026 }

```

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

@print@glossary

```

4027 \renewcommand{\@print@glossary}{%
4028 \makeatletter
4029 \@input{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4030 \IfFileExists{\jobname.\csname @glotype@\@glo@type @in\endcsname}%
4031 {}%
4032 {\glstrNoGlossaryWarning{\@glo@type}}%
4033 \ifglxindy
4034 \ifcsundef{\xdy@\@glo@type @language}%
4035 {%
4036 \edef\@do@auxoutstuff{%
4037 \noexpand\AtEndDocument{%
4038 \noexpand\immediate\noexpand\write\@auxout{%
4039 \string\providecommand\string\@xdylanguage[2]{}}%
4040 \noexpand\immediate\noexpand\write\@auxout{%
4041 \string\@xdylanguage{\@glo@type}{\@xdy@main@language}}%
4042 }%
4043 }%
4044 }%
4045 {%
4046 \edef\@do@auxoutstuff{%
4047 \noexpand\AtEndDocument{%
4048 \noexpand\immediate\noexpand\write\@auxout{%
4049 \string\providecommand\string\@xdylanguage[2]{}}%
4050 \noexpand\immediate\noexpand\write\@auxout{%
4051 \string\@xdylanguage{\@glo@type}{\csname @xdy@\@glo@type
4052 @language\endcsname}}%
4053 }%
4054 }%
4055 }%
4056 \@do@auxoutstuff
4057 \edef\@do@auxoutstuff{%
4058 \noexpand\AtEndDocument{%
4059 \noexpand\immediate\noexpand\write\@auxout{%
4060 \string\providecommand\string\@gls@codepage[2]{}}%
4061 \noexpand\immediate\noexpand\write\@auxout{%
4062 \string\@gls@codepage{\@glo@type}{\@gls@codepage}}%
4063 }%
4064 }%
4065 \@do@auxoutstuff
4066 \fi

```

```

4067 \renewcommand*{\@warn@nomakeglossaries}{%
4068   \GlossariesWarningNoLine{\string\makeglossaries\space
4069   hasn't been used,^^Jthe glossaries will not be updated}%
4070 }%
4071 }

```

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

`\GlsWarningHead` Header message.

```

4072 \newcommand{\GlsXtrNoGlsWarningHead}[2]{%
4073   This document is incomplete. The external file associated with
4074   the glossary '#1' (which should be called \texttt{#2})
4075   hasn't been created.%
4076 }

```

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

```

4077 \newcommand{\GlsXtrNoGlsWarningEmptyStart}{%
4078   This has probably happened because there are no entries defined
4079   in this glossary.%
4080 }

```

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

```

4081 \newcommand{\GlsXtrNoGlsWarningEmptyMain}{%
4082   If you don't want this glossary,
4083   add \texttt{nomain} to your package option list when you load
4084   \texttt{glossaries-extra.sty}. For example:%
4085 }

```

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

```

4086 \newcommand{\GlsXtrNoGlsWarningEmptyNotMain}[1]{%
4087   Did you forget to use \texttt{type=#1} when you defined your
4088   entries? If you tried to load entries into this glossary with
4089   \texttt{\string\loadglsentries} did you remember to use
4090   \texttt{[#1]} as the optional argument? If you did, check that
4091   the definitions in the file you loaded all had the type set
4092   to \texttt{\string\glsdefaulttype}.%
4093 }

```

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

```

4094 \newcommand{\GlsXtrNoGlsWarningCheckFile}[1]{%
4095   Check the contents of the file \texttt{#1}. If
4096   it's empty, that means you haven't indexed any of your entries in this
4097   glossary (using commands like \texttt{\string\gls} or
4098   \texttt{\string\glsadd}) so this list can't be generated.
4099   If the file isn't empty, the document build process hasn't been
4100   completed.%
4101 }

```



WarningAutoMake Message when automake option has been used.

```
4102 \newcommand{\GlsXtrNoGlsWarningAutoMake}[1]{%
4103   You may need to rerun \LaTeX. If you already have, it may be that
4104   \TeX's shell escape doesn't allow you to run
4105   \ifglxindy xindy\else makeindex\fi. Check the
4106   transcript file \texttt{\jobname.log}. If the shell escape is
4107   disabled, try one of the following:
4108
4109   \begin{itemize}
4110     \item Run the external (Lua) application:
4111
4112       \texttt{makeglossaries-lite.lua \string\jobname\string}
4113
4114     \item Run the external (Perl) application:
4115
4116       \texttt{makeglossaries \string\jobname\string}
4117   \end{itemize}
4118
4119   Then rerun \LaTeX\ on this document.
4120   \GlossariesExtraWarning{Rerun required to build the
4121   glossary '#1' or check TeX's shell escape allows
4122   you to run \ifglxindy xindy\else makeindex\fi}%
4123 }
```

WarningMismatch Mismatching \makenoidxglossaries.

```
4124 \newcommand{\GlsXtrNoGlsWarningMisMatch}{%
4125   You need to either replace \texttt{\string\makenoidxglossaries}
4126   with \texttt{\string\makeglossaries} or replace
4127   \texttt{\string\printglossary} (or \texttt{\string\printglossaries}) with
4128   \texttt{\string\printnoidxglossary}
4129   (or \texttt{\string\printnoidxglossaries}) and then rebuild
4130   this document.%
4131 }
```

WarningBuildInfo Build advice.

```
4132 \newcommand{\GlsXtrNoGlsWarningBuildInfo}{%
4133   Try one of the following:
4134   \begin{itemize}
4135     \item Add \texttt{automake} to your package option list when you load
4136           \texttt{glossaries-extra.sty}. For example:
4137
4138       \texttt{\string\usepackage[automake]%
4139       \glsopenbrace glossaries-extra\glsclosebrace}
4140
4141     \item Run the external (Lua) application:
4142
4143       \texttt{makeglossaries-lite.lua \string\jobname\string}
4144   \end{itemize}
```

```

4145 \item Run the external (Perl) application:
4146
4147 \texttt{makeglossaries \string\jobname\string}
4148 \end{itemize}
4149
4150 Then rerun \LaTeX\ on this document.%
4151 }

```

oGlsWarningTail Final paragraph.

```

4152 \newcommand{\GlsXtrNoGlsWarningTail}{%
4153 This message will be removed once the problem has been fixed.%
4154 }

```

GlsWarningNoOut No out file created. Build advice.

```

4155 \newcommand{\GlsXtrNoGlsWarningNoOut}[1]{%
4156 The file \texttt{#1} doesn't exist. This most likely means you haven't used
4157 \texttt{\string\makeglossaries} or you have used
4158 \texttt{\string\nofiles}. If this is just a draft version of the
4159 document, you can suppress this message using the
4160 \texttt{\nomissingglstext} package option.%
4161 }

```

glossarywarning

```

4162 \newcommand*{@@glsxtr@defaultnoglossarywarning}[1]{%
4163 \glossarysection[\glossarytoctitle]{\glossarytitle}
4164 \GlsXtrNoGlsWarningHead{#1}{\jobname.\csname @glo@type @in\endcsname}
4165 \par
4166 \glsxtrifemptyglossary{#1}%
4167 {%
4168 \GlsXtrNoGlsWarningEmptyStart\space
4169 \ifthenelse{\equal{#1}{main}}{\GlsXtrNoGlsWarningEmptyMain\par
4170 \medskip
4171 \noindent\texttt{\string\usepackage[nomain\ifglsacronym ,acronym\fi]%
4172 \glsopenbrace glossaries-extra\glsclosebrace}
4173 \medskip
4174 }%
4175 {\GlsXtrNoGlsWarningEmptyNotMain{#1}}%
4176 }%
4177 {%
4178 \IfFileExists{\jobname.\csname @glo@type @\@glo@type @out\endcsname}
4179 {%
4180 \GlsXtrNoGlsWarningCheckFile
4181 {\jobname.\csname @glo@type @\@glo@type @out\endcsname}
4182
4183 \ifglsautomake
4184
4185 \GlsXtrNoGlsWarningAutoMake{#1}
4186
4187 \else

```

```

4188
4189     \ifthenelse{\equal{#1}{main}}{%
4190     {%
4191         \GlsXtrNoGlsWarningEmptyMain\par
4192         \medskip
4193         \noindent\texttt{\string\usepackage[nomain]%
4194             \glsopenbrace glossaries-extra\glsclosebrace}
4195         \medskip
4196     }%
4197     }%
4198
4199     \ifdefequal\makeglossaries\@no@makeglossaries
4200     {%
4201         \GlsXtrNoGlsWarningMismatch
4202     }%
4203     {%
4204         \GlsXtrNoGlsWarningBuildInfo
4205     }%
4206     \fi
4207 }%
4208 {%
4209     \GlsXtrNoGlsWarningNoOut
4210     {\jobname.\csname @glotype@\@glo@type @out\endcsname}%
4211 }%
4212 }%
4213 \par
4214 \GlsXtrNoGlsWarningTail
4215 }

```

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

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

```

4216 \newcommand*{\glstrresourcefile}[2] [] {%

```

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

```

4217 \disable@keys{glossaries-extra.sty}{record}%
4218 \glstr@writefields
4219 \protected@write\@auxout{\glstrresourceinit}{\string\glstr@resource{#1}{#2}}%
4220 \let\@glstr@org@see@noindex\@gls@see@noindex
4221 \let\@gls@see@noindex\relax
4222 \IfFileExists{#2.glstex}%
4223 {%

```

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

```

4224 \edef\@bibgls@restreat{\noexpand\catcode\noexpand'\noexpand\@=\number\catcode'\@}%
4225 \makeatletter
4226 \@input{#2.glstex}%
4227 \@bibgls@restreat

```

```

4228 }%
4229 {%
4230   \GlossariesExtraWarning{No file ‘#2.glstex’}%
4231 }%
4232 \let\@gls@see@noindex\@glsxtr@org@see@noindex
4233 }
4234 \@onlypreamble\glsxtrresourcefile

xtrresourceinit  Code used during the protected write operation.
4235 \newcommand*{\glsxtrresourceinit}{%

trresourcecount
4236 \newcount\glsxtrresourcecount

trLoadResources  Short cut that uses \glsxtrresourcefile with \jobname as the mandatory argument.
4237 \newcommand*{\GlsXtrLoadResources}[1][]{%
4238   \ifnum\glsxtrresourcecount=0\relax
4239     \glsxtrresourcefile[#1]{\jobname}%
4240   \else
4241     \glsxtrresourcefile[#1]{\jobname-\the\glsxtrresourcecount}%
4242   \fi
4243   \advance\glsxtrresourcecount by 1\relax
4244 }

glsxtr@resource
4245 \newcommand*{\glsxtr@resource}[2]{%

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

xtr@texencoding
4247 \newcommand*{\glsxtr@texencoding}[1]{%

\glsxtr@langtag
4248 \newcommand*{\glsxtr@langtag}[1]{%

@pluralsuffixes
4249 \newcommand*{\glsxtr@pluralsuffixes}[4]{%

tr@shortcutsval
4250 \newcommand*{\glsxtr@shortcutsval}[1]{%

sxtr@linkprefix
4251 \newcommand*{\glsxtr@linkprefix}[1]{%

xtr@writefields  This information only needs to be written once, so disable it after it’s been used.
4252 \newcommand*{\glsxtr@writefields}{%

```

```

4253 \protected@write\@auxout{}%
4254   {\string\providecommand*{\string\glsxtr@fields}[1]{}}%
4255 \protected@write\@auxout{}%
4256   {\string\providecommand*{\string\glsxtr@resource}[2]{}}%
4257 \protected@write\@auxout{}%
4258   {\string\providecommand*{\string\glsxtr@pluralsuffixes}[4]{}}%
4259 \protected@write\@auxout{}%
4260   {\string\providecommand*{\string\glsxtr@shortcutsval}[1]{}}%
4261 \protected@write\@auxout{}%
4262   {\string\providecommand*{\string\glsxtr@linkprefix}[1]{}}%
4263 \protected@write\@auxout{}{\string\glsxtr@fields{\@gls@keymap}}%

```

```

4264 \protected@write\@auxout{}%
4265   {\string\providecommand*{\string\glsxtr@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 `\glsxtrresourcefile`.

```

4266 \ifdef\CurrentTrackedLanguageTag
4267 {%
4268   \protected@write\@auxout{}%
4269     {\string\glsxtr@langtag{\CurrentTrackedLanguageTag}}%
4270 }%
4271 {%
4272   \protected@write\@auxout{}{\string\glsxtr@pluralsuffixes
4273     {\glspluralsuffix}{\abbrvpluralsuffix}{\acrpluralsuffix}%
4274     {\glsxtrabbrvpluralsuffix}}%
4275 \ifdef\inputencodingname
4276 {%
4277   \protected@write\@auxout{}{\string\glsxtr@texencoding{\inputencodingname}}%
4278 }%
4279 {%

```

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.)

```

4280   \@ifpackageloaded{fontspec}%
4281     {\protected@write\@auxout{}{\string\glsxtr@texencoding{utf8}}}%
4282   {}%
4283 }%
4284 \protected@write\@auxout{}{\string\glsxtr@shortcutsval{\@glsxtr@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.

```

4285 \AtBeginDocument
4286   {\protected@write\@auxout{}{\string\glsxtr@linkprefix{\glolinkprefix}}}%
4287 \let\glsxtr@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.

```

4288 \ifglsautomake
4289 \IfFileExists{\jobname.aux}%
4290 {\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.

```

4291 \ifx\@gls@doautomake\@gls@doautomake@err
4292 \let\@gls@doautomake\relax
4293 \fi
4294 \fi
4295 }

```

do@automake@err

```

4296 \newcommand*{\@gls@doautomake@err}{%
4297 \PackageError{glossaries}{You must use
4298 \string\makeglossaries\space with automake=true}
4299 {%
4300 Either remove the automake=true setting or
4301 add \string\makeglossaries\space to your document preamble.%
4302 }%
4303 }

```

Allow locations specific to a particular counter to be recorded.

\glsxtr@record

```

4304 \newcommand*{\glsxtr@record}[5]{}

```

r@counterrecord Aux file command.

```

4305 \newcommand*{\glsxtr@counterrecord}[3]{%
4306 \glsxtrfieldlistgadd{#1}{record.#2}{#3}%
4307 }

```

unterrecordhook Hook used by \@glsxtr@dorecord.

```

4308 \newcommand*{\@glsxtr@counterrecordhook}{}

```

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

```

4309 \newcommand*{\GlsXtrRecordCounter}[1]{%
4310 \@glsxtr@recordcounter{#1}%
4311 }
4312 \@onlypreamble\GlsXtrRecordCounter

```

docounterrecord

```

4313 \newcommand*{\@glsxtr@docounterrecord}[1]{%
4314 \protected@write\@auxout{}{\string\glsxtr@counterrecord
4315 {\@gls@label}{#1}{\csuse{the#1}}}%
4316 }

```

`\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.

```

4317 \newcommand*\glxtrglossentry}[1]{%
4318   \glxtrtitleorpdforheading
4319   {\@glxtrglossentry{#1}}%
4320   {\glsentryname{#1}}%
4321   {\glxtrheadname{#1}}%
4322 }

```

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

```

4323 \newrobustcmd*\@glxtrglossentry}[1]{%
4324   \glxtrtitleorpdforheading
4325   {%
4326     \glsdodefexists{#1}%
4327     {%
4328       \begingroup
4329       \edef\glscurrententrylabel{\glsetoklabel{#1}}%
4330       \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4331       \ifglshasparent{#1}%
4332       {\glssubentryitem{#1}}%
4333       {\glsentryitem{#1}}%
4334       \glstarget{#1}{\glossentryname{#1}}%
4335     \endgroup
4336   }%
4337 }%
4338 {\glsentryname{#1}}%
4339 {\glxtrheadname{#1}}%
4340 }

```

`\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.

```

4341 \newcommand*\glxtrglossentryother}[3]{%
4342   \ifstrempy{#1}%
4343   {%
4344     \ifcsdef{glxtrhead#3}%
4345     {%
4346       \glxtrtitleorpdforheading
4347       {\@glxtrglossentryother{#2}{#3}{#1}}%
4348       {\@glsetentry@field{#2}{#3}}%
4349       {\csuse{glxtrhead#3}{#2}}%
4350     }%

```

```

4351   {%
4352       \glstrtitleorpdforheading
4353       {\@glstrglossentryother{#2}{#3}{#1}}%
4354       {\@gls@entry@field{#2}{#3}}%
4355       {\@gls@entry@field{\NoCaseChange{#2}}{#3}}%
4356   }%
4357 }%
4358 {%
4359     \glstrtitleorpdforheading
4360     {\@glstrglossentryother{#2}{#3}{#1}}%
4361     {\@gls@entry@field{#2}{#3}}%
4362     {#1}%
4363 }%
4364 }

```

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

```

4365 \newrobustcmd*{\@glstrglossentryother}[3]{%
4366     \glstrtitleorpdforheading
4367     {%
4368         \glsdoifexists{#1}%
4369         {%
4370             \begingroup
4371                 \edef\glscurrententrylabel{\glsdetoklabel{#1}}%
4372                 \edef\currentglossary{\glsentrytype{\glscurrententrylabel}}%
4373                 \ifglshasparent{#1}%
4374                 {\glssubentryitem{#1}}%
4375                 {\glsentryitem{#1}}%
4376                 \glstarget{#1}{\glossentrynameother{#1}{#2}}%
4377             \endgroup
4378         }%
4379     }%
4380     {\@gls@entry@field{#1}{#2}}%
4381     {#3}%
4382 }

```

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

```

4383 \newcommand*{\printunsrtglossary}{%
4384     \@ifstar\s@printunsrtglossary\@printunsrtglossary
4385 }

```

`\printunsrtglossary` Unstarred version.

```

4386 \newcommand*{\@printunsrtglossary}[1][ ]{%
4387     \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4388 }

```

`\printunsrtglossary` Starred version.

```

4389 \newcommand*{\s@printunsrtglossary}[2][ ]{%
4390     \begingroup

```



```

4391      #2%
4392      \@printglossary{type=\glsdefaulttype,#1}{\@print@unsrt@glossary}%
4393 \endgroup
4394 }

```

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

```

4395 \newcommand*\printunsrtglossaries{%
4396   \forallglossaries{\@glo@type}{\printunsrtglossary[type=\@glo@type]}%
4397 }

```

`\@unsrt@glossary`

```

4398 \newcommand*\@print@unsrt@glossary{%
4399   \glossarysection[\glossarytoctitle]{\glossarytitle}%
4400   \glossarypreamble
4401   check for empty list
4402   \glsxtrifemptyglossary{\@glo@type}%
4403   {%
4404     \GlossariesExtraWarning{No entries defined in glossary ‘\@glo@type’}%
4405   }%
4406   \key@ifundefined{glossentry}{group}%
4407   {\let\@gls@getgrouptitle\@gls@noidx@getgrouptitle}%
4408   {\let\@gls@getgrouptitle\@glsxtr@unsrt@getgrouptitle}%
4409   \def\@gls@currentlettergroup{}}%

```

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

```

4410   \def\@glsxtr@doglossary{%
4411     \begin{theglossary}%
4412     \glossaryheader
4413     \glsresetentrylist
4414   }%
4415   \expandafter\@for\expandafter\glscurrententrylabel\expandafter
4416     :\expandafter=\csname glo@list@\@glo@type\endcsname\do{%
4417     \ifdefempty{\glscurrententrylabel}
4418     {}%
4419     {%

```

Provide a hook (for example to measure width).

```

4420       \let\glsxtr@process\@firstofone
4421       \let\printunsrtglossaryskipentry
4422         \@glsxtr@printunsrtglossaryskipentry
4423       \printunsrtglossaryentryprocesshook{\glscurrententrylabel}%

```

Don't check group for child entries.

```

4424       \glsxtr@process
4425       {%
4426       \ifglshasparent{\glscurrententrylabel}{}%
4427       {%

```

```

4428         \@glsxtr@checkgroup\glscurrententrylabel
4429         \expandafter\appto\expandafter\@glsxtr@doglossary\expandafter
4430         {\@glsxtr@groupheading}%
4431     }%
4432     \eappto\@glsxtr@doglossary{%
4433         \noexpand\@printunsrt@glossary@handler{\glscurrententrylabel}}%
4434     }%
4435 }%
4436 }%
4437 \appto\@glsxtr@doglossary{\end{theglossary}}%
4438 \printunsrtglossarypredoglossary
4439 \@glsxtr@doglossary
4440 }%
4441 \glossarypostamble
4442 }

```

entryprocesshook

```

4443 \newcommand*{\printunsrtglossaryentryprocesshook}[1]{}

```

entryprocesshook

```

4444 \newcommand*{\printunsrtglossaryskipentry}{%
4445     \PackageError{glossaries-extra}{\string\printunsrtglossaryskipentry\space
4446 can only be used within \string\printunsrtglossaryentryprocesshook}{}%
4447 }

```

entryprocesshook

```

4448 \newcommand*{\@glsxtr@printunsrtglossaryskipentry}{%
4449     \let\glsxtr@process\gobble
4450 }

```

rypredoglossary

```

4451 \newcommand*{\printunsrtglossarypredoglossary}{}

```

lossary@handler

```

4452 \newcommand{\@printunsrt@glossary@handler}[1]{%
4453     \xdef\glscurrententrylabel{#1}%
4454     \printunsrtglossaryhandler\glscurrententrylabel
4455 }

```

glossaryhandler

```

4456 \newcommand{\printunsrtglossaryhandler}[1]{%
4457     \glsxtrunsrtdo{#1}%
4458 }

```

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.

```

4459 \newrobustcmd*{\glsxtriflabelinlist}[4]{%

```

```

4460 \protected@edef\@glxstr@doiflabelinlist{\noexpand\@glx@ifinlist{#1}{#2}}%
4461 \@glxstr@doiflabelinlist{#3}{#4}%
4462 }

```

srtglossaryunit

```

4463 \newcommand{\print@op@unsrtglossaryunit}[2] [] {%
4464   \s@printunsrtglossary[type=\glxdefaulttype,#1]{%
4465     \printunsrtglossaryunitsetup{#2}%
4466   }%
4467 }

```

ossaryunitsetup

```

4468 \newcommand*{\printunsrtglossaryunitsetup}[1] {%
4469   \renewcommand{\printunsrtglossaryhandler}[1] {%
4470     \glxstrfieldxifinlist{##1}{record.#1}{\csuse{the#1}}
4471     {\glxstrunsrtdo{##1}}%
4472   }%
4473 }%

```

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

```

4474 \ifcsundef{theH#1}%
4475 {%
4476   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{the#1}.\@gobble}%
4477 }%
4478 {%
4479   \renewcommand*{\@glxstrhypernameprefix}{record.#1.\csuse{theH#1}.\@gobble}%
4480 }%
4481 \renewcommand*{\glossarysection}[2] [] {}%
4482 \appto\glossarypostamble{\glxpar\medskip\glxpar}%
4483 }

```

srtglossaryunit

```

4484 \newcommand{\print@noop@unsrtglossaryunit}[2] [] {%
4485   \PackageError{glossaries-extra}{\string\printunsrtglossaryunit\space
4486     requires the record=only or record=alsoindex package option}{}%
4487 }

```

t@getgrouptitle

```

4488 \newrobustcmd*{\@glxstr@unsrt@getgrouptitle}[2] {%
4489   \protected@edef\@glxstr@titlelabel{\glxstr@grouptitle@#1}%
4490   \@onelevel@sanitize\@glxstr@titlelabel
4491   \ifcsdef{\@glxstr@titlelabel}
4492     {\letcs{#2}{\@glxstr@titlelabel}}%
4493     {\def#2{#1}}%
4494 }

```

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

```

4495 \newcommand{\glxstrunsrtdo}{\@glxstr@noidx@do}

```

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

```
4496 \newcommand*{\glsxtrgroupfield}{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 `\@glsxtr@doglossary` is being constructed rather than in the handler.

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

```
4497 \newcommand*{\@glsxtr@checkgroup}[1]{%
4498   \def\@glsxtr@groupheading{}%
4499   \key@ifundefined{glossentry}{group}%
4500   {%
4501     \letcs{\@gls@sort}{glo\@glsdetoklabel{#1}@sort}%
4502     \expandafter\glo@grabfirst\@gls@sort{}\@nil
4503   }%
4504   {%
4505     \protected@edef\@glo@thislettergrp{%
4506       \csuse{glo\@glsdetoklabel{#1}\@glsxtrgroupfield}}%
4507     }%
4508     \ifdefequal{\@glo@thislettergrp}{\@gls@currentlettergroup}%
4509     {}%
4510     {%
4511       \ifdefempty{\@gls@currentlettergroup}{}%
4512       {\def\@glsxtr@groupheading{\gls@groupskip}}%
4513       \eappto\@glsxtr@groupheading{%
4514         \noexpand\gls@groupheading{\expandonce\@glo@thislettergrp}%
4515       }%
4516     }%
4517     \let\@gls@currentlettergroup\@glo@thislettergrp
4518 }
```

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

```
4519 \newcommand{\@glsxtr@noidx@do}[1]{%
4520   \ifglssentryexists{#1}%
4521   {%
4522     \global\letcs{\@gls@loclist}{glo\@glsdetoklabel{#1}@loclist}%
4523     \global\letcs{\@gls@location}{glo\@glsdetoklabel{#1}@location}%
4524     \ifglshasparent{#1}%
4525     {%
4526       \gls@level=\csuse{glo\@glsdetoklabel{#1}@level}\relax
4527       \ifdefvoid{\@gls@location}%

```

```

4528     {%
4529         \ifdefvoid{\@gls@loclist}%
4530     {%
4531         \subglossentry{\gls@level}{#1}{}%
4532     }%
4533     {%
4534         \subglossentry{\gls@level}{#1}%
4535     {%
4536         \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4537     }%
4538     }%
4539     }%
4540     {%
4541         \subglossentry{\gls@level}{#1}{\glossaryentrynumbers{\@gls@location}}%
4542     }%
4543     }%
4544     {%

4545     \ifdefvoid{\@gls@location}%
4546     {%
4547         \ifdefvoid{\@gls@loclist}
4548         {%
4549             \glossentry{#1}{}%
4550         }%
4551         {%
4552             \glossentry{#1}%
4553             {%
4554                 \glossaryentrynumbers{\glsnoidxloclist{\@gls@loclist}}%
4555             }%
4556         }%
4557     }%
4558     {%
4559         \glossentry{#1}%
4560         {%
4561             \glossaryentrynumbers{\@gls@location}%
4562         }%
4563     }%
4564     }%
4565     }%
4566     {}%
4567 }

```

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

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

`\glsxtrnewgls`

```

4568 \newcount\@glsxtrnewgls@inner

```

(The default options supplied in *<options>* below could possibly be used to form the inner

control sequence name to help make it unique, but it might feasibly contain the value where the value might contain commands.)

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

```

4569 \newcommand*{\@glsxtrnewgls}[4]{%
4570   \ifdef{#3}%
4571   {%
4572     \PackageError{glossaries-extra}{Command \string#3\space already
4573 defined}{}%
4574   }%
4575   {%
4576     \ifcsdef{@#4like@#2}%
4577     {%
4578       \advance\@glsxtrnewgls@inner by \@ne
4579       \def\@glsxtrnewgls@innercsname{@#4like\number\@glsxtrnewgls@inner @#2}%
4580     }%
4581     {\def\@glsxtrnewgls@innercsname{@#4like@#2}}%
4582     \expandafter\newrobustcmd\expandafter*\expandafter
4583       #3\expandafter{\expandafter\@gls@hyp@opt\csname\@glsxtrnewgls@innercsname\endcsname}%
4584     \ifstrempy{#1}%
4585     {%
4586       \expandafter\newcommand\expandafter*\csname\@glsxtrnewgls@innercsname\endcsname[2][ ]{%
4587         \new@ifnextchar[%
4588           {\csname @#4@\endcsname{##1}{#2##2}}%
4589           {\csname @#4@\endcsname{##1}{#2##2} [ ]}%
4590         }%
4591       }%
4592     }%
4593     \expandafter\newcommand\expandafter*\csname\@glsxtrnewgls@innercsname\endcsname[2][ ]{%
4594       \new@ifnextchar[%
4595         {\csname @#4@\endcsname{#1,##1}{#2##2}}%
4596         {\csname @#4@\endcsname{#1,##1}{#2##2} [ ]}%
4597       }%
4598     }%
4599   }%
4600 }
```

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

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

```

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

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

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

`\lsxtrnewGLSlike` Provide a way to conveniently define commands that behave like `\GLS`, `\GLSpl` with a label prefix. The first argument prepends to the options and the second argument is the prefix.

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

`\glsxtrnewrgls` As `\glsxtrnewgls` but for `\rgls`.

```
4614 \newrobustcmd*{\glsxtrnewrgls}[3] [] {%
4615   \@glsxtrnewgls{#1}{#2}{#3}{rgls}%
4616 }
```

`\sxtrnewrglslike` As `\glsxtrnewglslike` but for `\rgls` etc.

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

`\sxtrnewrGLSlike` As `\glsxtrnewGLSlike` but for `\rGLS` etc.

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

Provide easy access to record count fields.

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

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

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

```
4632 \newcommand*{\GlsXtrRecordCount}[2] {%
```

```

4633 \ifcsdef{glo@\glxstrdetoklabel{#1}@recordcount.#2}%
4634 {\csname glo@\glxstrdetoklabel{#1}@recordcount.#2\endcsname}%
4635 {0}%
4636 }

```

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

```

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

```

**trdetoklocation**

```

4642 \newcommand*{\glxstrdetoklocation}[1]{#1}

```

**ablerecordcount**

```

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

```

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

```

4651 \newcommand*{\glxstrrecordtriggervalue}[1]{%
4652 \GlsXtrTotalRecordCount{#1}%
4653 }

```

**dCountAttribute**

```

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

```

**trifrecordtrigger**

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



```

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

```

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

```

4674 \newcommand*{\@glxtr@rglstrigger@record}[3]{%
4675   \edef\glslabel{\glsdetoklabel{#2}}%
4676   \let\@gls@link@label\glslabel
4677   \def\@glxtr@thevalue{%
4678     \def\@glxtr@theHvalue{\@glxtr@thevalue}%
4679     \def\@glsnumberformat{glstriggerrecordformat}%
4680     \edef\@gls@counter{\csname glo@\glslabel @counter\endcsname}%
4681     \edef\glstype{\csname glo@\glslabel @type\endcsname}%
4682     \def\@glxtr@thevalue{%
4683       \def\@glxtr@theHvalue{\@glxtr@thevalue}%
4684       \glxtrinitwrgloss
4685       \glslinkpresetkeys
4686       \setkeys{glslink}{#1}%
4687       \glslinkpostsetkeys
4688       \ifdefempty{\@glxtr@thevalue}%
4689       {%
4690         \@gls@saveentrycounter
4691       }%
4692       {%
4693         \let\thegl Sentrycounter\@glxtr@thevalue
4694         \def\theHgl Sentrycounter{\@glxtr@theHvalue}%
4695       }%
4696       \ifglxtrinitwrglossbefore
4697         \@do@wrglossary{#2}%
4698       \fi
4699       #3%
4700       \ifglxtrinitwrglossbefore
4701       \else
4702         \@do@wrglossary{#2}%
4703       \fi
4704       \ifKV@glslink@local
4705         \glslocalunset{#2}%
4706       \else
4707         \glsunset{#2}%
4708       \fi

```

```

4709 }

gerrecordformat Typically won't be used as it should be recognised as a special type of ignored location by
bib2gls.
4710 \newcommand*{\glstriggerrecordformat}[1]{

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

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

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

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

\@rglspl
4725 \newcommand*{\@rglspl}[2][{}]{%
4726   \new@ifnextchar[{\@rglspl@{#1}{#2}}{\@rglspl@{#1}{#2}[]}%
4727 }

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

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

```

```

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

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

\rGlspl
4750 \newrobustcmd*{\rGlspl}{\@glshyp@opt\rGlspl}

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

\@rGlspl@
4754 \def\@rGlspl@#1#2[#3]{%
4755 \glxtrifrecordtrigger{#2}%
4756 {%
4757 \@glxtr@rglstrigger@record{#1}{#2}{\rGlsplformat{#2}{#3}}%
4758 }%
4759 {%
4760 \@Glspl@{#1}{#2}[#3]%
4761 }%
4762 }%

\rGLS
4763 \newrobustcmd*{\rGLS}{\@glshyp@opt\rGLS}

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

\@rGLS@
4767 \def\@rGLS@#1#2[#3]{%
4768 \glxtrifrecordtrigger{#2}%
4769 {%
4770 \@glxtr@rglstrigger@record{#1}{#2}{\rGLSformat{#2}{#3}}%

```

```

4771 }%
4772 {%
4773   \@GLS@{#1}{#2}[#3]%
4774 }%
4775 }%

```

\rGLSpl

```

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

```

\@rGLSpl

```

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

```

\@rGLSpl@

```

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

```

\rglsformat

```

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

```

\rglsplformat

```

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

```

\rGlsformat

```

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

```

\rGlsplformat

```

4804 \newcommand*{\rGlsplformat}[2]{%

```

```

4805 \glsifregular{#1}
4806 {\Glsentryfirstplural{#1}}%
4807 {\ifglshaslong{#1}{\Glsentrylongplural{#1}}{\Glsentryfirstplural{#1}}}%#2%
4808 }

```

\rGLSformat

```

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

```

\rGLSplformat

```

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

```

## 1.4 Link Counting

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

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

```

4815 \newcommand{\@glsxtr@do@inc@linkcount}{%
  Does this entry have the linkcount attribute set?
4816 \glsifattribute{\glslabel}{linkcount}{true}%
4817 {%
  Does the counter exist?
4818 \ifcsdef{c@glxtr@linkcount@\glslabel}{}%
4819 {%
  Counter doesn’t exist, so define it.
4820 \newcounter{glxtr@linkcount@\glslabel}%
  If linkcountmaster is set, add to counter reset.
4821 \glshasattribute{\glslabel}{linkcountmaster}%
4822 {%
  Need to ensure values are fully expanded.
4823 \begingroup
4824 \edef\x{\endgroup\noexpand\@addtoreset{glxtr@linkcount@\glslabel}%
4825 {\glsggetattribute{\glslabel}{linkcountmaster}}}%
4826 \x
4827 }%

```

```

4828     {}%
4829   }%

  Increment counter:
4830   \glxstrinclinkcounter{glxstr@linkcount@glslabel}%
4831 }%
4832 {}%
4833 }

rlinkcounter  May be redefined to use \refstepcounter if required.
4834 \newcommand*{\glxstrinclinkcounter}[1]{\stepcounter{#1}}

linkCounterValue  Expands to the associated link counter register or 0 if not defined.
4835 \newcommand*{\GlsXtrLinkCounterValue}[1]{%
4836   \ifcsundef{c@glxstr@linkcount@#1}{0}{\csname c@glxstr@linkcount@#1\endcsname}%
4837 }

rTheLinkCounter  Expands to the display value of the associated link counter or 0 if not defined.
4838 \newcommand*{\GlsXtrTheLinkCounter}[1]{%
4839   \ifcsundef{theglsxtr@linkcount@#1}{0}%
4840   {\csname theglxstr@linkcount@#1\endcsname}%
4841 }

fLinkCounterDef  Tests if the counter has been defined
4842 \newcommand*{\GlsXtrIfLinkCounterDef}[3]{%
4843   \ifcsundef{theglsxtr@linkcount@#1}{#3}{#2}%
4844 }

LinkCounterName  Expands to the associated link counter name. (No check for existence.)
4845 \newcommand*{\GlsXtrLinkCounterName}[1]{glxstr@linkcount@#1}

ableLinkCounting  \GlsXtrEnableLinkCounting[master counter]{categories}

  Enable link counting for the given categories.
4846 \newcommand*{\GlsXtrEnableLinkCounting}[2][]{%
4847   \let\glxstr@inc@linkcount\@glxstr@do@inc@linkcount
4848   \@for\@glxstr@label:=#2\do
4849   {%
4850     \glsssetcategoryattribute{\@glxstr@label}{linkcount}{true}%
4851     \ifstrempy{#1}{}%
4852     {%
4853       \ifcsundef{c@#1}%
4854       {\@nocounterr{#1}}%
4855       {\glsssetcategoryattribute{\@glxstr@label}{linkcountmaster}{#1}}%
4856     }%
4857   }%
4858 }
4859 \@onlypreamble\GlsXtrEnableLinkCounting

```

## 1.5 Integration with glossaries-accsupp

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

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

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

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

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

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

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

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

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

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

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

```
4883 \newcommand*{\glsaccesstext}[1]{%
4884 \glstextaccessdisplay
4885 {%
4886 \glstrytext{#1}%
4887 }%
4888 {#1}%
4889 }
```

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

```

4890 \newcommand*{\Glsaccesstext}[1]{%
4891   \glstextaccessdisplay
4892   {%
4893     \Glsentrytext{#1}%
4894   }%
4895   {#1}%
4896 }
```

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

```

4897 \newcommand*{\GLSaccesstext}[1]{%
4898   \glstextaccessdisplay
4899   {%
4900     \mfirstucMakeUppercase{\Glsentrytext{#1}}%
4901   }%
4902   {#1}%
4903 }
```

`glsaccessplural` Display the plural value (no link and no check for existence).

```

4904 \newcommand*{\glsaccessplural}[1]{%
4905   \glspluralaccessdisplay
4906   {%
4907     \Glsentryplural{#1}%
4908   }%
4909   {#1}%
4910 }
```

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

```

4911 \newcommand*{\GLSaccessplural}[1]{%
4912   \glspluralaccessdisplay
4913   {%
4914     \Glsentryplural{#1}%
4915   }%
4916   {#1}%
4917 }
```

`GLSaccessplural` Display the plural value (no link and no check for existence) converted to upper case.

```

4918 \newcommand*{\GLSaccessplural}[1]{%
4919   \glspluralaccessdisplay
4920   {%
4921     \mfirstucMakeUppercase{\Glsentryplural{#1}}%
4922   }%
4923   {#1}%
4924 }
```

`\glsaccessfirst` Display the first value (no link and no check for existence).



```

4925 \newcommand*{\glsaccessfirst}[1]{%
4926   \glsfirstaccessdisplay
4927   {%
4928     \glsentryfirst{#1}%
4929   }%
4930   {#1}%
4931 }

```

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

```

4932 \newcommand*{\Glsaccessfirst}[1]{%
4933   \glsfirstaccessdisplay
4934   {%
4935     \Glsentryfirst{#1}%
4936   }%
4937   {#1}%
4938 }

```

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

```

4939 \newcommand*{\GLSaccessfirst}[1]{%
4940   \glsfirstaccessdisplay
4941   {%
4942     \mfirstucMakeUppercase{\glsentryfirst{#1}}%
4943   }%
4944   {#1}%
4945 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence).

```

4946 \newcommand*{\glsaccessfirstplural}[1]{%
4947   \glsfirstpluralaccessdisplay
4948   {%
4949     \glsentryfirstplural{#1}%
4950   }%
4951   {#1}%
4952 }

```

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

```

4953 \newcommand*{\Glsaccessfirstplural}[1]{%
4954   \glsfirstpluralaccessdisplay
4955   {%
4956     \Glsentryfirstplural{#1}%
4957   }%
4958   {#1}%
4959 }

```

`cessfirstplural` Display the firstplural value (no link and no check for existence) converted to upper case.

```

4960 \newcommand*{\GLSaccessfirstplural}[1]{%

```

```

4961 \glsfirstpluralaccessdisplay
4962 {%
4963     \mfirstucMakeUppercase{\glsentryfirstplural{#1}}%
4964 }%
4965 {#1}%
4966 }

```

**glsaccesssymbol** Display the symbol value (no link and no check for existence).

```

4967 \newcommand*{\glsaccesssymbol}[1]{%
4968     \glssymbolaccessdisplay
4969     {%
4970         \glsentrysymbol{#1}%
4971     }%
4972     {#1}%
4973 }

```

**Glsaccesssymbol** Display the symbol value (no link and no check for existence) with the first letter converted to upper case.

```

4974 \newcommand*{\Glsaccesssymbol}[1]{%
4975     \glssymbolaccessdisplay
4976     {%
4977         \Glsentrysymbol{#1}%
4978     }%
4979     {#1}%
4980 }

```

**GLSaccesssymbol** Display the symbol value (no link and no check for existence) converted to upper case.

```

4981 \newcommand*{\GLSaccesssymbol}[1]{%
4982     \glssymbolaccessdisplay
4983     {%
4984         \mfirstucMakeUppercase{\glsentrysymbol{#1}}%
4985     }%
4986     {#1}%
4987 }

```

**esssymbolplural** Display the symbolplural value (no link and no check for existence).

```

4988 \newcommand*{\glsaccesssymbolplural}[1]{%
4989     \glssymbolpluralaccessdisplay
4990     {%
4991         \glsentrysymbolplural{#1}%
4992     }%
4993     {#1}%
4994 }

```

**esssymbolplural** Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.

```

4995 \newcommand*{\Glsaccesssymbolplural}[1]{%
4996     \glssymbolpluralaccessdisplay

```

```

4997   {%
4998       \Glsentrysymbolplural{#1}%
4999   }%
5000   {#1}%
5001 }

```

**esssymbolplural** Display the symbolplural value (no link and no check for existence) converted to upper case.

```

5002 \newcommand*{\GLSaccesssymbolplural}[1]{%
5003     \glssymbolpluralaccessdisplay
5004     {%
5005         \mfirstucMakeUppercase{\glentrysymbolplural{#1}}%
5006     }%
5007     {#1}%
5008 }

```

**\glsaccessdesc** Display the desc value (no link and no check for existence).

```

5009 \newcommand*{\glsaccessdesc}[1]{%
5010     \glsdescriptionaccessdisplay
5011     {%
5012         \glentrydesc{#1}%
5013     }%
5014     {#1}%
5015 }

```

**\Glsaccessdesc** Display the desc value (no link and no check for existence) with the first letter converted to upper case.

```

5016 \newcommand*{\Glsaccessdesc}[1]{%
5017     \glsdescriptionaccessdisplay
5018     {%
5019         \Glsentrydesc{#1}%
5020     }%
5021     {#1}%
5022 }

```

**\GLSaccessdesc** Display the desc value (no link and no check for existence) converted to upper case.

```

5023 \newcommand*{\GLSaccessdesc}[1]{%
5024     \glsdescriptionaccessdisplay
5025     {%
5026         \mfirstucMakeUppercase{\glentrydesc{#1}}%
5027     }%
5028     {#1}%
5029 }

```

**ccessdescplural** Display the descplural value (no link and no check for existence).

```

5030 \newcommand*{\glsaccessdescplural}[1]{%
5031     \glsdescriptionpluralaccessdisplay
5032     {%
5033         \glentrydescplural{#1}%

```

```

5034     }%
5035     {#1}%
5036 }

```

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

```

5037 \newcommand*{\Glsaccessdescplural}[1]{%
5038   \glsdescriptionpluralaccessdisplay
5039   {%
5040     \Glsentrydescplural{#1}%
5041   }%
5042   {#1}%
5043 }

```

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

```

5044 \newcommand*{\GLSaccessdescplural}[1]{%
5045   \glsdescriptionpluralaccessdisplay
5046   {%
5047     \mfirstucMakeUppercase{\glsentrydescplural{#1}}%
5048   }%
5049   {#1}%
5050 }

```

`\glsaccessshort` Display the short form (no link and no check for existence).

```

5051 \newcommand*{\glsaccessshort}[1]{%
5052   \glsshortaccessdisplay
5053   {%
5054     \glsentryshort{#1}%
5055   }%
5056   {#1}%
5057 }

```

`\Glsaccessshort` Display the short form with first letter converted to uppercase (no link and no check for existence).

```

5058 \newcommand*{\Glsaccessshort}[1]{%
5059   \glsshortaccessdisplay
5060   {%
5061     \Glsentryshort{#1}%
5062   }%
5063   {#1}%
5064 }

```

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

```

5065 \newcommand*{\GLSAccessshort}[1]{%
5066   \glsshortaccessdisplay
5067   {%
5068     \mfirstucMakeUppercase{\glsentryshort{#1}}%
5069   }%

```

```

5070     {#1}%
5071 }

```

**lsaccessshortpl** Display the short plural form (no link and no check for existence).

```

5072 \newcommand*{\lsaccessshortpl}[1]{%
5073   \glshortpluralaccessdisplay
5074   {%
5075     \glentryshortpl{#1}%
5076   }%
5077   {#1}%
5078 }

```

**Lsaccessshortpl** Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```

5079 \newcommand*{\Lsaccessshortpl}[1]{%
5080   \glshortpluralaccessdisplay
5081   {%
5082     \Glentryshortpl{#1}%
5083   }%
5084   {#1}%
5085 }

```

**LSaccessshortpl** Display the shortplural value (no link and no check for existence) converted to upper case.

```

5086 \newcommand*{\LSaccessshortpl}[1]{%
5087   \glshortpluralaccessdisplay
5088   {%
5089     \mfirstucMakeUppercase{\glentryshortpl{#1}}%
5090   }%
5091   {#1}%
5092 }

```

**\glsaccesslong** Display the long form (no link and no check for existence).

```

5093 \newcommand*{\glsaccesslong}[1]{%
5094   \glslongaccessdisplay{\glentrylong{#1}}{#1}%
5095 }

```

**\Glsaccesslong** Display the long form (no link and no check for existence).

```

5096
5097 \newcommand*{\Glsaccesslong}[1]{%
5098   \glslongaccessdisplay{\Glentrylong{#1}}{#1}%
5099 }

```

**\GLSaccesslong** Display the long value (no link and no check for existence) converted to upper case.

```

5100 \newcommand*{\GLSaccesslong}[1]{%
5101   \glslongaccessdisplay
5102   {%
5103     \mfirstucMakeUppercase{\glentrylong{#1}}%
5104   }%

```

```

5105     {#1}%
5106 }

```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5107 \newcommand*{\glsaccesslongpl}[1]{%
5108     \glslongpluralaccessdisplay{\glsentrylongpl{#1}}{#1}%
5109 }

```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```

5110
5111 \newcommand*{\Glsaccesslongpl}[1]{%
5112     \glslongpluralaccessdisplay{\Glsentrylongpl{#1}}{#1}%
5113 }

```

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

```

5114 \newcommand*{\GLSaccesslongpl}[1]{%
5115     \glslongpluralaccessdisplay
5116     {%
5117         \mfirstucMakeUppercase{\glsentrylongpl{#1}}%
5118     }%
5119     {#1}%
5120 }

```

End of if part

```

5121 }
5122 {

```

No accessibility support. Just define these commands to do `\glsentry<xxx>`

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

```

5123 \newcommand*{\glsaccessname}[1]{\glsentryname{#1}}

```

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

```

5124 \newcommand*{\Glsaccessname}[1]{\Glsentryname{#1}}

```

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

```

5125 \newcommand*{\GLSaccessname}[1]{%
5126     \protect\mfirstucMakeUppercase{\glsentryname{#1}}}

```

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

```

5127 \newcommand*{\glsaccesstext}[1]{\glsentrytext{#1}}

```

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

```

5128 \newcommand*{\Glsaccesstext}[1]{\Glsentrytext{#1}}

```

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

5129 `\newcommand*{\GLSaccessstext}[1]{%`  
5130 `\protect\mfirstucMakeUppercase{\glstentrytext{#1}}}`

`glsaccessplural` Display the plural value (no link and no check for existence).

5131 `\newcommand*{\glsaccessplural}[1]{\glstentryplural{#1}}`

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

5132 `\newcommand*{\Glsaccessplural}[1]{\Glstentryplural{#1}}`

`GLSaccessplural` Display the plural value (no link and no check for existence). converted to upper case.

5133 `\newcommand*{\GLSaccessplural}[1]{%`  
5134 `\protect\mfirstucMakeUppercase{\glstentryplural{#1}}}`

`\glsaccessfirst` Display the first value (no link and no check for existence).

5135 `\newcommand*{\glsaccessfirst}[1]{\glstentryfirst{#1}}`

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

5136 `\newcommand*{\Glsaccessfirst}[1]{\Glstentryfirst{#1}}`

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

5137 `\newcommand*{\GLSaccessfirst}[1]{%`  
5138 `\protect\mfirstucMakeUppercase{\glstentryfirst{#1}}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence).

5139 `\newcommand*{\glsaccessfirstplural}[1]{\glstentryfirstplural{#1}}`

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

5140 `\newcommand*{\Glsaccessfirstplural}[1]{\Glstentryfirstplural{#1}}`

`cessfirstplural` Display the firstplural value (no link and no check for existence). converted to upper case.

5141 `\newcommand*{\GLSaccessfirstplural}[1]{%`  
5142 `\protect\mfirstucMakeUppercase{\glstentryfirstplural{#1}}}`

`glsaccesssymbol` Display the symbol value (no link and no check for existence).

5143 `\newcommand*{\glsaccesssymbol}[1]{\glstentrysymbol{#1}}`

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

5144 `\newcommand*{\Glsaccesssymbol}[1]{\Glstentrysymbol{#1}}`

`GLSaccesssymbol` Display the symbol value (no link and no check for existence). converted to upper case.

5145 `\newcommand*{\GLSaccesssymbol}[1]{%`  
5146 `\protect\mfirstucMakeUppercase{\glstentrysymbol{#1}}}`

esssymbolplural Display the symbolplural value (no link and no check for existence).  
5147 \newcommand\*{\glsaccesssymbolplural}[1]{\glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence) with the first letter converted to upper case.  
5148 \newcommand\*{\Glsaccesssymbolplural}[1]{\Glsentrysymbolplural{#1}}

esssymbolplural Display the symbolplural value (no link and no check for existence). converted to upper case.  
5149 \newcommand\*{\GLSaccesssymbolplural}[1]{%  
5150 \protect\mfirstucMakeUppercase{\glsentrysymbolplural{#1}}}

\glsaccessdesc Display the desc value (no link and no check for existence).  
5151 \newcommand\*{\glsaccessdesc}[1]{\glsentrydesc{#1}}

\Glsaccessdesc Display the desc value (no link and no check for existence) with the first letter converted to upper case.  
5152 \newcommand\*{\Glsaccessdesc}[1]{\Glsentrydesc{#1}}

\GLSaccessdesc Display the desc value (no link and no check for existence). converted to upper case.  
5153 \newcommand\*{\GLSaccessdesc}[1]{%  
5154 \protect\mfirstucMakeUppercase{\glsentrydesc{#1}}}

ccessdescplural Display the descplural value (no link and no check for existence).  
5155 \newcommand\*{\glsaccessdescplural}[1]{\glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence) with the first letter converted to upper case.  
5156 \newcommand\*{\Glsaccessdescplural}[1]{\Glsentrydescplural{#1}}

ccessdescplural Display the descplural value (no link and no check for existence). converted to upper case.  
5157 \newcommand\*{\GLSaccessdescplural}[1]{%  
5158 \protect\mfirstucMakeUppercase{\glsentrydescplural{#1}}}

\glsaccessshort Display the short form (no link and no check for existence).  
5159 \newcommand\*{\glsaccessshort}[1]{\glsentryshort{#1}}

\Glsaccessshort Display the short form with first letter converted to uppercase (no link and no check for existence).  
5160 \newcommand\*{\Glsaccessshort}[1]{\Glsentryshort{#1}}

\GLSaccessshort Display the short value (no link and no check for existence). converted to upper case.  
5161 \newcommand\*{\GLSaccessshort}[1]{%  
5162 \protect\mfirstucMakeUppercase{\glsentryshort{#1}}}

lsaccessshortpl Display the short plural form (no link and no check for existence).  
5163 \newcommand\*{\glsaccessshortpl}[1]{\glsentryshortpl{#1}}



`\lsaccessshortpl` Display the short plural form with first letter converted to uppercase (no link and no check for existence).

```
5164 \newcommand*{\Glsaccessshortpl}[1]{\Glsentryshortpl{#1}}
```

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

```
5165 \newcommand*{\GLSaccessshortpl}[1]{%
5166 \protect\mfirstucMakeUppercase{\glentryshortpl{#1}}}
```

`\glsaccesslong` Display the long form (no link and no check for existence).

```
5167 \newcommand*{\glsaccesslong}[1]{\glentrylong{#1}}
```

`\Glsaccesslong` Display the long form (no link and no check for existence).

```
5168 \newcommand*{\Glsaccesslong}[1]{\Glsentrylong{#1}}
```

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

```
5169 \newcommand*{\GLSaccesslong}[1]{%
5170 \protect\mfirstucMakeUppercase{\glentrylong{#1}}}
```

`\glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5171 \newcommand*{\glsaccesslongpl}[1]{\glentrylongpl{#1}}
```

`\Glsaccesslongpl` Display the long plural form (no link and no check for existence).

```
5172 \newcommand*{\Glsaccesslongpl}[1]{\Glsentrylongpl{#1}}
```

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

```
5173 \newcommand*{\GLSaccesslongpl}[1]{%
5174 \protect\mfirstucMakeUppercase{\glentrylongpl{#1}}}
```

End of else part

```
5175 }
```

## 1.6 Categories

`\glscategory` Add a new storage key that can be used to indicate a category. The default category is general.

```
5176 \glsaddstoragekey{category}{general}{\glscategory}
```

`\glsifcategory` Convenient shortcut to determine if an entry has the given category.

```
5177 \newcommand{\glsifcategory}[4]{%
5178 \ifglsfieldeq{#1}{category}{#2}{#3}{#4}%
5179 }
```

Categories can have attributes.

categoryattribute

`\glsssetcategoryattribute{<category>}{<attribute-label>}{<value>}`

Set (or override if already set) an attribute for the given category.

```
5180 \newcommand*{\glsssetcategoryattribute}[3]{%
5181   \csdef{@glssxtr@categoryattr@@#1@#2}{#3}%
5182 }
```

categoryattribute

`\glssgetcategoryattribute{<category>}{<attribute-label>}`

Get the value of the given attribute for the given category. Does nothing if the attribute isn't defined.

```
5183 \newcommand*{\glssgetcategoryattribute}[2]{%
5184   \csuse{@glssxtr@categoryattr@@#1@#2}%
5185 }
```

categoryattribute

`\glsshascategoryattribute{<category>}{<attribute-label>}{<true>}{<false>}`

Tests if the category has the given attribute set.

```
5186 \newcommand*{\glsshascategoryattribute}[4]{%
5187   \ifcsvoid{@glssxtr@categoryattr@@#1@#2}{#4}{#3}%
5188 }
```

\glsssetattribute

`\glsssetattribute{<entry label>}{<attribute-label>}{<value>}`

Short cut where the category label is obtained from the entry information.

```
5189 \newcommand*{\glsssetattribute}[3]{%
5190   \glsssetcategoryattribute{\glssccategory{#1}}{#2}{#3}%
5191 }
```

\glssgetattribute

`\glssgetattribute{<entry label>}{<attribute-label>}`

Short cut where the category label is obtained from the entry information.

```
5192 \newcommand*{\glssgetattribute}[2]{%
5193   \glssgetcategoryattribute{\glssccategory{#1}}{#2}%
5194 }
```

`\glshasattribute`

`\glshasattribute{<entry label>}{<attribute-label>}{<true>}{<false>}`

Short cut to test if the given attribute has been set where the category label is obtained from the entry information.

```
5195 \newcommand*{\glshasattribute}[4]{%
5196   \ifglentryexists{#1}%
5197   {\glshascategoryattribute{\glscategory{#1}}{#2}{#3}{#4}}%
5198   {#4}%
5199 }
```

`categoryattribute`

`\glcifcategoryattribute{<category>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

True if category has the attribute with the given value.

```
5200 \newcommand{\glcifcategoryattribute}[5]{%
5201   \ifcsundef{@glxtr@categoryattr@#1@#2}%
5202   {#5}%
5203   {\ifcsstring{@glxtr@categoryattr@#1@#2}{#3}{#4}{#5}}%
5204 }
```

`\glcifattribute`

`\glcifattribute{<entry label>}{<attribute-label>}{<value>}{<true part>}{<false part>}`

Short cut to determine if the given entry has a category with the given attribute set.

```
5205 \newcommand{\glcifattribute}[5]{%
5206   \ifglentryexists{#1}%
5207   {\glcifcategoryattribute{\glscategory{#1}}{#2}{#3}{#4}{#5}}%
5208   {#5}%
5209 }
```

Set attributes for the default general category:

```
5210 \glsetcategoryattribute{general}{regular}{true}
```

Acronyms are regular by default, since they're typically just treated like normal words.

```
5211 \glsetcategoryattribute{acronym}{regular}{true}
```

`regularcategory`

Convenient shortcut to create add the regular attribute.

```
5212 \newcommand*{\glsetregularcategory}[1]{%
5213   \glsetcategoryattribute{#1}{regular}{true}%
5214 }
```

`fregularcategory` `\glsifregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to true.

```
5215 \newcommand{\glsifregularcategory}[3]{%
5216   \glsifcategoryattribute{#1}{regular}{true}{#2}{#3}%
5217 }
```

`tregularcategory` `\glsifnotregularcategory{<category>}{<true part>}{<>false part>}`

Short cut to determine if a category has the regular attribute explicitly set to false.

```
5218 \newcommand{\glsifnotregularcategory}[3]{%
5219   \glsifcategoryattribute{#1}{regular}{false}{#2}{#3}%
5220 }
```

`\glsifregular` `\glsifregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to true.

```
5221 \newcommand{\glsifregular}[3]{%
5222   \glsifregularcategory{\glscategory{#1}}{#2}{#3}%
5223 }
```

`\glsifnotregular` `\glsifnotregular{<entry label>}{<true part>}{<>false part>}`

Short cut to determine if an entry has a regular attribute set to false.

```
5224 \newcommand{\glsifnotregular}[3]{%
5225   \glsifnotregularcategory{\glscategory{#1}}{#2}{#3}%
5226 }
```

`oreachincategory` `\glsforeachincategory[<glossary labels>]{<category-label>}{<glossary-cs>}{<label-cs>}{<body>}`

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category matches *<category-label>*. The control sequences *<glossary-cs>* and *<label-cs>* may be used in *<body>* to access the glossary label and entry label for the current iteration.

```
5227 \newcommand{\glsforeachincategory}[5][\@glo@types]{%
```

```

5228 \forallglossaries[#1]{#3}%
5229 {%
5230     \forallsentries[#3]{#4}%
5231     {%
5232         \glsifcategory{#4}{#2}{#5}{}%
5233     }%
5234 }%
5235 }

```

achwithattribute

```

\glsforeachwithattribute[<glossary labels>]{<attribute-label>}
{<attribute-value>}{<glossary-cs>}{<label-cs>}{<body>}

```

Iterates through all entries in all the glossaries (or just those listed in *<glossary labels>*) and does *<body>* if the category attribute *<attribute-label>* matches *<attribute-value>*. The control sequences *<glossary-cs>* and *<label-cs>* may be used in *<body>* to access the glossary label and entry label for the current iteration.

```

5236 \newcommand{\glsforeachwithattribute}[6][\@glo@types]{%
5237     \forallglossaries[#1]{#4}%
5238     {%
5239         \forallsentries[#4]{#5}%
5240         {%
5241             \glsifattribute{#5}{#2}{#3}{#6}{}%
5242         }%
5243     }%
5244 }

```

If `\newterm` has been defined, redefine it so that it automatically sets the category label to index and add `\glstrpostdescription`.

```

5245 \ifdef\newterm
5246 {%

```

\newterm

```

5247     \renewcommand*{\newterm}[2][ ]{%
5248         \newglossaryentry{#2}%
5249         {type=index,category=index,name={#2},%
5250          description={\glstrpostdescription\nopostdesc},#1}%
5251     }

```

Indexed terms are regular by default.

```

5252     \glsssetcategoryattribute{index}{regular}{true}

```

trpostdescindex

```

5253     \newcommand*{\glstrpostdescindex}{}

5254 }
5255 {}

```

If the symbols package option was used, define a similar command for symbols, but set the default sort to the label rather than the name as the symbols will typically contain commands that will confuse makeindex and xindy.

```
5256 \ifdef\printsymbols
5257 {%
```

`glsxtrnewsymbol` Unlike `\newterm`, this has a separate argument for the label (since the symbol will likely contain commands).

```
5258 \newcommand*{\glsxtrnewsymbol}[3] [] {%
5259 \newglossaryentry{#2}{name={#3},sort={#2},type=symbols,category=symbol,#1}%
5260 }
```

Symbols are regular by default.

```
5261 \glsssetcategoryattribute{symbol}{regular}{true}
```

`rpostdescsymbol`

```
5262 \newcommand*{\glsxtrpostdescsymbol}{}

5263 }
5264 {%
```

Similar for the numbers option.

```
5265 \ifdef\printnumbers
5266 {%
```

`glsxtrnewnumber`

```
5267 \ifdef\printnumbers
5268 \newcommand*{\glsxtrnewnumber}[3] [] {%
5269 \newglossaryentry{#2}{name={#3},sort={#2},type=numbers,category=number,#1}%
5270 }
```

Numbers are regular by default.

```
5271 \glsssetcategoryattribute{number}{regular}{true}
```

`rpostdescnumber`

```
5272 \newcommand*{\glsxtrpostdescnumber}{}

5273 }
5274 {%
```

`glsxtrsetcategory` Set the category for all listed labels. The first argument is the list of entry labels and the second argument is the category label.

```
5275 \newcommand*{\glsxtrsetcategory}[2] {%
5276 \@for\@glsxtr@label:=#1\do
5277 {%
5278 \glsfieldxddef{\@glsxtr@label}{category}{#2}%
5279 }%
5280 }
```

`categoryforall` Set the category for all entries in the listed glossaries. The first argument is the list of glossary labels and the second argument is the category label.

```

5281 \newcommand*{\glxtrsetcategoryforall}[2]{%
5282   \forallglossaries[#1]{\@glxtr@type}{%
5283     \forallglsentries[\@glxtr@type]{\@glxtr@label}%
5284     {%
5285       \glsfieldxdef{\@glxtr@label}{category}{#2}%
5286     }%
5287   }%
5288 }

```

`trfieldtitlecase` `\glxtrfieldtitlecase{<label>}{<field>}`

Apply title casing to the contents of the given field.

```

5289 \newcommand*{\glxtrfieldtitlecase}[2]{%
5290   \expandafter\glxtrfieldtitlecasesecs\expandafter
5291   {\csname glo@\glsdetoklabel{#1}@#2\endcsname}%
5292 }

```

`fieldtitlecasesecs` The command used by `\glxtrfieldtitlecase`. May be redefined to use a different command, for example, `\xcapitalisefmtwords`.

```

5293 \newcommand*{\glxtrfieldtitlecasesecs}[1]{\xcapitalisewords{#1}}

```

Provide a convenient way to modify glossary styles without having to define a new style just to convert the first letter of fields to upper case.

`\glossentrydesc` If the `glossdesc` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5294 \@ifpackageloaded{glossaries-accsupp}
5295 {
5296   \renewcommand*{\glossentrydesc}[1]{%
5297     \glsdoifexistsorwarn{#1}%
5298     {%
5299       \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossdescfont` attribute to determine the font applied.

```

5300   \glshasattribute{#1}{glossdescfont}%
5301   {%
5302     \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5303     \ifcsdef{\@glxtr@attrval}%
5304     {%
5305       \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5306     }%
5307     {%
5308       \GlossariesExtraWarning{Unknown control sequence name
5309       '\@glxtr@attrval' supplied in glossdescfont attribute

```

```

5310         for entry '#1'. Ignoring}%
5311         \let\@glxtr@glossdescfont\@firstofone
5312     }%
5313 }%
5314 {\let\@glxtr@glossdescfont\@firstofone}%
5315 \glsifattribute{#1}{glossdesc}{firstuc}%
5316 {%
5317     \@glxtr@glossdescfont{\Glsaccessdesc{#1}}%
5318 }%
5319 {%
5320     \glsifattribute{#1}{glossdesc}{title}%
5321     {%
5322         \@glxtr@do@titlecaps@warn
5323         \glsdescriptionaccessdisplay
5324         {%
5325             \@glxtr@glossdescfont{\glxtrfieldtitlecase{#1}{desc}}%
5326         }%
5327         {#1}%
5328     }%
5329     {%
5330         \@glxtr@glossdescfont{\glsaccessdesc{#1}}%
5331     }%
5332 }%
5333 }%
5334 }
5335 }
5336 {
5337 \renewcommand*{\glossentrydesc}[1]{%
5338     \glsdoifexistsorwarn{#1}%
5339     {%
5340         \glssetabbrvfmt{\glscategory{#1}}%
5341         \glsattribute{#1}{glossdescfont}%
5342         {%
5343             \edef\@glxtr@attrval{\glsgetattribute{#1}{glossdescfont}}%
5344             \ifcsdef{\@glxtr@attrval}%
5345             {%
5346                 \letcs{\@glxtr@glossdescfont}{\@glxtr@attrval}%
5347             }%
5348             {%
5349                 \GlossariesExtraWarning{Unknown control sequence name
5350                 '\@glxtr@attrval' supplied in glossdescfont attribute
5351                 for entry '#1'. Ignoring}%
5352                 \let\@glxtr@glossdescfont\@firstofone
5353             }%
5354         }%
5355         {\let\@glxtr@glossdescfont\@firstofone}%
5356         \glsifattribute{#1}{glossdesc}{firstuc}%
5357         {%
5358             \@glxtr@glossdescfont{\Glsentrydesc{#1}}%

```



```

5359     }%
5360     {%
5361         \glsifattribute{#1}{glossdesc}{title}%
5362         {%
5363             \@glsxtr@do@titlecaps@warn
5364             \@glsxtr@glossdescfont{\glsxtrfieldtitlecase{#1}{desc}}%
5365         }%
5366         {%
5367             \@glsxtr@glossdescfont{\glsentrydesc{#1}}%
5368         }%
5369     }%
5370 }%
5371 }
5372 }

```

`\glossentryname` If the `glossname` attribute is “firstuc” convert first letter to upper case. If the attribute is “title” use title case.

```

5373 \ifpackageloaded{glossaries-accsupp}
5374 {
5375     \renewcommand*{\glossentryname}[1]{%
5376         \@glsdoifexistsorwarn{#1}%
5377         {%
5378             \glsetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the `glossnamefont` attribute to determine the font applied.

```

5379     \glshasattribute{#1}{glossnamefont}%
5380     {%
5381         \edef\@glsxtr@attrval{\glsggetattribute{#1}{glossnamefont}}%
5382         \ifcsdef{\@glsxtr@attrval}%
5383         {%
5384             \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5385         }%
5386         {%
5387             \GlossariesExtraWarning{Unknown control sequence name
5388                 ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
5389                 for entry ‘#1’. Reverting to default \string\glsnamefont}%
5390             \let\@glsxtr@glossnamefont\glsnamefont
5391         }%
5392     }%
5393     {\let\@glsxtr@glossnamefont\glsnamefont}%
5394     \glsifattribute{#1}{glossname}{firstuc}%
5395     {%
5396         \glsnameaccessdisplay
5397         {%
5398             \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5399         }%
5400         {#1}%
5401     }%
5402     {%
5403         \glsifattribute{#1}{glossname}{title}%

```

```

5404      {%
5405      \@glstr@do@titlecaps@warn
5406      \glsnameaccessdisplay
5407      {%
5408      \@glstr@glossnamefont{\glstrfieldtitlecase{#1}{name}}%
5409      }%
5410      {#1}%
5411      }%
5412      {%
5413      \glsifattribute{#1}{glossname}{uc}%
5414      {%
5415      \glsnameaccessdisplay
5416      {%

```

Hide the label from the upper-casing command.

```

5417      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5418      \@glstr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5419      }%
5420      {#1}%
5421      }%
5422      {%
5423      \letcs{\glo@name}{glo@\glsdetoklabel{#1}@name}%
5424      \glsnameaccessdisplay
5425      {%
5426      \expandafter\@glstr@glossnamefont\expandafter{\glo@name}%
5427      }%
5428      {#1}%
5429      }%
5430      }%
5431      }%

```

Do post-name hook:

```

5432      \glstrpostnamehook{#1}%
5433      }%
5434  }
5435 }
5436 {
5437 \renewcommand*{\glossentryname}[1]{%
5438 \@glsdoifexistsorwarn{#1}%
5439 {%
5440 \glssetabbrvfmt{\glscategory{#1}}%
5441 \glsattribute{#1}{glossnamefont}%
5442 {%
5443 \edef\@glstr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5444 \ifcsdef{\@glstr@attrval}%
5445 {%
5446 \letcs{\@glstr@glossnamefont}{\@glstr@attrval}%
5447 }%
5448 {%
5449 \GlossariesExtraWarning{Unknown control sequence name

```

```

5450         ‘\@glsxtr@attrval’ supplied in glossnamefont attribute
5451         for entry ‘#1’. Reverting to default \string\glsnamefont}%
5452         \let\@glsxtr@glossnamefont\glsnamefont
5453     }%
5454 }%
5455 {\let\@glsxtr@glossnamefont\glsnamefont}%
5456 \glsifattribute{#1}{glossname}{firstuc}%
5457 {%
5458     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5459 }%
5460 {%
5461     \glsifattribute{#1}{glossname}{title}%
5462     {%
5463         \@glsxtr@do@titlecaps@warn
5464         \@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{name}}%
5465     }%
5466     {%
5467         \glsifattribute{#1}{glossname}{uc}%
5468         {%

```

Hide the label from the upper-casing command.

```

5469         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
5470         \@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}%
5471     }%
5472     {%

```

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.

```

5473         \letcs{\glo@name}{glo@glstdetoklabel{#1}@name}%
5474         \expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}%
5475     }%
5476 }%
5477 }%

```

Do post-name hook.

```

5478     \glsxtrpostnamehook{#1}%
5479 }%
5480 }
5481 }

```

**\Glossentryname** Redefine to set the abbreviation format and accessibility support.

```

5482 \@ifpackageloaded{glossaries-accsupp}
5483 {
5484     \renewcommand*{\Glossentryname}[1]{%
5485         \@glsdoifexistsorwarn{#1}%
5486         {%
5487             \glssetabbrvfmt{\glscategory{#1}}%

```

As from version 1.04, allow the glossnamefont attribute to determine the font applied.

```

5488         \glsifattribute{#1}{glossnamefont}%
5489         {%

```

```

5490     \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5491     \ifcsdef{\@glsxtr@attrval}%
5492     {%
5493         \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5494     }%
5495     {%
5496         \GlossariesExtraWarning{Unknown control sequence name
5497             '\@glsxtr@attrval' supplied in glossnamefont attribute
5498             for entry '#1'. Reverting to default \string\glsnamefont}%
5499         \let\@glsxtr@glossnamefont\glsnamefont
5500     }%
5501 }%
5502 {\let\@glsxtr@glossnamefont\glsnamefont}%
5503 \glsnameaccessdisplay
5504 {%
5505     \@glsxtr@glossnamefont{\Glsentryname{#1}}%
5506 }%
5507 {#1}%

```

Do post-name hook:

```

5508     \glsxtrpostnamehook{#1}%
5509 }%
5510 }
5511 }
5512 {
5513     \renewcommand*{\Glossentryname}[1]{%
5514         \@glsdoifexistsorwarn{#1}%
5515     {%
5516         \glssetabbrvfmt{\glscategory{#1}}%
5517         \glsattribute{#1}{glossnamefont}%
5518     {%
5519         \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5520         \ifcsdef{\@glsxtr@attrval}%
5521         {%
5522             \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5523         }%
5524         {%
5525             \GlossariesExtraWarning{Unknown control sequence name
5526                 '\@glsxtr@attrval' supplied in glossnamefont attribute
5527                 for entry '#1'. Reverting to default \string\glsnamefont}%
5528             \let\@glsxtr@glossnamefont\glsnamefont
5529         }%
5530     }%
5531     {\let\@glsxtr@glossnamefont\glsnamefont}%
5532     \@glsxtr@glossnamefont{\Glsentryname{#1}}%

```

Do post-name hook:

```

5533     \glsxtrpostnamehook{#1}%
5534 }%
5535 }

```

5536 }

Provide a convenient way to also index the entries using the standard `\index` mechanism. This may use different actual, encap and escape characters to those used for the glossaries.

`extrapostnamehook` Hook to append stuff after the name is displayed in the glossary. The argument is the entry's label.

```
5537 \newcommand*{\glxtrpostnamehook}[1]{%
5538   \let\@glsnumberformat\@glxtr@defaultnumberformat
5539   \glxtrdoautoindexname{#1}{indexname}%
```

Allow additional code regardless of category:

```
5540   \glsextrapostnamehook{#1}%
```

Allow categories to hook in here.

```
5541   \csuse{glxtrpostname\glscategory{#1}}%
5542 }
```

`trapostnamehook`

```
5543 \newcommand*{\glsextrapostnamehook}[1]{}%
```

`etaccessdisplay`

```
5544 \@ifpackageloaded{glossaries-accsupp}
5545 {
5546   \newcommand*{\glxtr@setaccessdisplay}[1]{%
5547     \ifcsdef{gls#1accessdisplay}%
5548       {\letcs\@glxtr@accessdisplay{gls#1accessdisplay}}%
5549     {%
```

This is essentially the reverse of `\@gls@fetchfield`, since the field supplied to `\glossentryname` has to be the internal label, but the `\gls<field>accessdisplay` commands use the key name.

```
5550     \edef\@gls@thisval{#1}%
5551     \@for\@gls@map:=\@gls@keymap\do{%
5552       \edef\@this@key{\expandafter\@secondoftwo\@gls@map}%
5553       \ifdefequal{\@this@key}{\@gls@thisval}%
5554       {%
5555         \edef\@gls@thisval{\expandafter\@firstoftwo\@gls@map}%
5556         \@endfortrue
5557       }%
5558     }%
5559   }%
5560   \ifcsdef{gls\@gls@thisval accessdisplay}%
5561   {\letcs\@glxtr@accessdisplay{gls\@gls@thisval accessdisplay}}%
5562   {\let\@glxtr@accessdisplay\@firstoftwo}%
5563 }%
5564 }
5565 }
5566 {%
5567   \newcommand*{\glxtr@setaccessdisplay}[1]{%
```

```

5568 \let\@glsxtr@accessdisplay\@firstoftwo}
5569 }

```

sentrynameother Provide a command that works like \glossentryname but accesses a different field (which must be supplied using its internal field label).

```

5570 \newrobustcmd*{\glossentrynameother}[2]{%
5571 \@glsdoifexistsorwarn{#1}%
5572 {%

```

Accessibility support:

```

5573 \glsxtr@setaccessdisplay{#2}%

```

Set the abbreviation format:

```

5574 \glssetabbrvfmt{\glscategory{#1}}%
5575 \glsattribute{#1}{glossnamefont}%
5576 {%
5577 \edef\@glsxtr@attrval{\glsgetattribute{#1}{glossnamefont}}%
5578 \ifcsdef{\@glsxtr@attrval}%
5579 {%
5580 \letcs{\@glsxtr@glossnamefont}{\@glsxtr@attrval}%
5581 }%
5582 {%
5583 \GlossariesExtraWarning{Unknown control sequence name
5584 '\@glsxtr@attrval' supplied in glossnamefont attribute
5585 for entry '#1'. Reverting to default \string\glsnamefont}%
5586 \let\@glsxtr@glossnamefont\glsnamefont
5587 }%
5588 }%
5589 {\let\@glsxtr@glossnamefont\glsnamefont}%
5590 \glsifattribute{#1}{glossname}{firstuc}%
5591 {%
5592 \@glsxtr@accessdisplay
5593 {\@glsxtr@glossnamefont{\@Gls@entry@field{#1}{#2}}}%
5594 {#1}%
5595 }%
5596 {%
5597 \glsifattribute{#1}{glossname}{title}%
5598 {%
5599 \@glsxtr@do@titlecaps@warn
5600 \@glsxtr@accessdisplay
5601 {\@glsxtr@glossnamefont{\glsxtrfieldtitlecase{#1}{#2}}}%
5602 {#1}%
5603 }%
5604 {%
5605 \glsifattribute{#1}{glossname}{uc}%
5606 {%
5607 \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
5608 \@glsxtr@accessdisplay
5609 {\@glsxtr@glossnamefont{\mfirstucMakeUppercase{\glo@name}}}%
5610 {#1}%

```

```

5611     }%
5612     {%
5613         \letcs{\glo@name}{\glo@\glsdetoklabel{#1}@#2}%
5614         \@glsxtr@accessdisplay
5615         {\expandafter\@glsxtr@glossnamefont\expandafter{\glo@name}}%
5616         {#1}%
5617     }%
5618 }%
5619 }%
    Do post-name hook.
5620     \glsxtrpostnamehook{#1}%
5621 }%
5622 }

```

`format@override` Determines if the format key should override the indexing attribute value.

```

5623 \newif\if@glsxtr@format@override
5624 \@glsxtr@format@overridefalse

```

If overriding is enabled, the `\glshypernumber` command will have to be redefined in the index to use `\hyperpage` instead.

`xFormatOverride`

```

5625 \@ifpackageloaded{hyperref}
5626 {
    If hyperref's hyperindex option is on, then hyperref will automatically add \hyperpage, so
    don't add it.
5627     \ifHy@hyperindex
5628         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5629             \@glsxtr@format@override true
5630             \appto\theindex{\let\glshypernumber\@firstofone}%
5631         }
5632     \else
5633         \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5634             \@glsxtr@format@override true
5635             \appto\theindex{\let\glshypernumber\hyperpage}%
5636         }
5637     \fi
5638 }
5639 {
5640     \newcommand*{\GlsXtrEnableIndexFormatOverride}{%
5641         \@glsxtr@format@override true
5642     }
5643 }
5644 \@onlypreamble\GlsXtrEnableIndexFormatOverride

```

`doautoindexname`

```

5645 \newcommand*{\glsxtrdoautoindexname}[2]{%
5646     \glshasattribute{#1}{#2}%
5647     {%

```

Escape any makeindex/xindy characters in the value of the name field. Take care with babel as this won't work if the category code has changed for those characters.

```
5648 \glsxtr@autoindex@setname{#1}%
```

If the attribute value is simply “true” don't add an encap, otherwise use the value as the encap.

```
5649 \protected@edef\glsxtr@attrval{\glsgetattribute{#1}{#2}}%
5650 \ifglsxtr@format@override

5651 \ifx\glsnumberformat\glsxtr@defaultnumberformat
5652 \else
5653 \let\glsxtr@attrval\glsnumberformat
5654 \fi
5655 \fi
5656 \ifdefstring{\glsxtr@attrval}{true}%
5657 {}%
5658 {\eappto\glo@name{\glsxtr@autoindex@encap\glsxtr@attrval}}%
5659 \expandafter\glsxtrautoindex\expandafter{\glo@name}%
5660 }%
5661 {}%
5662 }
```

glsxtrautoindex

```
5663 \newcommand*{\glsxtrautoindex}{\index}
```

toindex@setname Assign \glo@name for use with indexname attribute.

```
5664 \newcommand*{\glsxtr@autoindex@setname}[1]{%
5665 \protected@edef\glo@name{\glsxtrautoindexentry{#1}}%
5666 \glsxtrautoindexassignsort{\glo@sort}{#1}%
5667 \gls@checkmkidxchars\glo@sort
5668 \glsxtr@autoindex@doextra@esc\glo@sort
5669 \epreto\glo@name{\glo@sort\glsxtr@autoindex@at}%
5670 }
```

autoindexentry Command used for the actual part when auto-indexing.

```
5671 \newcommand*{\glsxtrautoindexentry}[1]{\string\glstryname{#1}}
```

indexassignsort Used to assign the sort value when auto-indexing.

```
5672 \newcommand*{\glsxtrautoindexassignsort}[2]{%
5673 \glsletentryfield{#1}{#2}{sort}%
5674 }
```

dex@doextra@esc

```
5675 \newcommand*{\glsxtr@autoindex@doextra@esc}[1]{%
```

Escape the escape character unless it has already been escaped.

```
5676 \ifx\glsxtr@autoindex@esc\gls@quotechar
5677 \else
5678 \def\gls@checkedmkidx{}%
5679 \edef\@glsxtr@checkspch{%
```



```

5680 \noexpand\@glsxtr@autoindex@escquote\expandonce{#1}%
5681 \noexpand\@empty\@glsxtr@autoindex@esc\noexpand\@nnil
5682 \@glsxtr@autoindex@esc\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5683 \@@glsxtr@checkspch
5684 \let#1\@gls@checkedmkidx\relax
5685 \fi

```

Escape actual character unless it has already been escaped.

```

5686 \ifx\@glsxtr@autoindex@at\@gls@actualchar
5687 \else
5688 \def\@gls@checkedmkidx{}%
5689 \edef\@@glsxtr@checkspch{%
5690 \noexpand\@glsxtr@autoindex@escat\expandonce{#1}%
5691 \noexpand\@empty\@glsxtr@autoindex@at\noexpand\@nnil
5692 \@glsxtr@autoindex@at\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5693 \@@glsxtr@checkspch
5694 \let#1\@gls@checkedmkidx\relax
5695 \fi

```

Escape level character unless it has already been escaped.

```

5696 \ifx\@glsxtr@autoindex@level\@gls@levelchar
5697 \else
5698 \def\@gls@checkedmkidx{}%
5699 \edef\@@glsxtr@checkspch{%
5700 \noexpand\@glsxtr@autoindex@esclevel\expandonce{#1}%
5701 \noexpand\@empty\@glsxtr@autoindex@level\noexpand\@nnil
5702 \@glsxtr@autoindex@level\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5703 \@@glsxtr@checkspch
5704 \let#1\@gls@checkedmkidx\relax
5705 \fi

```

Escape encap character unless it has already been escaped.

```

5706 \ifx\@glsxtr@autoindex@encap\@gls@encapchar
5707 \else
5708 \def\@gls@checkedmkidx{}%
5709 \edef\@@glsxtr@checkspch{%
5710 \noexpand\@glsxtr@autoindex@escencap\expandonce{#1}%
5711 \noexpand\@empty\@glsxtr@autoindex@encap\noexpand\@nnil
5712 \@glsxtr@autoindex@encap\noexpand\@empty\noexpand\@glsxtr@endescspch}%
5713 \@@glsxtr@checkspch
5714 \let#1\@gls@checkedmkidx\relax
5715 \fi
5716 }

```

The user commands here have a preamble-only restriction to ensure they are set before required and also to reduce the chances of complications caused by babel's shorthands.

tr@autoindex@at Actual character for use with \index.

```

5717 \newcommand*{\@glsxtr@autoindex@at}{\}

```

trSetActualChar Set the actual character.

```
5718 \newcommand*{\GlsXtrSetActualChar}[1]{%
5719   \gdef\@glstr@autoindex@at{#1}%
5720   \def\@glstr@autoindex@escat##1##2##3\@glstr@endescspch{%
5721     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escat}{##1}{##2}{##3}%
5722   }%
5723 }
5724 \@onlypreamble\GlsXtrSetActualChar
5725 \makeatother
5726 \GlsXtrSetActualChar{@}
5727 \makeatletter
```

autoindex@encap Encap character for use with \index.

```
5728 \newcommand*{\@glstr@autoindex@encap}{}
```

XtrSetEncapChar Set the encap character.

```
5729 \newcommand*{\GlsXtrSetEncapChar}[1]{%
5730   \gdef\@glstr@autoindex@encap{#1}%
5731   \def\@glstr@autoindex@escencap##1##2##3\@glstr@endescspch{%
5732     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escencap}{##1}{##2}{##3}%
5733   }%
5734 }
5735 \GlsXtrSetEncapChar{||}
5736 \@onlypreamble\GlsXtrSetEncapChar
```

autoindex@level Level character for use with \index.

```
5737 \newcommand*{\@glstr@autoindex@level}{}
```

XtrSetLevelChar Set the encap character.

```
5738 \newcommand*{\GlsXtrSetLevelChar}[1]{%
5739   \gdef\@glstr@autoindex@level{#1}%
5740   \def\@glstr@autoindex@esclevel##1##2##3\@glstr@endescspch{%
5741     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@esclevel}{##1}{##2}{##3}%
5742   }%
5743 }
5744 \GlsXtrSetLevelChar{!}
5745 \@onlypreamble\GlsXtrSetLevelChar
```

r@autoindex@esc Escape character for use with \index.

```
5746 \newcommand*{\@glstr@autoindex@esc}{""}
```

lsXtrSetEscChar Set the escape character.

```
5747 \newcommand*{\GlsXtrSetEscChar}[1]{%
5748   \gdef\@glstr@autoindex@esc{#1}%
5749   \def\@glstr@autoindex@escquote##1##2##3\@glstr@endescspch{%
5750     \@glstr@autoindex@escspch{#1}{\@glstr@autoindex@escquote}{##1}{##2}{##3}%
5751   }%
5752 }
```

```

5753 \GlsXtrSetEscChar{}
5754 \@onlypreamble\GlsXtrSetEscChar

```

Set if defined. (For example, if doc package has been loaded.) Actual character \actualchar:

```

5755 \ifdef\actualchar
5756 {\expandafter\GlsXtrSetActualChar\expandafter{\actualchar}}
5757 {}

```

Quote character \quotechar:

```

5758 \ifdef\quotechar
5759 {\expandafter\GlsXtrSetEscChar\expandafter{\quotechar}}
5760 {}

```

Level character \levelchar:

```

5761 \ifdef\levelchar
5762 {\expandafter\GlsXtrSetLevelChar\expandafter{\levelchar}}
5763 {}

```

Encap character \encapchar:

```

5764 \ifdef\encapchar
5765 {\expandafter\GlsXtrSetEncapChar\expandafter{\encapchar}}
5766 {}

```

leto@endescspch

```

5767 \def\@glxtr@gobbleto@endescspch#1\@glxtr@endescspch{}

```

toindex@esc@spch

```

\@glxtr@autoindex@escspch{<char>}{<cs>}{<pre>}{<mid>}{<post>}

```

```

5768 \newcommand*{\@glxtr@autoindex@escspch}[5]{%
5769   \@glstmpb=\expandafter{\@glstmpb\the\toks@}%
5770   \toks@={#3}%
5771   \ifx\@nnil#3\relax
5772     \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch#5\@glxtr@endescspch}%
5773   \else
5774     \ifx\@nnil#4\relax
5775       \edef\@glstmpb{\the\@glstmpb\the\toks@}%
5776       \def\@glxtr@checkspch{\@glxtr@gobbleto@endescspch
5777         #4#5\@glxtr@endescspch}%
5778     \else
5779       \edef\@glstmpb{\the\@glstmpb\the\toks@
5780         \@glxtr@autoindex@esc#1}%
5781       \def\@glxtr@checkspch{#2#5#1\@nnil#1\@glxtr@endescspch}%
5782     \fi
5783   \fi
5784   \@glxtr@checkspch
5785 }

```

`\Glossentrydesc` Redefine to set the abbreviation format and accessibility support.

```
5786 \renewcommand*{\Glossentrydesc}[1]{%
5787   \glsdoifexistsorwarn{#1}%
5788   {%
5789     \glssetabbrvfmt{\glscategory{#1}}%
5790     \Glsaccessdesc{#1}%
5791   }%
5792 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5793 \renewcommand*{\glossentrysymbol}[1]{%
5794   \glsdoifexistsorwarn{#1}%
5795   {%
5796     \glssetabbrvfmt{\glscategory{#1}}%
5797     \Glsaccesssymbol{#1}%
5798   }%
5799 }
```

`\glossentrysymbol` Redefine to set the abbreviation format and accessibility support.

```
5800 \renewcommand*{\Glossentrysymbol}[1]{%
5801   \glsdoifexistsorwarn{#1}%
5802   {%
5803     \glssetabbrvfmt{\glscategory{#1}}%
5804     \Glsaccesssymbol{#1}%
5805   }%
5806 }
```

Allow initials to be marked but only use the formatting for the tag in the glossary.

`\enableInitialTagging` Allow initial tagging. The first argument is a list of categories to apply this to. The second argument is the name of the command to use to tag the initials. This can't already be defined for safety unless the starred version is used.

```
5807 \newcommand*{\GlsXtrEnableInitialTagging}{%
5808   \@ifstar\s@glxtr@enabletagging\@glxtr@enabletagging
5809 }
5810 \@onlypreamble\GlsXtrEnableInitialTagging
```

`\r@enabletagging` Starred version undefines command.

```
5811 \newcommand*{\s@glxtr@enabletagging}[2]{%
5812   \undef#2%
5813   \@glxtr@enabletagging{#1}{#2}%
5814 }
```

`\r@enabletagging` Internal command.

```
5815 \newcommand*{\@glxtr@enabletagging}[2]{%
  Set attributes for categories given in the first argument.
5816   \@for\@glxtr@cat:=#1\do
5817   {%
```

```

5818 \ifdefempty\@glsxtr@cat
5819 {}%
5820 {\glssetcategoryattribute{\@glsxtr@cat}{tagging}{true}}%
5821 }%
5822 \newrobustcmd*#2[1]{##1}%
5823 \def\@glsxtr@taggingcs{#2}%
5824 \renewcommand*\@glsxtr@activate@initialtagging{%
5825 \let#2\@glsxtr@tag
5826 }%
5827 \ifundef\@gls@preglossaryhook
5828 {\GlossariesExtraWarning{Initial tagging requires at least
5829 glossaries.sty v4.19 to work correctly}}%
5830 {}%
5831 }

```

Are we using an old version of mfirstuc that has a bug in \capitalisewords? If so, patch it so we don't have a problem with a combination of tagging and title case.

fu@checkword@do If this command hasn't been defined, then we have pre v2.02 of mfirstuc

```

5832 \ifundef\mfu@checkword@do
5833 {
5834 \newcommand*{\mfu@checkword@do}[1]{%
5835 \ifdefstring{\mfu@checkword@arg}{#1}%
5836 {%
5837 \let\@mfu@domakefirstuc\@firstofone
5838 \listbreak
5839 }%
5840 }%
5841 }

```

\mfu@checkword \capitalisewords was introduced in mfirstuc v1.06. If \mfu@checkword hasn't been defined mfirstuc is too old to support the title case attribute.

```

5842 \ifundef\mfu@checkword
5843 {
5844 \newcommand{\@glsxtr@do@titlecaps@warn}{%
5845 \GlossariesExtraWarning{mfirstuc.sty too old. Title Caps
5846 support not available}%

```

One warning should suffice.

```

5847 \let\@glsxtr@do@titlecaps@warn\relax
5848 }
5849 }
5850 {
5851 \renewcommand*{\mfu@checkword}[1]{%
5852 \def\mfu@checkword@arg{#1}%
5853 \let\@mfu@domakefirstuc\makefirstuc
5854 \forlistloop\mfu@checkword@do\@mfu@nocaplist
5855 }
5856 }
5857 }

```

```

5858 {}% no patch required

@titlecaps@warn Do warning if title case not supported.
5859 \newcommand*{\@glxtr@do@titlecaps@warn}{}

@initialtagging Used in \printglossary but at least v4.19 of glossaries required.
5860 \newcommand*\@glxtr@activate@initialtagging{}

\@glxtr@tag Definition of tagging command when used in glossary.
5861 \newrobustcmd*{\@glxtr@tag}[1]{%
5862   \gl@ifattribute{\glscurrententrylabel}{tagging}{true}%
5863   {\glxtrtagfont{#1}}{#1}%
5864 }

\glxtrtagfont Used in the glossary.
5865 \newcommand*{\glxtrtagfont}[1]{\underline{#1}}

preglossaryhook This macro was introduced in glossaries version 4.19, so it may not be defined. If it hasn't
been defined this feature is unavailable. A check is added for the entry's existence to prevent
errors from occurring if the user removes an entry or changes the label, which can interrupt
the build process.
5866 \ifdef\@gl@preglossaryhook
5867 {
5868   \renewcommand*{\@gl@preglossaryhook}{%
5869     \@glxtr@activate@initialtagging

    Since the glossaries are automatically scoped, \@glxtr@org@postdescription shouldn't
    already be defined, but check anyway just as a precautionary measure.

5870     \ifundef\@glxtr@org@postdescription
5871     {%
5872       \let\@glxtr@org@postdescription\glspostdescription
5873       \renewcommand*{\glspostdescription}{%
5874         \ifglentryexists{\glscurrententrylabel}%
5875         {%
5876           \glxtrpostdescription
5877           \@glxtr@org@postdescription
5878         }%
5879       }%
5880     }%
5881     }%
5882     {}%

    Enable the options used by \@glxtrp:
5883     \glossxtrsetopts
5884   }%
5885 }
5886 {}

```

postdescription This command will only be used if `\@gls@preglossaryhook` is available *and* the glossary style uses `\glspostdescription` without modifying it. (`\nopostdesc` will suppress this.) The glossaries-extra-stylemods package will add the post description hook to all the predefined styles that don't include it.

```
5887 \newcommand*{\glsxtrpostdescription}{%
5888   \csuse{glsxtrpostdesc\glscategory{\glscurrententrylabel}}%
5889 }
```

postdescgeneral

```
5890 \newcommand*{\glsxtrpostdescgeneral}{}%
```

xtrpostdescterm

```
5891 \newcommand*{\glsxtrpostdescterm}{}%
```

postdescacronym

```
5892 \newcommand*{\glsxtrpostdescacronym}{}%
```

descabbreviation

```
5893 \newcommand*{\glsxtrpostdescabbreviation}{}%
```

glspostlinkhook Redefine the post link hook used by commands like `\gls` to make it easier for categories or attributes to modify this action. Since this hook occurs outside the existence check of commands like `\gls`, this needs to be checked again here. Do nothing if the entry hasn't been defined.

```
5894 \renewcommand*{\glspostlinkhook}{%
5895   \ifglsentryexists{\glslabel}{\glsxtrpostlinkhook}{}%
5896 }
```

xtrpostlinkhook The entry label should already be stored in `\glslabel` by `\@gls@link`.

```
5897 \newcommand*{\glsxtrpostlinkhook}{%
5898   \glsxtrdiscardperiod{\glslabel}%
5899   {\glsxtrpostlinkendsentence}%
5900   {\glsxtrifcustomdiscardperiod
5901     {\glsxtrifperiod{\glsxtrpostlinkendsentence}{\glsxtrpostlink}}%
5902     {\glsxtrpostlink}}%
5903   }%
5904 }
```

omdiscardperiod Allow user to provide a custom check. Should expand to #2 if no check is required otherwise expand to #1.

```
5905 \newcommand*{\glsxtrifcustomdiscardperiod}[2]{#2}
```

\glsxtrpostlink

```
5906 \newcommand*{\glsxtrpostlink}{%
5907   \csuse{glsxtrpostlink\glscategory{\glslabel}}%
5908 }
```

linkendsentence Done by \glxtrpostlinkhook if a full stop is discarded.

```

5909 \newcommand*{\glxtrpostlinkendsentence}{%
5910   \ifcsdef{glxtrpostlink\glscategory{\glslabel}}
5911   {%
5912     \csuse{glxtrpostlink\glscategory{\glslabel}}%
5913     .\spacefactor\sfcode'\. \relax
5914   }%
5915   {%
5916     \spacefactor\sfcode'\. \relax
5917   }%
5918 }

```

Put the full stop back.

Assume the full stop was discarded because the entry ends with a period, so adjust the spacefactor.

dDescOnFirstUse Provide a command for appending the description in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```

5919 \newcommand*{\glxtrpostlinkAddDescOnFirstUse}{%
5920   \glxtrifwasfirstuse{\space\glxtrparen{\glaccessdesc{\glslabel}}{}}%
5921 }

```

symbolOnFirstUse Provide a command for appending the symbol (if defined) in parentheses on first use, for the convenience of users wanting to add this to the post link hook.

```

5922 \newcommand*{\glxtrpostlinkAddSymbolOnFirstUse}{%
5923   \glxtrifwasfirstuse
5924   {%
5925     \ifglshassymbol{\glslabel}%
5926     {\space\glxtrparen{\glaccesssymbol{\glslabel}}}%
5927     {}%
5928   }%
5929   {}%
5930 }

```

trdiscardperiod Discard following period (if present) if the discardperiod attribute is true. If a period is discarded, do the second argument otherwise do the third argument. The entry label is in the first argument. Since this is designed for abbreviations that end with a period, check if the plural form was used (which typically won't end with a period).

```

5931 \newcommand*{\glxtrdiscardperiod}[3]{%
5932   \glxtrifwasfirstuse
5933   {%
5934     \gl@ifattribute{#1}{retainfirstuseperiod}{true}%
5935     {#3}%
5936     {%
5937       \gl@ifattribute{#1}{discardperiod}{true}%
5938       {%
5939         \gl@ifplural
5940         {%

```



```

5941      \glsifattribute{#1}{pluraldiscardperiod}{true}%
5942      {\glsxtrifperiod{#2}{#3}}%
5943      {#3}%
5944      }%
5945      {%
5946      \glsxtrifperiod{#2}{#3}%
5947      }%
5948      }%
5949      {#3}%
5950      }%
5951  }%
5952  {%
5953  \glsifattribute{#1}{discardperiod}{true}%
5954  {%
5955  \glsifplural
5956  {%
5957  \glsifattribute{#1}{pluraldiscardperiod}{true}%
5958  {\glsxtrifperiod{#2}{#3}}%
5959  {#3}%
5960  }%
5961  {%
5962  \glsxtrifperiod{#2}{#3}%
5963  }%
5964  }%
5965  {#3}%
5966  }%
5967  }

```

`\glsxtrifperiod` Make a convenient user command to check if the next character is a full stop (period). Works like `\@ifstar` but uses `\new@ifnextchar` rather than `\@ifnextchar`

```
5968 \newcommand*{\glsxtrifperiod}[1]{\new@ifnextchar.{\@firstoftwo{#1}}}
```

Sometimes it's useful to test if there's a punctuation character following the glossary entry.

`\glsxtr@punclist` List of characters identified as punctuation marks. (Be careful of babel shorthands!) This doesn't allow for punctuation marks made up from multiple characters (such as ' ').

```
5969 \newcommand*{\glsxtr@punclist}{.,:;?!}
```

`\punctuationmark` Add character to punctuation list.

```
5970 \newcommand*{\glsxtraddpunctuationmark}[1]{\appto\glsxtr@punclist{#1}}
```

`\punctuationmarks` Reset the punctuation list.

```
5971 \newcommand*{\glsxtrsetpunctuationmarks}[1]{\def\glsxtr@punclist{#1}}
```

`\glsxtrifpunc` `\glsxtrifnextpunc{<true part>}{<false part>}`

Test if this is followed by a punctuation mark. (Adapted from `\new@ifnextchar`.)

```
5972 \newcommand*{\glxtrifnextpunc}[2]{%
5973   \def\reserved@a{#1}%
5974   \def\reserved@b{#2}%
5975   \futurelet\@glspunc@token\glxtr@ifnextpunc
5976 }
```

`glxtr@ifnextpunc`

```
5977 \newcommand*{\glxtr@ifnextpunc}{%
5978   \glxtr@ifpunctoken{\@glspunc@token}{\let\reserved@b\reserved@a}{}%
5979   \reserved@b
5980 }
```

`glxtr@ifpunctoken` Test if the token given in the first argument is in the punctuation list.

```
5981 \newcommand*{\glxtr@ifpunctoken}[1]{%
5982   \expandafter\@glxtr@ifpunctoken\expandafter#1\glxtr@punclist\@nnil
5983 }
```

`glxtr@ifpunctoken`

```
5984 \def\@glxtr@ifpunctoken#1#2{%
5985   \let\reserved@d=#2%
5986   \ifx\reserved@d\@nnil
5987     \let\glxtr@next\@glxtr@notfoundinlist
5988   \else
5989     \ifx#1\reserved@d
5990       \let\glxtr@next\@glxtr@foundinlist
5991     \else
5992       \let\glxtr@next\@glxtr@ifpunctoken
5993     \fi
5994   \fi
5995   \glxtr@next#1%
5996 }
```

`glxtr@foundinlist`

```
5997 \def\@glxtr@foundinlist#1\@nnil{\@firstoftwo}
```

`@notfoundinlist`

```
5998 \def\@glxtr@notfoundinlist#1{\@secondoftwo}
```

`glxtr@dopostpunc`

```
\glxtr@dopostpunc{<code>}
```

If this is followed by a punctuation character, do `<code>` after the character otherwise do `<code>` before whatever comes next.

```
5999 \newcommand{\glxtr@dopostpunc}[1]{%
6000   \glxtrifnextpunc{\@glxtr@swaptwo{#1}}{#1}%
6001 }
```

@glxtr@swaptwo

```
6002 \newcommand{\@glxtr@swaptwo}[2]{#2#1}
```

## 1.7 Abbreviations

The “acronym” code from glossaries is misnamed as it’s more often used for other forms of abbreviations. This code corrects this inconsistency, but rather than just having synonyms, provide commands for abbreviations that have a similar, but not identical, underlying mechanism to acronyms.

If there’s a style for the given category, apply it.

```
6003 \define@key{glxtrabbrv}{category}{%
6004 \edef\glscategorylabel{#1}%
6005 \ifcsdef{@glsabbrv@current@#1}%
6006 {%
```

Warning should already have been issued.

```
6007 \let\@glxtr@orgwarndep\GlsXtrWarnDeprecatedAbbrStyle
6008 \let\GlsXtrWarnDeprecatedAbbrStyle\@gobbletwo
6009 \glxtr@applyabbrvstyle{\csname @glsabbrv@current@#1\endcsname}%
6010 \let\GlsXtrWarnDeprecatedAbbrStyle\@glxtr@orgwarndep
6011 }%
6012 {}%
6013 }
```

Save the short plural form. This may be needed before the entry is defined.

```
6014 \define@key{glxtrabbrv}{shortplural}{%
6015 \def\@gls@shortpl{#1}%
6016 }
```

Similarly for the long plural form.

```
6017 \define@key{glxtrabbrv}{longplural}{%
6018 \def\@gls@longpl{#1}%
6019 }
```

Token registers for the short plural and long plural, provided for use in the abbreviation style definitions.

\glsshortpltok

```
6020 \newtoks\glsshortpltok
```

\glslongpltok

```
6021 \newtoks\glslongpltok
```

`\glxtr@insertdots` Provided in case user wants to automatically insert dots between each letter of the abbreviation. This should be applied before defining the abbreviation to optimise the document build. (Otherwise, it would have to be done each time the short form is required, which is an unnecessary waste of time.) For this to work the short form must be expanded when passed to `\newabbreviation`. Note that explicitly using the short or shortplural keys will override this.

```

6022 \newcommand*{\@glxtr@insertdots}[2]{%
6023   \def#1{}%
6024   \@glxtr@insert@dots#1#2\@nnil
6025 }

```

xtr@insert@dots

```

6026 \newcommand*{\@glxtr@insert@dots}[2]{%
6027   \ifx\@nnil#2\relax
6028   \let\@glxtr@insert@dots@next\@gobble
6029   \else
6030   \ifx\relax#2\relax
6031   \else
6032     \appto#1{#2.}%
6033   \fi
6034   \let\@glxtr@insert@dots@next\@glxtr@insert@dots
6035   \fi
6036   \@glxtr@insert@dots@next#1%
6037 }

```

Similarly provide a way of replacing spaces with \glxtrwordsep, which first needs to be defined:

\glxtrwordsep

```

6038 \newcommand*{\glxtrwordsep}{\space}

```

Each word is marked with

\glxtrword

```

6039 \newcommand*{\glxtrword}[1]{#1}

```

tr@markwordseps

```

6040 \newcommand*{\@glxtr@markwordseps}[2]{%
6041   \def#1{}%
6042   \@glxtr@mark@wordseps#1#2 \@nnil
6043 }

```

r@mark@wordseps

```

6044 \def\@glxtr@mark@wordseps#1#2 #3{%
6045   \ifdefempty{#1}%
6046   {\def#1{\protect\glxtrword{#2}}}%
6047   {\appto#1{\protect\glxtrwordsep\protect\glxtrword{#2}}}%
6048   \ifx\@nnil#3\relax
6049   \let\@glxtr@mark@wordseps@next\relax
6050   \else
6051   \def\@glxtr@mark@wordseps@next{%
6052     \@glxtr@mark@wordseps#1#3}%
6053   \fi
6054   \@glxtr@mark@wordseps@next
6055 }

```

newabbreviation Define a new generic abbreviation.

```

6056 \newcommand*{\newabbreviation}[4] [] {%
6057   \glxstr@newabbreviation{#1}{#2}{#3}{#4}%
6058 }

```

newabbreviation Internal macro. (bib2gls has an option that needs to temporarily redefine \newabbreviation. This is just makes it easier to save and restore the original definition.)

```

6059 \newcommand*{\glxstr@newabbreviation}[4] {%
6060   \glscopylisttok{#1}%
6061   \glscopylabeltok{#2}%
6062   \glscopyshorttok{#3}%
6063   \glscopylongtok{#4}%

```

Save the original short and long values (before attribute settings modify them).

```

6064   \def\glxstrorgshort{#3}%
6065   \def\glxstrorglong{#4}%

```

Get the category.

```

6066   \def\glscategorylabel{abbreviation}%
6067   \glxstr@applyabbrvstyle{\@glscopy@current@abbreviation}%

```

Ignore the shortplural and longplural keys.

```

6068   \setkeys*{\glxstrabbrv}[shortplural,longplural]{#1}%

```

Set the default long plural

```

6069   \def\@glscopy@longpl{#4\glscopypluralsuffix}%
6070   \let\@glscopy@default@longpl\@glscopy@longpl

```

Has the markwords attribute been set?

```

6071   \glscopyifcategoryattribute{\glscopycategorylabel}{markwords}{true}%
6072   {%
6073     \@glscopy@markwordseps\@glscopy@long{#4}%
6074     \expandafter\def\expandafter\@glscopy@longpl\expandafter
6075       {\@glscopy@long\glscopypluralsuffix}%
6076     \let\@glscopy@default@longpl\@glscopy@longpl

```

Update \glscopylongtok.

```

6077     \expandafter\glscopylongtok\expandafter{\@glscopy@long}%
6078   }%
6079   {%

```

Has the markshortwords attribute been set? (Not compatible with insertdots.)

```

6080   \glscopyifcategoryattribute{\glscopycategorylabel}{markshortwords}{true}%
6081   {%
6082     \@glscopy@markwordseps\@glscopy@short{#3}%
6083   }%
6084   {%

```

Has the insertdots attribute been set?

```

6085   \glscopyifcategoryattribute{\glscopycategorylabel}{insertdots}{true}%
6086   {%
6087     \@glscopy@insertdots\@glscopy@short{#3}%

```

```

6088     \expandafter\glsshorttok\expandafter{\@gls@short\spacefactor1000 \relax}%
6089   }%
6090   {\def\@gls@short{#3}}%
6091 }%

```

Has the aposplural attribute been set? (Not compatible with noshortplural.)

```

6092 \glsifcategoryattribute{\glscategorylabel}{aposplural}{true}%
6093 {%
6094   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6095     '\abbrvpluralsuffix}%
6096 }%
6097 {%

```

Has the noshortplural attribute been set?

```

6098 \glsifcategoryattribute{\glscategorylabel}{noshortplural}{true}%
6099 {%
6100   \let\@gls@shortpl\@gls@short
6101 }%
6102 {%
6103   \expandafter\def\expandafter\@gls@shortpl\expandafter{\@gls@short
6104     '\abbrvpluralsuffix}%
6105 }%
6106 }%

```

Update \glsshorttok:

```

6107 \expandafter\glsshorttok\expandafter{\@gls@short}%

```

Hook for further customisation if required:

```

6108 \glstrnewabbrevpresetkeyhook{#1}{#2}{#3}%

```

Get the short and long plurals provided by user in optional argument to override defaults, if necessary. Ignore the category key (already obtained).

```

6109 \setkeys*{\glstrabbrv}[category]{#1}%

```

Has the plural been explicitly set?

```

6110 \ifx\@gls@default@longpl\@gls@longpl
6111 \else

```

Has the markwords attribute been set?

```

6112 \glsifcategoryattribute{\glscategorylabel}{markwords}{true}%
6113 {%
6114   \expandafter\@glstr@markwordseps\expandafter\@gls@longpl\expandafter
6115     {\@gls@longpl}%
6116 }%
6117 {}%
6118 \fi

```

Set the plural token registers so the values can be accessed by the abbreviation styles.

```

6119 \expandafter\glsshortpltok\expandafter{\@gls@shortpl}%
6120 \expandafter\glslongpltok\expandafter{\@gls@longpl}%

```

Do any extra setup provided by hook:

```

6121 \newabbreviationhook

```

Define this entry:

```
6122 \protected@edef\@do@newglossaryentry{%
6123   \noexpand\newglossaryentry{\the\glslabeltok}%
6124   {%
6125     type=\glstrabbrvtype,%
6126     category=abbreviation,%
6127     short={\the\glsshorttok},%
6128     shortplural={\the\glsshortpltok},%
6129     long={\the\glslongtok},%
6130     longplural={\the\glslongpltok},%
6131     name={\the\glsshorttok},%
6132     \CustomAbbreviationFields,%
6133     \the\glskeylisttok
6134   }%
6135 }%
6136 \@do@newglossaryentry
6137 \GlsXtrPostNewAbbreviation
6138 }
```

evpresetkeyhook Hook for extra stuff in \newabbreviation

```
6139 \newcommand*{\glstrnewabbspresetkeyhook}[3]{}%
```

NewAbbreviation Hook used by abbreviation styles.

```
6140 \newcommand*{\GlsXtrPostNewAbbreviation}{}%
```

bbreviationhook Hook for use with \newabbreviation.

```
6141 \newcommand*{\newabbreviationhook}{}%
```

reviationFields

```
6142 \newcommand*{\CustomAbbreviationFields}{}%
```

\glstrparen For the parenthetical styles.

```
6143 \newcommand*{\glstrparen}[1]{(#1)}
```

lsxtrfullformat Full format without case change.

```
6144 \newcommand*{\glstrfullformat}[2]{%
6145   \glsfirstlongfont{\glsaccesslong{#1}}#2\glstrfullsep{#1}%
6146   \glstrparen{\protect\glsfirstabbrvfont{\glsaccessshort{#1}}}%
6147 }
```

lsxtrfullformat Full format with case change.

```
6148 \newcommand*{\Glsxtrfullformat}[2]{%
6149   \glsfirstlongfont{\Glsaccesslong{#1}}#2\glstrfullsep{#1}%
6150   \glstrparen{\protect\glsfirstabbrvfont{\Glsaccessshort{#1}}}%
6151 }
```

`\glxtrfullplformat` Plural full format without case change.

```
6152 \newcommand*{\glxtrfullplformat}[2]{%
6153   \glsfirstlongfont{\glssaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6154   \glxtrparen{\protect\glsfirstabbrvfont{\glssaccessshortpl{#1}}}%
6155 }
```

`\glxtrfullplformat` Plural full format with case change.

```
6156 \newcommand*{\Glsxtrfullplformat}[2]{%
6157   \glsfirstlongfont{\Glsaccesslongpl{#1}}#2\glxtrfullsep{#1}%
6158   \glxtrparen{\protect\glsfirstabbrvfont{\Glsaccessshortpl{#1}}}%
6159 }
```

`\glxtrfullsep` Separator used by full format is a space by default. The argument is the entry's label.

```
6160 \newcommand*{\glxtrfullsep}[1]{\space}
```

In-line formats in case first use isn't compatible with `\gl Sentryfull` (for example, first use suppresses the long form or uses a footnote).

`\glxtrinlinefullformat` Full format without case change.

```
6161 \newcommand*{\glxtrinlinefullformat}{\glxtrfullformat}
```

`\Glsxtrinlinefullformat` Full format with case change.

```
6162 \newcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}
```

`\glxtrinlinefullplformat` Plural full format without case change.

```
6163 \newcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}
```

`\Glsxtrinlinefullplformat` Plural full format with case change.

```
6164 \newcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}
```

Redefine `\gl Sentryfull` 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 `\glxtrfull` set of commands instead.

`\gl Sentryfull`

```
6165 \renewcommand*{\gl Sentryfull}[1]{\glxtrinlinefullformat{#1}{}}
```

`\Glsentryfull`

```
6166 \renewcommand*{\Glsentryfull}[1]{\Glsxtrinlinefullformat{#1}{}}
```

`\gl Sentryfullpl`

```
6167 \renewcommand*{\gl Sentryfullpl}[1]{\glxtrinlinefullplformat{#1}{}}
```

`\Glsentryfullpl`

```
6168 \renewcommand*{\Glsentryfullpl}[1]{\Glsxtrinlinefullplformat{#1}{}}
```

`\gl sfirstabbrvfont` Font changing command used for the abbreviation on first use or in the full format.

```
6169 \newcommand*{\gl sfirstabbrvfont}[1]{\gl sfirstabbrvdefaultfont{#1}}
```



`bbrvdefaultfont` Font changing command used for the abbreviation on first use or in the full format.  
6170 `\newcommand*{\glsfirstabbrvdefaultfont}[1]{\glsabbrvfont{#1}}`

`\glsabbrvfont` Font changing command used for the abbreviation on subsequent use.  
6171 `\newcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{#1}}`

`bbrvdefaultfont`  
6172 `\newcommand*{\glsabbrvdefaultfont}[1]{#1}`

`\glslongfont` Font changing command used for the long form in commands like `\glsxtrlong`.  
6173 `\newcommand*{\glslongfont}[1]{\glslongdefaultfont{#1}}`

`longdefaultfont` Default font changing command used for the long form in commands like `\glsxtrlong`.  
6174 `\newcommand*{\glslongdefaultfont}[1]{#1}`

`lsfirstlongfont` Font changing command used for the long form on first use or in the full format.  
6175 `\newcommand*{\glsfirstlongfont}[1]{\glslongfont{#1}}`

`longdefaultfont`  
6176 `\newcommand*{\glsfirstlongdefaultfont}[1]{\glslongdefaultfont{#1}}`

`brvpluralsuffix` Default plural suffix. Allow an alternative default suffix for abbreviations.  
6177 `\newcommand*{\glsxtrabbrvpluralsuffix}{\glspluralsuffix}`

`brvpluralsuffix` Default plural suffix.  
6178 `\newcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}`

`\glsxtrfull` Full form (no case-change).  
6179 `\newrobustcmd*{\glsxtrfull}{\@gls@hyp@opt\ns@glsxtrfull}`  
6180 `\newcommand*{ns@glsxtrfull}[2][\%`  
6181 `\new@ifnextchar[{\@glsxtr@full{#1}{#2}}{\%`  
6182 `{\@glsxtr@full{#1}{#2}[]}\%`  
6183 `}`

`\@glsxtr@full` Low-level macro:  
6184 `\def\@glsxtr@full#1#2[#3]{\%`  
6185 `\glsdoifexists{#2}\%`  
6186 `{\%`  
6187 `\glssetabbrvfmt{\glscategory{#2}}\%`  
6188 `\let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper`  
6189 `\let\glsifplural\@secondoftwo`  
6190 `\let\gls caps case\@firstofthree`  
6191 `\let\glsinsert\@empty`  
6192 `\def\gls custom text{\glsxtr inline full format{#2}{#3}}\%`

What should `\glxtrifwasfirstuse` be set to here? Where the inline and display full forms are the same, this is essentially emulating first use, to it make sense for the postlink hook to pretend it was a first use instance. It makes less sense if the inline and display forms are different. Provide a hook to make it easier to reconfigure.

```
6193 \glxtrsetupfulldefs
6194 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6195 }%
6196 \glspostlinkhook
6197 }
```

`trsetupfulldefs`

```
6198 \newcommand*\glxtrsetupfulldefs{%
6199 \let\glxtrifwasfirstuse\@firstoftwo
6200 }
```

`\Glsxtrfull` Full form (first letter uppercase).

```
6201 \newrobustcmd*\Glsxtrfull{\@gls@hyp@opt\@ns@Glsxtrfull}
6202 \newcommand*\ns@Glsxtrfull[2][{}]{%
6203 \new@ifnextchar[\@Glsxtr@full{#1}{#2}}%
6204 {\@Glsxtr@full{#1}{#2}[]}%
6205 }
```

`\@Glsxtr@full` Low-level macro:

```
6206 \def\@Glsxtr@full#1#2[#3]{%
6207 \glsdoifexists{#2}%
6208 {%
6209 \glssetabbrvfmt{\glscategory{#2}}%
6210 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6211 \let\glsifplural\@secondoftwo
6212 \let\glsupcase\@secondofthree
6213 \let\glsinsert\@empty
6214 \def\glscustomtext{\Glsxtrinlinefullformat{#2}{#3}}%
6215 \glxtrsetupfulldefs
6216 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6217 }%
6218 \glspostlinkhook
6219 }
```

`\GLSxtrfull` Full form (all uppercase).

```
6220 \newrobustcmd*\GLSxtrfull{\@gls@hyp@opt\@ns@GLSxtrfull}
6221 \newcommand*\ns@GLSxtrfull[2][{}]{%
6222 \new@ifnextchar[\@GLSxtr@full{#1}{#2}}%
6223 {\@GLSxtr@full{#1}{#2}[]}%
6224 }
```

`\@GLSxtr@full` Low-level macro:

```
6225 \def\@GLSxtr@full#1#2[#3]{%
6226 \glsdoifexists{#2}%
6227 }
```

```

6227 {%
6228   \glssetabbrvfmt{\glscategory{#2}}%
6229   \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6230   \let\glsifplural\@secondoftwo
6231   \let\gls caps case\@thirdofthree
6232   \let\glsinsert\@empty
6233   \def\gls custom text{\mfirstucMakeUppercase{\glsxtr inline full format{#2}{#3}}}%
6234   \glsxtr setup full defs
6235   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6236 }%
6237 \gls post link hook
6238 }

```

`\glsxtrfullpl` Plural full form (no case-change).

```

6239 \newrobustcmd*{\glsxtrfullpl}{\@gls@hyp@opt\@ns@glsxtrfullpl}
6240 \newcommand*\ns@glsxtrfullpl[2][{}]{%
6241   \new@ifnextchar[{\@glsxtr@fullpl{#1}{#2}}%
6242     {\@glsxtr@fullpl{#1}{#2}[]}%
6243 }

```

`\@glsxtr@fullpl` Low-level macro:

```

6244 \def\@glsxtr@fullpl#1#2[#3]{%
6245   \glsdoifexists{#2}%
6246   {%
6247     \glssetabbrvfmt{\glscategory{#2}}%
6248     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6249     \let\glsifplural\@firstoftwo
6250     \let\gls caps case\@firstofthree
6251     \let\glsinsert\@empty
6252     \def\gls custom text{\glsxtr inline full pl format{#2}{#3}}%
6253     \glsxtr setup full defs
6254     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6255   }%
6256   \gls post link hook
6257 }

```

`\Glsxtrfullpl` Plural full form (first letter uppercase).

```

6258 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6259 \newcommand*\ns@Glsxtrfullpl[2][{}]{%
6260   \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6261     {\@Glsxtr@fullpl{#1}{#2}[]}%
6262 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6263 \def\@Glsxtr@fullpl#1#2[#3]{%
6264   \glsdoifexists{#2}%
6265   {%
6266     \glssetabbrvfmt{\glscategory{#2}}%
6267     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper

```

```

6268 \let\glsifplural\@firstoftwo
6269 \let\glsifscapscase\@secondofthree
6270 \let\glsinsert\@empty
6271 \def\glscustomtext{\Glsxtrinlinefullplformat{#2}{#3}}%
6272 \glsxtrsetupfulldefs
6273 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6274 }%
6275 \glspostlinkhook
6276 }

```

`\Glsxtrfullpl` Plural full form (all upper case).

```

6277 \newrobustcmd*{\Glsxtrfullpl}{\@gls@hyp@opt\@ns@Glsxtrfullpl}
6278 \newcommand*\ns@Glsxtrfullpl[2][]{%
6279 \new@ifnextchar[{\@Glsxtr@fullpl{#1}{#2}}%
6280 {\@Glsxtr@fullpl{#1}{#2}[]}%
6281 }

```

`\@Glsxtr@fullpl` Low-level macro:

```

6282 \def\@Glsxtr@fullpl#1#2[#3]{%
6283 \glsdoifexists{#2}%
6284 {%
6285 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6286 \let\glsifplural\@firstoftwo
6287 \let\glsifscapscase\@thirdofthree
6288 \let\glsinsert\@empty
6289 \def\glscustomtext{%
6290 \mfirstucMakeUppercase{\glsxtrinlinefullplformat{#2}{#3}}}%
6291 \glsxtrsetupfulldefs
6292 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6293 }%
6294 \glspostlinkhook
6295 }

```

The short and long forms work in a similar way to acronyms.

`\glsxtrshort`

```

6296 \newrobustcmd*{\glsxtrshort}{\@gls@hyp@opt\@ns@glsxtrshort}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6297 \newcommand*\ns@glsxtrshort[2][]{%
6298 \new@ifnextchar[{\@glsxtrshort{#1}{#2}}{\@glsxtrshort{#1}{#2}[]}%
6299 }

```

Read in the final optional argument:

```

6300 \def\@glsxtrshort#1#2[#3]{%
6301 \glsdoifexists{#2}%
6302 {%

```

Need to make sure `\glsabbrvfont` is set correctly.

```

6303 \glssetabbrvfmt{\glscategory{#2}}%

```

```

6304 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6305 \let\glxtrifwasfirstuse\@secondoftwo
6306 \let\gl@sifplural\@secondoftwo
6307 \let\glscapscase\@firstofthree
6308 \let\gl$insert\@empty
6309 \def\glscustomtext{%
6310 \gl@sabbrvfont{\gl@saccessshort{#2}\ifglxtrininsertinside#3\fi}%
6311 \ifglxtrininsertinside\else#3\fi
6312 }%
6313 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6314 }%
6315 \glspostlinkhook
6316 }

```

\Glsxtrshort

```

6317 \newrobustcmd*{\Glsxtrshort}{\@gl@hyp@opt\@ns@Glsxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
6318 \newcommand*{\ns@Glsxtrshort}[2][{}]{%
6319 \new@ifnextchar[{\@Glsxtrshort{#1}{#2}}{\@Glsxtrshort{#1}{#2}[]}%
6320 }
    Read in the final optional argument:
6321 \def\@Glsxtrshort#1#2[#3]{%
6322 \gl@doifexists{#2}%
6323 {%
6324 \gl@setabbrvfmt{\gl@category{#2}}%
6325 \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6326 \let\glxtrifwasfirstuse\@secondoftwo
6327 \let\gl@sifplural\@secondoftwo
6328 \let\glscapscase\@secondofthree
6329 \let\gl$insert\@empty
6330 \def\glscustomtext{%
6331 \gl@sabbrvfont{\Glsaccessshort{#2}\ifglxtrininsertinside#3\fi}%
6332 \ifglxtrininsertinside\else#3\fi
6333 }%
6334 \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6335 }%
6336 \glspostlinkhook
6337 }

```

\GLSxtrshort

```

6338 \newrobustcmd*{\GLSxtrshort}{\@gl@hyp@opt\@ns@GLSxtrshort}
    Define the un-starred form. Need to determine if there is a final optional argument
6339 \newcommand*{\ns@GLSxtrshort}[2][{}]{%
6340 \new@ifnextchar[{\@GLSxtrshort{#1}{#2}}{\@GLSxtrshort{#1}{#2}[]}%
6341 }
    Read in the final optional argument:
6342 \def\@GLSxtrshort#1#2[#3]{%

```

```

6343 \glsdoifexists{#2}%
6344 {%
6345   \glssetabbrvfmt{\glscategory{#2}}%
6346   \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6347   \let\glxtrifwasfirstuse\@secondoftwo
6348   \let\gl@ifplural\@secondoftwo
6349   \let\glscapscase\@thirdofthree
6350   \let\glinsert\@empty
6351   \def\glscustomtext{%
6352     \mfirstucMakeUppercase
6353     {\glabbrvfont{\glaccessshort{#2}\ifglxtrininsertinside#3\fi}%
6354     \ifglxtrininsertinside\else#3\fi
6355   }%
6356   }%
6357   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6358   }%
6359   \glspostlinkhook
6360 }

```

`\glxtrlong`

```

6361 \newrobustcmd*{\glxtrlong}{\@gl@hyp@opt\@ns@glxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6362 \newcommand*{\ns@glxtrlong}[2] []{%
6363   \new@ifnextchar[{\@glxtrlong{#1}{#2}}{\@glxtrlong{#1}{#2} []}%
6364 }
    Read in the final optional argument:
6365 \def\@glxtrlong#1#2[#3]{%
6366   \glsdoifexists{#2}%
6367   {%
6368     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6369     \let\glxtrifwasfirstuse\@secondoftwo
6370     \let\gl@ifplural\@secondoftwo
6371     \let\glscapscase\@firstofthree
6372     \let\glinsert\@empty
6373     \def\glscustomtext{%
6374       \glslongfont{\glaccesslong{#2}\ifglxtrininsertinside#3\fi}%
6375       \ifglxtrininsertinside\else#3\fi
6376     }%
6377     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6378     }%
6379     \glspostlinkhook
6380 }

```

`\Glsxtrlong`

```

6381 \newrobustcmd*{\Glsxtrlong}{\@gl@hyp@opt\@ns@Glsxtrlong}
    Define the un-starred form. Need to determine if there is a final optional argument
6382 \newcommand*{\ns@Glsxtrlong}[2] []{%

```

```

6383 \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6384 }

```

Read in the final optional argument:

```

6385 \def\@GLSxtrlong#1#2[#3]{%
6386   \glsdoifexists{#2}%
6387   {%
6388     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6389     \let\glxtrifwasfirstuse\@secondoftwo
6390     \let\gl@sifplural\@secondoftwo
6391     \let\glscapscase\@secondofthree
6392     \let\gl$insert\@empty
6393     \def\glscustomtext{%
6394       \glslongfont{\gl@saccesslong{#2}\ifglxtrininsertinside#3\fi}%
6395       \ifglxtrininsertinside\else#3\fi
6396     }%
6397     \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6398   }%
6399   \glspostlinkhook
6400 }

```

\GLSxtrlong

```

6401 \newrobustcmd*{\GLSxtrlong}{\@gl@hyp@opt\@ns@GLSxtrlong}

```

Define the un-starred form. Need to determine if there is a final optional argument

```

6402 \newcommand*{\ns@GLSxtrlong}[2] []{%
6403   \new@ifnextchar[{\@GLSxtrlong{#1}{#2}}{\@GLSxtrlong{#1}{#2} []}%
6404 }

```

Read in the final optional argument:

```

6405 \def\@GLSxtrlong#1#2[#3]{%
6406   \glsdoifexists{#2}%
6407   {%
6408     \let\do@gl@link@checkfirsthyper\@gl@link@nocheckfirsthyper
6409     \let\glxtrifwasfirstuse\@secondoftwo
6410     \let\gl@sifplural\@secondoftwo
6411     \let\glscapscase\@thirdofthree
6412     \let\gl$insert\@empty
6413     \def\glscustomtext{%
6414       \mfirstucMakeUppercase
6415       {\glslongfont{\gl@saccesslong{#2}\ifglxtrininsertinside#3\fi}%
6416       \ifglxtrininsertinside\else#3\fi
6417     }%
6418   }%
6419   \@gl@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6420 }%
6421 \glspostlinkhook
6422 }

```

Plural short forms:

\glstrshortpl

```
6423 \newrobustcmd*{\glstrshortpl}{\@gls@hyp@opt\@ns@glstrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6424 \newcommand*{\ns@glstrshortpl}[2] [] {%
6425   \new@ifnextchar[{\@glstrshortpl{#1}{#2}}{\@glstrshortpl{#1}{#2} []}%
6426 }

    Read in the final optional argument:
6427 \def\@glstrshortpl#1#2[#3] {%
6428   \glsdoifexists{#2}%
6429   {%
6430     \glssetabbrvfmt{\glscategory{#2}}%
6431     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6432     \let\glstrifwasfirstuse\@secondoftwo
6433     \let\glsifplural\@firstoftwo
6434     \let\glscapscase\@firstofthree
6435     \let\glsinsert\@empty
6436     \def\glscustomtext{%
6437       \glsabbrvfont{\glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
6438       \ifglstrinsertinside\else#3\fi
6439     }%
6440     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6441   }%
6442   \glspostlinkhook
6443 }
```

\Glsxtrshortpl

```
6444 \newrobustcmd*{\Glsxtrshortpl}{\@gls@hyp@opt\@ns@Glsxtrshortpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6445 \newcommand*{\ns@Glsxtrshortpl}[2] [] {%
6446   \new@ifnextchar[{\@Glsxtrshortpl{#1}{#2}}{\@Glsxtrshortpl{#1}{#2} []}%
6447 }

    Read in the final optional argument:
6448 \def\@Glsxtrshortpl#1#2[#3] {%
6449   \glsdoifexists{#2}%
6450   {%
6451     \glssetabbrvfmt{\glscategory{#2}}%
6452     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6453     \let\glstrifwasfirstuse\@secondoftwo
6454     \let\glsifplural\@firstoftwo
6455     \let\glscapscase\@secondofthree
6456     \let\glsinsert\@empty
6457     \def\glscustomtext{%
6458       \Glsabbrvfont{\Glsaccessshortpl{#2}\ifglstrinsertinside#3\fi}%
6459       \ifglstrinsertinside\else#3\fi
6460     }%
6461     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6462   }%
```



```

6463 \glspostlinkhook
6464 }

```

\GLSxtrshortpl

```

6465 \newrobustcmd*{\GLSxtrshortpl}{\@gls@hyp@opt\@ns@GLSxtrshortpl}

  Define the un-starred form. Need to determine if there is a final optional argument
6466 \newcommand*{\ns@GLSxtrshortpl}[2] [] {%
6467   \new@ifnextchar[{\@GLSxtrshortpl{#1}{#2}}{\@GLSxtrshortpl{#1}{#2} []}%
6468 }

  Read in the final optional argument:
6469 \def\@GLSxtrshortpl#1#2[#3] {%
6470   \glsdoifexists{#2}%
6471   {%
6472     \glssetabbrvfmt{\glscategory{#2}}%
6473     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6474     \let\glsxtrifwasfirstuse\@secondoftwo
6475     \let\glsifplural\@firstoftwo
6476     \let\glsapscase\@thirdofthree
6477     \let\glsinsert\@empty
6478     \def\glscustomtext{%
6479       \mfirstucMakeUppercase
6480       {\glsabbrvfont{\glsaccessshortpl{#2}\ifglsxtrininsertinside#3\fi}%
6481       \ifglsxtrininsertinside\else#3\fi
6482     }%
6483   }%
6484   \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6485 }%
6486 \glspostlinkhook
6487 }

```

Plural long forms:

\glsxtrlongpl

```

6488 \newrobustcmd*{\glsxtrlongpl}{\@gls@hyp@opt\@ns@glsxtrlongpl}

  Define the un-starred form. Need to determine if there is a final optional argument
6489 \newcommand*{\ns@glsxtrlongpl}[2] [] {%
6490   \new@ifnextchar[{\@glsxtrlongpl{#1}{#2}}{\@glsxtrlongpl{#1}{#2} []}%
6491 }

  Read in the final optional argument:
6492 \def\@glsxtrlongpl#1#2[#3] {%
6493   \glsdoifexists{#2}%
6494   {%
6495     \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6496     \let\glsxtrifwasfirstuse\@secondoftwo
6497     \let\glsifplural\@firstoftwo
6498     \let\glsapscase\@firstofthree
6499     \let\glsinsert\@empty

```

```

6500 \def\glscustomtext{%
6501 \glslongfont{\glssaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
6502 \ifglxtrinsertinside\else#3\fi
6503 }%
6504 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6505 }%
6506 \glspostlinkhook
6507 }

```

\Glsxtrlongpl

```

6508 \newrobustcmd*{\Glsxtrlongpl}{\@gls@hyp@opt\@ns@Glsxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6509 \newcommand*{\ns@Glsxtrlongpl}[2] [] {%
6510 \new@ifnextchar[{\@Glsxtrlongpl{#1}{#2}}{\@Glsxtrlongpl{#1}{#2} []}%
6511 }
    Read in the final optional argument:
6512 \def\@Glsxtrlongpl#1#2[#3]{%
6513 \glsdofexists{#2}%
6514 {%
6515 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6516 \let\glxtrifwasfirstuse\@secondoftwo
6517 \let\glsifplural\@firstoftwo
6518 \let\glscapscase\@secondofthree
6519 \let\glsinsert\@empty
6520 \def\glscustomtext{%
6521 \glslongfont{\glssaccesslongpl{#2}\ifglxtrinsertinside#3\fi}%
6522 \ifglxtrinsertinside\else#3\fi
6523 }%
6524 \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6525 }%
6526 \glspostlinkhook
6527 }

```

\GLSxtrlongpl

```

6528 \newrobustcmd*{\GLSxtrlongpl}{\@gls@hyp@opt\@ns@GLSxtrlongpl}
    Define the un-starred form. Need to determine if there is a final optional argument
6529 \newcommand*{\ns@GLSxtrlongpl}[2] [] {%
6530 \new@ifnextchar[{\@GLSxtrlongpl{#1}{#2}}{\@GLSxtrlongpl{#1}{#2} []}%
6531 }
    Read in the final optional argument:
6532 \def\@GLSxtrlongpl#1#2[#3]{%
6533 \glsdofexists{#2}%
6534 {%
6535 \let\do@gls@link@checkfirsthyper\@gls@link@nocheckfirsthyper
6536 \let\glxtrifwasfirstuse\@secondoftwo
6537 \let\glsifplural\@firstoftwo
6538 \let\glscapscase\@thirdofthree

```

```

6539 \let\glsinsert\@empty
6540 \def\glscustomtext{%
6541     \mfirstucMakeUppercase
6542     {\glslongfont{\glsaccesslongpl{#2}\ifglstrinsertinside#3\fi}%
6543     \ifglstrinsertinside\else#3\fi
6544     }%
6545     }%
6546     \@gls@link[#1]{#2}{\csname gls@\glstype @entryfmt\endcsname}%
6547 }%
6548 \glspostlinkhook
6549 }

```

`\glsetabbrvfmt` Set the current format for the given category (or the abbreviation category if unset).

```

6550 \newcommand*{\glsetabbrvfmt}[1]{%
6551     \ifcsdef{@glsabbrv@current@#1}%
6552     {\glstr@applyabbrvfmt{\csname @glsabbrv@current@#1\endcsname}}%
6553     {\glstr@applyabbrvfmt{\@glsabbrv@current@abbreviation}}%
6554 }

```

`\gluseabbrvfont` Provide a way to use the abbreviation font for a given category for arbitrary text.

```

6555 \newrobustcmd*{\gluseabbrvfont}[2]{\glsetabbrvfmt{#2}\glsabbrvfont{#1}}

```

`\gluselongfont` Provide a way to use the long font for a given category for arbitrary text.

```

6556 \newrobustcmd*{\gluselongfont}[2]{\glsetabbrvfmt{#2}\glslongfont{#1}}

```

`\glstrgenabbrvfmt` Similar to `\glsgenacfmt`, but for abbreviations.

```

6557 \newcommand*{\glstrgenabbrvfmt}{%
6558     \ifdefempty\glscustomtext
6559     {%
6560         \ifglssused\glslabel
6561         {%

```

Subsequent use:

```

6562         \glsifplural
6563         {%

```

Subsequent plural form:

```

6564         \glscapscase
6565         {%

```

Subsequent plural form, don't adjust case:

```

6566         \glstrsubsequentplfmt{\glslabel}{\glsinsert}%
6567     }%
6568     {%

```

Subsequent plural form, make first letter upper case:

```

6569         \Glsxtrsubsequentplfmt{\glslabel}{\glsinsert}%
6570     }%
6571     {%

```

Subsequent plural form, all caps:

```
6572      \mfirstucMakeUppercase
6573      {\glxtrsubsequentplfmt{\glslabel}{\glsinsert}}%
6574      }%
6575      }%
6576      {%
```

Subsequent singular form

```
6577      \glscapscase
6578      {%
```

Subsequent singular form, don't adjust case:

```
6579      \glxtrsubsequentfmt{\glslabel}{\glsinsert}%
6580      }%
6581      {%
```

Subsequent singular form, make first letter upper case:

```
6582      \Glsxtrsubsequentfmt{\glslabel}{\glsinsert}%
6583      }%
6584      {%
```

Subsequent singular form, all caps:

```
6585      \mfirstucMakeUppercase
6586      {\glxtrsubsequentfmt{\glslabel}{\glsinsert}}%
6587      }%
6588      }%
6589      }%
6590      {%
```

First use:

```
6591      \glsifplural
6592      {%
```

First use plural form:

```
6593      \glscapscase
6594      {%
```

First use plural form, don't adjust case:

```
6595      \glxtrfullplformat{\glslabel}{\glsinsert}%
6596      }%
6597      {%
```

First use plural form, make first letter upper case:

```
6598      \Glsxtrfullplformat{\glslabel}{\glsinsert}%
6599      }%
6600      {%
```

First use plural form, all caps:

```
6601      \mfirstucMakeUppercase
6602      {\glxtrfullplformat{\glslabel}{\glsinsert}}%
6603      }%
6604      }%
6605      {%
```

First use singular form

```
6606      \glscapscase
6607      {%
```

First use singular form, don't adjust case:

```
6608      \glxtrfullformat{\glslabel}{\glsinsert}%
6609      }%
6610      {%
```

First use singular form, make first letter upper case:

```
6611      \Glsxtrfullformat{\glslabel}{\glsinsert}%
6612      }%
6613      {%
```

First use singular form, all caps:

```
6614      \mfirstucMakeUppercase
6615      {\glxtrfullformat{\glslabel}{\glsinsert}}%
6616      }%
6617      }%
6618      }%
6619      }%
6620      {%
```

User supplied text.

```
6621      \glscustomtext
6622      }%
6623 }
```

trsubsequentfmt Subsequent use format (singular no case change).

```
6624 \newcommand*{\glxtrsubsequentfmt}[2]{%
6625   \glsabbrvfont{\glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
6626   \ifglxtrininsertinside \else#2\fi
6627 }
6628 \let\glxtrdefaultsubsequentfmt\glxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural no case change).

```
6629 \newcommand*{\glxtrsubsequentplfmt}[2]{%
6630   \glsabbrvfont{\glsaccessshortpl{#1}\ifglxtrininsertinside #2\fi}%
6631   \ifglxtrininsertinside \else#2\fi
6632 }
6633 \let\glxtrdefaultsubsequentplfmt\glxtrsubsequentplfmt
```

trsubsequentfmt Subsequent use format (singular, first letter uppercase).

```
6634 \newcommand*{\Glsxtrsubsequentfmt}[2]{%
6635   \glsabbrvfont{\Glsaccessshort{#1}\ifglxtrininsertinside #2\fi}%
6636   \ifglxtrininsertinside \else#2\fi
6637 }
6638 \let\Glsxtrdefaultsubsequentfmt\Glsxtrsubsequentfmt
```

subsequentplfmt Subsequent use format (plural, first letter uppercase).

```

6639 \newcommand*{\Glsxtrsubsequentplfmt}[2]{%
6640   \glsabbrvfont{\Glsaccessshortpl{#1}\ifglxtrinsertinside #2\fi}%
6641   \ifglxtrinsertinside \else#2\fi
6642 }
6643 \let\Glsxtrdefaultsubsequentplfmt\Glsxtrsubsequentplfmt

```

### 1.7.1 Abbreviation Styles Setup

breviationstyle

```

6644 \newcommand*{\setabbreviationstyle}[2][abbreviation]{%
6645   \ifcsundef{@glsabbrv@dispstyle@setup@#2}%
6646   {%
6647     \PackageError{glossaries-extra}{Undefined abbreviation style ‘#2’}{}%
6648   }%
6649   {%

```

Have abbreviations already been defined for this category?

```

6650     \ifcsstring{@glsabbrv@current@#1}{#2}%
6651     {%

```

Style already set.

```

6652     }%
6653     {%
6654       \def\@glxtr@dostylewarn{%
6655         \glsforeachincategory{#1}{\@gls@type}{\@gls@label}%
6656         {%
6657           \def\@glxtr@dostylewarn{\GlossariesWarning{Abbreviation
6658             style has been switched \MessageBreak
6659             for category ‘#1’, \MessageBreak
6660             but there have already been entries \MessageBreak
6661             defined for this category. Unwanted \MessageBreak
6662             side-effects may result}}%
6663           \@endfortrue
6664         }%
6665         \@glxtr@dostylewarn

```

Set up the style for the given category.

```

6666         \csdef{@glsabbrv@current@#1}{#2}%
6667         \glxtr@applyabbrvstyle{#2}%
6668       }%
6669     }%
6670 }

```

applyabbrvstyle Apply the abbreviation style without existence check.

```

6671 \newcommand*{\glxtr@applyabbrvstyle}[1]{%
6672   \csuse{@glsabbrv@dispstyle@setup@#1}%
6673   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6674 }

```

r@applyabbrvfmt Only apply the style formats.

```
6675 \newcommand*{\glxtr@applyabbrvfmt}[1]{%
6676   \csuse{@glsabbrv@dispstyle@fmts@#1}%
6677 }
```

breivationstyle This is different from \newacronymstyle. The first argument is the label, the second argument sets the information required when defining the new abbreviation and the third argument sets the commands used to display the full format.

```
6678 \newcommand*{\newabbreviationstyle}[3]{%
6679   \ifcsdef{@glsabbrv@dispstyle@setup@#1}
6680     {%
6681       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ already
6682         defined}{}%
6683     }%
6684     {%
6685       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
6686   \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6687   #2}%
6688   \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```
6689   \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6690   \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6691   \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6692   \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%

```

Reset \glxtrsubsequentfmt etc in case a style changes this.

```
6693   \let\glxtrsubsequentfmt\glxtrdefaultsubsequentfmt
6694   \let\glxtrsubsequentplfmt\glxtrdefaultsubsequentplfmt
6695   \let\Glsxtrsubsequentfmt\Glsxtrdefaultsubsequentfmt
6696   \let\Glsxtrsubsequentplfmt\Glsxtrdefaultsubsequentplfmt
6697   #3}%
6698 }%
6699 }
```

breivationstyle

```
6700 \newcommand*{\renewabbreviationstyle}[3]{%
6701   \ifcsundef{@glsabbrv@dispstyle@setup@#1}
6702     {%
6703       \PackageError{glossaries-extra}{Abbreviation style ‘#1’ not defined}{}%
6704     }%
6705     {%
6706       \csdef{@glsabbrv@dispstyle@setup@#1}{%

```

Initialise hook to do nothing. The style may change this.

```
6707   \renewcommand*{\GlsXtrPostNewAbbreviation}{}%
6708   #2}%
6709   \csdef{@glsabbrv@dispstyle@fmts@#1}{%

```

Assume in-line form is the same as first use. The style may change this.

```

6710 \renewcommand*{\glxtrinlinefullformat}{\glxtrfullformat}%
6711 \renewcommand*{\Glsxtrinlinefullformat}{\Glsxtrfullformat}%
6712 \renewcommand*{\glxtrinlinefullplformat}{\glxtrfullplformat}%
6713 \renewcommand*{\Glsxtrinlinefullplformat}{\Glsxtrfullplformat}%
6714 #3}%
6715 }%
6716 }

```

**abbreviationstyle** Define a synonym for an abbreviation style. The first argument is the new name. The second argument is the original style's name.

```

6717 \newcommand*{\letabbreviationstyle}[2]{%
6718 \csletcs{@glsabbrv@dispstyle@setup@#1}{@glsabbrv@dispstyle@setup@#2}%
6719 \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6720 }

```

**deprecated@abbrstyle** `\@glxtr@deprecated@abbrstyle{<old-name>}{<new-name>}`

Define a synonym for a deprecated abbreviation style.

```

6721 \newcommand*{\@glxtr@deprecated@abbrstyle}[2]{%
6722 \csdef{@glsabbrv@dispstyle@setup@#1}{%
6723 \GlsXtrWarnDeprecatedAbbrStyle{#1}{#2}%
6724 \csuse{@glsabbrv@dispstyle@setup@#2}%
6725 }%
6726 \csletcs{@glsabbrv@dispstyle@fmts@#1}{@glsabbrv@dispstyle@fmts@#2}%
6727 }

```

**deprecatedAbbrStyle** Generate warning for deprecated style use.

```

6728 \newcommand*{\GlsXtrWarnDeprecatedAbbrStyle}[2]{%
6729 \GlossariesExtraWarning{Deprecated abbreviation style name ‘#1’,
6730 use ‘#2’ instead}%
6731 }

```

**UseAbbrStyleSetup**

```

6732 \newcommand*{\GlsXtrUseAbbrStyleSetup}[1]{%
6733 \ifcsundef{@glsabbrv@dispstyle@setup@#1}%
6734 {%
6735 \PackageError{glossaries-extra}%
6736 {Unknown abbreviation style definitions ‘#1’}{}%
6737 }%
6738 {%
6739 \csname @glsabbrv@dispstyle@setup@#1\endcsname
6740 }%
6741 }

```



seAbbrStyleFmts

```
6742 \newcommand*{\GlsXtrUseAbbrStyleFmts}[1]{%
6743   \ifcsundef{@glsabbrv@dispstyle@fmts@#1}%
6744   {%
6745     \PackageError{glossaries-extra}%
6746     {Unknown abbreviation style formats ‘#1’}{}%
6747   }%
6748   {%
6749     \csname @glsabbrv@dispstyle@fmts@#1\endcsname
6750   }%
6751 }
```

### 1.7.2 Predefined Styles (Default Font)

Define some common styles. These will set the first, firstplural, text and plural keys, even if the regular attribute isn't set to "true". If this attribute is set, commands like `\gls` will use them as per a regular entry, otherwise those keys will be ignored unless explicitly invoked by the user with commands like `\glsfirst`. In order for the first letter uppercase versions to work correctly, `\glsxtrfullformat` needs to be expanded when those keys are set. The final optional argument of `\glsfirst` will behave differently to the final optional argument of `\gls` with some styles.

xtrinsertinside Switch to determine if the insert text should be inside or outside the font changing command.  
The default is outside.

```
6752 \newif\ifglsxtrinsertinside
6753 \glsxtrinsertinsidefalse
```

trlongshortname

```
6754 \newcommand*{\glsxtrlongshortname}{%
6755   \protect\glsabbrvfont{\the\glsshorttok}%
6756 }
```

long-short

```
6757 \newabbreviationstyle{long-short}%
6758 {%
6759   \renewcommand*{\CustomAbbreviationFields}{%
6760     name={\glsxtrlongshortname},
6761     sort={\the\glsshorttok},
6762     first={\protect\glsfirstlongfont{\the\glslongtok}%
6763       \protect\glsxtrfullsep{\the\glslabeltok}%
6764       \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6765     firstplural={\protect\glsfirstlongfont{\the\glslongtok}%
6766       \protect\glsxtrfullsep{\the\glslabeltok}%
6767       \glsxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%
6768     plural={\protect\glsabbrvfont{\the\glsshortpltok}},%
6769     description={\the\glslongtok}}%
```

Unset the regular attribute if it has been set.

```
6770 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6771   \glshasattribute{\the\glslabeltok}{regular}%
6772   {%
6773     \glsssetattribute{\the\glslabeltok}{regular}{false}%
6774   }%
6775   {}%
6776 }%
6777 }%
6778 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
6779 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6780 \renewcommand*{\glssabrvfont}[1]{\glssabrvdefaultfont{##1}}%
6781 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvdefaultfont{##1}}%
6782 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongdefaultfont{##1}}%
6783 \renewcommand*{\glsslongfont}[1]{\glsslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
6784 \renewcommand*{\glsxtrfullformat}[2]{%
6785   \glssfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6786   \ifglsxtrininsertinside\else##2\fi
6787   \glsxtrfullsep{##1}%
6788   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}%
6789 }%
6790 \renewcommand*{\glsxtrfullplformat}[2]{%
6791   \glssfirstlongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6792   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6793   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}%
6794 }%
6795 \renewcommand*{\Glsxtrfullformat}[2]{%
6796   \glssfirstlongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
6797   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6798   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshort{##1}}}%
6799 }%
6800 \renewcommand*{\Glsxtrfullplformat}[2]{%
6801   \glssfirstlongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
6802   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
6803   \glsxtrparen{\glssfirstabbrvfont{\glssaccessshortpl{##1}}}%
6804 }%
6805 }
```

Set this as the default style for general abbreviations:

```
6806 \setabbreviationstyle{long-short}
```

ngshortdescsort

```
6807 \newcommand*{\glsxtrlongshortdescsort}{%
6808   \expandonce\glsxtrorglong\space (\expandonce\glsxtrorgshort)%
6809 }
```

ngshortdescname

```
6810 \newcommand*{\glxtrlongshortdescname}{%
6811   \protect\glslongfont{\the\glslongtok}
6812   \glxtrparen{\protect\glsabbrvfont{\the\glsshorttok}}%
6813 }
```

long-short-desc User supplies description. The long form is included in the name.

```
6814 \newabbreviationstyle{long-short-desc}%
6815 {%
6816   \renewcommand*{\CustomAbbreviationFields}{%
6817     name={\glxtrlongshortdescname},
6818     sort={\glxtrlongshortdescsort},%
6819     first={\protect\glsfirstlongfont{\the\glslongtok}%
6820       \protect\glxtrfullsep{\the\glslabeltok}%
6821       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshorttok}}},%
6822     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}%
6823       \protect\glxtrfullsep{\the\glslabeltok}%
6824       \glxtrparen{\protect\glsfirstabbrvfont{\the\glsshortpltok}}},%

```

The text key should only have the short form.

```
6825   text={\protect\glsabbrvfont{\the\glsshorttok}},%
6826   plural={\protect\glsabbrvfont{\the\glsshortpltok}}%
6827 }%
```

Unset the regular attribute if it has been set.

```
6828 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6829   \glshasattribute{\the\glslabeltok}{regular}%
6830   {%
6831     \glissetattribute{\the\glslabeltok}{regular}{false}%
6832   }%
6833   {}%
6834 }%
6835 }%
6836 {%
6837   \GlsXtrUseAbbrStyleFmts{long-short}%
6838 }
```

trshortlongname

```
6839 \newcommand*{\glxtrshortlongname}{%
6840   \protect\glsabbrvfont{\the\glsshorttok}%
6841 }
```

short-long Short form followed by long form in parenthesis on first use.

```
6842 \newabbreviationstyle{short-long}%
6843 {%
6844   \renewcommand*{\CustomAbbreviationFields}{%
6845     name={\glxtrshortlongname},
6846     sort={\the\glsshorttok},
6847     description={\the\glslongtok},%

```

```

6848 first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6849 \protect\glsxtrfullsep{\the\glslabeltok}%
6850 \glsxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6851 firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6852 \protect\glsxtrfullsep{\the\glslabeltok}%
6853 \glsxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6854 plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

6855 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6856 \glshasattribute{\the\glslabeltok}{regular}%
6857 {%
6858 \glssetattribute{\the\glslabeltok}{regular}{false}%
6859 }%
6860 {}%
6861 }%
6862 }%
6863 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

6864 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
6865 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6866 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6867 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
6868 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

6869 \renewcommand*{\glsxtrfullformat}[2]{%
6870 \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
6871 \ifglsxtrinsertinside\else##2\fi
6872 \glsxtrfullsep{##1}%
6873 \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6874 }%
6875 \renewcommand*{\glsxtrfullplformat}[2]{%
6876 \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
6877 \ifglsxtrinsertinside\else##2\fi
6878 \glsxtrfullsep{##1}%
6879 \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6880 }%
6881 \renewcommand*{\Glsxtrfullformat}[2]{%
6882 \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrinsertinside##2\fi}%
6883 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
6884 \glsxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
6885 }%
6886 \renewcommand*{\Glsxtrfullplformat}[2]{%
6887 \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrinsertinside##2\fi}%
6888 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
6889 \glsxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
6890 }%
6891 }

```

ortlongdescsort

```
6892 \newcommand*{\glxtrshortlongdescsort}{\the\glsshorttok}
```

ortlongdescname

```
6893 \newcommand*{\glxtrshortlongdescname}{%
6894   \protect\glsabbrvfont{\the\glsshorttok}
6895   \glxtrparen{\protect\glslongfont{\the\glslongtok}}}%
6896 }
```

short-long-desc User supplies description. The long form is included in the name.

```
6897 \newabbreviationstyle{short-long-desc}%
6898 {%
6899   \renewcommand*{\CustomAbbreviationFields}{%
6900     name={\glxtrshortlongdescname},
6901     sort={\glxtrshortlongdescsort},
6902     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6903       \protect\glxtrfullsep{\the\glslabeltok}%
6904       \glxtrparen{\protect\glsfirstlongfont{\the\glslongtok}}},%
6905     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6906       \protect\glxtrfullsep{\the\glslabeltok}%
6907       \glxtrparen{\protect\glsfirstlongfont{\the\glslongpltok}}},%
6908     text={\protect\glsabbrvfont{\the\glsshorttok}},%
6909     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
6910 }%
```

Unset the regular attribute if it has been set.

```
6911 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6912   \glshasattribute{\the\glslabeltok}{regular}%
6913   {%
6914     \glissetattribute{\the\glslabeltok}{regular}{false}%
6915   }%
6916   {}%
6917 }%
6918 }%
6919 {%
6920   \GlsXtrUseAbbrStyleFmts{short-long}%
6921 }
```

ongfootnotefont Only used by the “footnote” styles.

```
6922 \newcommand*{\glsfirstlongfootnotefont}[1]{\glslongfootnotefont{#1}}%
```

ongfootnotefont Only used by the “footnote” styles.

```
6923 \newcommand*{\glslongfootnotefont}[1]{\glslongdefaultfont{#1}}%
```

xtrabbrvfootnote

```
\glxtrabbrvfootnote{<label>}{<long>}
```

Command used by footnote abbreviation styles. The default definition ignores the first argument. The second argument *<long>* includes the font changing command and may be the singular or plural form, depending on the command that was used (for example, `\gls` or `\glspl`).

```
6924 \newcommand*{\glstrabbrvfootnote}[2]{\footnote{#2}}
```

xtrfootnotename

```
6925 \newcommand*{\glstrfootnotename}{%
6926   \protect\glsabbrvfont{\the\glsshorttok}%
6927 }
```

footnote Short form followed by long form in footnote on first use.

```
6928 \newabbreviationstyle{footnote}%
6929 {%
6930   \renewcommand*{\CustomAbbreviationFields}{%
6931     name={\glstrfootnotename},
6932     sort={\the\glsshorttok},
6933     description={\the\glslongtok},%

6934     first={\protect\glsfirstabbrvfont{\the\glsshorttok}%
6935       \protect\glstrabbrvfootnote{\the\glslabeltok}%
6936       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
6937     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}%
6938       \protect\glstrabbrvfootnote{\the\glslabeltok}%
6939       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%

6940     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
6941 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
6942   \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
6943   \glshasattribute{\the\glslabeltok}{regular}%
6944   {%
6945     \glssetattribute{\the\glslabeltok}{regular}{false}%
6946   }%
6947   {}%
6948 }%
6949 }%
6950 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```
6951 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
6952 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
6953 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
6954 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
6955 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

6956 \renewcommand*{\glxtrfullformat}[2]{%
6957   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6958   \ifglxtrininsertinside\else##2\fi
6959   \protect\glxtrabbrvfootnote{##1}%
6960   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6961 }%
6962 \renewcommand*{\glxtrfullplformat}[2]{%
6963   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6964   \ifglxtrininsertinside\else##2\fi
6965   \protect\glxtrabbrvfootnote{##1}%
6966   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6967 }%
6968 \renewcommand*{\Glsxtrfullformat}[2]{%
6969   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6970   \ifglxtrininsertinside\else##2\fi
6971   \protect\glxtrabbrvfootnote{##1}%
6972   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6973 }%
6974 \renewcommand*{\Glsxtrfullplformat}[2]{%
6975   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6976   \ifglxtrininsertinside\else##2\fi
6977   \protect\glxtrabbrvfootnote{##1}%
6978   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6979 }%

```

The first use full form and the inline full form use the short (long) style.

```

6980 \renewcommand*{\glxtrinlinefullformat}[2]{%
6981   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6982   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6983   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6984 }%
6985 \renewcommand*{\glxtrinlinefullplformat}[2]{%
6986   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6987   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6988   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6989 }%
6990 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
6991   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
6992   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6993   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
6994 }%
6995 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
6996   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
6997   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
6998   \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
6999 }%
7000 }

```

short-footnote

```
7001 \letabbreviationstyle{short-footnote}{footnote}
```

postfootnote    Similar to the above but the footnote is placed afterwards, outside the link. This avoids nested links and can also move the footnote marker after any following punctuation mark. Pre v1.07 included `\footnote` in the first keys, which was incorrect as it caused duplicate footnotes.

```
7002 \newabbreviationstyle{postfootnote}%
7003 {%
7004   \renewcommand*{\CustomAbbreviationFields}{%
7005     name={\glxstrfootnotename},
7006     sort={\the\glsshorttok},
7007     description={\the\glslongtok},%
7008     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},%
7009     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},%
7010     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
7011   }
```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```
7011   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7012     \csdef{glxstrpostlink\glscategorylabel}{%
7013       \glxtrifwasfirstuse
7014       {%
7015         \glxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
7016           {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7017         }%
7018       }%
7019     }%
7020     \glshasattribute{\the\glslabeltok}{regular}%
7021     {%
7022       \glissetattribute{\the\glslabeltok}{regular}{false}%
7023     }%
7024   }%
7025 }
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
7015     \glxtrdopostpunc{\protect\glsxtrabbrvfootnote{\glslabel}%
7016       {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
7017     }%
7018   }%
7019 }%
7020 \glshasattribute{\the\glslabeltok}{regular}%
7021 {%
7022   \glissetattribute{\the\glslabeltok}{regular}{false}%
7023 }%
7024 {}%
7025 }%
```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```
7026 \renewcommand*{\glxtrsetupfulldefs}{%
7027   \let\glxtrifwasfirstuse\@secondoftwo
7028 }%
7029 }%
7030 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
7031 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7032 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7033 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
```



```

7034 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7035 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7036 \renewcommand*{\glsxtrfullformat}[2]{%
7037   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7038   \ifglsxtrininsertinside\else##2\fi
7039 }%
7040 \renewcommand*{\glsxtrfullplformat}[2]{%
7041   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7042   \ifglsxtrininsertinside\else##2\fi
7043 }%
7044 \renewcommand*{\Glsxtrfullformat}[2]{%
7045   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7046   \ifglsxtrininsertinside\else##2\fi
7047 }%
7048 \renewcommand*{\Glsxtrfullplformat}[2]{%
7049   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7050   \ifglsxtrininsertinside\else##2\fi
7051 }%

```

The first use full form and the inline full form use the short (long) style.

```

7052 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7053   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7054   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7055   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7056 }%
7057 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7058   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7059   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7060   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7061 }%
7062 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7063   \glsfirstabbrvfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
7064   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7065   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7066 }%
7067 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7068   \glsfirstabbrvfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
7069   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7070   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7071 }%
7072 }

```

rt-postfootnote

```

7073 \letabbreviationstyle{short-postfootnote}{postfootnote}

```

shortnolongname

```

7074 \newcommand*{\glsxtrshortnolongname}{%
7075   \protect\glsabbrvfont{\the\glsshorttok}%

```

7076 }

**short** Provide a style that only displays the short form on first use, but the short and long form can be displayed with the “full” commands that use the inline format. If the user supplies a description, the long form won’t be displayed in the predefined glossary styles, but the post description hook can be employed to automatically insert it.

```

7077 \newabbreviationstyle{short}%
7078 {%
7079   \renewcommand*{\CustomAbbreviationFields}{%
7080     name={\glstrshortnolongname},
7081     sort={\the\glsshorttok},
7082     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7083     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7084     text={\protect\glsabbrvfont{\the\glsshorttok}},
7085     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7086     description={\the\glslongtok}}%
7087   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7088     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7089 }%
7090 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7091 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
7092 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7093 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7094 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7095 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

7096 \renewcommand*{\glxtrinlinefullformat}[2]{%
7097   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7098   \ifglxtrininsertinside##2\fi}%
7099   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7100   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7101 }%
7102 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7103   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
7104   \ifglxtrininsertinside##2\fi}%
7105   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7106   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7107 }%
7108 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7109   \protect\glsfirstabbrvfont{\glsaccessshort{##1}}%
7110   \ifglxtrininsertinside##2\fi}%
7111   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7112   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7113 }%
7114 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7115   \protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}%
7116   \ifglxtrininsertinside##2\fi}%

```

```

7117     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7118     \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7119 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

7120 \renewcommand*{\glxtrfullformat}[2]{%
7121     \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7122     \ifglxtrinsertinside\else##2\fi
7123 }%
7124 \renewcommand*{\glxtrfullplformat}[2]{%
7125     \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7126     \ifglxtrinsertinside\else##2\fi
7127 }%
7128 \renewcommand*{\Glsxtrfullformat}[2]{%
7129     \glsfirstabbrvfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7130     \ifglxtrinsertinside\else##2\fi
7131 }%
7132 \renewcommand*{\Glsxtrfullplformat}[2]{%
7133     \glsfirstabbrvfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7134     \ifglxtrinsertinside\else##2\fi
7135 }%
7136 }

```

Set this as the default style for acronyms:

```

7137 \setabbreviationstyle[acronym]{short}

```

short-nolong

```

7138 \letabbreviationstyle{short-nolong}{short}

```

rt-nolong-noreg Like short-nolong but doesn't set the regular attribute.

```

7139 \newabbreviationstyle{short-nolong-noreg}%
7140 {%
7141     \GlsXtrUseAbbrStyleSetup{short-nolong}%

```

Unset the regular attribute if it has been set.

```

7142 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7143     \glshasattribute{\the\glslabeltok}{regular}%
7144     {%
7145         \glissetattribute{\the\glslabeltok}{regular}{false}%
7146     }%
7147     {}%
7148 }%
7149 }%
7150 {%
7151     \GlsXtrUseAbbrStyleFmts{short-nolong}%
7152 }

```

trshortdescname

```

7153 \newcommand*{\glxtrshortdescname}{%
7154   \protect\glsabbrvfont{\the\glsshorttok}%
7155 }

```

short-desc The user must supply the description in this style. The long form is added to the name. The short style (possibly with the post-description hooks set) might be a better option.

```

7156 \newabbreviationstyle{short-desc}%
7157 {%
7158   \renewcommand*{\CustomAbbreviationFields}{%
7159     name={\glxtrshortdescname},
7160     sort={\the\glsshorttok},
7161     first={\protect\glsfirstabbrvfont{\the\glsshorttok}},
7162     firstplural={\protect\glsfirstabbrvfont{\the\glsshortpltok}},
7163     text={\protect\glsabbrvfont{\the\glsshorttok}},
7164     plural={\protect\glsabbrvfont{\the\glsshortpltok}},
7165     description={\the\glslongtok}}%
7166   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7167     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7168 }%
7169 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7170 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%
7171 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvdefaultfont{##1}}%
7172 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7173 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7174 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7175 \renewcommand*{\glxtrinlinefullformat}[2]{%
7176   \glsfirstabbrvfont{\glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
7177   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7178   \glxtrparen{\glsfirstlongfont{\glsaccesslong{##1}}}%
7179 }%
7180 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7181   \glsfirstabbrvfont{\glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
7182   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7183   \glxtrparen{\glsfirstlongfont{\glsaccesslongpl{##1}}}%
7184 }%
7185 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7186   \glsfirstabbrvfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
7187   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7188   \glxtrparen{\glsfirstlongfont{\Glsaccesslong{##1}}}%
7189 }%
7190 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7191   \glsfirstabbrvfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
7192   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7193   \glxtrparen{\glsfirstlongfont{\Glsaccesslongpl{##1}}}%
7194 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

7195 \renewcommand*{\glxtrfullformat}[2]{%
7196   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7197   \ifglxtrinsertinside\else##2\fi
7198 }%
7199 \renewcommand*{\glxtrfullplformat}[2]{%
7200   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7201   \ifglxtrinsertinside\else##2\fi
7202 }%
7203 \renewcommand*{\Glsxtrfullformat}[2]{%
7204   \glsfirstabbrvfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7205   \ifglxtrinsertinside\else##2\fi
7206 }%
7207 \renewcommand*{\Glsxtrfullplformat}[2]{%
7208   \glsfirstabbrvfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7209   \ifglxtrinsertinside\else##2\fi
7210 }%
7211 }

```

ort-nolong-desc

```

7212 \letabbreviationstyle{short-nolong-desc}{short-desc}

```

long-desc-noreg Like short-nolong-desc but doesn't set the regular attribute.

```

7213 \newabbreviationstyle{short-nolong-desc-noreg}%
7214 {%
7215   \GlsXtrUseAbbrStyleSetup{short-nolong-desc}%
7216   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7217     \glshasattribute{\the\glslabeltok}{regular}%
7218     {%
7219       \glsssetattribute{\the\glslabeltok}{regular}{false}%
7220     }%
7221     {}%
7222   }%
7223 }%
7224 {%
7225   \GlsXtrUseAbbrStyleFmts{short-nolong-desc}%
7226 }

```

nolong-short Similar to short-nolong but the full form shows the long form followed by the short form in parentheses.

```

7227 \newabbreviationstyle{nolong-short}%
7228 {%
7229   \GlsXtrUseAbbrStyleSetup{short-nolong}%
7230 }%
7231 {%
7232   \GlsXtrUseAbbrStyleFmts{short-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

7233 \renewcommand*{\glxtrinlinefullformat}[2]{%
7234   \protect\glsfirstlongfont{\glsaccesslong{##1}%
7235     \ifglxtrinsertinside##2\fi}%
7236   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7237   \glxtrparen{\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7238 }%
7239 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7240   \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
7241     \ifglxtrinsertinside##2\fi}%
7242   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7243   \glxtrparen{\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7244 }%
7245 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7246   \protect\glsfirstlongfont{\glsaccesslong{##1}%
7247     \ifglxtrinsertinside##2\fi}%
7248   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7249   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshort{##1}}}%
7250 }%
7251 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7252   \protect\glsfirstlongfont{\glsaccesslongpl{##1}%
7253     \ifglxtrinsertinside##2\fi}%
7254   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7255   \glxtrparen{\glsfirstabbrvfont{\Glsaccessshortpl{##1}}}%
7256 }%
7257 }

```

`long-short-noreg` Like `long-short` but doesn't set the regular attribute.

```

7258 \newabbreviationstyle{long-short-noreg}%
7259 {%
7260   \GlsXtrUseAbbrStyleSetup{long-short}%

```

Unset the regular attribute if it has been set.

```

7261 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7262   \glshasattribute{\the\glslabeltok}{regular}%
7263   {%
7264     \glssetattribute{\the\glslabeltok}{regular}{false}%
7265   }%
7266   {}%
7267 }%
7268 }%
7269 {%
7270   \GlsXtrUseAbbrStyleFmts{long-short}%
7271 }

```

`noshortdescname`

```

7272 \newcommand*{\glxtrlongnoshortdescname}{%
7273   \protect\glslongfont{\the\glslongtok}%
7274 }

```

long-desc Provide a style that only displays the long form, but the long and short form can be displayed with the “full” commands that use the inline format. The predefined glossary styles won’t show the short form. The user must supply a description for this style.

```

7275 \newabbreviationstyle{long-desc}%
7276 {%
7277   \renewcommand*{\CustomAbbreviationFields}{%
7278     name={\glstrlongnoshortdescname},
7279     sort={\the\glslongtok},
7280     first={\protect\glsfirstlongfont{\the\glslongtok}},
7281     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7282     text={\glslongfont{\the\glslongtok}},
7283     plural={\glslongfont{\the\glslongpltok}}}%
7284   }%
7285   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7286     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7287 }%
7288 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

7289 \renewcommand*{\abbrvpluralsuffix}{\glstrabbrvpluralsuffix}%
7290 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvdefaultfont{##1}}%
7291 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvdefaultfont{##1}}%
7292 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7293 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7294 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
7295   \glslongfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7296   \ifglsxtrininsertinside \else##2\fi
7297 }%
7298 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
7299   \glslongfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7300   \ifglsxtrininsertinside \else##2\fi
7301 }%
7302 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7303   \glslongfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
7304   \ifglsxtrininsertinside \else##2\fi
7305 }%
7306 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7307   \glslongfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
7308   \ifglsxtrininsertinside \else##2\fi
7309 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7310 \renewcommand*{\glsxtrinlinefullformat}[2]{%
7311   \glsfirstlongfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
7312   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
7313   \glsxtrparen{\protect\glsfirstabbrvfont{\glssaccessshort{##1}}}%
7314 }%
7315 \renewcommand*{\glsxtrinlinefullplformat}[2]{%

```

```

7316 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7317 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7318 \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7319 }%
7320 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7321 \glsfirstlongfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7322 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7323 \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
7324 }%
7325 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7326 \glsfirstlongfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7327 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7328 \glxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
7329 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

7330 \renewcommand*{\glxtrfullformat}[2]{%
7331 \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7332 \ifglxtrinsertinside\else##2\fi
7333 }%
7334 \renewcommand*{\glxtrfullplformat}[2]{%
7335 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7336 \ifglxtrinsertinside\else##2\fi
7337 }%
7338 \renewcommand*{\Glsxtrfullformat}[2]{%
7339 \glsfirstlongfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7340 \ifglxtrinsertinside\else##2\fi
7341 }%
7342 \renewcommand*{\Glsxtrfullplformat}[2]{%
7343 \glsfirstlongfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7344 \ifglxtrinsertinside\else##2\fi
7345 }%
7346 }

```

`ng-noshort-desc` Provide a synonym that matches similar styles.

```
7347 \letabbreviationstyle{long-noshort-desc}{long-desc}
```

`short-desc-noreg` Like `long-noshort-desc` but doesn't set the regular attribute.

```

7348 \newabbreviationstyle{long-noshort-desc-noreg}%
7349 {%
7350 \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%

```

Unset the regular attribute if it has been set.

```

7351 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7352 \glshasattribute{\the\glslabeltok}{regular}%
7353 {%
7354 \glssetattribute{\the\glslabeltok}{regular}{false}%
7355 }%
7356 {}%

```



```

7357 }%
7358 }%
7359 {%
7360   \GlsXtrUseAbbrStyleFmts{long-noshort-desc}%
7361 }

```

longnoshortname

```

7362 \newcommand*{\glxtrlongnoshortname}{%
7363   \protect\glssabrvfont{\the\glsshorttok}%
7364 }

```

long It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

7365 \newabbreviationstyle{long}%
7366 {%
7367   \renewcommand*{\CustomAbbreviationFields}{%
7368     name={\glxtrlongnoshortname},
7369     sort={\the\glsshorttok},
7370     first={\protect\glsfirstlongfont{\the\glslongtok}},
7371     firstplural={\protect\glsfirstlongfont{\the\glslongpltok}},
7372     text={\glslongfont{\the\glslongtok}},
7373     plural={\glslongfont{\the\glslongpltok}},%
7374     description={\the\glslongtok}%
7375   }%
7376   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7377     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
7378 }%
7379 {%
7380   \GlsXtrUseAbbrStyleFmts{long-desc}%
7381 }

```

long-noshort Provide a synonym that matches similar styles.

```

7382 \letabbreviationstyle{long-noshort}{long}

```

g-noshort-noreg Like long-noshort but doesn't set the regular attribute.

```

7383 \newabbreviationstyle{long-noshort-noreg}%
7384 {%
7385   \GlsXtrUseAbbrStyleSetup{long-noshort}%
7386   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7387     \glshasattribute{\the\glslabeltok}{regular}%
7388     {%
7389       \glsssetAttribute{\the\glslabeltok}{regular}{false}%
7390     }%
7391     {}%
7392   }%
7393 }%

```

```

7394 {%
7395   \GlsXtrUseAbbrStyleFmts{long-noshort}%
7396 }

```

### 1.7.3 Predefined Styles (Small Capitals)

These styles use `\textsc` for the short form.

```

\glxtrscfont   Maintained for backward-compatibility.
7397 \newcommand*{\glxtrscfont}[1]{\textsc{#1}}

\glsabbrvscfont Added for consistent naming.
7398 \newcommand*{\glsabbrvscfont}{\glxtrscfont}

sxtrfirstscfont Maintained for backward-compatibility.
7399 \newcommand*{\glsxtrfirstscfont}[1]{\glsabbrvscfont{#1}}

irstabbrvscfont Added for consistent naming.
7400 \newcommand*{\glfirstabbrvscfont}{\glsxtrfirstscfont}

```

and for the default short form suffix:

```

\glxtrscsuffix
7401 \newcommand*{\glxtrscsuffix}{\glstextup{\glxtrabbrvpluralsuffix}}

long-short-sc
7402 \newabbreviationstyle{long-short-sc}%
7403 {%
7404   \renewcommand*{\CustomAbbreviationFields}{%
7405     name={\glxtrlongshortname},
7406     sort={\the\glsshorttok},
7407     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7408       \protect\glxtrfullsep{\the\glslabeltok}%
7409       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7410     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7411       \protect\glxtrfullsep{\the\glslabeltok}%
7412       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7413     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}},%
7414     description={\the\glslongtok}}%
7415   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7416     \glshasattribute{\the\glslabeltok}{regular}%
7417     {%
7418       \glissetattribute{\the\glslabeltok}{regular}{false}%
7419     }%
7420   }%
7421 }%
7422 }%
7423 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7424 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7425 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7426 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
```

Use the default long fonts.

```
7427 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7428 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
7429 \renewcommand*{\glxtrfullformat}[2]{%
7430   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7431   \ifglxtrinsertinside\else##2\fi
7432   \glxtrfullsep{##1}%
7433   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7434 }%
7435 \renewcommand*{\glxtrfullplformat}[2]{%
7436   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7437   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7438   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7439 }%
7440 \renewcommand*{\Glsxtrfullformat}[2]{%
7441   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7442   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7443   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7444 }%
7445 \renewcommand*{\Glsxtrfullplformat}[2]{%
7446   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7447   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7448   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7449 }%
7450 }
```

g-short-sc-desc

```
7451 \newabbreviationstyle{long-short-sc-desc}%
7452 {%
7453   \renewcommand*{\CustomAbbreviationFields}{%
7454     name={\glxtrlongshortdescname},
7455     sort={\glxtrlongshortdescsort},%
7456     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
7457       \protect\glxtrfullsep{\the\glslabeltok}%
7458       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshorttok}}},%
7459     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
7460       \protect\glxtrfullsep{\the\glslabeltok}%
7461       \glxtrparen{\protect\glsfirstabbrvscfont{\the\glsshortpltok}}},%
7462     text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7463     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}%
7464   }%
```

Unset the regular attribute if it has been set.

```

7465 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7466   \glshasattribute{\the\glslabeltok}{regular}%
7467   {%
7468     \glissetattribute{\the\glslabeltok}{regular}{false}%
7469   }%
7470   {}%
7471 }%
7472 }%
7473 {%

```

As long-short-sc style:

```

7474 \GlsXtrUseAbbrStyleFmts{long-short-sc}%
7475 }

```

Now the short (long) version

```

7476 \newabbreviationstyle{short-sc-long}%
7477 {%
7478   \renewcommand*{\CustomAbbreviationFields}{%
7479     name={\glxtrshortlongname},
7480     sort={\the\glsshorttok},
7481     description={\the\glslongtok},%
7482     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7483       \protect\glxtrfullsep{\the\glslabeltok}%
7484       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7485     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7486       \protect\glxtrfullsep{\the\glslabeltok}%
7487       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7488     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}

```

Unset the regular attribute if it has been set.

```

7489 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7490   \glshasattribute{\the\glslabeltok}{regular}%
7491   {%
7492     \glissetattribute{\the\glslabeltok}{regular}{false}%
7493   }%
7494   {}%
7495 }%
7496 }%
7497 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7498 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7499 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7500 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7501 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7502 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

7503 \renewcommand*{\glxtrfullformat}[2]{%
7504   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7505   \ifglxtrininsertinside\else##2\fi

```

```

7506 \glxtrfullsep{##1}%
7507 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7508 }%
7509 \renewcommand*{\glxtrfullplformat}[2]{%
7510 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7511 \ifglxtrininsertinside\else##2\fi
7512 \glxtrfullsep{##1}%
7513 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7514 }%
7515 \renewcommand*{\Glsxtrfullformat}[2]{%
7516 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7517 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7518 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7519 }%
7520 \renewcommand*{\Glsxtrfullplformat}[2]{%
7521 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7522 \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7523 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7524 }%
7525 }

```

As before but user provides description

```

7526 \newabbreviationstyle{short-sc-long-desc}%
7527 {%
7528 \renewcommand*{\CustomAbbreviationFields}{%
7529 name={\glxtrshortlongdescname},
7530 sort={\glxtrshortlongdescsort},
7531 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}%
7532 \protect\glxtrfullsep{\the\glslabeltok}%
7533 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
7534 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7535 \protect\glxtrfullsep{\the\glslabeltok}%
7536 \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
7537 text={\protect\glsabbrvscfont{\the\glsshorttok}},%
7538 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7539 }%

```

Unset the regular attribute if it has been set.

```

7540 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7541 \glshasattribute{\the\glslabeltok}{regular}%
7542 {%
7543 \glissetattribute{\the\glslabeltok}{regular}{false}%
7544 }%
7545 {}%
7546 }%
7547 }%
7548 {%

```

As short-sc-long style:

```

7549 \GlsXtrUseAbbrStyleFmts{short-sc-long}%

```

7550 }

short-sc

```
7551 \newabbreviationstyle{short-sc}%
7552 {%
7553   \renewcommand*{\CustomAbbreviationFields}{%
7554     name={\glxtrshortnolongname},
7555     sort={\the\glsshorttok},
7556     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7557     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7558     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7559     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7560     description={\the\glslongtok}}%
7561   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7562     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7563 }%
7564 {%
```

Use smallcaps and adjust the plural suffix to revert to upright.

```
7565 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7566 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7567 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7568 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7569 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The inline full form displays the short form followed by the long form in parentheses.

```
7570 \renewcommand*{\glxtrinlinefullformat}[2]{%
7571   \protect\glsfirstabbrvscfont{\glsaccessshort{##1}}%
7572   \ifglxtrininsertinside##2\fi}%
7573   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7574   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
7575 }%
7576 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7577   \protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}%
7578   \ifglxtrininsertinside##2\fi}%
7579   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7580   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
7581 }%
7582 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7583   \protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}%
7584   \ifglxtrininsertinside##2\fi}%
7585   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7586   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
7587 }%
7588 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7589   \protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}%
7590   \ifglxtrininsertinside##2\fi}%
7591   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7592   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
7593 }%
```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

7594 \renewcommand*{\glxstrfullformat}[2]{%
7595   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7596   \ifglxtrinsertinside\else##2\fi
7597 }%
7598 \renewcommand*{\glxstrfullplformat}[2]{%
7599   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7600   \ifglxtrinsertinside\else##2\fi
7601 }%
7602 \renewcommand*{\Glsxtrfullformat}[2]{%
7603   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7604   \ifglxtrinsertinside\else##2\fi
7605 }%
7606 \renewcommand*{\Glsxtrfullplformat}[2]{%
7607   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7608   \ifglxtrinsertinside\else##2\fi
7609 }%
7610 }

```

short-sc-nolong

```

7611 \letabbreviationstyle{short-sc-nolong}{short-sc}

```

short-sc-desc

```

7612 \newabbreviationstyle{short-sc-desc}{%
7613 {%
7614   \renewcommand*{\CustomAbbreviationFields}{%
7615     name={\glxstrshortdescname},
7616     sort={\the\glsshorttok},
7617     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},
7618     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},
7619     text={\protect\glsabbrvscfont{\the\glsshorttok}},
7620     plural={\protect\glsabbrvscfont{\the\glsshortpltok}},
7621     description={\the\glslongtok}}%
7622   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7623     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7624 }%
7625 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7626 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7627 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7628 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7629 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7630 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

7631 \renewcommand*{\glxtrinlinefullformat}[2]{%
7632   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7633   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7634 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7635 }%
7636 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7637 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7638 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7639 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7640 }%
7641 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7642 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7643 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7644 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
7645 }%
7646 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7647 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7648 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7649 \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
7650 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

7651 \renewcommand*{\glxtrfullformat}[2]{%
7652 \glsfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7653 \ifglxtrinsertinside\else##2\fi
7654 }%
7655 \renewcommand*{\glxtrfullplformat}[2]{%
7656 \glsfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7657 \ifglxtrinsertinside\else##2\fi
7658 }%
7659 \renewcommand*{\Glsxtrfullformat}[2]{%
7660 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7661 \ifglxtrinsertinside\else##2\fi
7662 }%
7663 \renewcommand*{\Glsxtrfullplformat}[2]{%
7664 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7665 \ifglxtrinsertinside\else##2\fi
7666 }%
7667 }

```

-sc-nolong-desc

```
7668 \letabbreviationstyle{short-sc-nolong-desc}{short-sc-desc}
```

nolong-short-sc

```

7669 \newabbreviationstyle{nolong-short-sc}%
7670 {%
7671 \GlsXtrUseAbbrStyleSetup{short-sc-nolong}%
7672 }%
7673 {%
7674 \GlsXtrUseAbbrStyleFmts{short-sc-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.



```

7675 \renewcommand*{\glxtrinlinefullformat}[2]{%
7676   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
7677     \ifglxtrinsertinside##2\fi}%
7678   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7679   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7680 }%
7681 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7682   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
7683     \ifglxtrinsertinside##2\fi}%
7684   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7685   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7686 }%
7687 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7688   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
7689     \ifglxtrinsertinside##2\fi}%
7690   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7691   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7692 }%
7693 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7694   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
7695     \ifglxtrinsertinside##2\fi}%
7696   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7697   \glxtrparen{\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7698 }%
7699 }

```

long-noshort-sc The smallcaps font will only be used if the short form is explicitly invoked through commands like `\glxtrshort`.

```

7700 \newabbreviationstyle{long-noshort-sc}%
7701 {%
7702   \renewcommand*{\CustomAbbreviationFields}{%
7703     name={\glxtrlongnoshortname},
7704     sort={\the\glsshorttok},
7705     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
7706     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
7707     text={\protect\glslongdefaultfont{\the\glslongtok}},
7708     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
7709     description={\the\glslongtok}%
7710   }%
7711   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7712     \glssetattribute{\the\glslabeltok}{regular}{true}}%
7713 }%
7714 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7715 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7716 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7717 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7718 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7719 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7720 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7721   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7722   \ifglxtrinsertinside \else##2\fi
7723 }%
7724 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7725   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7726   \ifglxtrinsertinside \else##2\fi
7727 }%
7728 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7729   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7730   \ifglxtrinsertinside \else##2\fi
7731 }%
7732 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7733   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7734   \ifglxtrinsertinside \else##2\fi
7735 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7736 \renewcommand*{\glxtrinlinefullformat}[2]{%
7737   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7738   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7739   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7740 }%
7741 \renewcommand*{\glxtrinlinefullplformat}[2]{%
7742   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7743   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7744   \glxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7745 }%
7746 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7747   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7748   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7749   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshort{##1}}}%
7750 }%
7751 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7752   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7753   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7754   \glxtrparen{\protect\glsfirstabbrvscfont{\Glsaccessshortpl{##1}}}%
7755 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

7756 \renewcommand*{\glxtrfullformat}[2]{%
7757   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7758   \ifglxtrinsertinside\else##2\fi
7759 }%
7760 \renewcommand*{\glxtrfullplformat}[2]{%
7761   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7762   \ifglxtrinsertinside\else##2\fi
7763 }%

```

```

7764 \renewcommand*{\Glsxtrfullformat}[2]{%
7765   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7766   \ifglxtrinsertinside\else##2\fi
7767 }%
7768 \renewcommand*{\Glsxtrfullplformat}[2]{%
7769   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
7770   \ifglxtrinsertinside\else##2\fi
7771 }%
7772 }

```

long-sc Backward compatibility:

```

7773 \@glxtr@deprecated@abbrstyle{long-sc}{long-noshort-sc}

```

noshort-sc-desc The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```

7774 \newabbreviationstyle{long-noshort-sc-desc}%
7775 {%
7776   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
7777 }%
7778 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7779 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7780 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7781 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7782 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
7783 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

7784 \renewcommand*{\glxtrsubsequentfmt}[2]{%
7785   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7786   \ifglxtrinsertinside \else##2\fi
7787 }%
7788 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
7789   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7790   \ifglxtrinsertinside \else##2\fi
7791 }%
7792 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
7793   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
7794   \ifglxtrinsertinside \else##2\fi
7795 }%
7796 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
7797   \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
7798   \ifglxtrinsertinside \else##2\fi
7799 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

7800 \renewcommand*{\glxtrinlinelinefullformat}[2]{%
7801   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
7802   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%

```

```

7803 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7804 }%
7805 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
7806 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7807 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7808 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7809 }%
7810 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7811 \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7812 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7813 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshort{##1}}}%
7814 }%
7815 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7816 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7817 \ifglsxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
7818 \glsxtrparen{\protect\glsfirstabbrvscfont{\glsaccessshortpl{##1}}}%
7819 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

7820 \renewcommand*{\glsxtrfullformat}[2]{%
7821 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7822 \ifglsxtrinsertinside\else##2\fi
7823 }%
7824 \renewcommand*{\glsxtrfullplformat}[2]{%
7825 \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7826 \ifglsxtrinsertinside\else##2\fi
7827 }%
7828 \renewcommand*{\Glsxtrfullformat}[2]{%
7829 \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
7830 \ifglsxtrinsertinside\else##2\fi
7831 }%
7832 \renewcommand*{\Glsxtrfullplformat}[2]{%
7833 \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
7834 \ifglsxtrinsertinside\else##2\fi
7835 }%
7836 }

```

long-desc-sc Backward compatibility:

```

7837 \@glsxtr@deprecated@abbrstyle{long-desc-sc}{long-noshort-sc-desc}

```

ort-sc-footnote

```

7838 \newabbreviationstyle{short-sc-footnote}%
7839 {%
7840 \renewcommand*{\CustomAbbreviationFields}{%
7841 name={\glsxtrfootnotename},
7842 sort={\the\glsshorttok},
7843 description={\the\glslongtok},%
7844 first={\protect\glsfirstabbrvscfont{\the\glsshorttok}}%

```

```

7845 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7846 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
7847 firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}%
7848 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
7849 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
7850 plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%

```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```

7851 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
7852 \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
7853 \glsattribute{\the\glslabeltok}{regular}%
7854 {%
7855 \glssetattribute{\the\glslabeltok}{regular}{false}%
7856 }%
7857 {}%
7858 }%
7859 }%
7860 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7861 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7862 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvscfont{##1}}%
7863 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvscfont{##1}}%
7864 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
7865 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

7866 \renewcommand*{\glxtrfullformat}[2]{%
7867 \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7868 \ifglxtrinsertinside\else##2\fi
7869 \protect\glxtrabbrvfootnote{##1}%
7870 {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7871 }%
7872 \renewcommand*{\glxtrfullplformat}[2]{%
7873 \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7874 \ifglxtrinsertinside\else##2\fi
7875 \protect\glxtrabbrvfootnote{##1}%
7876 {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7877 }%
7878 \renewcommand*{\Glsxtrfullformat}[2]{%
7879 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7880 \ifglxtrinsertinside\else##2\fi
7881 \protect\glxtrabbrvfootnote{##1}%
7882 {\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7883 }%
7884 \renewcommand*{\Glsxtrfullplformat}[2]{%
7885 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7886 \ifglxtrinsertinside\else##2\fi
7887 \protect\glxtrabbrvfootnote{##1}%
7888 {\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%

```

7889 }%

The first use full form and the inline full form use the short (long) style.

```
7890 \renewcommand*\glxstrinlinefullformat}[2]{%
7891   \glsfirstabbrvscfont{\glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
7892   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7893   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
7894 }%
7895 \renewcommand*\glxstrinlinefullplformat}[2]{%
7896   \glsfirstabbrvscfont{\glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
7897   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7898   \glxstrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
7899 }%
7900 \renewcommand*\Glsxtrinlinefullformat}[2]{%
7901   \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxstrinsertinside##2\fi}%
7902   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7903   \glxstrparen{\glsfirstlongfootnotefont{\Glsaccesslong{##1}}}%
7904 }%
7905 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
7906   \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxstrinsertinside##2\fi}%
7907   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
7908   \glxstrparen{\glsfirstlongfootnotefont{\Glsaccesslongpl{##1}}}%
7909 }%
7910 }
```

footnote-sc Backward compatibility:

```
7911 \@glxstr@deprecated@abbrstyle{footnote-sc}{short-sc-footnote}
```

sc-postfootnote

```
7912 \newabbreviationstyle{short-sc-postfootnote}%
7913 {%
7914   \renewcommand*\CustomAbbreviationFields{%
7915     name={\glxstrfootnotename},
7916     sort={\the\glsshorttok},
7917     description={\the\glslongtok},%
7918     first={\protect\glsfirstabbrvscfont{\the\glsshorttok}},%
7919     firstplural={\protect\glsfirstabbrvscfont{\the\glsshortpltok}},%
7920     plural={\protect\glsabbrvscfont{\the\glsshortpltok}}}%
7921 }
```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```
7921 \renewcommand*\GlsXtrPostNewAbbreviation{%
7922   \csdef{glxstrpostlink\glscategorylabel}{%
7923     \glxstrifwasfirstuse
7924     {%
```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```
7925     \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}}%
7926     {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}%
7927   }
```

```

7927     }%
7928     {}%
7929     }%
7930     \glshasattribute{\the\glslabeltok}{regular}%
7931     {%
7932         \glissetattribute{\the\glslabeltok}{regular}{false}%
7933     }%
7934     {}%
7935     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

7936 \renewcommand*{\glxtrsetupfulldefs}{%
7937     \let\glxtrifwasfirstuse\@secondoftwo
7938 }%
7939 }%
7940 {%

```

Use smallcaps and adjust the plural suffix to revert to upright.

```

7941 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrscsuffix}%
7942 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvscfont{##1}}%
7943 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvscfont{##1}}%
7944 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongfootnotefont{##1}}%
7945 \renewcommand*{\glsslongfont}[1]{\glsslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

7946 \renewcommand*{\glxtrfullformat}[2]{%
7947     \glssfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7948     \ifglxtrininsertinside\else##2\fi
7949 }%
7950 \renewcommand*{\glxtrfullplformat}[2]{%
7951     \glssfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7952     \ifglxtrininsertinside\else##2\fi
7953 }%
7954 \renewcommand*{\Glsxtrfullformat}[2]{%
7955     \glssfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7956     \ifglxtrininsertinside\else##2\fi
7957 }%
7958 \renewcommand*{\Glsxtrfullplformat}[2]{%
7959     \glssfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
7960     \ifglxtrininsertinside\else##2\fi
7961 }%

```

The first use full form and the inline full form use the short (long) style.

```

7962 \renewcommand*{\glxtrininlinefullformat}[2]{%
7963     \glssfirstabbrvscfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
7964     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
7965     \glxtrparen{\glssfirstlongfootnotefont{\glssaccesslong{##1}}}%
7966 }%
7967 \renewcommand*{\glxtrininlinefullplformat}[2]{%
7968     \glssfirstabbrvscfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

7969 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7970 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7971 }%
7972 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
7973 \glsfirstabbrvscfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
7974 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7975 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
7976 }%
7977 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
7978 \glsfirstabbrvscfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
7979 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
7980 \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
7981 }%
7982 }

```

postfootnote-sc Backward compatibility:

```

7983 \@glxtr@deprecated@abbrstyle{postfootnote-sc}{short-sc-postfootnote}

```

### 1.7.4 Predefined Styles (Fake Small Capitals)

These styles require the relsize package, which must be loaded by the user. These styles all use:

\glxtrsmfont Maintained for backward compatibility.

```

7984 \newcommand*{\glxtrsmfont}[1]{\textsmaller{#1}}

```

\glssabbrvsmfont Added for consistent naming.

```

7985 \newcommand*{\glssabbrvsmfont}{\glxtrsmfont}

```

glxtrfirstsmfont Maintained for backward compatibility.

```

7986 \newcommand*{\glxtrfirstsmfont}[1]{\glssabbrvsmfont{#1}}

```

glssfirstabbrvsmfont Added for consistent naming.

```

7987 \newcommand*{\glssfirstabbrvsmfont}{\glxtrfirstsmfont}

```

and for the default short form suffix:

\glxtrsmssuffix

```

7988 \newcommand*{\glxtrsmssuffix}{\glssabbrvpluralsuffix}

```

long-short-sm

```

7989 \newabbreviationstyle{long-short-sm}%
7990 {%
7991 \renewcommand*{\CustomAbbreviationFields}{%
7992 name={\glsslongshortname},
7993 sort={\the\glssshorttok},
7994 first={\protect\glsfirstlongdefaultfont{\the\glsslongtok}%
7995 \protect\glssfullsep{\the\glsslabeltok}%

```



```

7996 \glxtrparen{\protect\glfirstabbrvsmfont{\the\glsshorttok}}},%
7997 firstplural={\protect\glfirstlongdefaultfont{\the\glslongpltok}}%
7998 \protect\glxtrfullsep{\the\glslabeltok}%
7999 \glxtrparen{\protect\glfirstabbrvsmfont{\the\glsshortpltok}}},%
8000 plural={\protect\glabbrvsmfont{\the\glsshortpltok}}},%
8001 description={\the\glslongtok}}%
8002 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8003 \glshasattribute{\the\glslabeltok}{regular}}%
8004 {%
8005 \glissetattribute{\the\glslabeltok}{regular}{false}}%
8006 }%
8007 {}%
8008 }%
8009 }%
8010 {%
8011 \renewcommand*\glabbrvfont[1]{\glabbrvsmfont{##1}}%
8012 \renewcommand*\glfirstabbrvfont[1]{\glfirstabbrvsmfont{##1}}%
8013 \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%

```

Use the default long fonts.

```

8014 \renewcommand*\glfirstlongfont[1]{\glfirstlongdefaultfont{##1}}%
8015 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8016 \renewcommand*\glxtrfullformat[2]{%
8017 \glfirstlongdefaultfont{\glaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8018 \ifglxtrinsertinside\else##2\fi
8019 \glxtrfullsep{##1}%
8020 \glxtrparen{\glfirstabbrvsmfont{\glaccessshort{##1}}}%
8021 }%
8022 \renewcommand*\glxtrfullplformat[2]{%
8023 \glfirstlongdefaultfont{\glaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8024 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8025 \glxtrparen{\glfirstabbrvsmfont{\glaccessshortpl{##1}}}%
8026 }%
8027 \renewcommand*\Glsxtrfullformat[2]{%
8028 \glfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8029 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8030 \glxtrparen{\glfirstabbrvsmfont{\glaccessshort{##1}}}%
8031 }%
8032 \renewcommand*\Glsxtrfullplformat[2]{%
8033 \glfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8034 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8035 \glxtrparen{\glfirstabbrvsmfont{\glaccessshortpl{##1}}}%
8036 }%
8037 }

```

g-short-sm-desc

```

8038 \newabbreviationstyle{long-short-sm-desc}%
8039 {%

```

```

8040 \renewcommand*{\CustomAbbreviationFields}{%
8041   name={\glxtrlongshortdescname},
8042   sort={\glxtrlongshortdescsort},%
8043   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8044     \protect\glxtrfullsep{\the\glslabeltok}%
8045     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshorttok}}},%
8046   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8047     \protect\glxtrfullsep{\the\glslabeltok}%
8048     \glxtrparen{\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}},%
8049   text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
8050   plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
8051 }%

```

Unset the regular attribute if it has been set.

```

8052 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8053   \glshasattribute{\the\glslabeltok}{regular}%
8054   {%
8055     \glissetattribute{\the\glslabeltok}{regular}{false}%
8056   }%
8057   {}%
8058 }%
8059 }%
8060 {%

```

As long-short-sm style:

```

8061 \GlsXtrUseAbbrStyleFmts{long-short-sm}%
8062 }

```

short-sm-long Now the short (long) version

```

8063 \newabbreviationstyle{short-sm-long}%
8064 {%
8065   \renewcommand*{\CustomAbbreviationFields}{%
8066     name={\glxtrshortlongname},
8067     sort={\the\glsshorttok},
8068     description={\the\glslongtok},%
8069     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8070       \protect\glxtrfullsep{\the\glslabeltok}%
8071       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8072     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8073       \protect\glxtrfullsep{\the\glslabeltok}%
8074       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8075     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8076 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8077   \glshasattribute{\the\glslabeltok}{regular}%
8078   {%
8079     \glissetattribute{\the\glslabeltok}{regular}{false}%
8080   }%
8081   {}%

```

```

8082 }%
8083 }%
8084 {%
8085 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8086 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8087 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8088 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8089 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8090 \renewcommand*\glsxtrfullformat[2]{%
8091 \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8092 \ifglsxtrininsertinside\else##2\fi
8093 \glsxtrfullsep{##1}}%
8094 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8095 }%
8096 \renewcommand*\glsxtrfullplformat[2]{%
8097 \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8098 \ifglsxtrininsertinside\else##2\fi
8099 \glsxtrfullsep{##1}}%
8100 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8101 }%
8102 \renewcommand*\Glsxtrfullformat[2]{%
8103 \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8104 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8105 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8106 }%
8107 \renewcommand*\Glsxtrfullplformat[2]{%
8108 \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8109 \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}}%
8110 \glsxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8111 }%
8112 }

```

rt-sm-long-desc As before but user provides description

```

8113 \newabbreviationstyle{short-sm-long-desc}%
8114 {%
8115 \renewcommand*\CustomAbbreviationFields{%
8116 name={\glsxtrshortlongdescname},
8117 sort={\glsxtrshortlongdescsort},
8118 first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}}%
8119 \protect\glsxtrfullsep{\the\glslabeltok}}%
8120 \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8121 firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}}%
8122 \protect\glsxtrfullsep{\the\glslabeltok}}%
8123 \glsxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8124 text={\protect\glsabbrvsmfont{\the\glsshorttok}},%
8125 plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
8126 }%

```

Unset the regular attribute if it has been set.

```

8127 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8128   \glshasattribute{\the\glslabeltok}{regular}%
8129   {%
8130     \glissetattribute{\the\glslabeltok}{regular}{false}%
8131   }%
8132   {}%
8133 }%
8134 }%
8135 {%

```

As short-sm-long style:

```

8136 \GlsXtrUseAbbrStyleFmts{short-sm-long}%
8137 }

```

short-sm

```

8138 \newabbreviationstyle{short-sm}%
8139 {%
8140   \renewcommand*{\CustomAbbreviationFields}{%
8141     name={\glxtrshortnolongname},
8142     sort={\the\glsshorttok},
8143     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8144     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8145     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8146     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8147     description={\the\glslongtok}}%
8148   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8149     \glissetattribute{\the\glslabeltok}{regular}{true}}%
8150 }%
8151 {%
8152   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8153   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8154   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8155   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8156   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8157 \renewcommand*{\glxtrinlinefullformat}[2]{%
8158   \protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}%
8159   \ifglxtrininsertinside##2\fi}%
8160   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8161   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8162 }%
8163 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8164   \protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}%
8165   \ifglxtrininsertinside##2\fi}%
8166   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8167   \glxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8168 }%

```

```

8169 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8170   \protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}%
8171   \ifglxtrinsertinside##2\fi}%
8172   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8173   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslong{##1}}}%
8174 }%
8175 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8176   \protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}%
8177   \ifglxtrinsertinside##2\fi}%
8178   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8179   \glxtrparen{\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}}}%
8180 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

8181 \renewcommand*{\glxtrfullformat}[2]{%
8182   \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglxtrinsertinside##2\fi}%
8183   \ifglxtrinsertinside\else##2\fi
8184 }%
8185 \renewcommand*{\glxtrfullplformat}[2]{%
8186   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglxtrinsertinside##2\fi}%
8187   \ifglxtrinsertinside\else##2\fi
8188 }%
8189 \renewcommand*{\Glsxtrfullformat}[2]{%
8190   \glsfirstabbrvsmfont{\Glsaccessshort{##1}}\ifglxtrinsertinside##2\fi}%
8191   \ifglxtrinsertinside\else##2\fi
8192 }%
8193 \renewcommand*{\Glsxtrfullplformat}[2]{%
8194   \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}\ifglxtrinsertinside##2\fi}%
8195   \ifglxtrinsertinside\else##2\fi
8196 }%
8197 }

```

short-sm-nolong

```

8198 \letabbreviationstyle{short-sm-nolong}{short-sm}

```

short-sm-desc

```

8199 \newabbreviationstyle{short-sm-desc}{%
8200 {%
8201   \renewcommand*{\CustomAbbreviationFields}{%
8202     name={\glxtrshortdesname},
8203     sort={\the\glsshorttok},
8204     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},
8205     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},
8206     text={\protect\glsabbrvsmfont{\the\glsshorttok}},
8207     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}},
8208     description={\the\glslongtok}}%
8209 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8210   \glssetattribute{\the\glslabeltok}{regular}{true}}%

```

```

8211 }%
8212 {%
8213   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8214   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8215   \renewcommand*\abbrvpluralsuffix{\protect\glsxtrmsuffix}%
8216   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8217   \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8218   \renewcommand*\glsxtrinlinefullformat[2]{%
8219     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8220     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8221     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8222   }%
8223   \renewcommand*\glsxtrinlinefullplformat[2]{%
8224     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8225     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8226     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8227   }%
8228   \renewcommand*\Glsxtrinlinefullformat[2]{%
8229     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8230     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8231     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8232   }%
8233   \renewcommand*\Glsxtrinlinefullplformat[2]{%
8234     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8235     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8236     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8237   }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

8238   \renewcommand*\glsxtrfullformat[2]{%
8239     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8240     \ifglsxtrininsertinside\else##2\fi
8241   }%
8242   \renewcommand*\glsxtrfullplformat[2]{%
8243     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8244     \ifglsxtrininsertinside\else##2\fi
8245   }%
8246   \renewcommand*\Glsxtrfullformat[2]{%
8247     \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8248     \ifglsxtrininsertinside\else##2\fi
8249   }%
8250   \renewcommand*\Glsxtrfullplformat[2]{%
8251     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8252     \ifglsxtrininsertinside\else##2\fi
8253   }%
8254 }

```

-sm-nolong-desc

```
8255 \letabbreviationstyle{short-sm-nolong-desc}{short-sm-desc}
```

nolong-short-sm

```
8256 \newabbreviationstyle{nolong-short-sm}%
8257 {%
8258   \GlsXtrUseAbbrStyleSetup{short-sm-nolong}%
8259 }%
8260 {%
8261   \GlsXtrUseAbbrStyleFmts{short-sm-nolong}%
```

The inline full form displays the long form followed by the short form in parentheses.

```
8262 \renewcommand*{\glxtrinlinefullformat}[2]{%
8263   \protect\glsfirstlongdefaultfont{\glsaccesslong{##1}%
8264     \ifglxtrininsertinside##2\fi}%
8265   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8266   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8267 }%
8268 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8269   \protect\glsfirstlongdefaultfont{\glsaccesslongpl{##1}%
8270     \ifglxtrininsertinside##2\fi}%
8271   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8272   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8273 }%
8274 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8275   \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%
8276     \ifglxtrininsertinside##2\fi}%
8277   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8278   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8279 }%
8280 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8281   \protect\glsfirstlongdefaultfont{\Glsaccesslongpl{##1}%
8282     \ifglxtrininsertinside##2\fi}%
8283   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8284   \glxtrparen{\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8285 }%
8286 }
```

long-noshort-sm The smallcaps font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
8287 \newabbreviationstyle{long-noshort-sm}%
8288 {%
8289   \renewcommand*{\CustomAbbreviationFields}{%
8290     name={\glxtrlongnoshortname},
8291     sort={\the\glsshorttok},
8292     first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
8293     firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
8294     text={\protect\glslongdefaultfont{\the\glslongtok}},
8295     plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
```

```

8296     description={\the\glslongtok}%
8297 }%
8298 \renewcommand*\GlsXtrPostNewAbbreviation{%
8299     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8300 }%
8301 {%
8302 \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8303 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8304 \renewcommand*\abbrvpluralsuffix{\protect\glsxtrsmsuffix}%
8305 \renewcommand*\glsfirstlongfont[1]{\glsfirstlongdefaultfont{##1}}%
8306 \renewcommand*\glslongfont[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

8307 \renewcommand*\glsxtrsubsequentfmt}[2]{%
8308     \glslongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8309     \ifglsxtrininsertinside \else##2\fi
8310 }%
8311 \renewcommand*\glsxtrsubsequentplfmt}[2]{%
8312     \glslongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8313     \ifglsxtrininsertinside \else##2\fi
8314 }%
8315 \renewcommand*\Glsxtrsubsequentfmt}[2]{%
8316     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
8317     \ifglsxtrininsertinside \else##2\fi
8318 }%
8319 \renewcommand*\Glsxtrsubsequentplfmt}[2]{%
8320     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
8321     \ifglsxtrininsertinside \else##2\fi
8322 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8323 \renewcommand*\glsxtrinlinefullformat}[2]{%
8324     \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8325     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8326     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8327 }%
8328 \renewcommand*\glsxtrinlinefullplformat}[2]{%
8329     \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8330     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8331     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8332 }%
8333 \renewcommand*\Glsxtrinlinefullformat}[2]{%
8334     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
8335     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8336     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8337 }%
8338 \renewcommand*\Glsxtrinlinefullplformat}[2]{%
8339     \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
8340     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8341     \glsxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%

```



8342 }%

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```
8343 \renewcommand*{\glxtrfullformat}[2]{%
8344   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8345   \ifglxtrinsertinside\else##2\fi
8346 }%
8347 \renewcommand*{\glxtrfullplformat}[2]{%
8348   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8349   \ifglxtrinsertinside\else##2\fi
8350 }%
8351 \renewcommand*{\Glsxtrfullformat}[2]{%
8352   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8353   \ifglxtrinsertinside\else##2\fi
8354 }%
8355 \renewcommand*{\Glsxtrfullplformat}[2]{%
8356   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8357   \ifglxtrinsertinside\else##2\fi
8358 }%
8359 }
```

long-sm Backward compatibility:

```
8360 \@glxtr@deprecated@abbrstyle{long-sm}{long-noshort-sm}
```

noshort-sm-desc The smaller font will only be used if the short form is explicitly invoked through commands like \glsshort.

```
8361 \newabbreviationstyle{long-noshort-sm-desc}%
8362 {%
8363   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
8364 }%
8365 {%
8366   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvsmfont{##1}}%
8367   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvsmfont{##1}}%
8368   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtrsmsuffix}%
8369   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8370   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%
```

The format for subsequent use (not used when the regular attribute is set).

```
8371 \renewcommand*{\glxtrsubsequentfmt}[2]{%
8372   \glslongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8373   \ifglxtrinsertinside \else##2\fi
8374 }%
8375 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
8376   \glslongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
8377   \ifglxtrinsertinside \else##2\fi
8378 }%
8379 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
8380   \glslongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8381   \ifglxtrinsertinside \else##2\fi
```

```

8382 }%
8383 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
8384   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
8385   \ifglxtrinsertinside \else##2\fi
8386 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

8387 \renewcommand*{\glxtrinlinefullformat}[2]{%
8388   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8389   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8390   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshort{##1}}}%
8391 }%
8392 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8393   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8394   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8395   \glxtrparen{\protect\glsfirstabbrvsmfont{\glsaccessshortpl{##1}}}%
8396 }%
8397 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8398   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8399   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8400   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshort{##1}}}%
8401 }%
8402 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8403   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8404   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8405   \glxtrparen{\protect\glsfirstabbrvsmfont{\Glsaccessshortpl{##1}}}%
8406 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

8407 \renewcommand*{\glxtrfullformat}[2]{%
8408   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8409   \ifglxtrinsertinside\else##2\fi
8410 }%
8411 \renewcommand*{\glxtrfullplformat}[2]{%
8412   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8413   \ifglxtrinsertinside\else##2\fi
8414 }%
8415 \renewcommand*{\Glsxtrfullformat}[2]{%
8416   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
8417   \ifglxtrinsertinside\else##2\fi
8418 }%
8419 \renewcommand*{\Glsxtrfullplformat}[2]{%
8420   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
8421   \ifglxtrinsertinside\else##2\fi
8422 }%
8423 }

```

long-desc-sm Backward compatibility:

```

8424 \@glxtr@deprecated@abbrstyle{long-desc-sm}{long-noshort-sm-desc}

```

ort-sm-footnote

```
8425 \newabbreviationstyle{short-sm-footnote}%
8426 {%
8427   \renewcommand*{\CustomAbbreviationFields}{%
8428     name={\glxtrfootnotename},
8429     sort={\the\glsshorttok},
8430     description={\the\glslongtok},%
8431     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}%
8432       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8433       {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
8434     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}%
8435       \protect\glxtrabbrvfootnote{\the\glslabeltok}%
8436       {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
8437     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%
8438   }
```

Switch off hyperlinks on first use to prevent nested hyperlinks, and unset the regular attribute if it has been set.

```
8438   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8439     \glssetattribute{\the\glslabeltok}{nohyperfirst}{true}%
8440     \glsattribute{\the\glslabeltok}{regular}%
8441     {%
8442       \glssetattribute{\the\glslabeltok}{regular}{false}%
8443     }%
8444   }%
8445 }%
8446 }%
8447 {%
8448   \renewcommand*\glsabbrvfont[1]{\glsabbrvsmfont{##1}}%
8449   \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvsmfont{##1}}%
8450   \renewcommand*\abbrvpluralsuffix{\protect\glxtrmsuffix}%
8451   \renewcommand*\glsfirstlongfont[1]{\glsfirstlongfootnotefont{##1}}%
8452   \renewcommand*\glslongfont[1]{\glslongfootnotefont{##1}}%
8453 }
```

The full format displays the short form followed by the long form as a footnote.

```
8453 \renewcommand*\glxtrfullformat[2]{%
8454   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8455   \ifglxtrininsertinside\else##2\fi
8456   \protect\glxtrabbrvfootnote{##1}%
8457   {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8458 }%
8459 \renewcommand*\glxtrfullplformat[2]{%
8460   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8461   \ifglxtrininsertinside\else##2\fi
8462   \protect\glxtrabbrvfootnote{##1}%
8463   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8464 }%
8465 \renewcommand*\Glsxtrfullformat[2]{%
8466   \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8467   \ifglxtrininsertinside\else##2\fi
8468   \protect\glxtrabbrvfootnote{##1}%
8469 }
```

```

8469     {\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8470 }%
8471 \renewcommand*{\Glsxtrfullplformat}[2]{%
8472   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8473   \ifglsxtrininsertinside\else##2\fi
8474   \protect\glsxtrabbrvfootnote{##1}%
8475   {\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8476 }%

```

The first use full form and the inline full form use the short (long) style.

```

8477 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8478   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8479   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8480   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8481 }%
8482 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8483   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8484   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8485   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8486 }%
8487 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8488   \glsfirstabbrvsmfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
8489   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8490   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8491 }%
8492 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8493   \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
8494   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8495   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8496 }%
8497 }

```

**footnote-sm** Backward compatibility:

```

8498 \@glsxtr@deprecated@abbrstyle{footnote-sm}{short-sm-footnote}

```

**sm-postfootnote**

```

8499 \newabbreviationstyle{short-sm-postfootnote}%
8500 {%
8501   \renewcommand*{\CustomAbbreviationFields}{%
8502     name={\glsxtrfootnotename},
8503     sort={\the\glsshorttok},
8504     description={\the\glslongtok},%
8505     first={\protect\glsfirstabbrvsmfont{\the\glsshorttok}},%
8506     firstplural={\protect\glsfirstabbrvsmfont{\the\glsshortpltok}},%
8507     plural={\protect\glsabbrvsmfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

8508 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8509   \csdef{glsxtrpostlink\glscategorylabel}{%

```

```

8510     \glxtrifwasfirstuse
8511     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

8512     \glxtrdopostpunc{\protect\glxtrabbrvfootnote{\glslabel}%
8513     {\glsfirstlongfootnotefont{\gl Sentrylong{\glslabel}}}}%
8514     }%
8515     {%
8516     }%
8517     \glshasattribute{\the\glslabeltok}{regular}%
8518     {%
8519     \glissetattribute{\the\glslabeltok}{regular}{false}%
8520     }%
8521     }%
8522     }%

```

The footnote needs to be suppressed in the inline form, so `\glxtrfull` must set the first use switch off.

```

8523 \renewcommand*\glxtrsetupfulldefs{%
8524 \let\glxtrifwasfirstuse\@secondoftwo
8525 }%
8526 }%
8527 {%
8528 \renewcommand*\glxtrabbrvfont[1]{\glxtrabbrvsmfont{##1}}%
8529 \renewcommand*\glxtrfirstabbrvfont[1]{\glxtrfirstabbrvsmfont{##1}}%
8530 \renewcommand*\glxtrabbrvpluralsuffix{\protect\glxtrrmsuffix}%
8531 \renewcommand*\glxtrfirstlongfont[1]{\glxtrfirstlongfootnotefont{##1}}%
8532 \renewcommand*\glxtrlongfont[1]{\glxtrlongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

8533 \renewcommand*\glxtrfullformat[2]{%
8534 \glxtrfirstabbrvsmfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
8535 \ifglxtrininsertinside\else##2\fi
8536 }%
8537 \renewcommand*\glxtrfullplformat[2]{%
8538 \glxtrfirstabbrvsmfont{\glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8539 \ifglxtrininsertinside\else##2\fi
8540 }%
8541 \renewcommand*\Glxtrfullformat[2]{%
8542 \glxtrfirstabbrvsmfont{\Glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%
8543 \ifglxtrininsertinside\else##2\fi
8544 }%
8545 \renewcommand*\Glxtrfullplformat[2]{%
8546 \glxtrfirstabbrvsmfont{\Glxtraccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8547 \ifglxtrininsertinside\else##2\fi
8548 }%

```

The first use full form and the inline full form use the short (long) style.

```

8549 \renewcommand*\glxtrininlinefullformat[2]{%
8550 \glxtrfirstabbrvsmfont{\glxtraccessshort{##1}\ifglxtrininsertinside##2\fi}%

```

```

8551     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8552     \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8553 }%
8554 \renewcommand*{\glxtrinlinefullplformat}[2]{%
8555     \glsfirstabbrvsmfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8556     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8557     \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8558 }%
8559 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8560     \glsfirstabbrvsmfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8561     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8562     \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
8563 }%
8564 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8565     \glsfirstabbrvsmfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8566     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8567     \glxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
8568 }%
8569 }

```

postfootnote-sm Backward compatibility:

```

8570 \@glxtr@deprecated@abbrstyle{postfootnote-sm}{short-sm-postfootnote}

```

### 1.7.5 Predefined Styles (Emphasized)

These styles use `\emph` for the short form.

`\glsabbrvemfont`

```

8571 \newcommand*{\glsabbrvemfont}[1]{\emph{##1}}%

```

`\glsfirstabbrvemfont`

```

8572 \newcommand*{\glsfirstabbrvemfont}[1]{\glsabbrvemfont{##1}}%

```

The default short form suffix:

`\glxtremsuffix`

```

8573 \newcommand*{\glxtremsuffix}{\glxtrabbrvpluralsuffix}

```

`\glsfirstlongemfont` Only used by the “long-em” styles.

```

8574 \newcommand*{\glsfirstlongemfont}[1]{\glslongemfont{##1}}%

```

`\glslongemfont` Only used by the “long-em” styles.

```

8575 \newcommand*{\glslongemfont}[1]{\emph{##1}}%

```

`\long-short-em` The long form is just set in the default long font.

```

8576 \newabbreviationstyle{long-short-em}%
8577 {%
8578     \renewcommand*{\CustomAbbreviationFields}{%

```

```

8579   name={\glxtrlongshortname},
8580   sort={\the\glsshorttok},
8581   first={\protect\glsfirstlongdefaultfont{\the\glslongtok}%
8582     \protect\glxtrfullsep{\the\glslabeltok}%
8583     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshorttok}}},%
8584   firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}%
8585     \protect\glxtrfullsep{\the\glslabeltok}%
8586     \glxtrparen{\protect\glsfirstabbrvemfont{\the\glsshortpltok}}},%
8587   plural={\protect\glsabbrvemfont{\the\glsshortpltok}}},%
8588   description={\the\glslongtok}}%
8589 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8590   \glshasattribute{\the\glslabeltok}{regular}}%
8591   {%
8592     \glissetattribute{\the\glslabeltok}{regular}{false}}%
8593   }%
8594   {}%
8595 }%
8596 }%
8597 {%
8598   \renewcommand*{\glsabbrvfont[1]{\glsabbrvemfont{##1}}}%
8599   \renewcommand*{\glsfirstabbrvfont[1]{\glsfirstabbrvemfont{##1}}}%
8600   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%

```

Use the default long fonts.

```

8601   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8602   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8603   \renewcommand*{\glxtrfullformat}[2]{%
8604     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8605     \ifglxtrininsertinside\else##2\fi
8606     \glxtrfullsep{##1}%
8607     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
8608   }%
8609   \renewcommand*{\glxtrfullplformat}[2]{%
8610     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8611     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8612     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8613   }%
8614   \renewcommand*{\Glsxtrfullformat}[2]{%
8615     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%
8616     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8617     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
8618   }%
8619   \renewcommand*{\Glsxtrfullplformat}[2]{%
8620     \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
8621     \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8622     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8623   }%
8624 }

```

g-short-em-desc

```
8625 \newabbreviationstyle{long-short-em-desc}%
8626 {%
8627   \renewcommand*{\CustomAbbreviationFields}{%
8628     name={\glxtrlongshortdescname},
8629     sort={\glxtrlongshortdescsort},%
8630     first={\protect\glslongdefaultfont{\the\glslongtok}%
8631       \protect\glxtrfullsep{\the\glslabeltok}%
8632       \glxtrparen{\protect\glslongfirstabbrvfont{\the\glsshorttok}}},%
8633     firstplural={\protect\glslongdefaultfont{\the\glslongpltok}%
8634       \protect\glxtrfullsep{\the\glslabeltok}%
8635       \glxtrparen{\protect\glslongfirstabbrvfont{\the\glsshortpltok}}},%
8636     text={\protect\glslongfirstabbrvfont{\the\glsshorttok}},%
8637     plural={\protect\glslongfirstabbrvfont{\the\glsshortpltok}}}%
8638   }%
```

Unset the regular attribute if it has been set.

```
8639   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8640     \glshasattribute{\the\glslabeltok}{regular}%
8641     {%
8642       \glissetattribute{\the\glslabeltok}{regular}{false}%
8643     }%
8644   }%
8645 }%
8646 }%
8647 {%
```

As long-short-em style:

```
8648   \GlsXtrUseAbbrStyleFmts{long-short-em}%
8649 }
```

ong-em-short-em

```
8650 \newabbreviationstyle{long-em-short-em}%
8651 {%
8652   \renewcommand*{\CustomAbbreviationFields}{%
8653     name={\glxtrlongshortname},
8654     sort={\the\glsshorttok},
8655     first={\protect\glslongfirstlongemfont{\the\glslongtok}%
8656       \protect\glxtrfullsep{\the\glslabeltok}%
8657       \glxtrparen{\protect\glslongfirstabbrvfont{\the\glsshorttok}}},%
8658     firstplural={\protect\glslongfirstlongemfont{\the\glslongpltok}%
8659       \protect\glxtrfullsep{\the\glslabeltok}%
8660       \glxtrparen{\protect\glslongfirstabbrvfont{\the\glsshortpltok}}},%
8661     plural={\protect\glslongfirstabbrvfont{\the\glsshortpltok}},%
8662     description={\protect\glslongemfont{\the\glslongtok}}}%
8663   }
```

Unset the regular attribute if it has been set.

```
8663   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8664     \glshasattribute{\the\glslabeltok}{regular}%
8665     {%
8666       \glissetattribute{\the\glslabeltok}{regular}{false}%
8667     }%
8668   }%
```



```

8664 \glshasattribute{\the\glslabeltok}{regular}%
8665 {%
8666 \glissetattribute{\the\glslabeltok}{regular}{false}%
8667 }%
8668 {}%
8669 }%
8670 }%
8671 {%
8672 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8673 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
8674 \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvemfont{##1}}%
8675 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlongemfont{##1}}%
8676 \renewcommand*{\glsslongfont}[1]{\glsslongemfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8677 \renewcommand*{\glssxtrfullformat}[2]{%
8678 \glssfirstlongemfont{\glssaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8679 \ifglssxtrininsertinside\else##2\fi
8680 \glssxtrfullsep{##1}%
8681 \glssxtrparen{\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
8682 }%
8683 \renewcommand*{\glssxtrfullplformat}[2]{%
8684 \glssfirstlongemfont{\glssaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
8685 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8686 \glssxtrparen{\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8687 }%
8688 \renewcommand*{\Glsxtrfullformat}[2]{%
8689 \glssfirstlongemfont{\Glsaccesslong{##1}\ifglssxtrininsertinside##2\fi}%
8690 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8691 \glssxtrparen{\glssfirstabbrvemfont{\glssaccessshort{##1}}}%
8692 }%
8693 \renewcommand*{\Glsxtrfullplformat}[2]{%
8694 \glssfirstlongemfont{\Glsaccesslongpl{##1}\ifglssxtrininsertinside##2\fi}%
8695 \ifglssxtrininsertinside\else##2\fi\glssxtrfullsep{##1}%
8696 \glssxtrparen{\glssfirstabbrvemfont{\glssaccessshortpl{##1}}}%
8697 }%
8698 }

```

m-short-em-desc

```

8699 \newabbreviationstyle{long-em-short-em-desc}%
8700 {%
8701 \renewcommand*{\CustomAbbreviationFields}{%
8702 name={\glssxtrlongshortdescname},
8703 sort={\glssxtrlongshortdescsort},%
8704 first={\protect\glssfirstlongemfont{\the\glsslongtok}}%
8705 \protect\glssxtrfullsep{\the\glslabeltok}%
8706 \glssxtrparen{\protect\glssfirstabbrvemfont{\the\glssshorttok}}},%
8707 firstplural={\protect\glssfirstlongemfont{\the\glsslongpltok}}%
8708 \protect\glssxtrfullsep{\the\glslabeltok}%
8709 \glssxtrparen{\protect\glssfirstabbrvemfont{\the\glssshortpltok}}},%

```

```

8710     text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8711     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8712 }%

```

Unset the regular attribute if it has been set.

```

8713 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8714   \glshasattribute{\the\glslabeltok}{regular}%
8715   {%
8716     \glssetattribute{\the\glslabeltok}{regular}{false}%
8717   }%
8718   {}}%
8719 }%
8720 }%
8721 {%
8722   \GlsXtrUseAbbrStyleFmts{long-em-short-em}%
8723 }

```

short-em-long Now the short (long) version

```

8724 \newabbreviationstyle{short-em-long}%
8725 {%
8726   \renewcommand*{\CustomAbbreviationFields}{%
8727     name={\glstrshortlongname},
8728     sort={\the\glsshorttok},
8729     description={\the\glslongtok},%
8730     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}}%
8731     \protect\glstrfullsep{\the\glslabeltok}%
8732     \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8733     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}}%
8734     \protect\glstrfullsep{\the\glslabeltok}%
8735     \glstrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8736     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

8737 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8738   \glshasattribute{\the\glslabeltok}{regular}%
8739   {%
8740     \glssetattribute{\the\glslabeltok}{regular}{false}%
8741   }%
8742   {}}%
8743 }%
8744 }%
8745 {%

```

Mostly as short-long style:

```

8746 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8747 \renewcommand*{\glsabbrvfont[1]}{\glsabbrvemfont{##1}}%
8748 \renewcommand*{\glsfirstabbrvfont[1]}{\glsfirstabbrvemfont{##1}}%
8749 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8750 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

8751 \renewcommand*{\glxtrfullformat}[2]{%
8752   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8753   \ifglxtrininsertinside\else##2\fi
8754   \glxtrfullsep{##1}%
8755   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
8756 }%
8757 \renewcommand*{\glxtrfullplformat}[2]{%
8758   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8759   \ifglxtrininsertinside\else##2\fi
8760   \glxtrfullsep{##1}%
8761   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8762 }%
8763 \renewcommand*{\Glsxtrfullformat}[2]{%
8764   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8765   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8766   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslong{##1}}}%
8767 }%
8768 \renewcommand*{\Glsxtrfullplformat}[2]{%
8769   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8770   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
8771   \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8772 }%
8773 }

```

rt-em-long-desc As before but user provides description

```

8774 \newabbreviationstyle{short-em-long-desc}%
8775 {%
8776   \renewcommand*{\CustomAbbreviationFields}{%
8777     name={\glxtrshortlongdescname},
8778     sort={\glxtrshortlongdescsort},
8779     first={\protect\glsfirstabbrvemfont{\the\glssshorttok}%
8780       \protect\glxtrfullsep{\the\glslabeltok}%
8781       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongtok}}},%
8782     firstplural={\protect\glsfirstabbrvemfont{\the\glssshortpltok}%
8783       \protect\glxtrfullsep{\the\glslabeltok}%
8784       \glxtrparen{\protect\glsfirstlongdefaultfont{\the\glslongpltok}}},%
8785     text={\protect\glssabbrvemfont{\the\glssshorttok}},%
8786     plural={\protect\glssabbrvemfont{\the\glssshortpltok}}%
8787   }%

```

Unset the regular attribute if it has been set.

```

8788 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8789   \glshasattribute{\the\glslabeltok}{regular}%
8790   {%
8791     \glsssetAttribute{\the\glslabeltok}{regular}{false}%
8792   }%
8793   {}%
8794 }%
8795 }%
8796 {%

```

```

8797 \GlsXtrUseAbbrStyleFmts{short-em-long}%
8798 }

```

hort-em-long-em

```

8799 \newabbreviationstyle{short-em-long-em}%
8800 {%
    \glslongemfont is used in the description since \glsdesc doesn't set the style.
8801 \renewcommand*{\CustomAbbreviationFields}{%
8802     name={\glxtrshortlongname},
8803     sort={\the\glsshorttok},
8804     description={\protect\glslongemfont{\the\glslongtok}},%
8805     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8806         \protect\glxtrfullsep{\the\glslabeltok}%
8807         \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8808     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8809         \protect\glxtrfullsep{\the\glslabeltok}%
8810         \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8811     plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%

    Unset the regular attribute if it has been set.
8812 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8813     \glshasattribute{\the\glslabeltok}{regular}%
8814     {%
8815         \glissetattribute{\the\glslabeltok}{regular}{false}%
8816     }%
8817     {}%
8818 }%
8819 }%
8820 {%
8821 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
8822 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8823 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8824 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongemfont{##1}}%
8825 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

    The first use full form and the inline full form are the same for this style.
8826 \renewcommand*{\glxtrfullformat}[2]{%
8827     \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
8828     \ifglxtrininsertinside\else##2\fi
8829     \glxtrfullsep{##1}%
8830     \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8831 }%
8832 \renewcommand*{\glxtrfullplformat}[2]{%
8833     \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
8834     \ifglxtrininsertinside\else##2\fi
8835     \glxtrfullsep{##1}%
8836     \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8837 }%
8838 \renewcommand*{\GlsXtrFullFormat}[2]{%

```

```

8839 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8840 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8841 \glxtrparen{\glsfirstlongemfont{\glsaccesslong{##1}}}%
8842 }%
8843 \renewcommand*{\Glsxtrfullplformat}[2]{%
8844 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8845 \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8846 \glxtrparen{\glsfirstlongemfont{\glsaccesslongpl{##1}}}%
8847 }%
8848 }

```

em-long-em-desc

```

8849 \newabbreviationstyle{short-em-long-em-desc}%
8850 {%
8851 \renewcommand*{\CustomAbbreviationFields}{%
8852 name={\glxtrshortlongdescname},%
8853 sort={\glxtrshortlongdescsort},%
8854 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
8855 \protect\glxtrfullsep{\the\glslabeltok}%
8856 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongtok}}},%
8857 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
8858 \protect\glxtrfullsep{\the\glslabeltok}%
8859 \glxtrparen{\protect\glsfirstlongemfont{\the\glslongpltok}}},%
8860 text={\protect\glsabbrvemfont{\the\glsshorttok}},%
8861 plural={\protect\glsabbrvemfont{\the\glsshortpltok}}}%
8862 }%

```

Unset the regular attribute if it has been set.

```

8863 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8864 \glshasattribute{\the\glslabeltok}{regular}%
8865 {%
8866 \glissetattribute{\the\glslabeltok}{regular}{false}%
8867 }%
8868 {}%
8869 }%
8870 }%
8871 {%
8872 \GlsXtrUseAbbrStyleFmts{short-em-long-em}%
8873 }

```

short-em

```

8874 \newabbreviationstyle{short-em}%
8875 {%
8876 \renewcommand*{\CustomAbbreviationFields}{%
8877 name={\glxtrshortnolongname},
8878 sort={\the\glsshorttok},
8879 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8880 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8881 text={\protect\glsabbrvemfont{\the\glsshorttok}},
8882 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},

```

```

8883   description={\the\glslongtok}}%
8884   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8885     \glssetattribute{\the\glslabeltok}{regular}{true}}%
8886 }%
8887 {%
8888   \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8889   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8890   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8891   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8892   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short form followed by the long form in parentheses.

```

8893   \renewcommand*{\glsxtrinlinefullformat}[2]{%
8894     \protect\glsfirstabbrvemfont{\glsaccesssshort{##1}}%
8895     \ifglsxtrininsertinside##2\fi}%
8896     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8897     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8898   }%
8899   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8900     \protect\glsfirstabbrvemfont{\glsaccesssshortpl{##1}}%
8901     \ifglsxtrininsertinside##2\fi}%
8902     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8903     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8904   }%

8905   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8906     \protect\glsfirstabbrvemfont{\Glsaccesssshort{##1}}%
8907     \ifglsxtrininsertinside##2\fi}%
8908     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8909     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8910   }%
8911   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8912     \protect\glsfirstabbrvemfont{\Glsaccesssshortpl{##1}}%
8913     \ifglsxtrininsertinside##2\fi}%
8914     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
8915     \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8916   }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

8917   \renewcommand*{\glsxtrfullformat}[2]{%
8918     \glsfirstabbrvemfont{\glsaccesssshort{##1}}\ifglsxtrininsertinside##2\fi}%
8919     \ifglsxtrininsertinside\else##2\fi
8920   }%
8921   \renewcommand*{\glsxtrfullplformat}[2]{%
8922     \glsfirstabbrvemfont{\glsaccesssshortpl{##1}}\ifglsxtrininsertinside##2\fi}%
8923     \ifglsxtrininsertinside\else##2\fi
8924   }%
8925   \renewcommand*{\Glsxtrfullformat}[2]{%
8926     \glsfirstabbrvemfont{\Glsaccesssshort{##1}}\ifglsxtrininsertinside##2\fi}%

```

```

8927 \ifglxtrinsertinside\else##2\fi
8928 }%
8929 \renewcommand*{\Glsxtrfullplformat}[2]{%
8930 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8931 \ifglxtrinsertinside\else##2\fi
8932 }%
8933 }

```

short-em-nolong

```

8934 \letabbreviationstyle{short-em-nolong}{short-em}

```

short-em-desc

```

8935 \newabbreviationstyle{short-em-desc}%
8936 {%
8937 \renewcommand*{\CustomAbbreviationFields}{%
8938 name={\glsxtrshortdesname},
8939 sort={\the\glsshorttok},
8940 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},
8941 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},
8942 text={\protect\glsabbrvemfont{\the\glsshorttok}},
8943 plural={\protect\glsabbrvemfont{\the\glsshortpltok}},
8944 description={\the\glslongtok}}%
8945 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
8946 \glssetattribute{\the\glslabeltok}{regular}{true}}%
8947 }%
8948 {%
8949 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtremsuffix}%
8950 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
8951 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
8952 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
8953 \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The inline full form displays the short format followed by the long form in parentheses.

```

8954 \renewcommand*{\glsxtrinlinefullformat}[2]{%
8955 \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8956 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8957 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8958 }%
8959 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
8960 \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8961 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8962 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslongpl{##1}}}%
8963 }%
8964 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
8965 \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8966 \ifglxtrinsertinside\else##2\fi\glsxtrfullsep{##1}%
8967 \glsxtrparen{\glsfirstlongdefaultfont{\glsaccesslong{##1}}}%
8968 }%
8969 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
8970 \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%

```

```

8971     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
8972     \glxtrparen{\glsfirstlongdefaultfont{\glssaccesslongpl{##1}}}%
8973 }%

```

The first use full form only displays the short form, but it typically won't be used as the regular attribute is set by this style.

```

8974 \renewcommand*{\glxtrfullformat}[2]{%
8975     \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8976     \ifglxtrinsertinside\else##2\fi
8977 }%
8978 \renewcommand*{\glxtrfullplformat}[2]{%
8979     \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8980     \ifglxtrinsertinside\else##2\fi
8981 }%
8982 \renewcommand*{\Glsxtrfullformat}[2]{%
8983     \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrinsertinside##2\fi}%
8984     \ifglxtrinsertinside\else##2\fi
8985 }%
8986 \renewcommand*{\Glsxtrfullplformat}[2]{%
8987     \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrinsertinside##2\fi}%
8988     \ifglxtrinsertinside\else##2\fi
8989 }%
8990 }

```

-em-nolong-desc

```

8991 \letabbreviationstyle{short-em-nolong-desc}{short-em-desc}

```

nolong-short-em

```

8992 \newabbreviationstyle{nolong-short-em}%
8993 {%
8994     \GlsXtrUseAbbrStyleSetup{short-em-nolong}%
8995 }%
8996 {%
8997     \GlsXtrUseAbbrStyleFmts{short-em-nolong}%

```

The inline full form displays the long form followed by the short form in parentheses.

```

8998 \renewcommand*{\glxtrinlinefullformat}[2]{%
8999     \protect\glsfirstlongdefaultfont{\glssaccesslong{##1}%
9000         \ifglxtrinsertinside##2\fi}%
9001     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9002     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9003 }%
9004 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9005     \protect\glsfirstlongdefaultfont{\glssaccesslongpl{##1}%
9006         \ifglxtrinsertinside##2\fi}%
9007     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9008     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9009 }%
9010 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9011     \protect\glsfirstlongdefaultfont{\Glsaccesslong{##1}%

```



```

9012     \ifglxtrinsertinside##2\fi}%
9013     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9014     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9015 }%
9016 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9017     \protect\glsfirstlongdefaultfont{\glssaccesslongpl{##1}%
9018         \ifglxtrinsertinside##2\fi}%
9019     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9020     \glxtrparen{\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9021 }%
9022 }

```

long-noshort-em The short form is explicitly invoked through commands like \glssshort.

```

9023 \newabbreviationstyle{long-noshort-em}%
9024 {%
9025     \renewcommand*{\CustomAbbreviationFields}{%
9026         name={\glxtrlongnoshortname},
9027         sort={\the\glssshorttok},
9028         first={\protect\glsfirstlongdefaultfont{\the\glslongtok}},
9029         firstplural={\protect\glsfirstlongdefaultfont{\the\glslongpltok}},
9030         text={\protect\glslongdefaultfont{\the\glslongtok}},
9031         plural={\protect\glslongdefaultfont{\the\glslongpltok}},%
9032         description={\the\glslongtok}%
9033     }%
9034     \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9035         \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9036 }%
9037 {%
9038     \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9039     \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9040     \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9041     \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9042     \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9043     \renewcommand*{\glxtrsubsequentfmt}[2]{%
9044         \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9045         \ifglxtrinsertinside \else##2\fi
9046     }%
9047     \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9048         \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9049         \ifglxtrinsertinside \else##2\fi
9050     }%
9051     \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9052         \glslongdefaultfont{\glssaccesslong{##1}\ifglxtrinsertinside ##2\fi}%
9053         \ifglxtrinsertinside \else##2\fi
9054     }%
9055     \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9056         \glslongdefaultfont{\glssaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9057         \ifglxtrinsertinside \else##2\fi

```

9058 }%

The inline full form displays the long format followed by the short form in parentheses.

```
9059 \renewcommand*{\glxstrinlinefullformat}[2]{%
9060   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9061   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9062   \glxstrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9063 }%
9064 \renewcommand*{\glxstrinlinefullplformat}[2]{%
9065   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9066   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9067   \glxstrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9068 }%
9069 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9070   \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9071   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9072   \glxstrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9073 }%
9074 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9075   \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9076   \ifglxstrinsertinside\else##2\fi\glxstrfullsep{##1}%
9077   \glxstrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9078 }%
```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```
9079 \renewcommand*{\glxstrfullformat}[2]{%
9080   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9081   \ifglxstrinsertinside\else##2\fi
9082 }%
9083 \renewcommand*{\glxstrfullplformat}[2]{%
9084   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9085   \ifglxstrinsertinside\else##2\fi
9086 }%
9087 \renewcommand*{\Glsxtrfullformat}[2]{%
9088   \glsfirstlongdefaultfont{\glsaccesslong{##1}\ifglxstrinsertinside##2\fi}%
9089   \ifglxstrinsertinside\else##2\fi
9090 }%
9091 \renewcommand*{\Glsxtrfullplformat}[2]{%
9092   \glsfirstlongdefaultfont{\glsaccesslongpl{##1}\ifglxstrinsertinside##2\fi}%
9093   \ifglxstrinsertinside\else##2\fi
9094 }%
9095 }
```

long-em Backward compatibility:

```
9096 \@glxstr@deprecated@abbrstyle{long-em}{long-noshort-em}
```

g-em-noshort-em The short form is explicitly invoked through commands like \glsshort.

```
9097 \newabbreviationstyle{long-em-noshort-em}%
9098 {%
```

```

9099 \renewcommand*{\CustomAbbreviationFields}{%
9100   name={\glxstrlongnosshortname},
9101   sort={\the\glssshorttok},
9102   first={\protect\glstrfirstlongemfont{\the\glslongtok}},
9103   firstplural={\protect\glstrfirstlongemfont{\the\glslongpltok}},
9104   text={\protect\glslongemfont{\the\glslongtok}},
9105   plural={\protect\glslongemfont{\the\glslongpltok}},%
9106   description={\protect\glslongemfont{\the\glslongtok}}%
9107 }%
9108 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9109   \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9110}%
9111{%
9112 \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9113 \renewcommand*{\glssabbrvfont}[1]{\glssabbrvemfont{##1}}%
9114 \renewcommand*{\glstrfirstabbrvfont}[1]{\glstrfirstabbrvemfont{##1}}%
9115 \renewcommand*{\glstrfirstlongfont}[1]{\glstrfirstlongemfont{##1}}%
9116 \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9117 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9118   \glslongemfont{\glssaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9119   \ifglxtrininsertinside \else##2\fi
9120}%
9121 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9122   \glslongemfont{\glssaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9123   \ifglxtrininsertinside \else##2\fi
9124}%
9125 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9126   \glslongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside ##2\fi}%
9127   \ifglxtrininsertinside \else##2\fi
9128}%
9129 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9130   \glslongemfont{\Glsaccesslongpl{##1}\ifglxtrininsertinside ##2\fi}%
9131   \ifglxtrininsertinside \else##2\fi
9132}%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9133 \renewcommand*{\glxtrininlinefullformat}[2]{%
9134   \glstrfirstlongemfont{\glssaccesslong{##1}\ifglxtrininsertinside##2\fi}%
9135   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9136   \glxtrparen{\protect\glstrfirstabbrvemfont{\glssaccessshort{##1}}}%
9137}%
9138 \renewcommand*{\glxtrininlinefullplformat}[2]{%
9139   \glstrfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrininsertinside##2\fi}%
9140   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9141   \glxtrparen{\protect\glstrfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9142}%
9143 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9144   \glstrfirstlongemfont{\Glsaccesslong{##1}\ifglxtrininsertinside##2\fi}%

```

```

9145     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9146     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshort{##1}}}%
9147 }%
9148 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9149     \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9150     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9151     \glxtrparen{\protect\glsfirstabbrvemfont{\glssaccessshortpl{##1}}}%
9152 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

9153 \renewcommand*{\glxtrfullformat}[2]{%
9154     \glsfirstlongemfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9155     \ifglxtrinsertinside\else##2\fi
9156 }%
9157 \renewcommand*{\glxtrfullplformat}[2]{%
9158     \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9159     \ifglxtrinsertinside\else##2\fi
9160 }%
9161 \renewcommand*{\Glsxtrfullformat}[2]{%
9162     \glsfirstlongemfont{\glssaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9163     \ifglxtrinsertinside\else##2\fi
9164 }%
9165 \renewcommand*{\Glsxtrfullplformat}[2]{%
9166     \glsfirstlongemfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9167     \ifglxtrinsertinside\else##2\fi
9168 }%
9169 }

```

`noshort-em-noreg` Like `long-em-noshort-em` but doesn't set the regular attribute.

```

9170 \newabbreviationstyle{long-em-noshort-em-noreg}%
9171 {%
9172     \GlsXtrUseAbbrStyleSetup{long-em-noshort-em}%

```

Unset the regular attribute if it has been set.

```

9173 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9174     \glshasattribute{\the\glslabeltok}{regular}%
9175     {%
9176         \glsssetattribute{\the\glslabeltok}{regular}{false}%
9177     }%
9178     {}%
9179 }%
9180 }%
9181 {%
9182     \GlsXtrUseAbbrStyleFmts{long-em-noshort-em}%
9183 }

```

`noshort-em-desc` The emphasized font will only be used if the short form is explicitly invoked through commands like `\glssshort`.

```

9184 \newabbreviationstyle{long-noshort-em-desc}%

```

```

9185 {%
9186   \GlsXtrUseAbbrStyleSetup{long-noshort-desc}%
9187 }%
9188 {%
9189   \renewcommand*{\abbrvpluralsuffix}{\protect\glstremssuffix}%
9190   \renewcommand*{\glssabrvfont}[1]{\glssabrvemfont{##1}}%
9191   \renewcommand*{\glsfirstabrvfont}[1]{\glsfirstabrvemfont{##1}}%
9192   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongdefaultfont{##1}}%
9193   \renewcommand*{\glslongfont}[1]{\glslongdefaultfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9194   \renewcommand*{\glsxtrsubsequentfmt}[2]{%
9195     \glslongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9196     \ifglsxtrininsertinside \else##2\fi
9197   }%
9198   \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
9199     \glslongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9200     \ifglsxtrininsertinside \else##2\fi
9201   }%
9202   \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
9203     \glslongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside ##2\fi}%
9204     \ifglsxtrininsertinside \else##2\fi
9205   }%
9206   \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9207     \glslongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside ##2\fi}%
9208     \ifglsxtrininsertinside \else##2\fi
9209   }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9210   \renewcommand*{\glsxtrinlinefullformat}[2]{%
9211     \glsfirstlongdefaultfont{\glssaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9212     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9213     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshort{##1}}}%
9214   }%
9215   \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9216     \glsfirstlongdefaultfont{\glssaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9217     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9218     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshortpl{##1}}}%
9219   }%
9220   \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9221     \glsfirstlongdefaultfont{\Glsaccesslong{##1}\ifglsxtrininsertinside##2\fi}%
9222     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9223     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshort{##1}}}%
9224   }%
9225   \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9226     \glsfirstlongdefaultfont{\Glsaccesslongpl{##1}\ifglsxtrininsertinside##2\fi}%
9227     \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9228     \glsxtrparen{\protect\glsfirstabrvemfont{\glssaccessshortpl{##1}}}%
9229   }%

```

The first use full form only displays the long form, but it typically won't be used as the regular

attribute is set by this style.

```

9230 \renewcommand*{\glxstrfullformat}[2]{%
9231   \glstrfirstlongdefaultfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
9232   \ifglstrinsertinside\else##2\fi
9233 }%
9234 \renewcommand*{\glxstrfullplformat}[2]{%
9235   \glstrfirstlongdefaultfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
9236   \ifglstrinsertinside\else##2\fi
9237 }%
9238 \renewcommand*{\Glsxtrfullformat}[2]{%
9239   \glstrfirstlongdefaultfont{\glstraccesslong{##1}\ifglstrinsertinside##2\fi}%
9240   \ifglstrinsertinside\else##2\fi
9241 }%
9242 \renewcommand*{\Glsxtrfullplformat}[2]{%
9243   \glstrfirstlongdefaultfont{\glstraccesslongpl{##1}\ifglstrinsertinside##2\fi}%
9244   \ifglstrinsertinside\else##2\fi
9245 }%
9246 }

```

long-desc-em Backward compatibility:

```

9247 \@glxtr@deprecated@abbrstyle{long-desc-em}{long-noshort-em-desc}

```

noshort-em-desc The short form is explicitly invoked through commands like `\glsshort`. The long form is emphasized.

```

9248 \newabbreviationstyle{long-em-noshort-em-desc}%
9249 {%
9250   \renewcommand*{\CustomAbbreviationFields}{%
9251     name={\glxtrlongnoshortdescname},
9252     sort={\the\glslongtok},
9253     first={\protect\glstrfirstlongemfont{\the\glslongtok}},
9254     firstplural={\protect\glstrfirstlongemfont{\the\glslongpltok}},
9255     text={\glslongemfont{\the\glslongtok}},
9256     plural={\glslongemfont{\the\glslongpltok}}}%
9257   }%
9258   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9259     \glsssetAttribute{\the\glslabeltok}{regular}{true}}%
9260 }%
9261 {%
9262   \renewcommand*{\abbrvpluralsuffix}{\protect\glxstremsuffix}%
9263   \renewcommand*{\glabbrvfont}[1]{\glabbrvemfont{##1}}%
9264   \renewcommand*{\glstrfirstabbrvfont}[1]{\glstrfirstabbrvemfont{##1}}%
9265   \renewcommand*{\glstrfirstlongfont}[1]{\glstrfirstlongemfont{##1}}%
9266   \renewcommand*{\glslongfont}[1]{\glslongemfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9267 \renewcommand*{\glxtrsubsequentfmt}[2]{%
9268   \glslongemfont{\glstraccesslong{##1}\ifglstrinsertinside ##2\fi}%
9269   \ifglstrinsertinside \else##2\fi
9270 }%

```

```

9271 \renewcommand*{\glxtrsubsequentplfmt}[2]{%
9272   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9273   \ifglxtrinsertinside \else##2\fi
9274 }%
9275 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9276   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9277   \ifglxtrinsertinside \else##2\fi
9278 }%
9279 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
9280   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside ##2\fi}%
9281   \ifglxtrinsertinside \else##2\fi
9282 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9283 \renewcommand*{\glxtrinlinefullformat}[2]{%
9284   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9285   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9286   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9287 }%
9288 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9289   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9290   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9291   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9292 }%
9293 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9294   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9295   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9296   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshort{##1}}}%
9297 }%
9298 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9299   \glsfirstlongemfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9300   \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9301   \glxtrparen{\protect\glsfirstabbrvemfont{\glsaccessshortpl{##1}}}%
9302 }%

```

The first use full form only displays the long form, but it typically won't be used as the regular attribute is set by this style.

```

9303 \renewcommand*{\glxtrfullformat}[2]{%
9304   \glsfirstlongemfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9305   \ifglxtrinsertinside\else##2\fi
9306 }%
9307 \renewcommand*{\glxtrfullplformat}[2]{%
9308   \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9309   \ifglxtrinsertinside\else##2\fi
9310 }%
9311 \renewcommand*{\Glsxtrfullformat}[2]{%
9312   \glsfirstlongemfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9313   \ifglxtrinsertinside\else##2\fi
9314 }%
9315 \renewcommand*{\Glsxtrfullplformat}[2]{%

```

```

9316 \glsfirstlongemfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9317 \ifglxtrinsertinside\else##2\fi
9318 }%
9319 }

```

t-em-desc-noreg Like long-em-noshort-em-desc but doesn't set the regular attribute.

```

9320 \newabbreviationstyle{long-em-noshort-em-desc-noreg}%
9321 {%
9322 \GlsXtrUseAbbrStyleSetup{long-em-noshort-em-desc}%
    Unset the regular attribute if it has been set.
9323 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9324 \glshasattribute{\the\glslabeltok}{regular}%
9325 {%
9326 \glissetattribute{\the\glslabeltok}{regular}{false}%
9327 }%
9328 {}%
9329 }%
9330 }%
9331 {%
9332 \GlsXtrUseAbbrStyleFmts{long-em-noshort-em-desc}%
9333 }

```

ort-em-footnote

```

9334 \newabbreviationstyle{short-em-footnote}%
9335 {%
9336 \renewcommand*{\CustomAbbreviationFields}{%
9337 name={\glxtrfootnotename},
9338 sort={\the\glsshorttok},
9339 description={\the\glslongtok},%
9340 first={\protect\glsfirstabbrvemfont{\the\glsshorttok}%
9341 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9342 {\protect\glsfirstlongfootnotefont{\the\glslongtok}}},%
9343 firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}%
9344 \protect\glxtrabbrvfootnote{\the\glslabeltok}%
9345 {\protect\glsfirstlongfootnotefont{\the\glslongpltok}}},%
9346 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.
9347 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9348 \glissetattribute{\the\glslabeltok}{nohyperfirst}{true}%
9349 \glshasattribute{\the\glslabeltok}{regular}%
9350 {%
9351 \glissetattribute{\the\glslabeltok}{regular}{false}%
9352 }%
9353 {}%
9354 }%
9355 }%
9356 {%

```



```

9357 \renewcommand*{\abbrvpluralsuffix}{\protect\glxxtremsuffix}%
9358 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvemfont{##1}}%
9359 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvemfont{##1}}%
9360 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9361 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form followed by the long form as a footnote.

```

9362 \renewcommand*{\glxtrfullformat}[2]{%
9363   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9364   \ifglxtrininsertinside\else##2\fi
9365   \protect\glxtrabbrvfootnote{##1}%
9366   {\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9367 }%
9368 \renewcommand*{\glxtrfullplformat}[2]{%
9369   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9370   \ifglxtrininsertinside\else##2\fi
9371   \protect\glxtrabbrvfootnote{##1}%
9372   {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9373 }%
9374 \renewcommand*{\Glsxtrfullformat}[2]{%
9375   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9376   \ifglxtrininsertinside\else##2\fi
9377   \protect\glxtrabbrvfootnote{##1}%
9378   {\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9379 }%
9380 \renewcommand*{\Glsxtrfullplformat}[2]{%
9381   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9382   \ifglxtrininsertinside\else##2\fi
9383   \protect\glxtrabbrvfootnote{##1}%
9384   {\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9385 }%

```

The first use full form and the inline full form use the short (long) style.

```

9386 \renewcommand*{\glxtrininlinefullformat}[2]{%
9387   \glsfirstabbrvemfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9388   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9389   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9390 }%
9391 \renewcommand*{\glxtrininlinefullplformat}[2]{%
9392   \glsfirstabbrvemfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9393   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9394   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9395 }%
9396 \renewcommand*{\Glsxtrininlinefullformat}[2]{%
9397   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9398   \ifglxtrininsertinside\else##2\fi\glxtrfullsep{##1}%
9399   \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslong{##1}}}%
9400 }%
9401 \renewcommand*{\Glsxtrininlinefullplformat}[2]{%
9402   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%

```

```

9403     \ifglxtrinsertinside\else##2\fi\glxtrfullsep{##1}%
9404     \glxtrparen{\glsfirstlongfootnotefont{\glssaccesslongpl{##1}}}%
9405 }%
9406 }

```

footnote-em Backward compatibility:

```

9407 \@glxtr@deprecated@abbrstyle{footnote-em}{short-em-footnote}

```

em-postfootnote

```

9408 \newabbreviationstyle{short-em-postfootnote}%
9409 {%
9410   \renewcommand*{\CustomAbbreviationFields}{%
9411     name={\glxtrfootnotename},
9412     sort={\the\glsshorttok},
9413     description={\the\glslongtok},%
9414     first={\protect\glsfirstabbrvemfont{\the\glsshorttok}},%
9415     firstplural={\protect\glsfirstabbrvemfont{\the\glsshortpltok}},%
9416     plural={\protect\glssabbrvemfont{\the\glsshortpltok}}}%

```

Make this category insert a footnote after the link if this was the first use, and unset the regular attribute if it has been set.

```

9417 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9418   \csdef{glxtrpostlink\glscategorylabel}{%
9419     \glxtrifwasfirstuse
9420     {%

```

Needs the specific font command here as the style may have been lost by the time the footnote occurs.

```

9421       \glxtrdopostpunc{\protect\glxtrabbrvffootnote{\glslabel}%
9422       {\glsfirstlongfootnotefont{\glsentrylong{\glslabel}}}}%
9423     }%
9424   }%
9425 }%
9426 \glshasattribute{\the\glslabeltok}{regular}%
9427 {%
9428   \glissetattribute{\the\glslabeltok}{regular}{false}%
9429 }%
9430 {}%
9431 }%

```

The footnote needs to be suppressed in the inline form, so \glxtrfull must set the first use switch off.

```

9432 \renewcommand*{\glxtrsetupfulldefs}{%
9433   \let\glxtrifwasfirstuse\@secondoftwo
9434 }%
9435 }%
9436 {%
9437   \renewcommand*{\abbrvpluralsuffix}{\protect\glxtremsuffix}%
9438   \renewcommand*{\glssabbrvf}[1]{\glssabbrvemfont{##1}}%
9439   \renewcommand*{\glsfirstabbrvf}[1]{\glsfirstabbrvemfont{##1}}%

```

```

9440 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongfootnotefont{##1}}%
9441 \renewcommand*{\glslongfont}[1]{\glslongfootnotefont{##1}}%

```

The full format displays the short form. The long form is deferred.

```

9442 \renewcommand*{\glsxtrfullformat}[2]{%
9443   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9444   \ifglsxtrininsertinside\else##2\fi
9445 }%
9446 \renewcommand*{\glsxtrfullplformat}[2]{%
9447   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9448   \ifglsxtrininsertinside\else##2\fi
9449 }%
9450 \renewcommand*{\Glsxtrfullformat}[2]{%
9451   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9452   \ifglsxtrininsertinside\else##2\fi
9453 }%
9454 \renewcommand*{\Glsxtrfullplformat}[2]{%
9455   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9456   \ifglsxtrininsertinside\else##2\fi
9457 }%

```

The first use full form and the inline full form use the short (long) style.

```

9458 \renewcommand*{\glsxtrinlinefullformat}[2]{%
9459   \glsfirstabbrvemfont{\glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9460   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9461   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9462 }%
9463 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
9464   \glsfirstabbrvemfont{\glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9465   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9466   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9467 }%
9468 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9469   \glsfirstabbrvemfont{\Glsaccessshort{##1}\ifglsxtrininsertinside##2\fi}%
9470   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9471   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslong{##1}}}%
9472 }%
9473 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9474   \glsfirstabbrvemfont{\Glsaccessshortpl{##1}\ifglsxtrininsertinside##2\fi}%
9475   \ifglsxtrininsertinside\else##2\fi\glsxtrfullsep{##1}%
9476   \glsxtrparen{\glsfirstlongfootnotefont{\glsaccesslongpl{##1}}}%
9477 }%
9478 }

```

postfootnote-em Backward compatibility:

```

9479 \@glsxtr@deprecated@abbrstyle{postfootnote-em}{short-em-postfootnote}

```

### 1.7.6 Predefined Styles (User Parentheses Hook)

These styles allow the user to adjust the parenthetical forms. These styles all test for the existence of the field given by:

`glxtruserfield` Default is the `useri` field.

```
9480 \newcommand*{\glxtruserfield}{useri}
```

`glxtruserparen` The format of the parenthetical information. The first argument is the long/short form. The second argument is the entry's label. If `\glscurrentfieldvalue` has been defined, then we have at least glossaries v4.23, which makes it easier for the user to adjust this.

```
9481 \ifdef\glscurrentfieldvalue
9482 {
9483   \newcommand*{\glxtruserparen}[2]{%
9484     \glxtrfullsep{#2}%
9485     \glxtrparen
9486     {#1\ifglshasfield{\glxtruserfield}{#2}{, \glscurrentfieldvalue}{}}%
9487   }
9488 }
9489 {
9490   \newcommand*{\glxtruserparen}[2]{%
9491     \glxtrfullsep{#2}%
9492     \glxtrparen
9493     {#1\ifglshasfield{\glxtruserfield}{#2}{, \@glo@thisvalue}{}}%
9494   }
9495 }
```

Font used for short form:

`lsabbrvuserfont`

```
9496 \newcommand*{\glabbrvuserfont}[1]{\glabbrvdefaultfont{#1}}
```

Font used for short form on first use:

`stabbrvuserfont`

```
9497 \newcommand*{\glfirstabbrvuserfont}[1]{\glabbrvuserfont{#1}}
```

Font used for long form:

`glslonguserfont`

```
9498 \newcommand*{\glslonguserfont}[1]{\glslongdefaultfont{#1}}
```

Font used for long form on first use:

`rstlonguserfont`

```
9499 \newcommand*{\glfirstlonguserfont}[1]{\glslonguserfont{#1}}
```

The default short form suffix:

`lsxtrusersuffix`

```
9500 \newcommand*{\glxtrusersuffix}{\glxtrabbrvpluralsuffix}
```

long-short-user

```
9501 \newabbreviationstyle{long-short-user}%
9502 {%
9503   \renewcommand*{\CustomAbbreviationFields}{%
9504     name={\glxtrlongshortname},
9505     sort={\the\glsshorttok},
9506     first={\protect\glsfirstlonguserfont{\the\glslongtok}%
9507       \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9508     {\the\glslabeltok}},%
9509     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9510       \protect\glxtruserparen
9511       {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9512     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9513     description={\protect\glslonguserfont{\the\glslongtok}}}%
9514 }
```

Unset the regular attribute if it has been set.

```
9514 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9515   \glshasattribute{\the\glslabeltok}{regular}%
9516   {%
9517     \glissetattribute{\the\glslabeltok}{regular}{false}%
9518   }%
9519   {}%
9520 }%
9521 }%
9522 {%
```

In case the user wants to mix and match font styles, these are redefined here.

```
9523 \renewcommand*{\abbrevpluralsuffix}{\glxtrusersuffix}%
9524 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9525 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9526 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9527 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%
```

The first use full form and the inline full form are the same for this style.

```
9528 \renewcommand*{\glxtrfullformat}[2]{%
9529   \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9530   \ifglxtrinsertinside\else##2\fi
9531   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9532 }%
9533 \renewcommand*{\glxtrfullplformat}[2]{%
9534   \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9535   \ifglxtrinsertinside\else##2\fi
9536   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9537 }%
9538 \renewcommand*{\Glsxtrfullformat}[2]{%
9539   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9540   \ifglxtrinsertinside\else##2\fi
9541   \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9542 }%
```

```

9543 \renewcommand*{\Glsxtrfullplformat}[2]{%
9544   \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9545   \ifglxtrinsertinside\else##2\fi
9546   \glxtruserparen{\glsfirstabbrvuserfont{\Glsaccessshortpl{##1}}}{##1}%
9547 }%
9548 }

```

-postshort-user Like long-short-user but defers the parenthetical matter to after the link.

```

9549 \newabbreviationstyle{long-postshort-user}%
9550 {%
9551   \renewcommand*{\CustomAbbreviationFields}{%
9552     name={\glxtrlongshortname},
9553     sort={\the\glsshorttok},
9554     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9555     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9556     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9557     description={\protect\glslonguserfont{\the\glslongtok}}}%
9558 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9559   \csdef{glxtrpostlink\glscategorylabel}{%
9560     \glxtrifwasfirstuse
9561     {%
9562       \glxtruserparen
9563       {\glsfirstabbrvuserfont{\glstryshort{\glslabel}}}%
9564       {\glslabel}%
9565     }%
9566     {}%
9567   }%
9568   \glshasattribute{\the\glslabeltok}{regular}%
9569   {%
9570     \glissetattribute{\the\glslabeltok}{regular}{false}%
9571   }%
9572   {}%
9573 }%
9574 }%
9575 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9576 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9577 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9578 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9579 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9580 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9581 \renewcommand*{\glxtrfullformat}[2]{%
9582   \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9583   \ifglxtrinsertinside\else##2\fi
9584 }%
9585 \renewcommand*{\glxtrfullplformat}[2]{%

```

```

9586 \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9587 \ifglxtrinsertinside\else##2\fi
9588 }%
9589 \renewcommand*{\Glsxtrfullformat}[2]{%
9590 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9591 \ifglxtrinsertinside\else##2\fi
9592 }%
9593 \renewcommand*{\Glsxtrfullplformat}[2]{%
9594 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9595 \ifglxtrinsertinside\else##2\fi
9596 }%

```

In-line format:

```

9597 \renewcommand*{\glxtrinlinefullformat}[2]{%
9598 \glsfirstlonguserfont{\glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9599 \ifglxtrinsertinside\else##2\fi
9600 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9601 }%
9602 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9603 \glsfirstlonguserfont{\glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9604 \ifglxtrinsertinside\else##2\fi
9605 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9606 }%
9607 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9608 \glsfirstlonguserfont{\Glsaccesslong{##1}\ifglxtrinsertinside##2\fi}%
9609 \ifglxtrinsertinside\else##2\fi
9610 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshort{##1}}}{##1}%
9611 }%
9612 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9613 \glsfirstlonguserfont{\Glsaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
9614 \ifglxtrinsertinside\else##2\fi
9615 \glxtruserparen{\glsfirstabbrvuserfont{\glsaccessshortpl{##1}}}{##1}%
9616 }%
9617 }

```

ortuserdesname

```

9618 \newcommand*{\glxtrlongshortuserdesname}{%
9619 \protect\glslonguserfont{\the\glslongtok}%
9620 \protect\glxtruserparen
9621 {\protect\glsabbrvuserfont{\the\glsshorttok}}{\the\glslabeltok}%
9622 }

```

short-user-desc Like long-postshort-user but the user supplies the description.

```

9623 \newabbreviationstyle{long-postshort-user-desc}%
9624 {%
9625 \renewcommand*{\CustomAbbreviationFields}{%
9626 name={\glxtrlongshortuserdesname},
9627 sort={\the\glslongtok},
9628 first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9629 firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%

```

```

9630     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9631     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%
9632 }%
9633 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9634   \csdef{glsxtrpostlink\glscategorylabel}{%
9635     \glsxtrifwasfirstuse
9636     {%
9637       \glsxtruserparen
9638       {\glsfirstabbrvuserfont{\glsentryshort{\glslabel}}}%
9639       {\glslabel}}%
9640     }%
9641     {}%
9642   }%
9643   \glshasattribute{\the\glslabeltok}{regular}%
9644   {%
9645     \glssetattribute{\the\glslabeltok}{regular}{false}%
9646   }%
9647   {}%
9648 }%
9649 }%
9650 {%
9651   \GlsXtrUseAbbrStyleFmts{long-postshort-user}%
9652 }

```

t-postlong-user Like short-long-user but defers the parenthetical matter to after the link.

```

9653 \newabbreviationstyle{short-postlong-user}%
9654 {%
9655   \renewcommand*{\CustomAbbreviationFields}{%
9656     name={\glsxtrshortlongname},
9657     sort={\the\glsshorttok},
9658     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9659     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}},%
9660     plural={\protect\glsabbrvuserfont{\the\glsshortpltok}},%
9661     description={\protect\glslonguserfont{\the\glslongtok}}}%
9662 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9663   \csdef{glsxtrpostlink\glscategorylabel}{%
9664     \glsxtrifwasfirstuse
9665     {%
9666       \glsxtruserparen
9667       {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9668       {\glslabel}}%
9669     }%
9670     {}%
9671   }%
9672   \glshasattribute{\the\glslabeltok}{regular}%
9673   {%
9674     \glssetattribute{\the\glslabeltok}{regular}{false}%
9675   }%
9676   {}%

```



```

9677 }%
9678 }%
9679 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9680 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9681 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvuserfont{##1}}%
9682 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvuserfont{##1}}%
9683 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonguserfont{##1}}%
9684 \renewcommand*{\glslongfont}[1]{\glslonguserfont{##1}}%

```

First use full form:

```

9685 \renewcommand*{\glxtrfullformat}[2]{%
9686   \glsfirstabbrvuserfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
9687   \ifglxtrininsertinside\else##2\fi
9688 }%
9689 \renewcommand*{\glxtrfullplformat}[2]{%
9690   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
9691   \ifglxtrininsertinside\else##2\fi
9692 }%
9693 \renewcommand*{\Glsxtrfullformat}[2]{%
9694   \glsfirstabbrvuserfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
9695   \ifglxtrininsertinside\else##2\fi
9696 }%
9697 \renewcommand*{\Glsxtrfullplformat}[2]{%
9698   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
9699   \ifglxtrininsertinside\else##2\fi
9700 }%

```

In-line format:

```

9701 \renewcommand*{\glxtrinlinefullformat}[2]{%
9702   \glsfirstabbrvuserfont{\glssaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
9703   \ifglxtrininsertinside\else##2\fi
9704   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9705 }%
9706 \renewcommand*{\glxtrinlinefullplformat}[2]{%
9707   \glsfirstabbrvuserfont{\glssaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
9708   \ifglxtrininsertinside\else##2\fi
9709   \glxtruserparen{\glsfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9710 }%
9711 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
9712   \glsfirstabbrvuserfont{\Glsaccessshort{##1}}\ifglxtrininsertinside##2\fi}%
9713   \ifglxtrininsertinside\else##2\fi
9714   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9715 }%
9716 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
9717   \glsfirstabbrvuserfont{\Glsaccessshortpl{##1}}\ifglxtrininsertinside##2\fi}%
9718   \ifglxtrininsertinside\else##2\fi
9719   \glxtruserparen{\glsfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9720 }%
9721 }

```

onguserdescname

```
9722 \newcommand*{\glxtrshortlonguserdescname}{%
9723   \protect\glsabbrvuserfont{\the\glsshorttok}%
9724   \protect\glxtruserparen
9725   {\protect\glsfirstlonguserfont{\the\glslongptok}}%
9726   {\the\glslabeltok}%
9727 }
```

long-user-desc Like short-postlong-user but leaves the user to specify the description.

```
9728 \newabbreviationstyle{short-postlong-user-desc}%
9729 {%
9730   \renewcommand*{\CustomAbbreviationFields}{%
9731     name={\glxtrshortlonguserdescname},
9732     sort={\the\glsshorttok},
9733     first={\protect\glsfirstlonguserfont{\the\glslongtok}},%
9734     firstplural={\protect\glsfirstlonguserfont{\the\glslongptok}},%
9735     text={\protect\glsabbrvuserfont{\the\glsshorttok}},%
9736     plural={\protect\glsabbrvuserfont{\the\glsshortptok}}%
9737   }%
9738   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9739     \csdef{glxtrpostlink\glscategorylabel}{%
9740       \glxtrifwasfirstuse
9741       {%
9742         \glxtruserparen
9743         {\glsfirstlonguserfont{\glsentrylong{\glslabel}}}%
9744         {\glslabel}%
9745       }%
9746     }%
9747   }%
9748   \glshasattribute{\the\glslabeltok}{regular}%
9749   {%
9750     \glissetattribute{\the\glslabeltok}{regular}{false}%
9751   }%
9752   {%
9753   }%
9754 }%
9755 {%
9756   \GlsXtrUseAbbrStyleFmts{short-postlong-user}%
9757 }
```

short-user-desc

```
9758 \newabbreviationstyle{long-short-user-desc}%
9759 {%
9760   \renewcommand*{\CustomAbbreviationFields}{%
9761     name={\glxtrlongshortuserdescname},
9762     sort={\glxtrlongshortdescsort},%
9763     first={\protect\glsfirstlonguserfont{\the\glslongtok}}%
```

```

9764     \protect\glxtruserparen{\protect\glsfirstabbrvuserfont{\the\glsshorttok}}}%
9765     {\the\glslabeltok}},%
9766     firstplural={\protect\glsfirstlonguserfont{\the\glslongpltok}%
9767     \protect\glxtruserparen
9768     {\protect\glsfirstabbrvuserfont{\the\glsshortpltok}}{\the\glslabeltok}},%
9769     text={\protect\glsabbrvfont{\the\glsshorttok}},%
9770     plural={\protect\glsabbrvfont{\the\glsshortpltok}}}%
9771 }%

```

Unset the regular attribute if it has been set.

```

9772 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9773   \glshasattribute{\the\glslabeltok}{regular}%
9774   {%
9775     \glissetattribute{\the\glslabeltok}{regular}{false}%
9776   }%
9777   {}%
9778 }%
9779 }%
9780 {%
9781   \GlsXtrUseAbbrStyleFmts{long-short-user}%
9782 }

```

short-long-user

```

9783 \newabbreviationstyle{short-long-user}%
9784 {%

```

\glslonguserfont is used in the description since \glsdesc doesn't set the style.

```

9785 \renewcommand*{\CustomAbbreviationFields}{%
9786   name={\glxtrshortlongname},
9787   sort={\the\glsshorttok},
9788   description={\protect\glslonguserfont{\the\glslongtok}},%
9789   first={\protect\glsfirstabbrvuserfont{\the\glsshorttok}%
9790     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongtok}}}%
9791     {\the\glslabeltok}},%
9792   firstplural={\protect\glsfirstabbrvuserfont{\the\glsshortpltok}%
9793     \protect\glxtruserparen{\protect\glsfirstlonguserfont{\the\glslongpltok}}}%
9794     {\the\glslabeltok}},%
9795   plural={\protect\glsabbrvuserfont{\the\glsshortpltok}}}%

```

Unset the regular attribute if it has been set.

```

9796 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9797   \glshasattribute{\the\glslabeltok}{regular}%
9798   {%
9799     \glissetattribute{\the\glslabeltok}{regular}{false}%
9800   }%
9801   {}%
9802 }%
9803 }%
9804 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9805 \renewcommand*{\abbrvpluralsuffix}{\glxtrusersuffix}%
9806 \renewcommand*{\glssabrvfont[1]}{\glssabrvuserfont{##1}}%
9807 \renewcommand*{\glssfirstabrvfont}[1]{\glssfirstabrvuserfont{##1}}%
9808 \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonguserfont{##1}}%
9809 \renewcommand*{\glsslongfont}[1]{\glsslonguserfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9810 \renewcommand*{\glxtrfullformat}[2]{%
9811   \glssfirstabrvuserfont{\glssaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9812   \ifglxtrininsertinside\else##2\fi
9813   \glxtruserparen{\glssfirstlonguserfont{\glssaccesslong{##1}}}{##1}%
9814 }%
9815 \renewcommand*{\glxtrfullplformat}[2]{%
9816   \glssfirstabrvuserfont{\glssaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9817   \ifglxtrininsertinside\else##2\fi
9818   \glxtruserparen{\glssfirstlonguserfont{\glssaccesslongpl{##1}}}{##1}%
9819 }%
9820 \renewcommand*{\Glsxtrfullformat}[2]{%
9821   \glssfirstabrvuserfont{\Glsaccessshort{##1}\ifglxtrininsertinside##2\fi}%
9822   \ifglxtrininsertinside\else##2\fi
9823   \glxtruserparen{\glssfirstlonguserfont{\Glsaccesslong{##1}}}{##1}%
9824 }%
9825 \renewcommand*{\Glsxtrfullplformat}[2]{%
9826   \glssfirstabrvuserfont{\Glsaccessshortpl{##1}\ifglxtrininsertinside##2\fi}%
9827   \ifglxtrininsertinside\else##2\fi
9828   \glxtruserparen{\glssfirstlonguserfont{\Glsaccesslongpl{##1}}}{##1}%
9829 }%
9830 }

```

-long-user-desc

```

9831 \newabbreviationstyle{short-long-user-desc}%
9832 {%
9833   \renewcommand*{\CustomAbbreviationFields}{%
9834     name={\glxtrshortlonguserdescname},
9835     sort={\glxtrshortlongdescsort},%
9836     first={\protect\glssfirstabrvuserfont{\the\glssshorttok}}%
9837     \protect\glxtruserparen{\protect\glssfirstlonguserfont{\the\glsslongtok}}%
9838     {\the\glsslabeltok}},%
9839     firstplural={\protect\glssfirstabrvuserfont{\the\glssshortpltok}}%
9840     \protect\glxtruserparen{\protect\glssfirstlonguserfont{\the\glsslongpltok}}%
9841     {\the\glsslabeltok}},%
9842     text={\protect\glssabrvfont{\the\glssshorttok}},%
9843     plural={\protect\glssabrvfont{\the\glssshortpltok}}%
9844   }%

```

Unset the regular attribute if it has been set.

```

9845 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9846   \glshasattribute{\the\glsslabeltok}{regular}%

```

```

9847   {%
9848     \glsetattribute{\the\glslabeltok}{regular}{false}%
9849   }%
9850   {}%
9851 }%
9852}%
9853{%
9854  \GlsXtrUseAbbrStyleFmts{short-long-user}%
9855 }

```

### 1.7.7 Predefined Styles (Hyphen)

These styles are designed to work with the `keywords` 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 `\glstrlongset` `\glsinsert` to empty with the entire link-text stored in `\glscustomtext`.

`\trifhyphenstart` Checks if the argument starts with a hyphen. The argument may be `\glsinsert` so check for that and expand.

```

9856 \newrobustcmd*{\glstrifhyphenstart}[3]{%
9857   \ifx\glsinsert#1\relax
9858     \expandafter\@glstrifhyphenstart#1\relax\relax
9859     \@end@glstrifhyphenstart{#2}{#3}%
9860   \else
9861     \@glstrifhyphenstart#1\relax\relax\@end@glstrifhyphenstart{#2}{#3}%
9862   \fi
9863 }

```

`\trifhyphenstart`

```

9864 \def\@glstrifhyphenstart#1#2\@end@glstrifhyphenstart#3#4{%
9865   \ifx-#1\relax#3\else #4\fi
9866 }

```

`\trlonghyphenshort`

```
\glstrlonghyphenshort{<label>}{<long>}{<short>}{<insert>}
```

The `<long>` and `<short>` arguments may be the plural form. The `<long>` argument may also be the first letter uppercase form.

```
9867 \newcommand*{\glstrlonghyphenshort}[4]{%
```

Grouping is needed to localise the redefinitions.

```
9868 {%
```

If `<insert>` starts with a hyphen, redefine `\glstrwordsep` to a hyphen. The inserted material is also inserted into the parenthetical part. (The inserted material is grouped as a precautionary measure.) No change is made to `\glstrwordsep` if `<insert>` doesn't start with a hyphen.

```
9869   \glstrifhyphenstart{#4}{\def\glstrwordsep{-}}{}%
```

```

9870 \glsfirstlonghyphenfont{#2\ifglxtrinsertinside{#4}\fi}%
9871 \ifglxtrinsertinside\else{#4}\fi
9872 \glxtrfullsep{#1}%
9873 \glxtrparen{\glsfirstabbrvhyphenfont{#3\ifglxtrinsertinside{#4}\fi}%
9874 \ifglxtrinsertinside\else{#4}\fi}%
9875 }%
9876 }

```

abbrvhyphenfont

```

9877 \newcommand*{\glsabbrvhyphenfont}{\glsabbrvdefaultfont}%

```

abbrvhyphenfont

```

9878 \newcommand*{\glsfirstabbrvhyphenfont}{\glsabbrvhyphenfont}%

```

slonghyphenfont

```

9879 \newcommand*{\glslonghyphenfont}{\glslongdefaultfont}%

```

tlonghyphenfont

```

9880 \newcommand*{\glsfirstlonghyphenfont}{\glslonghyphenfont}%

```

The default short form suffix:

xtrhyphensuffix

```

9881 \newcommand*{\glxtrhyphensuffix}{\glxtrabbrvpluralsuffix}

```

en-short-hyphen Designed for use with the markwords attribute.

```

9882 \newabbreviationstyle{long-hyphen-short-hyphen}%
9883 {%
9884 \renewcommand*{\CustomAbbreviationFields}{%
9885 name={\glxtrlongshortname},
9886 sort={\the\glsshorttok},
9887 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}%
9888 \protect\glxtrfullsep{\the\glslabeltok}%
9889 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}},%
9890 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}%
9891 \protect\glxtrfullsep{\the\glslabeltok}%
9892 \glxtrparen{\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}},%
9893 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}},%
9894 description={\protect\glslonghyphenfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```

9895 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9896 \glshasattribute{\the\glslabeltok}{regular}%
9897 {%
9898 \glssetattribute{\the\glslabeltok}{regular}{false}%
9899 }%
9900 {}%
9901 }%
9902 }%

```

```

9903 {%
9904   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
9905   \renewcommand*{\glssabbrvfont}[1]{\glssabbrvhyphenfont{##1}}%
9906   \renewcommand*{\glssfirstabbrvfont}[1]{\glssfirstabbrvhyphenfont{##1}}%
9907   \renewcommand*{\glssfirstlongfont}[1]{\glssfirstlonghyphenfont{##1}}%
9908   \renewcommand*{\glsslongfont}[1]{\glsslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

9909   \renewcommand*{\glxtrfullformat}[2]{%
9910     \glxtrlonghyphenshort{##1}{\glssaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9911   }%
9912   \renewcommand*{\glxtrfullplformat}[2]{%
9913     \glxtrlonghyphenshort{##1}{\glssaccesslongpl{##1}}%
9914     {\glssaccessshortpl{##1}}{##2}%
9915   }%
9916   \renewcommand*{\Glsxtrfullformat}[2]{%
9917     \glxtrlonghyphenshort{##1}{\Glsaccesslong{##1}}{\glssaccessshort{##1}}{##2}%
9918   }%
9919   \renewcommand*{\Glsxtrfullplformat}[2]{%
9920     \glxtrlonghyphenshort{##1}{\Glsaccesslongpl{##1}}%
9921     {\glssaccessshortpl{##1}}{##2}%
9922   }%
9923 }

```

ort-hyphen-desc Like long-hyphen-short-hyphen but the description must be supplied by the user.

```

9924 \newabbreviationstyle{long-hyphen-short-hyphen-desc}%
9925 {%
9926   \renewcommand*{\CustomAbbreviationFields}{%
9927     name={\glxtrlongshortdescname},
9928     sort={\glxtrlongshortdescsort},
9929     first={\protect\glssfirstlonghyphenfont{\the\glsslongtok}}%
9930     \protect\glxtrfullsep{\the\glsslabeltok}%
9931     \glxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshorttok}}},%
9932     firstplural={\protect\glssfirstlonghyphenfont{\the\glsslongpltok}}%
9933     \protect\glxtrfullsep{\the\glsslabeltok}%
9934     \glxtrparen{\protect\glssfirstabbrvhyphenfont{\the\glssshortpltok}}},%
9935     text={\protect\glssabbrvhyphenfont{\the\glssshorttok}},%
9936     plural={\protect\glssabbrvhyphenfont{\the\glssshortpltok}}}%
9937   }%

```

Unset the regular attribute if it has been set.

```

9938   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9939     \glshasattribute{\the\glsslabeltok}{regular}%
9940     {%
9941       \glsssetAttribute{\the\glsslabeltok}{regular}{false}%
9942     }%
9943   }%
9944 }%
9945 }%
9946 {%

```

```

9947 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%
9948 }

```

onghyphennoshort

```
\glxtrlonghyphennoshort{<label>}{<long>}{<insert>}
```

```

9949 \newcommand*{\glxtrlonghyphennoshort}[3]{%

```

Grouping is needed to localise the redefinitions.

```

9950 {%

```

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.

```

9951 \glxtrifhyphenstart{#3}{\def\glxtrwordsep{-}}{%
9952 \glsfirslonghyphenfont{#2\ifglxtrininsertinside{#3}\fi}%
9953 \ifglxtrininsertinside\else{#3}\fi
9954 }%
9955 }

```

short-desc-noreg

This version doesn't show the short form (except explicitly with `\glxtrshort`). Since `\glxtrshort` doesn't support the hyphen switch, the short form just uses the default short-form font command. This style won't work with the regular as the regular form isn't flexible enough.

```

9956 \newabbreviationstyle{long-hyphen-noshort-desc-noreg}%
9957 {%
9958 \renewcommand*{\CustomAbbreviationFields}{%
9959   name={\glxtrlongnoshortdescname},
9960   sort={\expandonce\glxtrorglong},
9961   first={\protect\glsfirslonghyphenfont{\the\glslongtok}},%
9962   firstplural={\protect\glsfirslonghyphenfont{\the\glslongpltok}},%
9963   plural={\protect\glslonghyphenfont{\the\glslongpltok}}%
9964 }%

```

Unset the regular attribute if it has been set.

```

9965 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
9966 \glshasattribute{\the\glslabeltok}{regular}%
9967 {%
9968 \glissetattribute{\the\glslabeltok}{regular}{false}%
9969 }%
9970 {}%
9971 }%
9972 }%
9973 {%
9974 \GlsXtrUseAbbrStyleFmts{long-hyphen-short-hyphen}%

```

In case the user wants to mix and match font styles, these are redefined here.

```

9975 \renewcommand*{\abbrvpluralsuffix}{\glxtrabbrvpluralsuffix}%

```



```

9976 \renewcommand*\glsabbrvfont[1]{\glsabbrvdefaultfont{##1}}%
9977 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvdefaultfont{##1}}%
9978 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
9979 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

The format for subsequent use (not used when the regular attribute is set).

```

9980 \renewcommand*\glsxtrsubsequentfmt[2]{%
9981   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9982 }%
9983 \renewcommand*\glsxtrsubsequentplfmt[2]{%
9984   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9985 }%
9986 \renewcommand*\Glsxtrsubsequentfmt[2]{%
9987   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
9988 }%
9989 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
9990   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
9991 }%

```

The inline full form displays the long format followed by the short form in parentheses.

```

9992 \renewcommand*\glsxtrinlinefullformat[2]{%
9993   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
9994   \glsxtrfullsep{##1}%
9995   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
9996 }%
9997 \renewcommand*\glsxtrinlinefullplformat[2]{%
9998   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
9999   \glsxtrfullsep{##1}%
10000   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
10001 }%
10002 \renewcommand*\Glsxtrinlinefullformat[2]{%
10003   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10004   \glsxtrfullsep{##1}%
10005   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshort{##1}}}%
10006 }%
10007 \renewcommand*\Glsxtrinlinefullplformat[2]{%
10008   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10009   \glsxtrfullsep{##1}%
10010   \glsxtrparen{\protect\glsfirstabbrvfont{\glsaccessshortpl{##1}}}%
10011 }%

```

The first use full form only displays the long form.

```

10012 \renewcommand*\glsxtrfullformat[2]{%
10013   \glsxtrlonghyphennoshort{##1}{\glsaccesslong{##1}}{##2}%
10014 }%
10015 \renewcommand*\glsxtrfullplformat[2]{%
10016   \glsxtrlonghyphennoshort{##1}{\glsaccesslongpl{##1}}{##2}%
10017 }%
10018 \renewcommand*\Glsxtrfullformat[2]{%
10019   \glsxtrlonghyphennoshort{##1}{\Glsaccesslong{##1}}{##2}%
10020 }%

```

```

10021 \renewcommand*{\Glsxtrfullplformat}[2]{%
10022   \glsxtrlonghyphennoshort{##1}{\Glsaccesslongpl{##1}}{##2}%
10023 }%
10024 }

```

n-noshort-noreg It doesn't really make a great deal of sense to have a long-only style that doesn't have a description (unless no glossary is required), but the best course of action here is to use the short form as the name and the long form as the description.

```

10025 \newabbreviationstyle{long-hyphen-noshort-noreg}%
10026 {%
10027   \renewcommand*{\CustomAbbreviationFields}{%
10028     name={\glsxtrlongnoshortname},
10029     sort={\the\glsshorttok},
10030     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10031     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10032     text={\protect\glslonghyphenfont{\the\glslongtok}},%
10033     plural={\protect\glslonghyphenfont{\the\glslongpltok}},%
10034     description={\the\glslongtok}%
10035   }%

```

Unset the regular attribute if it has been set.

```

10036 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10037   \glshasattribute{\the\glslabeltok}{regular}%
10038   {%
10039     \glissetattribute{\the\glslabeltok}{regular}{false}%
10040   }%
10041   {}%
10042 }%
10043 }%
10044 {%
10045   \GlsXtrUseAbbrStyleFmts{long-desc}%
10046 }

```

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.

```

10047 \newcommand*{\glsxtrlonghyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10048 {%
10049   \glsxtrifhyphenstart{#3}{\def\glsxtrwordsep{-}}{}%
10050   \glsfirstlonghyphenfont{#1}%
10051 }%
10052 }

```

posthyphenshort

`\glxtrposthyphenshort{<label>}{<insert>}`

Used in the post-link hook for the long-hyphen-postshort-hyphen style. Much like `\glxtrlonghyphenshort` but omits the *<long>* part. This always uses the singular short form.

```
10053 \newcommand*{\glxtrposthyphenshort}[2]{%
10054   {%
10055     \glxtrifhyphenstart{#2}{\def\glxtrwordsep{-}}{}}%
10056     \ifglxtrininsertinside{\glsfirstlonghyphenfont{#2}}\else{#2}\fi
10057     \glxtrfullsep{#1}%
10058     \glxtrparen
10059     {\glsfirstabbrvhyphenfont{\glstryshort{#1}}\ifglxtrininsertinside{#2}\fi}%
10060     \ifglxtrininsertinside\else{#2}\fi
10061   }%
10062 }%
10063 }
```

hyphensubsequent

`\glxtrposthyphensubsequent{<label>}{<insert>}`

Format in the post-link hook for subsequent use. The label is ignored by default.

```
10064 \newcommand*{\glxtrposthyphensubsequent}[2]{%
10065   \glsshortfont{\ifglxtrininsertinside {#2}\fi}%
10066   \ifglxtrininsertinside \else{#2}\fi
10067 }
```

postshort-hyphen

Like long-hyphen-short-hyphen but shifts the insert and parenthetical material to the post-link hook.

```
10068 \newabbreviationstyle{long-hyphen-postshort-hyphen}%
10069 {%
10070   \renewcommand*{\CustomAbbreviationFields}{%
10071     name={\glxtrlongshortname},
10072     sort={\the\glsshorttok},
10073     first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10074     firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10075     plural={\protect\glsshorthyphenfont{\the\glsshortpltok}},%
10076     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10077   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10078     \csdef{glxtrpostlink\glscategorylabel}{%
10079       \glxtrifwasfirstuse
10080       {%
10081         \glxtrposthyphenshort{\glslabel}{\glsininsert}%
10082       }%
10083     }%
```

Put the insertion into the post-link:

```

10084      \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10085    }%
10086  }%
10087  \glsattribute{\the\glslabeltok}{regular}%
10088  {%
10089    \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10090  }%
10091  {}%
10092 }%
10093 }%
10094 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10095 \renewcommand*\abbrvpluralsuffix{\glxtrabbrvpluralsuffix}%
10096 \renewcommand*\glsabbrvfont[1]{\glsabbrvhyphenfont{##1}}%
10097 \renewcommand*\glsfirstabbrvfont[1]{\glsfirstabbrvhyphenfont{##1}}%
10098 \renewcommand*\glsfirstlongfont[1]{\glsfirstlonghyphenfont{##1}}%
10099 \renewcommand*\glslongfont[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10100 \renewcommand*\glxtrsubsequentfmt[2]{%
10101   \glsabbrvfont{\glsaccessshort{##1}}%
10102 }%
10103 \renewcommand*\glxtrsubsequentplfmt[2]{%
10104   \glsabbrvfont{\glsaccessshortpl{##1}}%
10105 }%
10106 \renewcommand*\Glsxtrsubsequentfmt[2]{%
10107   \glsabbrvfont{\Glsaccessshort{##1}}%
10108 }%
10109 \renewcommand*\Glsxtrsubsequentplfmt[2]{%
10110   \glsabbrvfont{\Glsaccessshortpl{##1}}%
10111 }%

```

First use full form:

```

10112 \renewcommand*\glxtrfullformat[2]{%
10113   \glxtrlonghyphen{\glsaccesslong{##1}}{##1}{##2}%
10114 }%
10115 \renewcommand*\glxtrfullplformat[2]{%
10116   \glxtrlonghyphen{\glsaccesslongpl{##1}}{##1}{##2}%
10117 }%
10118 \renewcommand*\Glsxtrfullformat[2]{%
10119   \glxtrlonghyphen{\Glsaccesslong{##1}}{##1}{##2}%
10120 }%
10121 \renewcommand*\Glsxtrfullplformat[2]{%
10122   \glxtrlonghyphen{\Glsaccesslongpl{##1}}{##1}{##2}%
10123 }%

```

In-line format.

```

10124 \renewcommand*\glxtrinlinefullformat[2]{%
10125   \glsfirstlonghyphenfont{\glsaccesslong{##1}}%
10126   \ifglxtrininsertinside{##2}\fi}%

```

```

10127 \ifglxtrinsertinside \else{##2}\fi
10128 }%
10129 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10130 \glsfirstlonghyphenfont{\glsaccesslongpl{##1}%
10131 \ifglxtrinsertinside{##2}\fi}%
10132 \ifglxtrinsertinside \else{##2}\fi
10133 }%
10134 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10135 \glsfirstlonghyphenfont{\Glsaccesslong{##1}%
10136 \ifglxtrinsertinside{##2}\fi}%
10137 \ifglxtrinsertinside \else{##2}\fi
10138 }%
10139 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10140 \glsfirstlonghyphenfont{\Glsaccesslongpl{##1}%
10141 \ifglxtrinsertinside{##2}\fi}%
10142 \ifglxtrinsertinside \else{##2}\fi
10143 }%
10144 }

```

**ort-hyphen-desc** Like long-hyphen-postshort-hyphen but the description must be supplied by the user.

```

10145 \newabbreviationstyle{long-hyphen-postshort-hyphen-desc}%
10146 {%
10147 \renewcommand*{\CustomAbbreviationFields}{%
10148 name={\glxtrlongshortdescname},
10149 sort={\glxtrlongshortdescsort},%
10150 first={\protect\glsfirstlonghyphenfont{\the\glslongtok}},%
10151 firstplural={\protect\glsfirstlonghyphenfont{\the\glslongpltok}},%
10152 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10153 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10154 }%
10155 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10156 \csdef{glxtrpostlink\glscategorylabel}{%
10157 \glxtrifwasfirstuse
10158 {%
10159 \glxtrposthyphenshort{\glslabel}{\glsinsert}%
10160 }%
10161 {%

```

Put the insertion into the post-link:

```

10162 \glxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10163 }%
10164 }%
10165 \glshasattribute{\the\glslabeltok}{regular}%
10166 {%
10167 \glssetattribute{\the\glslabeltok}{regular}{false}%
10168 }%
10169 {}%
10170 }%
10171 }%
10172 {%

```

```

10173 \GlsXtrUseAbbrStyleFmts{long-hyphen-postshort-hyphen}%
10174 }

```

rshorthyphenlong

```
\glxtrshorthyphenlong{<label>}{<short>}{<long>}{<insert>}
```

The *<long>* and *<short>* arguments may be the plural form. The *<long>* argument may also be the first letter uppercase form.

```
10175 \newcommand*{\glxtrshorthyphenlong}[4]{%
```

Grouping is needed to localise the redefinitions.

```
10176 {%
```

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.)

```

10177 \glxtrifhyphenstart{#4}{\def\glxtrwordsep{-}}{}%
10178 \glsfirstabbrvhyphenfont{#2\ifglxtrininsertinside{#4}\fi}%
10179 \ifglxtrininsertinside\else{#4}\fi
10180 \glxtrfullsep{#1}%
10181 \glxtrparen{\glsfirstlonghyphenfont{#3\ifglxtrininsertinside{#4}\fi}%
10182 \ifglxtrininsertinside\else{#4}\fi}%
10183 }%
10184 }

```

hen-long-hyphen

Designed for use with the `markwords` attribute.

```

10185 \newabbreviationstyle{short-hyphen-long-hyphen}%
10186 {%
10187 \renewcommand*{\CustomAbbreviationFields}{%
10188 name={\glxtrshortlongname},
10189 sort={\the\glsshorttok},
10190 first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}%
10191 \protect\glxtrfullsep{\the\glslabeltok}%
10192 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10193 firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}%
10194 \protect\glxtrfullsep{\the\glslabeltok}%
10195 \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10196 plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}}},%
10197 description={\protect\glslonghyphenfont{\the\glslongtok}}}

```

Unset the regular attribute if it has been set.

```

10198 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10199 \glshasattribute{\the\glslabeltok}{regular}%
10200 {%
10201 \glissetattribute{\the\glslabeltok}{regular}{false}%
10202 }%
10203 {}%
10204 }%

```

```

10205 }%
10206 {%
10207   \renewcommand*{\abbrvpluralsuffix}{\glxtrhyphensuffix}%
10208   \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10209   \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10210   \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10211   \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

The first use full form and the inline full form are the same for this style.

```

10212   \renewcommand*{\glxtrfullformat}[2]{%
10213     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\glsaccesslong{##1}}{##2}%
10214   }%
10215   \renewcommand*{\glxtrfullplformat}[2]{%
10216     \glxtrshorthyphenlong{##1}%
10217     {\glsaccessshortpl{##1}}{\glsaccesslongpl{##1}}{##2}%
10218   }%
10219   \renewcommand*{\Glsxtrfullformat}[2]{%
10220     \glxtrshorthyphenlong{##1}{\glsaccessshort{##1}}{\Glsaccesslong{##1}}{##2}%
10221   }%
10222   \renewcommand*{\Glsxtrfullplformat}[2]{%
10223     \glxtrshorthyphenlong{##1}%
10224     {\glsaccessshortpl{##1}}{\Glsaccesslongpl{##1}}{##2}%
10225   }%
10226 }

```

ong-hyphen-desc Like short-hyphen-long-hyphen but the description must be supplied by the user.

```

10227 \newabbreviationstyle{short-hyphen-long-hyphen-desc}%
10228 {%
10229   \renewcommand*{\CustomAbbreviationFields}{%
10230     name={\glxtrshortlongdescname},
10231     sort={\glxtrshortlongdescsort},
10232     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}}%
10233     \protect\glxtrfullsep{\the\glslabeltok}%
10234     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongtok}}},%
10235     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}}%
10236     \protect\glxtrfullsep{\the\glslabeltok}%
10237     \glxtrparen{\protect\glsfirstlonghyphenfont{\the\glslongpltok}}},%
10238     text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10239     plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}%
10240   }%

```

Unset the regular attribute if it has been set.

```

10241   \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10242     \glshasattribute{\the\glslabeltok}{regular}%
10243     {%
10244       \glissetattribute{\the\glslabeltok}{regular}{false}%
10245     }%
10246   }%
10247 }%
10248 }%

```

```

10249 {%
10250   \GlsXtrUseAbbrStyleFmts{short-hyphen-long-hyphen}%
10251 }

```

`\glstrshorthyphen` `\glstrshorthyphen{<short>}{<label>}{<insert>}`

Used by short-hyphen-postlong-hyphen. The `<insert>` is check to determine if it starts with a hyphen but isn't used here as it's moved to the post-link hook.

```

10252 \newcommand*{\glstrshorthyphen}[3]{%

```

Grouping is needed to localise the redefinitions.

```

10253 {%
10254   \glstrifhyphenstart{#3}{\def\glstrwordsep{-}}{}%
10255   \glsfirstabbrvhyphenfont{#1}%
10256 }%
10257 }

```

`\glstrposthyphenlong` `\glstrposthyphenlong{<label>}{<insert>}`

Used in the post-link hook for the short-hyphen-postlong-hyphen style. Much like `\glstrshorthyphenlong` but omits the `<short>` part. This always uses the singular long form.

```

10258 \newcommand*{\glstrposthyphenlong}[2]{%
10259 {%
10260   \glstrifhyphenstart{#2}{\def\glstrwordsep{-}}{}%
10261   \ifglstrinsertinside{\glsfirstabbrvhyphenfont{#2}}\else{#2}\fi
10262   \glstrfullsep{#1}%
10263   \glstrparen
10264   {\glsfirstlonghyphenfont{\glsentrylong{#1}}\ifglstrinsertinside{#2}\fi}%
10265   \ifglstrinsertinside\else{#2}\fi
10266 }%
10267 }%
10268 }

```

`postlong-hyphen` Like short-hyphen-long-hyphen but shifts the insert and parenthetical material to the post-link hook.

```

10269 \newabbreviationstyle{short-hyphen-postlong-hyphen}%
10270 {%
10271   \renewcommand*{\CustomAbbreviationFields}{%
10272     name={\glstrshortlongname},
10273     sort={\the\glsshorttok},
10274     first={\protect\glsfirstabbrvhyphenfont{\the\glsshorttok}},%
10275     firstplural={\protect\glsfirstabbrvhyphenfont{\the\glsshortpltok}},%
10276     plural={\protect\glssabbrvhyphenfont{\the\glsshortpltok}},%

```



```

10277     description={\protect\glslonghyphenfont{\the\glslongtok}}}%
10278 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10279     \csdef{glsxtrpostlink\glscategorylabel}{%
10280         \glsxtrifwasfirstuse
10281         {%
10282             \glsxtrposthyphenlong{\glslabel}{\glsinsert}%
10283         }%
10284     }%

```

Put the insertion into the post-link:

```

10285         \glsxtrposthyphensubsequent{\glslabel}{\glsinsert}%
10286     }%
10287 }%
10288 \glsattribute{\the\glslabeltok}{regular}%
10289 {%
10290     \glssetattribute{\the\glslabeltok}{regular}{false}%
10291 }%
10292 {}%
10293 }%
10294 }%
10295 {%

```

In case the user wants to mix and match font styles, these are redefined here.

```

10296 \renewcommand*{\abbrvpluralsuffix}{\glsxtrabbrvpluralsuffix}%
10297 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvhyphenfont{##1}}%
10298 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvhyphenfont{##1}}%
10299 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlonghyphenfont{##1}}%
10300 \renewcommand*{\glslongfont}[1]{\glslonghyphenfont{##1}}%

```

Subsequent use needs to omit the insertion:

```

10301 \renewcommand*{\glsxtrsubsequentfmt}[2]{%
10302     \glsabbrvfont{\glsaccessshort{##1}}}%
10303 }%
10304 \renewcommand*{\glsxtrsubsequentplfmt}[2]{%
10305     \glsabbrvfont{\glsaccessshortpl{##1}}}%
10306 }%
10307 \renewcommand*{\Glsxtrsubsequentfmt}[2]{%
10308     \glsabbrvfont{\Glsaccessshort{##1}}}%
10309 }%
10310 \renewcommand*{\Glsxtrsubsequentplfmt}[2]{%
10311     \glsabbrvfont{\Glsaccessshortpl{##1}}}%
10312 }%

```

First use full form:

```

10313 \renewcommand*{\glsxtrfullformat}[2]{%
10314     \glsxtrshorthyphen{\glsaccessshort{##1}}{##1}{##2}%
10315 }%
10316 \renewcommand*{\glsxtrfullplformat}[2]{%
10317     \glsxtrshorthyphen{\glsaccessshortpl{##1}}{##1}{##2}%
10318 }%
10319 \renewcommand*{\Glsxtrfullformat}[2]{%

```

```

10320 \glstrshortshyphen{\Glsaccessshort{##1}}{##1}{##2}%
10321 }%
10322 \renewcommand*{\Glsxtrfullplformat}[2]{%
10323 \glstrshortshyphen{\Glsaccessshortpl{##1}}{##1}{##2}%
10324 }%

```

In-line format. Commands like `\glstrfull` set `\glinsert` to empty. The entire link-text (provided by the following commands) is stored in `\glscustomtext`.

```

10325 \renewcommand*{\glxtrinlinefullformat}[2]{%
10326 \glfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10327 \ifglxtrininsertinside{##2}\fi}%
10328 \ifglxtrininsertinside \else{##2}\fi
10329 }%
10330 \renewcommand*{\glxtrinlinefullplformat}[2]{%
10331 \glfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10332 \ifglxtrininsertinside{##2}\fi}%
10333 \ifglxtrininsertinside \else{##2}\fi
10334 }%
10335 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10336 \glfirstabbrvhyphenfont{\Glsaccessshort{##1}}%
10337 \ifglxtrininsertinside{##2}\fi}%
10338 \ifglxtrininsertinside \else{##2}\fi
10339 }%
10340 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10341 \glfirstabbrvhyphenfont{\Glsaccessshortpl{##1}}%
10342 \ifglxtrininsertinside{##2}\fi}%
10343 \ifglxtrininsertinside \else{##2}\fi
10344 }%
10345 }

```

`ong-hyphen-desc` Like `short-hyphen-postlong-hyphen` but the description must be supplied by the user.

```

10346 \newabbreviationstyle{short-hyphen-postlong-hyphen-desc}%
10347 {%
10348 \renewcommand*{\CustomAbbreviationFields}{%
10349 name={\glstrshortlongdescname},
10350 sort={\glstrshortlongdescsort},%
10351 first={\protect\glfirstabbrvhyphenfont{\the\glsshorttok}},%
10352 firstplural={\protect\glfirstabbrvhyphenfont{\the\glsshortpltok}},%
10353 text={\protect\glsabbrvhyphenfont{\the\glsshorttok}},%
10354 plural={\protect\glsabbrvhyphenfont{\the\glsshortpltok}}}%
10355 }%
10356 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10357 \csdef{glxtrpostlink\glscategorylabel}{%
10358 \glxtrifwasfirstuse
10359 {%
10360 \glxtrposthyphenlong{\glslabel}{\glinsert}%
10361 }%
10362 }%

```

Put the insertion into the post-link:

```

10363      \glstrposthyphensubsequent{\glslabel}{\glsinsert}%
10364    }%
10365  }%
10366  \glshasattribute{\the\glslabeltok}{regular}%
10367  {%
10368    \glissetattribute{\the\glslabeltok}{regular}{false}%
10369  }%
10370  {}%
10371 }%
10372 }%
10373 {%
10374   \GlsXtrUseAbbrStyleFmts{short-hyphen-postlong-hyphen}%
10375 }

```

### 1.7.8 Predefined Styles (No Short on First Use)

These styles show only the long form on first use and only the short form on subsequent use.

lsabbrvonlyfont

```
10376 \newcommand*{\glsabbrvonlyfont}{\glsabbrvdefaultfont}%
```

stabbrvonlyfont

```
10377 \newcommand*{\glsfirstabbrvonlyfont}{\glsabbrvonlyfont}%
```

glslongonlyfont

```
10378 \newcommand*{\glslongonlyfont}{\glslongdefaultfont}%
```

rstlongonlyfont

```
10379 \newcommand*{\glsfirstlongonlyfont}{\glslongonlyfont}%
```

The default short form suffix:

lsxtronlysuffix

```
10380 \newcommand*{\glsxtronlysuffix}{\glsxtrabbrvpluralsuffix}
```

\glsxtronlyname The default name format for this style.

```

10381 \newcommand*{\glsxtronlyname}{%
10382   \protect\glsabbrvonlyfont{\the\glsshorttok}%
10383 }

```

only-short-only

```

10384 \newabbreviationstyle{long-only-short-only}%
10385 {%
10386   \renewcommand*{\CustomAbbreviationFields}{%
10387     name={\glsxtronlyname},
10388     sort={\the\glsshorttok},
10389     first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10390     firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10391     plural={\protect\glsabbrvonlyfont{\the\glsshortpltok}},%
10392     description={\protect\glslongonlyfont{\the\glslongtok}}}%

```

Unset the regular attribute if it has been set.

```
10393 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10394   \glshasattribute{\the\glslabeltok}{regular}%
10395   {%
10396     \glsetattribute{\the\glslabeltok}{regular}{false}%
10397   }%
10398   {}%
10399 }%
10400}%
10401{%
10402 \renewcommand*{\abbrvpluralsuffix}{\protect\glsxtronlysuffix}%
10403 \renewcommand*{\glsabbrvfont}[1]{\glsabbrvonlyfont{##1}}%
10404 \renewcommand*{\glsfirstabbrvfont}[1]{\glsfirstabbrvonlyfont{##1}}%
10405 \renewcommand*{\glsfirstlongfont}[1]{\glsfirstlongonlyfont{##1}}%
10406 \renewcommand*{\glslongfont}[1]{\glslongonlyfont{##1}}%
```

The first use full form doesn't show the short form.

```
10407 \renewcommand*{\glsxtrfullformat}[2]{%
10408   \glsfirstlongonlyfont{\glssaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
10409   \ifglsxtrinsertinside\else##2\fi
10410 }%
10411 \renewcommand*{\glsxtrfullplformat}[2]{%
10412   \glsfirstlongonlyfont{\glssaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
10413   \ifglsxtrinsertinside\else##2\fi
10414 }%
10415 \renewcommand*{\Glsxtrfullformat}[2]{%
10416   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
10417   \ifglsxtrinsertinside\else##2\fi
10418 }%
10419 \renewcommand*{\Glsxtrfullplformat}[2]{%
10420   \glsfirstlongonlyfont{\Glsaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
10421   \ifglsxtrinsertinside\else##2\fi
10422 }%
```

The inline full form does show the short form.

```
10423 \renewcommand*{\glsxtrinlinefullformat}[2]{%
10424   \glsfirstlongonlyfont{\glssaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
10425   \ifglsxtrinsertinside\else##2\fi
10426   \glsxtrfullsep{##1}%
10427   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glssaccessshort{##1}}}%
10428 }%
10429 \renewcommand*{\glsxtrinlinefullplformat}[2]{%
10430   \glsfirstlongonlyfont{\glssaccesslongpl{##1}\ifglsxtrinsertinside##2\fi}%
10431   \ifglsxtrinsertinside\else##2\fi
10432   \glsxtrfullsep{##1}%
10433   \glsxtrparen{\protect\glsfirstabbrvonlyfont{\glssaccessshortpl{##1}}}%
10434 }%
10435 \renewcommand*{\Glsxtrinlinefullformat}[2]{%
10436   \glsfirstlongonlyfont{\Glsaccesslong{##1}\ifglsxtrinsertinside##2\fi}%
10437   \ifglsxtrinsertinside\else##2\fi
```

```

10438 \glxtrfullsep{##1}%
10439 \glxtrparen{\protect\glsfirstabbrvonlyfont{\glssaccessshortpl{##1}}}%
10440 }%
10441 \renewcommand*{\Glsxtrinlinefullplformat}[2]{%
10442 \glsfirstlongonlyfont{\glssaccesslongpl{##1}\ifglxtrinsertinside##2\fi}%
10443 \ifglxtrinsertinside\else##2\fi
10444 \glxtrfullsep{##1}%
10445 \glxtrparen{\protect\glsfirstabbrvonlyfont{\glssaccessshortpl{##1}}}%
10446 }%
10447 }

```

xtronlydescsort

```

10448 \newcommand*{\glxtronlydescsort}{\the\glslongtok}

```

xtronlydescname

```

10449 \newcommand*{\glxtronlydescname}{%
10450 \protect\glslongfont{\the\glslongtok}%
10451 }

```

short-only-desc

```

10452 \newabbreviationstyle{long-only-short-only-desc}%
10453 {%
10454 \renewcommand*{\CustomAbbreviationFields}{%
10455 name={\glxtronlydescname},
10456 sort={\glxtronlydescsort},%
10457 first={\protect\glsfirstlongonlyfont{\the\glslongtok}},%
10458 firstplural={\protect\glsfirstlongonlyfont{\the\glslongpltok}},%
10459 text={\protect\glssabbrvonlyfont{\the\glssshorttok}},%
10460 plural={\protect\glssabbrvonlyfont{\the\glssshortpltok}}}%
10461 }%

```

Unset the regular attribute if it has been set.

```

10462 \renewcommand*{\GlsXtrPostNewAbbreviation}{%
10463 \glshasattribute{\the\glslabeltok}{regular}%
10464 {%
10465 \glsssetAttribute{\the\glslabeltok}{regular}{false}%
10466 }%
10467 {}%
10468 }%
10469 }%
10470 {%
10471 \GlsXtrUseAbbrStyleFmts{long-only-short-only}%
10472 }

```

## 1.8 Using Entries in Headings

There are four main problems with using entries in sectioning commands: they can mess with the first use flag if they end up in the table of contents, they can add unwanted numbers to the entry's location list, the label is corrupted if used inside `\MakeUppercase` (which

is used by the default headings style) and they need to be expandable for PDF bookmarks. The glossaries package therefore recommends the use of the expandable commands, such as `\glsentryshort`, instead but this doesn't reflect the formatting since it doesn't include `\glsabbrvfont`. The commands below are an attempt to get around these problems.

The PDF bookmark issue can easily be fixed with `hyperref`'s `\texorpdfstring` which can simply use the expandable command in the PDF string case. The  $\TeX$  string case can now use `\glsxtrshort` with the `noindex` key set, which prevents the unwanted additions to the location list, and the `hyper` key set to false, which prevents the problem of nested links. This just leaves one thing left that needs to be dealt with, and that's what to do if the heading style uses `\MakeUppercase`.

Note that `glossaries` automatically loads `textcase`, so the label can be protected from case change with `textcase`'s `\NoCaseChange`. This means that we don't have a problem provided the page style uses `\MakeTextUppercase`, but the default heading page style uses `\MakeUppercase`.

To get around this, save the original definition of `\markboth` and `\markright` and adjust it so that `\MakeUppercase` is temporarily redefined to `\MakeTextUppercase`. Some packages or classes redefine these commands, so we can't just assume they still have the original kernel definition.

`\markright` Save original definition:

```
10473 \let\@glsxtr@org@markright\markright
```

Redefine (grouping not added in case it interferes with the original code):

```
10474 \renewcommand*{\markright}[1]{%
```

```
10475   \glsxtrmarkhook
```

```
10476   \@glsxtr@org@markright{\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
10477   \glsxtrrestoremarkhook
```

```
10478 }
```

`\markboth` Save original definition:

```
10479 \let\@glsxtr@org@markboth\markboth
```

Redefine (grouping not added in case it interferes with the original code):

```
10480 \renewcommand*{\markboth}[2]{%
```

```
10481   \glsxtrmarkhook
```

```
10482   \@glsxtr@org@markboth
```

```
10483     {\@glsxtrinmark#1\@glsxtrnotinmark}%
```

```
10484     {\@glsxtrinmark#2\@glsxtrnotinmark}%
```

```
10485   \glsxtrrestoremarkhook
```

```
10486 }
```

Also do this for `\@starttoc`

`\@starttoc` Save original definition:

```
10487 \let\@glsxtr@org@@starttoc\@starttoc
```

Redefine:

```
10488 \renewcommand*{\@starttoc}[1]{%
```

```

10489 \glsxtrmarkhook
10490 \@glsxtrinmark
10491 \@glsxtr@org@starttoc{#1}%
10492 \@glsxtrnotinmark
10493 \glsxtrrestoremarkhook
10494 }

```

If this causes a problem provide a simple way of switching back to the original definitions:

sxtrRevertMarks

```

10495 \newcommand*{\glsxtrRevertMarks}{%
10496   \let\markright\@glsxtr@org@markright
10497   \let\markboth\@glsxtr@org@markboth
10498   \let\starttoc\@glsxtr@org@starttoc
10499 }

```

\glsxtrifinmark

```

10500 \newcommand*{\glsxtrifinmark}[2]{#2}

```

\@glsxtrinmark

```

10501 \newrobustcmd*{\@glsxtrinmark}{%
10502   \let\glsxtrifinmark\@firstoftwo
10503 }

```

glsxtrnotinmark

```

10504 \newrobustcmd*{\@glsxtrnotinmark}{%
10505   \let\glsxtrifinmark\@secondoftwo
10506 }

```

eorpdforheading

```

10507 \ifdef\teorpdfstring
10508 {
10509   \newcommand*{\glsxtrtitleorpdforheading}[3]{\teorpdfstring{#1}{#2}}
10510 }
10511 {
10512   \newcommand*{\glsxtrtitleorpdforheading}[3]{#1}
10513 }

```

\glsxtrmarkhook Hook used in new definition of \markboth and \markright to make some changes to apply to the marks:

```

10514 \newcommand*{\glsxtrmarkhook}{%

```

Save current definitions:

```

10515   \let\@glsxtr@org@MakeUppercase\MakeUppercase
10516   \let\@glsxtr@org@glsxtrtitleorpdforheading\glsxtrtitleorpdforheading
10517   \let\@glsxtr@org@glsxtrtitleshort\glsxtrtitleshort
10518   \let\@glsxtr@org@glsxtrtitleshortpl\glsxtrtitleshortpl
10519   \let\@glsxtr@org@Glsxtrtitleshort\Glsxtrtitleshort
10520   \let\@glsxtr@org@Glsxtrtitleshortpl\Glsxtrtitleshortpl

```

```

10521 \let\@glsxtr@org@glsxtrtitlename\glsxtrtitlename
10522 \let\@glsxtr@org@Glsxtrtitlename\Glsxtrtitlename
10523 \let\@glsxtr@org@glsxtrtitletext\glsxtrtitletext
10524 \let\@glsxtr@org@Glsxtrtitletext\Glsxtrtitletext
10525 \let\@glsxtr@org@glsxtrtitleplural\glsxtrtitleplural
10526 \let\@glsxtr@org@Glsxtrtitleplural\Glsxtrtitleplural
10527 \let\@glsxtr@org@glsxtrtitlefirst\glsxtrtitlefirst
10528 \let\@glsxtr@org@Glsxtrtitlefirst\Glsxtrtitlefirst
10529 \let\@glsxtr@org@glsxtrtitlefirstplural\glsxtrtitlefirstplural
10530 \let\@glsxtr@org@Glsxtrtitlefirstplural\Glsxtrtitlefirstplural
10531 \let\@glsxtr@org@glsxtrtitlelong\glsxtrtitlelong
10532 \let\@glsxtr@org@glsxtrtitlelongpl\glsxtrtitlelongpl
10533 \let\@glsxtr@org@Glsxtrtitlelong\Glsxtrtitlelong
10534 \let\@glsxtr@org@Glsxtrtitlelongpl\Glsxtrtitlelongpl
10535 \let\@glsxtr@org@glsxtrtitlefull\glsxtrtitlefull
10536 \let\@glsxtr@org@glsxtrtitlefullpl\glsxtrtitlefullpl
10537 \let\@glsxtr@org@Glsxtrtitlefull\Glsxtrtitlefull
10538 \let\@glsxtr@org@Glsxtrtitlefullpl\Glsxtrtitlefullpl

```

#### New definitions

```

10539 \let\glsxtrifinmark\@firstoftwo
10540 \let\MakeUppercase\MakeTextUppercase
10541 \let\glsxtrtitleorpdforheading\@thirdofthree
10542 \let\glsxtrtitleshort\glsxtrheadshort
10543 \let\glsxtrtitleshortpl\glsxtrheadshortpl
10544 \let\Glsxtrtitleshort\Glsxtrheadshort
10545 \let\Glsxtrtitleshortpl\Glsxtrheadshortpl
10546 \let\glsxtrtitlename\glsxtrheadname
10547 \let\Glsxtrtitlename\Glsxtrheadname
10548 \let\glsxtrtitletext\glsxtrheadtext
10549 \let\Glsxtrtitletext\Glsxtrheadtext
10550 \let\glsxtrtitleplural\glsxtrheadplural
10551 \let\Glsxtrtitleplural\Glsxtrheadplural
10552 \let\glsxtrtitlefirst\glsxtrheadfirst
10553 \let\Glsxtrtitlefirst\Glsxtrheadfirst
10554 \let\glsxtrtitlefirstplural\glsxtrheadfirstplural
10555 \let\Glsxtrtitlefirstplural\Glsxtrheadfirstplural
10556 \let\glsxtrtitlelong\glsxtrheadlong
10557 \let\glsxtrtitlelongpl\glsxtrheadlongpl
10558 \let\Glsxtrtitlelong\Glsxtrheadlong
10559 \let\Glsxtrtitlelongpl\Glsxtrheadlongpl
10560 \let\glsxtrtitlefull\glsxtrheadfull
10561 \let\glsxtrtitlefullpl\glsxtrheadfullpl
10562 \let\Glsxtrtitlefull\Glsxtrheadfull
10563 \let\Glsxtrtitlefullpl\Glsxtrheadfullpl
10564 }

```

`\restoremarkhook` Hook used in new definition of `\markboth` and `\markright` to restore the modified definitions. (This is in case the original `\markboth` and `\markright` shouldn't be grouped for some reason. There already is some grouping within those original definitions, but some of



the code lies outside that grouping, and possibly there's a reason for it.)

```

10565 \newcommand*{\glxtrrestoremarkhook}{%
10566   \let\glxtrifinmark\@secondoftwo
10567   \let\MakeUppercase\@glxtr@org@MakeUppercase
10568   \let\glxtrtitleorpdforheading\@glxtr@org@glxtrtitleorpdforheading
10569   \let\glxtrtitleshort\@glxtr@org@glxtrtitleshort
10570   \let\glxtrtitleshortpl\@glxtr@org@glxtrtitleshortpl
10571   \let\Glsxtrtitleshort\@glxtr@org@Glsxtrtitleshort
10572   \let\Glsxtrtitleshortpl\@glxtr@org@Glsxtrtitleshortpl
10573   \let\glxtrtitlename\@glxtr@org@glxtrtitlename
10574   \let\Glsxtrtitlename\@glxtr@org@Glsxtrtitlename
10575   \let\glxtrtitletext\@glxtr@org@glxtrtitletext
10576   \let\Glsxtrtitletext\@glxtr@org@Glsxtrtitletext
10577   \let\glxtrtitleplural\@glxtr@org@glxtrtitleplural
10578   \let\Glsxtrtitleplural\@glxtr@org@Glsxtrtitleplural
10579   \let\glxtrtitlefirst\@glxtr@org@glxtrtitlefirst
10580   \let\Glsxtrtitlefirst\@glxtr@org@Glsxtrtitlefirst
10581   \let\glxtrtitlefirstplural\@glxtr@org@glxtrtitlefirstplural
10582   \let\Glsxtrtitlefirstplural\@glxtr@org@Glsxtrtitlefirstplural
10583   \let\glxtrtitlelong\@glxtr@org@glxtrtitlelong
10584   \let\glxtrtitlelongpl\@glxtr@org@glxtrtitlelongpl
10585   \let\Glsxtrtitlelong\@glxtr@org@Glsxtrtitlelong
10586   \let\Glsxtrtitlelongpl\@glxtr@org@Glsxtrtitlelongpl
10587   \let\glxtrtitlefull\@glxtr@org@glxtrtitlefull
10588   \let\glxtrtitlefullpl\@glxtr@org@glxtrtitlefullpl
10589   \let\Glsxtrtitlefull\@glxtr@org@Glsxtrtitlefull
10590   \let\Glsxtrtitlefullpl\@glxtr@org@Glsxtrtitlefullpl
10591 }

```

Instead of using one document-wide conditional, use headuc attribute to determine whether or not to use the all upper case form.

**glxtrheadshort** Command used to display short form in the page header.

```

10592 \newcommand*{\glxtrheadshort}[1]{%
10593   \protect\NoCaseChange
10594   {%
10595     \gl@ifattribute{#1}{headuc}{true}%
10596     {%
10597       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10598     }%
10599     {%
10600       \glxtrshort[noindex,hyper=false]{#1}[]%
10601     }%
10602   }%
10603 }

```

**lsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents.

```

10604 \newrobustcmd*{\lsxtrtitleshort}[1]{%
10605   \glxtrshort[noindex,hyper=false]{#1}[]%

```

10606 }

**sxtrheadshortpl** Command used to display plural short form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrshortpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```
10607 \newcommand*{\glsxtrheadshortpl}[1]{%
10608   \protect\NoCaseChange
10609   {%
10610     \gl@ifattribute{#1}{headuc}{true}%
10611     {%
10612       \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10613     }%
10614     {%
10615       \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10616     }%
10617   }%
10618 }
```

**xtrtitleshortpl** Command to display plural short form of abbreviation in section title and table of contents.

```
10619 \newrobustcmd*{\glsxtrtitleshortpl}[1]{%
10620   \glsxtrshortpl[noindex,hyper=false]{#1}[]%
10621 }
```

**Glsxtrheadshort** Command used to display short form in the page header with the first letter converted to upper case.

```
10622 \newcommand*{\Glsxtrheadshort}[1]{%
10623   \protect\NoCaseChange
10624   {%
10625     \gl@ifattribute{#1}{headuc}{true}%
10626     {%
10627       \GLSxtrshort[noindex,hyper=false]{#1}[]%
10628     }%
10629     {%
10630       \Glsxtrshort[noindex,hyper=false]{#1}[]%
10631     }%
10632   }%
10633 }
```

**lxsxtrtitleshort** Command to display short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```
10634 \newrobustcmd*{\Glsxtrtitleshort}[1]{%
10635   \Glsxtrshort[noindex,hyper=false]{#1}[]%
10636 }
```

**sxtrheadshortpl** Command used to display plural short form in the page header with the first letter converted to upper case.

```
10637 \newcommand*{\Glsxtrheadshortpl}[1]{%
10638   \protect\NoCaseChange
```

```

10639 {%
10640   \glsifattribute{#1}{headuc}{true}%
10641   {%
10642     \GLSxtrshortpl[noindex,hyper=false]{#1}[]%
10643   }%
10644   {%
10645     \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10646   }%
10647 }%
10648 }

```

`xtrtitleshortpl` Command to display plural short form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10649 \newrobustcmd*{\Glsxtrtitleshortpl}[1]{%
10650   \Glsxtrshortpl[noindex,hyper=false]{#1}[]%
10651 }

```

`\glsxtrheadname` As above but for the name value.

```

10652 \newcommand*{\glsxtrheadname}[1]{%
10653   \protect\NoCaseChange
10654   {%
10655     \glsifattribute{#1}{headuc}{true}%
10656     {%
10657       \GLSname[noindex,hyper=false]{#1}[]%
10658     }%
10659     {%
10660       \glsname[noindex,hyper=false]{#1}[]%
10661     }%
10662   }%
10663 }

```

`glsxtrtitlename` Command to display name value in section title and table of contents.

```

10664 \newrobustcmd*{\glsxtrtitlename}[1]{%
10665   \glsname[noindex,hyper=false]{#1}[]%
10666 }

```

`\Glsxtrheadname` First letter converted to upper case

```

10667 \newcommand*{\Glsxtrheadname}[1]{%
10668   \protect\NoCaseChange
10669   {%
10670     \glsifattribute{#1}{headuc}{true}%
10671     {%
10672       \GLSname[noindex,hyper=false]{#1}[]%
10673     }%
10674     {%
10675       \Glsname[noindex,hyper=false]{#1}[]%
10676     }%
10677   }%
10678 }

```

`Glsxtrtitlename` Command to display name value in section title and table of contents with the first letter changed to upper case.

```
10679 %\changes{1.21}{2017-11-03}{new}
10680 \newrobustcmd*{\Glsxtrtitlename}[1]{%
10681   \Glsname[noindex,hyper=false]{#1}[]%
10682 }
```

`\glsxtrheadtext` As above but for the text value.

```
10683 \newcommand*{\glsxtrheadtext}[1]{%
10684   \protect\NoCaseChange
10685   {%
10686     \glsifattribute{#1}{headuc}{true}%
10687     {%
10688       \GLStext[noindex,hyper=false]{#1}[]%
10689     }%
10690     {%
10691       \glstext[noindex,hyper=false]{#1}[]%
10692     }%
10693   }%
10694 }
```

`glsxtrtitletext` Command to display text value in section title and table of contents.

```
10695 \newrobustcmd*{\glsxtrtitletext}[1]{%
10696   \glstext[noindex,hyper=false]{#1}[]%
10697 }
```

`\Glsxtrheadtext` First letter converted to upper case

```
10698 \newcommand*{\Glsxtrheadtext}[1]{%
10699   \protect\NoCaseChange
10700   {%
10701     \glsifattribute{#1}{headuc}{true}%
10702     {%
10703       \GLStext[noindex,hyper=false]{#1}[]%
10704     }%
10705     {%
10706       \Glstext[noindex,hyper=false]{#1}[]%
10707     }%
10708   }%
10709 }
```

`Glsxtrtitletext` Command to display text value in section title and table of contents with the first letter changed to upper case.

```
10710 \newrobustcmd*{\Glsxtrtitletext}[1]{%
10711   \Glstext[noindex,hyper=false]{#1}[]%
10712 }
```

`lsxtrheadplural` As above but for the plural value.

```
10713 \newcommand*{\lsxtrheadplural}[1]{%
```

```

10714 \protect\NoCaseChange
10715 {%
10716   \glsifattribute{#1}{headuc}{true}%
10717   {%
10718     \GLSplural[noindex,hyper=false]{#1}[]%
10719   }%
10720   {%
10721     \Glsplural[noindex,hyper=false]{#1}[]%
10722   }%
10723 }%
10724 }

```

`sxtrtitleplural` Command to display plural value in section title and table of contents.

```

10725 \newrobustcmd*{\glsxtrtitleplural}[1]{%
10726   \Glsplural[noindex,hyper=false]{#1}[]%
10727 }

```

`lsxtrheadplural` Convert first letter to upper case.

```

10728 \newcommand*{\Glsxtrheadplural}[1]{%
10729   \protect\NoCaseChange
10730   {%
10731     \glsifattribute{#1}{headuc}{true}%
10732     {%
10733       \GLSplural[noindex,hyper=false]{#1}[]%
10734     }%
10735     {%
10736       \Glsplural[noindex,hyper=false]{#1}[]%
10737     }%
10738   }%
10739 }

```

`sxtrtitleplural` Command to display plural value in section title and table of contents with the first letter changed to upper case.

```

10740 \newrobustcmd*{\Glsxtrtitleplural}[1]{%
10741   \Glsplural[noindex,hyper=false]{#1}[]%
10742 }

```

`glsxtrheadfirst` As above but for the first value.

```

10743 \newcommand*{\glsxtrheadfirst}[1]{%
10744   \protect\NoCaseChange
10745   {%
10746     \glsifattribute{#1}{headuc}{true}%
10747     {%
10748       \GLSfirst[noindex,hyper=false]{#1}[]%
10749     }%
10750     {%
10751       \glsfirst[noindex,hyper=false]{#1}[]%
10752     }%
10753   }%

```

10754 }

**lsxtrtitlefirst** Command to display first value in section title and table of contents.

```
10755 \newrobustcmd*{\lsxtrtitlefirst}[1]{%
10756   \glsfirst[noindex,hyper=false]{#1}[]%
10757 }
```

**Glsxtrheadfirst** First letter converted to upper case

```
10758 \newcommand*{\Glsxtrheadfirst}[1]{%
10759   \protect\NoCaseChange
10760   {%
10761     \glsifattribute{#1}{headuc}{true}%
10762     {%
10763       \GLSfirst[noindex,hyper=false]{#1}[]%
10764     }%
10765     {%
10766       \Glsfirst[noindex,hyper=false]{#1}[]%
10767     }%
10768   }%
10769 }
```

**lsxtrtitlefirst** Command to display first value in section title and table of contents with the first letter changed to upper case.

```
10770 \newrobustcmd*{\Glsxtrtitlefirst}[1]{%
10771   \Glsfirst[noindex,hyper=false]{#1}[]%
10772 }
```

**headfirstplural** As above but for the firstplural value.

```
10773 \newcommand*{\glsxtrheadfirstplural}[1]{%
10774   \protect\NoCaseChange
10775   {%
10776     \glsifattribute{#1}{headuc}{true}%
10777     {%
10778       \GLSfirstplural[noindex,hyper=false]{#1}[]%
10779     }%
10780     {%
10781       \glsfirstplural[noindex,hyper=false]{#1}[]%
10782     }%
10783   }%
10784 }
```

**itlefirstplural** Command to display firstplural value in section title and table of contents.

```
10785 \newrobustcmd*{\glsxtrtitlefirstplural}[1]{%
10786   \glsfirstplural[noindex,hyper=false]{#1}[]%
10787 }
```

**headfirstplural** First letter converted to upper case

```
10788 \newcommand*{\Glsxtrheadfirstplural}[1]{%
```

```

10789 \protect\NoCaseChange
10790 {%
10791   \glsifattribute{#1}{headuc}{true}%
10792   {%
10793     \GLSfirstplural[noindex,hyper=false]{#1}[]%
10794   }%
10795   {%
10796     \Glsfirstplural[noindex,hyper=false]{#1}[]%
10797   }%
10798 }%
10799 }

```

`\titlefirstplural` Command to display first value in section title and table of contents with the first letter changed to upper case.

```

10800 \newrobustcmd*{\Glsxtrtitlefirstplural}[1]{%
10801   \GLSfirstplural[noindex,hyper=false]{#1}[]%
10802 }

```

`\glxtrheadlong` Command used to display long form in the page header.

```

10803 \newcommand*{\glxtrheadlong}[1]{%
10804   \protect\NoCaseChange
10805   {%
10806     \glsifattribute{#1}{headuc}{true}%
10807     {%
10808       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10809     }%
10810     {%
10811       \glxtrlong[noindex,hyper=false]{#1}[]%
10812     }%
10813   }%
10814 }

```

`\glxtrtitlelong` Command to display long form of abbreviation in section title and table of contents.

```

10815 \newrobustcmd*{\glxtrtitlelong}[1]{%
10816   \glxtrlong[noindex,hyper=false]{#1}[]%
10817 }

```

`\lsxtrheadlongpl` Command used to display plural long form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrlongpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10818 \newcommand*{\glxtrheadlongpl}[1]{%
10819   \protect\NoCaseChange
10820   {%
10821     \glsifattribute{#1}{headuc}{true}%
10822     {%
10823       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10824     }%
10825     {%

```

```

10826      \glxstrlongpl[noindex,hyper=false]{#1}[]%
10827    }%
10828  }%
10829 }

```

`sxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents.

```

10830 \newrobustcmd*{\glxstrtitlelongpl}[1]{%
10831   \glxstrlongpl[noindex,hyper=false]{#1}[]%
10832 }

```

`\Glsxtrheadlong` Command used to display long form in the page header with the first letter converted to upper case.

```

10833 \newcommand*{\Glsxtrheadlong}[1]{%
10834   \protect\NoCaseChange
10835   {%
10836     \glsifattribute{#1}{headuc}{true}%
10837     {%
10838       \GLSxtrlong[noindex,hyper=false]{#1}[]%
10839     }%
10840     {%
10841       \Glsxtrlong[noindex,hyper=false]{#1}[]%
10842     }%
10843   }%
10844 }

```

`Glsxtrtitlelong` Command to display long form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10845 \newrobustcmd*{\Glsxtrtitlelong}[1]{%
10846   \Glsxtrlong[noindex,hyper=false]{#1}[]%
10847 }

```

`lsxtrheadlongpl` Command used to display plural long form in the page header with the first letter converted to upper case.

```

10848 \newcommand*{\Glsxtrheadlongpl}[1]{%
10849   \protect\NoCaseChange
10850   {%
10851     \glsifattribute{#1}{headuc}{true}%
10852     {%
10853       \GLSxtrlongpl[noindex,hyper=false]{#1}[]%
10854     }%
10855     {%
10856       \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10857     }%
10858   }%
10859 }

```

`sxstrtitlelongpl` Command to display plural long form of abbreviation in section title and table of contents with the first letter converted to upper case.



```

10860 \newrobustcmd*{\Glsxtrtitlelongpl}[1]{%
10861   \Glsxtrlongpl[noindex,hyper=false]{#1}[]%
10862 }

```

**\glxtrheadfull** Command used to display full form in the page header.

```

10863 \newcommand*{\glxtrheadfull}[1]{%
10864   \protect\NoCaseChange
10865   {%
10866     \glsifattribute{#1}{headuc}{true}%
10867     {%
10868       \GLSxtrfull[noindex,hyper=false]{#1}[]%
10869     }%
10870     {%
10871       \glxtrfull[noindex,hyper=false]{#1}[]%
10872     }%
10873   }%
10874 }

```

**glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents.

```

10875 \newrobustcmd*{\glsxtrtitlefull}[1]{%
10876   \glxtrfull[noindex,hyper=false]{#1}[]%
10877 }

```

**lsxtrheadfullpl** Command used to display plural full form in the page header. If you want the text converted to upper case, this needs to be redefined to use `\GLSxtrfullpl` instead. If you are using a smallcaps style, the default fonts don't provide italic smallcaps.

```

10878 \newcommand*{\lsxtrheadfullpl}[1]{%
10879   \protect\NoCaseChange
10880   {%
10881     \glsifattribute{#1}{headuc}{true}%
10882     {%
10883       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10884     }%
10885     {%
10886       \glxtrfullpl[noindex,hyper=false]{#1}[]%
10887     }%
10888   }%
10889 }

```

**sxtrtitlefullpl** Command to display plural full form of abbreviation in section title and table of contents.

```

10890 \newrobustcmd*{\sxtrtitlefullpl}[1]{%
10891   \glxtrfullpl[noindex,hyper=false]{#1}[]%
10892 }

```

**\Glsxtrheadfull** Command used to display full form in the page header with the first letter converted to upper case.

```

10893 \newcommand*{\Glsxtrheadfull}[1]{%
10894   \protect\NoCaseChange

```

```

10895 {%
10896   \glsifattribute{#1}{headuc}{true}%
10897   {%
10898     \GLSxtrfull[noindex,hyper=false]{#1}[]%
10899   }%
10900   {%
10901     \GLSxtrfull[noindex,hyper=false]{#1}[]%
10902   }%
10903 }%
10904 }

```

**\Glsxtrtitlefull** Command to display full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10905 \newrobustcmd*{\Glsxtrtitlefull}[1]{%
10906   \GLSxtrfull[noindex,hyper=false]{#1}[]%
10907 }

```

**\Glsxtrheadfullpl** Command used to display plural full form in the page header with the first letter converted to upper case.

```

10908 \newcommand*{\Glsxtrheadfullpl}[1]{%
10909   \protect\NoCaseChange
10910   {%
10911     \glsifattribute{#1}{headuc}{true}%
10912     {%
10913       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10914     }%
10915     {%
10916       \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10917     }%
10918   }%
10919 }

```

**\Glsxtrtitlefullpl** Command to display plural full form of abbreviation in section title and table of contents with the first letter converted to upper case.

```

10920 \newrobustcmd*{\Glsxtrtitlefullpl}[1]{%
10921   \GLSxtrfullpl[noindex,hyper=false]{#1}[]%
10922 }

```

**\glsfmtshort** Provide a way of using the formatted short form in section headings. If hyperref has been loaded, use `\texorpdfstring` for convenience in PDF bookmarks.

```

10923 \ifdef\texorpdfstring
10924 {
10925   \newcommand*{\glsfmtshort}[1]{%
10926     \texorpdfstring
10927     {\glsxtrtitleshort{#1}}%
10928     {\glsentryshort{#1}}%
10929   }
10930 }

```

```

10931 {
10932   \newcommand*{\glsfmtshort}[1]{%
10933     \glstrtitleshort{#1}}
10934 }

```

Similarly for the plural version.

`\glsfmtshortpl`

```

10935 \ifdef\teorpdfstring
10936 {
10937   \newcommand*{\glsfmtshortpl}[1]{%
10938     \teorpdfstring
10939       {\glstrtitleshortpl{#1}}%
10940       {\glstryshortpl{#1}}%
10941   }
10942 }
10943 {
10944   \newcommand*{\glsfmtshortpl}[1]{%
10945     \glstrtitleshortpl{#1}}
10946 }

```

The case-changing version isn't suitable for PDF bookmarks, so the PDF alternative uses the non-case-changing version.

`\Glsfmtshort` Singular form (first letter uppercase).

```

10947 \ifdef\teorpdfstring
10948 {
10949   \newcommand*{\Glsfmtshort}[1]{%
10950     \teorpdfstring
10951       {\Glsxtrtitleshort{#1}}%
10952       {\glstryshort{#1}}%
10953   }
10954 }
10955 {
10956   \newcommand*{\Glsfmtshort}[1]{%
10957     \Glsxtrtitleshort{#1}}
10958 }

```

`\Glsfmtshortpl` Plural form (first letter uppercase).

```

10959 \ifdef\teorpdfstring
10960 {
10961   \newcommand*{\Glsfmtshortpl}[1]{%
10962     \teorpdfstring
10963       {\Glsxtrtitleshortpl{#1}}%
10964       {\glstryshortpl{#1}}%
10965   }
10966 }
10967 {
10968   \newcommand*{\Glsfmtshortpl}[1]{%
10969     \Glsxtrtitleshortpl{#1}}

```

10970 }

`\glsfmtname` As above but for the name value.

```
10971 \ifdef\texorpdfstring
10972 {
10973   \newcommand*\glsfmtname[1]{%
10974     \texorpdfstring
10975       {\glsxtrtitlename{#1}}%
10976       {\glsentryname{#1}}%
10977   }
10978 }
10979 {
10980   \newcommand*\glsfmtname[1]{%
10981     \glsxtrtitlename{#1}}
10982 }
```

`\Glsfmtname` First letter converted to upper case.

```
10983 \ifdef\texorpdfstring
10984 {
10985   \newcommand*\Glsfmtname[1]{%
10986     \texorpdfstring
10987       {\Glsxtrtitlename{#1}}%
10988       {\glsentryname{#1}}%
10989   }
10990 }
10991 {
10992   \newcommand*\Glsfmtname[1]{%
10993     \Glsxtrtitlename{#1}}
10994 }
```

`\glsfmttext` As above but for the text value.

```
10995 \ifdef\texorpdfstring
10996 {
10997   \newcommand*\glsfmttext[1]{%
10998     \texorpdfstring
10999       {\glsxtrtitletext{#1}}%
11000       {\glsentrytext{#1}}%
11001   }
11002 }
11003 {
11004   \newcommand*\glsfmttext[1]{%
11005     \glsxtrtitletext{#1}}
11006 }
```

`\Glsfmttext` First letter converted to upper case.

```
11007 \ifdef\texorpdfstring
11008 {
11009   \newcommand*\Glsfmttext[1]{%
11010     \texorpdfstring
```

```

11011     {\Glsxtrtitletext{#1}}%
11012     {\glsentrytext{#1}}%
11013   }
11014 }
11015 {
11016   \newcommand*{\Glsfmttext}[1]{%
11017     \Glsxtrtitletext{#1}}
11018 }

```

`\glsfmtplural` As above but for the plural value.

```

11019 \ifdef\texorpdfstring
11020 {
11021   \newcommand*{\glsfmtplural}[1]{%
11022     \texorpdfstring
11023     {\Glsxtrtitleplural{#1}}%
11024     {\glsentryplural{#1}}%
11025   }
11026 }
11027 {
11028   \newcommand*{\glsfmtplural}[1]{%
11029     \Glsxtrtitleplural{#1}}
11030 }

```

`\Glsfmtplural` First letter converted to upper case.

```

11031 \ifdef\texorpdfstring
11032 {
11033   \newcommand*{\Glsfmtplural}[1]{%
11034     \texorpdfstring
11035     {\Glsxtrtitleplural{#1}}%
11036     {\glsentryplural{#1}}%
11037   }
11038 }
11039 {
11040   \newcommand*{\Glsfmtplural}[1]{%
11041     \Glsxtrtitleplural{#1}}
11042 }

```

`\glsfmtfirst` As above but for the first value.

```

11043 \ifdef\texorpdfstring
11044 {
11045   \newcommand*{\glsfmtfirst}[1]{%
11046     \texorpdfstring
11047     {\Glsxtrtitlefirst{#1}}%
11048     {\glsentryfirst{#1}}%
11049   }
11050 }
11051 {
11052   \newcommand*{\glsfmtfirst}[1]{%
11053     \Glsxtrtitlefirst{#1}}

```

11054 }

`\Glsfmtfirst` First letter converted to upper case.

```
11055 \ifdef\texorpdfstring
11056 {
11057   \newcommand*\Glsfmtfirst}[1]{%
11058     \texorpdfstring
11059     {\Glsxtrtitlefirst{#1}}%
11060     {\glstryfirst{#1}}%
11061   }
11062 }
11063 {
11064   \newcommand*\Glsfmtfirst}[1]{%
11065     \Glsxtrtitlefirst{#1}}
11066 }
```

`\glsfmtfirstpl` As above but for the firstplural value.

```
11067 \ifdef\texorpdfstring
11068 {
11069   \newcommand*\glsfmtfirstpl}[1]{%
11070     \texorpdfstring
11071     {\glsxtrtitlefirstplural{#1}}%
11072     {\glstryfirstplural{#1}}%
11073   }
11074 }
11075 {
11076   \newcommand*\glsfmtfirstpl}[1]{%
11077     \glsxtrtitlefirstplural{#1}}
11078 }
```

`\Glsfmtfirstpl` First letter converted to upper case.

```
11079 \ifdef\texorpdfstring
11080 {
11081   \newcommand*\Glsfmtfirstpl}[1]{%
11082     \texorpdfstring
11083     {\Glsxtrtitlefirstplural{#1}}%
11084     {\glstryfirstplural{#1}}%
11085   }
11086 }
11087 {
11088   \newcommand*\Glsfmtfirstpl}[1]{%
11089     \Glsxtrtitlefirstplural{#1}}
11090 }
```

`\glsfmtlong` As above but for the long value.

```
11091 \ifdef\texorpdfstring
11092 {
11093   \newcommand*\glsfmtlong}[1]{%
11094     \texorpdfstring
```

```

11095     {\glsxtrtitlelong{#1}}%
11096     {\glsentrylong{#1}}%
11097   }
11098 }
11099 {
11100   \newcommand*{\glsfmtlong}[1]{%
11101     \glsxtrtitlelong{#1}}
11102 }

```

`\Glsfmtlong` First letter converted to upper case.

```

11103 \ifdef\txorpdfstring
11104 {
11105   \newcommand*{\Glsfmtlong}[1]{%
11106     \txorpdfstring
11107     {\Glsxtrtitlelong{#1}}%
11108     {\glsentrylong{#1}}%
11109   }
11110 }
11111 {
11112   \newcommand*{\Glsfmtlong}[1]{%
11113     \Glsxtrtitlelong{#1}}
11114 }

```

`\glsfmtlongpl` As above but for the longplural value.

```

11115 \ifdef\txorpdfstring
11116 {
11117   \newcommand*{\glsfmtlongpl}[1]{%
11118     \txorpdfstring
11119     {\glsxtrtitlelongpl{#1}}%
11120     {\glsentrylongpl{#1}}%
11121   }
11122 }
11123 {
11124   \newcommand*{\glsfmtlongpl}[1]{%
11125     \glsxtrtitlelongpl{#1}}
11126 }

```

`\Glsfmtlongpl` First letter converted to upper case.

```

11127 \ifdef\txorpdfstring
11128 {
11129   \newcommand*{\Glsfmtlongpl}[1]{%
11130     \txorpdfstring
11131     {\Glsxtrtitlelongpl{#1}}%
11132     {\glsentrylongpl{#1}}%
11133   }
11134 }
11135 {
11136   \newcommand*{\Glsfmtlongpl}[1]{%
11137     \Glsxtrtitlelongpl{#1}}

```

11138 }

`\glsfmtfull` In-line full format.

```
11139 \ifdef\texorpdfstring
11140 {
11141   \newcommand*\glsfmtfull[1]{%
11142     \texorpdfstring
11143     {\glsxtrtitlefull{#1}}%
11144     {\glsxtrinlinefullformat{#1}-{}}%
11145   }
11146 }
11147 {
11148   \newcommand*\glsfmtfull[1]{%
11149     \glsxtrtitlefull{#1}}
11150 }
```

`\Glsfmtfull` First letter converted to upper case.

```
11151 \ifdef\texorpdfstring
11152 {
11153   \newcommand*\Glsfmtfull[1]{%
11154     \texorpdfstring
11155     {\Glsxtrtitlefull{#1}}%
11156     {\Glsxtrinlinefullformat{#1}-{}}%
11157   }
11158 }
11159 {
11160   \newcommand*\Glsfmtfull[1]{%
11161     \Glsxtrtitlefull{#1}}
11162 }
```

`\glsfmtfullpl` In-line full plural format.

```
11163 \ifdef\texorpdfstring
11164 {
11165   \newcommand*\glsfmtfullpl[1]{%
11166     \texorpdfstring
11167     {\glsxtrtitlefullpl{#1}}%
11168     {\glsxtrinlinefullplformat{#1}-{}}%
11169   }
11170 }
11171 {
11172   \newcommand*\glsfmtfullpl[1]{%
11173     \glsxtrtitlefullpl{#1}}
11174 }
```

`\Glsfmtfullpl` First letter converted to upper case.

```
11175 \ifdef\texorpdfstring
11176 {
11177   \newcommand*\Glsfmtfullpl[1]{%
11178     \texorpdfstring
```



```

11179     {\Glsxtrtitlefullpl{#1}}}%
11180     {\Glsxtrinlinefullplformat{#1}-{}}}%
11181   }
11182 }
11183 {
11184   \newcommand*{\Glsfmtfullpl}[1]{%
11185     \Glsxtrtitlefullpl{#1}}
11186 }

```

## 1.9 Multi-Lingual Support

Add the facility to load language modules, if they are installed, but none are provided with this package.

`sariesExtraLang`

```

11187 \newcommand*{\RequireGlossariesExtraLang}[1]{%
11188   \@ifundefined{ver@glossariesxtr-#1.ldf}{\input{glossariesxtr-#1.ldf}}{}%
11189 }

```

`sariesExtraLang`

```

11190 \newcommand*{\ProvidesGlossariesExtraLang}[1]{%
11191   \ProvidesFile{glossariesxtr-#1.ldf}%
11192 }

```

Load any required language modules that are available. This doesn't generate any warning if none are found, since they're not essential. (The only command that really needs defining for the document is `\abbreviationsname`, which can simply be redefined. However, with `bib2gls` it might be useful to provide custom rules for a particular locale.)

`xtr@loaddialect` The dialect label should be stored in `\this@dialect` before using this command.

```

11193 \newcommand{\glxtr@loaddialect}{%
11194   \IfTrackedLanguageFileExists{\this@dialect}%
11195   {glossariesxtr-}% prefix
11196   {.ldf}%
11197   {%
11198     \RequireGlossariesExtraLang{\CurrentTrackedTag}%
11199   }%
11200   {}% not found

```

If `glossaries-extra-bib2gls` has been loaded, `\@glxtrdialecthook` will check for the associated script, otherwise it will do nothing.

```

11201   \@glxtrdialecthook
11202 }

11203 \@ifpackageloaded{tracklang}
11204 {%
11205   \AnyTrackedLanguages
11206   {%

```

```

11207 \ForEachTrackedDialect{\this@dialect}{\glstr@loaddialect}%
11208 }%
11209 {}%
11210 }
11211 {}

```

Load glossaries-extra-stylemods if required.

```

11212 \@glstr@redefstyles
and set the style:
11213 \@glstr@do@style

```

## 1.10 glossaries-extra-bib2gls.sty

This package provides additional support for bib2gls and is automatically loaded by the record option.

```

11214 \NeedsTeXFormat{LaTeX2e}
11215 \ProvidesPackage{glossaries-extra-bib2gls}[2018/03/06 v1.28 (NLCT)]

```

These are some convenient macros for use with custom rules.

`\glshex`

```

11216 \newcommand*{\glshex}{\string\u}

```

`\providecommand` For use in @preamble, this behaves like `\providecommand` in the document but like `\renewcommand` in bib2gls.

```

11217 \newcommand*{\glstrprovidecommand}{\providecommand}

```

Provide missing Greek letters for use in maths mode. These are recognised by bib2gls and will be mapped to the Mathematical Greek Italic letters. This ensures that the Greek letters that have the same shape as Latin letters are kept with the other Greek letters. These commands use an upright font for capitals and italic for lower case to provide a better match with the other Greek symbols provided by the kernel.

`\Alpha`

```

11218 \providecommand{\Alpha}{\mathrm{A}}

```

`\Beta`

```

11219 \providecommand{\Beta}{\mathrm{B}}

```

`\Epsilon`

```

11220 \providecommand{\Epsilon}{\mathrm{E}}

```

`\Zeta`

```

11221 \providecommand{\Zeta}{\mathrm{Z}}

```

`\Eta`

```

11222 \providecommand{\Eta}{\mathrm{H}}

```

```

\Iota
11223 \providecommand*\Iota{\mathrm{I}}

\Kappa
11224 \providecommand*\Kappa{\mathrm{K}}

\Mu
11225 \providecommand*\Mu{\mathrm{M}}

\Nu
11226 \providecommand*\Nu{\mathrm{N}}

\Omicron
11227 \providecommand*\Omicron{\mathrm{O}}

\Rho
11228 \providecommand*\Rho{\mathrm{P}}

\Tau
11229 \providecommand*\Tau{\mathrm{T}}

\Chi
11230 \providecommand*\Chi{\mathrm{X}}

\Digamma
11231 \providecommand*\Digamma{\mathrm{F}}

\omicron
11232 \providecommand*\omicron{\mathit{o}}

        Provide corresponding upright characters if upgreek has been loaded.
11233 \@ifpackageloaded{upgreek}%
11234 {

\Upalpha
11235 \providecommand*\Upalpha{\mathrm{A}}

\Upbeta
11236 \providecommand*\Upbeta{\mathrm{B}}

\Upepsilon
11237 \providecommand*\Upepsilon{\mathrm{E}}

\Upzeta
11238 \providecommand*\Upzeta{\mathrm{Z}}

```

```

\Upeta
11239 \providecommand*\Upeta{\mathrm{H}}

\Upiota
11240 \providecommand*\Upiota{\mathrm{I}}

\Upkappa
11241 \providecommand*\Upkappa{\mathrm{K}}

\Upmu
11242 \providecommand*\Upmu{\mathrm{M}}

\Upnu
11243 \providecommand*\Upnu{\mathrm{N}}

\Upomicron
11244 \providecommand*\Upomicron{\mathrm{O}}

\Uprho
11245 \providecommand*\Uprho{\mathrm{P}}

\Uptau
11246 \providecommand*\Uptau{\mathrm{T}}

\Upchi
11247 \providecommand*\Upchi{\mathrm{X}}

\upomicron
11248 \providecommand*\upomicron{\mathrm{o}}

11249 }%
11250 {}% upgreek.sty not loaded

```

This package provides some basic rules but is not intended for complete coverage of all locales. The CLDR should provide the appropriate locale-sensitive rules. These macros are primarily to help construct custom rules to include, for example, Greek maths symbols mixed with Latin. For the full rule syntax, see the Java API for [RuleBaseCollator](#)

If you want to provide a rule-block for a particular locale to allow for customization within that locale, create a file called `glossariesxtr-<tag>.ldf` (where *<tag>* identifies the locale) and add similar commands. See the description of `\IfTrackedLanguageFileExists` in the `tracklang` manual for the allowed forms of *<tag>*. The simplest is to just use the root language label. The file will then be automatically loaded by `glossaries-extra` if the document has support for that language.

When combining these blocks of rules, remember to separate them with the appropriate character. For example:

```
sort-rule={\glxtrcontrolrules
```

```

;\glxtrspacerules
;\glxtrnonprintablerules
;\glxtrcombiningdiacriticrules
,\glxtrhyphenrules
<\glxtrgeneralpuncrules
<\glxtrdigitrules
<\glxtrfractionrules
<\glxtrGeneralLatinIVrules
<\glxtrMathItalicGreekIrules
}

```

**xtrcontrolrules** These are control characters that are usually placed at the start of a rule in the ‘ignored characters’ section. \string is used for punctuation characters in case they’ve been made active.

```

11251 \newcommand*{\glxtrcontrolrules}{%
11252 \string'\glshex 200B\string'\string=\glshex 200C\string=\glshex 200D
11253 \string=\glshex 200E\string=\glshex 200F\string=\glshex 0000\string=\glshex 0001
11254 \string=\glshex 0002\string=\glshex 0003\string=\glshex 0004\string=\glshex 0005
11255 \string=\glshex 0006\string=\glshex 0007\string=\glshex 0008
11256 \string=\string'\glshex 0009\string'\string=\string'\glshex 000B\string'
11257 \string=\glshex 000E\string=\glshex 000F\string=\string'\glshex
11258 0010\string'\string=\glshex 0011
11259 \string=\glshex 0012\string=\glshex 0013\string=\glshex 0014\string=\glshex 0015
11260 \string=\glshex 0016\string=\glshex 0017\string=\glshex 0018\string=\glshex 0019
11261 \string=\glshex 001A\string=\glshex 001B\string=\glshex 001C\string=\glshex 001D
11262 \string=\glshex 001E\string=\glshex 001F\string=\glshex 007F\string=\glshex 0080
11263 \string=\glshex 0081\string=\glshex 0082\string=\glshex 0083\string=\glshex 0084
11264 \string=\glshex 0085\string=\glshex 0086\string=\glshex 0087\string=\glshex 0088
11265 \string=\glshex 0089\string=\glshex 008A\string=\glshex 008B\string=\glshex 008C
11266 \string=\glshex 008D\string=\glshex 008E\string=\glshex 008F\string=\glshex 0090
11267 \string=\glshex 0091\string=\glshex 0092\string=\glshex 0093\string=\glshex 0094
11268 \string=\glshex 0095\string=\glshex 0096\string=\glshex 0097\string=\glshex 0098
11269 \string=\glshex 0099\string=\glshex 009A\string=\glshex 009B\string=\glshex 009C
11270 \string=\glshex 009D\string=\glshex 009E\string=\glshex 009F
11271 }

```

**lsxtrspacerules** These are space characters.

```

11272 \newcommand*{\glxtrspacerules}{%
11273 \string' \string'\string;
11274 \string'\glshex 00A0\string'\string;
11275 \string'\glshex 2000\string'\string;
11276 \string'\glshex 2001\string'\string;
11277 \string'\glshex 2002\string'\string;
11278 \string'\glshex 2003\string'\string;
11279 \string'\glshex 2004\string'\string;
11280 \string'\glshex 2005\string'\string;
11281 \string'\glshex 2006\string'\string;
11282 \string'\glshex 2007\string'\string;
11283 \string'\glshex 2008\string'\string;
11284 \string'\glshex 2009\string'\string;

```

```

11285 \string'\glshex 200A\string'\string;
11286 \string'\glshex 3000\string'
11287 }

```

nprintablerules These are non-printable characters (BOM, tabs, line feed and carriage return).

```

11288 \newcommand*{\glxtrnonprintablerules}{%
11289 \string'\glshex FEFF\string'\string;
11290 \string'\glshex 000A\string'\string;
11291 \string'\glshex 0009\string'\string;
11292 \string'\glshex 000C\string'\string;
11293 \string'\glshex 000B\string'
11294 }

```

gdiacriticrules Combining diacritic marks. This is split into multiple macros.

```

11295 \newcommand*{\glxtrcombiningdiacriticrules}{%
11296 \glxtrcombiningdiacriticIrules\string;
11297 \glxtrcombiningdiacriticIIrules\string;
11298 \glxtrcombiningdiacriticIIIrules\string;
11299 \glxtrcombiningdiacriticIVrules
11300 }

```

diacriticIrules First set of combining diacritic marks.

```

11301 \newcommand*{\glxtrcombiningdiacriticIrules}{%
11302 \glshex 0301\string;% combining acute
11303 \glshex 0300\string;% combining grave
11304 \glshex 0306\string;% combining breve
11305 \glshex 0302\string;% combining circumflex
11306 \glshex 030C\string;% combining caron
11307 \glshex 030A\string;% combining ring
11308 \glshex 030D\string;% combining vertical line above
11309 \glshex 0308\string;% combining diaeresis
11310 \glshex 030B\string;% combining double acute
11311 \glshex 0303\string;% combining tilde
11312 \glshex 0307\string;% combining dot above
11313 \glshex 0304% combining macron
11314 }

```

iacriticIIrules Second set of combining diacritic marks.

```

11315 \newcommand*{\glxtrcombiningdiacriticIIrules}{%
11316 \glshex 0337\string;% combining short solidus overlay
11317 \glshex 0327\string;% combining cedilla
11318 \glshex 0328\string;% combining ogonek
11319 \glshex 0323\string;% combining dot below
11320 \glshex 0332\string;% combining low line
11321 \glshex 0305\string;% combining overline
11322 \glshex 0309\string;% combining hook above
11323 \glshex 030E\string;% combining double vertical line above
11324 \glshex 030F\string;% combining double grave accent
11325 \glshex 0310\string;% combining candrabindu

```

11326 \glshex 0311\string;% combining inverted breve  
 11327 \glshex 0312\string;% combining turned comma above  
 11328 \glshex 0313\string;% combining comma above  
 11329 \glshex 0314\string;% combining reversed comma above  
 11330 \glshex 0315\string;% combining comma above right  
 11331 \glshex 0316\string;% combining grave accent below  
 11332 \glshex 0317% combining acute accent below  
 11333 }

acriticIIIrules Third set of combining diacritic marks.

11334 \newcommand\*{\glxtrcombiningdiacriticIIIrules}{%  
 11335 \glshex 0318\string;% combining left tack below  
 11336 \glshex 0319\string;% combining right tack below  
 11337 \glshex 031A\string;% combining left angle above  
 11338 \glshex 031B\string;% combining horn  
 11339 \glshex 031C\string;% combining left half ring below  
 11340 \glshex 031D\string;% combining up tack below  
 11341 \glshex 031E\string;% combining down tack below  
 11342 \glshex 031F\string;% combining plus sign below  
 11343 \glshex 0320\string;% combining minus sign below  
 11344 \glshex 0321\string;% combining palatalized hook below  
 11345 \glshex 0322\string;% combining retroflex hook below  
 11346 \glshex 0324\string;% combining diaeresis below  
 11347 \glshex 0325\string;% combining ring below  
 11348 \glshex 0326\string;% combining comma below  
 11349 \glshex 0329\string;% combining vertical line below  
 11350 \glshex 032A\string;% combining bridge below  
 11351 \glshex 032B\string;% combining inverted double arch below  
 11352 \glshex 032C\string;% combining caron below  
 11353 \glshex 032D\string;% combining circumflex accent below  
 11354 \glshex 032E\string;% combining breve below  
 11355 \glshex 032F\string;% combining inverted breve below  
 11356 \glshex 0330\string;% combining tilde below  
 11357 \glshex 0331\string;% combining macron below  
 11358 \glshex 0333\string;% combining double low line  
 11359 \glshex 0334\string;% combining tilde overlay  
 11360 \glshex 0335\string;% combining short stroke overlay  
 11361 \glshex 0336\string;% combining long stroke overlay  
 11362 \glshex 0338\string;% combining long solidus overlay  
 11363 \glshex 0339\string;% combining combining right half ring below  
 11364 \glshex 033A\string;% combining inverted bridge below  
 11365 \glshex 033B\string;% combining square below  
 11366 \glshex 033C\string;% combining seagull below  
 11367 \glshex 033D\string;% combining x above  
 11368 \glshex 033E\string;% combining vertical tilde  
 11369 \glshex 033F\string;% combining double overline  
 11370 \glshex 0342\string;% combining Greek perispomeni  
 11371 \glshex 0344\string;% combining Greek dialytika tonos  
 11372 \glshex 0345\string;% combining Greek ypogegrammeni

```

11373 \glshex 0360\string;% combining double tilde
11374 \glshex 0361\string;% combining double inverted breve
11375 \glshex 0483\string;% combining Cyrillic titlo
11376 \glshex 0484\string;% combining Cyrillic palatalization
11377 \glshex 0485\string;% combining Cyrillic dasia pneumata
11378 \glshex 0486% combining Cyrillic psili pneumata
11379 }

```

iacriticIVrules Fourth set of combining diacritic marks.

```

11380 \newcommand*{\glxstrcombingdiacriticIVrules}{%
11381 \glshex 20D0\string;% combining left harpoon above
11382 \glshex 20D1\string;% combining right harpoon above
11383 \glshex 20D2\string;% combining long vertical line overlay
11384 \glshex 20D3\string;% combining short vertical line overlay
11385 \glshex 20D4\string;% combining anticlockwise arrow above
11386 \glshex 20D5\string;% combining clockwise arrow above
11387 \glshex 20D6\string;% combining left arrow above
11388 \glshex 20D7\string;% combining right arrow above
11389 \glshex 20D8\string;% combining ring overlay
11390 \glshex 20D9\string;% combining clockwise ring overlay
11391 \glshex 20DA\string;% combining anticlockwise ring overlay
11392 \glshex 20DB\string;% combining three dots above
11393 \glshex 20DC\string;% combining four dots above
11394 \glshex 20DD\string;% combining enclosing circle
11395 \glshex 20DE\string;% combining enclosing square
11396 \glshex 20DF\string;% combining enclosing diamond
11397 \glshex 20E0\string;% combining enclosing circle backslash
11398 \glshex 20E1% combining left right arrow above
11399 }

```

sxtrhyphenrules Hyphens.

```

11400 \newcommand*{\glxstrhyphenrules}{%
11401 \string'\string-\string'\string;% ASCII hyphen
11402 \glshex 00AD\string;% soft hyphen
11403 \glshex 2010\string;% hyphen
11404 \glshex 2011\string;% non-breaking hyphen
11405 \glshex 2012\string;% figure dash
11406 \glshex 2013\string;% en dash
11407 \glshex 2014\string;% em dash
11408 \glshex 2015\string;% horizontal bar
11409 \glshex 2212\string=\glshex 207B\string=\glshex 208B% minus sign
11410 }

```

eneralpuncrules General punctuation.

```

11411 \newcommand*{\glxstrgeneralpuncrules}{%
11412 \glxstrgeneralpuncIrules
11413 \string<\glxstrcurrencyrules
11414 \string<\glxstrgeneralpuncIIrules
11415 }

```



neralpuncIrules First set of general punctuation.

```
11416 \newcommand*{\glxtrgeneralpuncIrules}{%
11417 \string'\glshex 005F\string'% underscore
11418 \string<\glshex 00AF% macron
11419 \string<\string'\glshex 002C\string'% comma
11420 \string<\string'\glshex 003B\string'% semi-colon
11421 \string<\string'\glshex 003A\string'% colon
11422 \string<\string'\glshex 0021\string'% exclamation mark
11423 \string<\glshex 00A1% inverted exclamation mark
11424 \string<\string'\glshex 003F\string'% question mark
11425 \string<\glshex 00BF% inverted question mark
11426 \string<\string'\glshex 002F\string'% solidus
11427 \string<\string'\glshex 002E\string'% full stop
11428 \string<\glshex 00B4% acute accent
11429 \string<\string'\glshex 0060\string'% grave accent
11430 \string<\string'\glshex 005E\string'% circumflex accent
11431 \string<\glshex 00A8% diaeresis
11432 \string<\string'\glshex 007E\string'% tilde
11433 \string<\glshex 00B7% middle dot
11434 \string<\glshex 00B8% cedilla
11435 \string<\string'\glshex 0027\string'% straight apostrophe
11436 \string<\string'\glshex 0022\string'% straight double quote
11437 \string<\glshex 00AB% left guillemet
11438 \string<\glshex 00BB% right guillemet
11439 \string<\string'\glshex 0028\string'% left parenthesis
11440 \string=<\glshex 207D\string=<\glshex 208D% super/subscript left parenthesis
11441 \string<\string'\glshex 0029\string'% right parenthesis
11442 \string=<\glshex 207E\string=<\glshex 208E% super/subscript right parenthesis
11443 \string<\string'\glshex 005B\string'% left square bracket
11444 \string<\string'\glshex 005D\string'% right square bracket
11445 \string<\string'\glshex 007B\string'% left curly bracket
11446 \string<\string'\glshex 007D\string'% right curly bracket
11447 \string<\glshex 00A7% section sign
11448 \string<\glshex 00B6% pilcrow sign
11449 \string<\glshex 00A9% copyright sign
11450 \string<\glshex 00AE% registered sign
11451 \string<\string'\glshex 0040\string'% at sign
11452 }
```

trcurrencyrules General punctuation.

```
11453 \newcommand*{\glxtrcurrencyrules}{%
11454 \glshex 00A4% currency sign
11455 \string<\glshex 0E3F% Thai currency symbol baht
11456 \string<\glshex 00A2% cent sign
11457 \string<\glshex 20A1% colon sign
11458 \string<\glshex 20A2% cruzeiro sign
11459 \string<\string'\glshex 0024\string'% dollar sign
11460 \string<\glshex 20AB% dong sign
11461 \string<\glshex 20AC% euro sign
```

```

11462 \string<\glshex 20A3% French franc sign
11463 \string<\glshex 20A4% lira sign
11464 \string<\glshex 20A5% mill sign
11465 \string<\glshex 20A6% naira sign
11466 \string<\glshex 20A7% peseta sign
11467 \string<\glshex 00A3% pound sign
11468 \string<\glshex 20A8% rupee sign
11469 \string<\glshex 20AA% new sheqel sign
11470 \string<\glshex 20A9% won sign
11471 \string<\glshex 00A5% yen sign
11472 }

```

eralpuncIIrules Second set of general punctuation.

```

11473 \newcommand*{\glxtrgeneralpuncIIrules}{%
11474 \string'\glshex 002A\string'% asterisk
11475 \string<\string'\glshex 005C\string'% backslash
11476 \string<\string'\glshex 0026\string'% ampersand
11477 \string<\string'\glshex 0023\string'% hash sign
11478 \string<\string'\glshex 0025\string'% percent sign
11479 \string<\string'\glshex 002B\string'% plus sign
11480 \string=\glshex 207A\string=\glshex 208A% super/subscript plus sign
11481 \string<\glshex 00B1% plus-minus sign
11482 \string<\glshex 00F7% division sign
11483 \string<\glshex 00D7% multiplication sign
11484 \string<\string'\glshex 003C\string'% less-than sign
11485 \string<\string'\glshex 003D\string'% equals sign
11486 \string<\string'\glshex 003E\string'% greater-than sign
11487 \string<\glshex 00AC% not sign
11488 \string<\string'\glshex 007C\string'% vertical bar (pipe)
11489 \string<\glshex 00A6% broken bar
11490 \string<\glshex 00B0% degree sign
11491 \string<\glshex 00B5% micron sign
11492 }

```

eralLatinIrules Basic Latin alphabet.

```

11493 \newcommand*{\glxtrGeneralLatinIrules}{%
11494 \glxtrLatinA
11495 \string<b,B%
11496 \string<c,C%
11497 \string<d,D%
11498 \string<\glxtrLatinE
11499 \string<f,F%
11500 \string<g,G%
11501 \string<\glxtrLatinH
11502 \string<\glxtrLatinI
11503 \string<j,J%
11504 \string<\glxtrLatinK
11505 \string<\glxtrLatinL
11506 \string<\glxtrLatinM

```

```

11507 \string<\glxtrLatinN
11508 \string<\glxtrLatinO
11509 \string<\glxtrLatinP
11510 \string<q,Q%
11511 \string<r,R%
11512 \string<\glxtrLatinS
11513 \string<\glxtrLatinT
11514 \string<u,U%
11515 \string<v,V%
11516 \string<w,W%
11517 \string<\glxtrLatinX
11518 \string<y,Y%
11519 \string<z,Z
11520 }

```

raLatinIIrules General Latin alphabet (eth between D and E, ß treated as SS).

```

11521 \newcommand*{\glxtrGeneralLatinIIrules}{%
11522 \glxtrLatinA
11523 \string<b,B%
11524 \string<c,C%
11525 \string<d,D%
11526 \string<\glxtrLatinEth
11527 \string<\glxtrLatinE
11528 \string<f,F%
11529 \string<g,G%
11530 \string<\glxtrLatinH
11531 \string<\glxtrLatinI
11532 \string<j,J%
11533 \string<\glxtrLatinK
11534 \string<\glxtrLatinL
11535 \string<\glxtrLatinM
11536 \string<\glxtrLatinN
11537 \string<\glxtrLatinO
11538 \string<\glxtrLatinP
11539 \string<q,Q%
11540 \string<r,R%
11541 \string<\glxtrLatinS
11542 \string& SS \string, \glxtrLatinEszettSs
11543 \string<\glxtrLatinT
11544 \string<u,U%
11545 \string<v,V%
11546 \string<w,W%
11547 \string<\glxtrLatinX
11548 \string<y,Y%
11549 \string<z,Z%
11550 }

```

alLatinIIIrules General Latin alphabet (eth between D and E, ß treated as SZ).

```

11551 \newcommand*{\glxtrGeneralLatinIIIrules}{%

```

```

11552 \glxtrLatinA
11553 \string<b,B%
11554 \string<c,C%
11555 \string<d,D%
11556 \string<\glxtrLatinEth
11557 \string<\glxtrLatinE
11558 \string<f,F%
11559 \string<g,G%
11560 \string<\glxtrLatinH
11561 \string<\glxtrLatinI
11562 \string<j,J%
11563 \string<\glxtrLatinK
11564 \string<\glxtrLatinL
11565 \string<\glxtrLatinM
11566 \string<\glxtrLatinN
11567 \string<\glxtrLatinO
11568 \string<\glxtrLatinP
11569 \string<q,Q%
11570 \string<r,R%
11571 \string<\glxtrLatinS
11572 \string& SZ, \glxtrLatinEszettSz
11573 \string<\glxtrLatinT
11574 \string<u,U%
11575 \string<v,V%
11576 \string<w,W%
11577 \string<\glxtrLatinX
11578 \string<y,Y%
11579 \string<z,Z%
11580 }

```

`\glxtrGeneralLatinIVrules` General Latin alphabet ( $\mathcal{A}$  treated as AE and  $\mathcal{E}$  treated as OE,  $\mathcal{P}$  treated as TH,  $\mathcal{B}$  treated as SS, eth between D and E).

```

11581 \newcommand*{\glxtrGeneralLatinIVrules}{%
11582 \glxtrLatinA
11583 \string& AE , \glxtrLatinAELigature
11584 \string<b,B%
11585 \string<c,C%
11586 \string<d,D%
11587 \string<\glxtrLatinEth
11588 \string<\glxtrLatinE
11589 \string<f,F%
11590 \string<g,G%
11591 \string<\glxtrLatinH
11592 \string<\glxtrLatinI
11593 \string<j,J%
11594 \string<\glxtrLatinK
11595 \string<\glxtrLatinL
11596 \string<\glxtrLatinM
11597 \string<\glxtrLatinN

```

```

11598 \string<\glxtrLatinO
11599 \string& OE , \glxtrLatinOELigature
11600 \string<\glxtrLatinP
11601 \string<q,Q%
11602 \string<r,R%
11603 \string<\glxtrLatinS
11604 \string& SS , \glxtrLatinEszettSs
11605 \string<\glxtrLatinT
11606 \string& th =\glshex 00DE
11607 \string& TH =\glshex 00FE
11608 \string<u,U%
11609 \string<v,V%
11610 \string<w,W%
11611 \string<\glxtrLatinX
11612 \string<y,Y%
11613 \string<z,Z%
11614 }

```

eralLatinVrules General Latin alphabet (eth between D and E, ß treated as SS, Þ treated as TH).

```

11615 \newcommand*{\glxtrGeneralLatinVrules}{%
11616 \glxtrLatinA
11617 \string<b,B%
11618 \string<c,C%
11619 \string<d,D%
11620 \string<\glxtrLatinEth
11621 \string<\glxtrLatinE
11622 \string<f,F%
11623 \string<g,G%
11624 \string<\glxtrLatinH
11625 \string<\glxtrLatinI
11626 \string<j,J%
11627 \string<\glxtrLatinK
11628 \string<\glxtrLatinL
11629 \string<\glxtrLatinM
11630 \string<\glxtrLatinN
11631 \string<\glxtrLatinO
11632 \string<\glxtrLatinP
11633 \string<q,Q%
11634 \string<r,R%
11635 \string<\glxtrLatinS
11636 \string& SS , \glxtrLatinEszettSs
11637 \string<\glxtrLatinT
11638 \string& th =\glshex 00DE
11639 \string& TH =\glshex 00FE
11640 \string<u,U%
11641 \string<v,V%
11642 \string<w,W%
11643 \string<\glxtrLatinX
11644 \string<y,Y%

```

```

11645 \string<z,Z%
11646 }

```

alLatinVIrules General Latin alphabet (eth between D and E, ß treated as SZ, Þ treated as TH).

```

11647 \newcommand*{\glxtrGeneralLatinVIrules}{%
11648 \glxtrLatinA
11649 \string<b,B%
11650 \string<c,C%
11651 \string<d,D%
11652 \string<\glxtrLatinEth
11653 \string<\glxtrLatinE
11654 \string<f,F%
11655 \string<g,G%
11656 \string<\glxtrLatinH
11657 \string<\glxtrLatinI
11658 \string<j,J%
11659 \string<\glxtrLatinK
11660 \string<\glxtrLatinL
11661 \string<\glxtrLatinM
11662 \string<\glxtrLatinN
11663 \string<\glxtrLatinO
11664 \string<\glxtrLatinP
11665 \string<q,Q%
11666 \string<r,R%
11667 \string<\glxtrLatinS
11668 \string& SZ , \glxtrLatinEszettSz
11669 \string<\glxtrLatinT
11670 \string& th =\glshex 00DE
11671 \string& TH =\glshex 00FE
11672 \string<u,U%
11673 \string<v,V%
11674 \string<w,W%
11675 \string<\glxtrLatinX
11676 \string<y,Y%
11677 \string<z,Z%
11678 }

```

alLatinVIIrules General Latin alphabet (Æ between A and B, eth between D and E, insular G as G, Œ between O and P, long S equivalent to S, Þ between T and U and wynn as W).

```

11679 \newcommand*{\glxtrGeneralLatinVIIrules}{%
11680 \glxtrLatinA
11681 \string<\glxtrLatinAELigature
11682 \string<b,B%
11683 \string<c,C%
11684 \string<d,D%
11685 \string<\glxtrLatinEth
11686 \string<\glxtrLatinE
11687 \string<f,F%
11688 \string<\glxtrLatinInsularG

```

```

11689 \string<\glxtrLatinH
11690 \string<\glxtrLatinI
11691 \string<j,J%
11692 \string<\glxtrLatinK
11693 \string<\glxtrLatinL
11694 \string<\glxtrLatinM
11695 \string<\glxtrLatinN
11696 \string<\glxtrLatinO
11697 \string<\glxtrLatinOELigature
11698 \string<\glxtrLatinP
11699 \string<q,Q%
11700 \string<r,R%
11701 \string<\glshex 017F=\glxtrLatinS % s and long s
11702 \string<\glxtrLatinT
11703 \string<\glxtrLatinThorn
11704 \string<u,U%
11705 \string<v,V%
11706 \string< w\string=\glshex 01BF, W\string=\glshex 01F7
11707 \string<\glxtrLatinX
11708 \string<y,Y%
11709 \string<z,Z%
11710 }

```

11LatinVIIIrules General Latin alphabet (Æ treated as AE and Ætreated as OE, Ptreated as TH, ß treated as SS, eth treated as D, Ø treated as O, L treated as L).

```

11711 \newcommand*{\glxtrGeneralLatinVIIIrules}{%
11712 \glxtrLatinA
11713 \string& AE , \glxtrLatinAELigature
11714 \string<b,B%
11715 \string<c,C%
11716 \string<\glshex 00F0\string;d,\glshex 00D0\string;D% D and eth
11717 \string<\glxtrLatinE
11718 \string<f,F%
11719 \string<g,G%
11720 \string<\glxtrLatinH
11721 \string<\glxtrLatinI
11722 \string<j,J%
11723 \string<\glxtrLatinK
11724 \string<\glshex 0142\string=\glxtrLatinL\string=\glshex 0141% L and \L
11725 \string<\glxtrLatinM
11726 \string<\glxtrLatinN
11727 \string<\glshex 00F8\string=\glxtrLatinO\string=\glshex 00D8% O and \O
11728 \string& OE , \glxtrLatinOELigature
11729 \string<\glxtrLatinP
11730 \string<q,Q%
11731 \string<r,R%
11732 \string<\glxtrLatinS
11733 \string& SS , \glxtrLatinEszettSs
11734 \string<\glxtrLatinT

```

```

11735 \string& th =\glshex 00DE
11736 \string& TH =\glshex 00FE
11737 \string<u,U%
11738 \string<v,V%
11739 \string<w,W%
11740 \string<\glsxtrLatinX
11741 \string<y,Y%
11742 \string<z,Z%
11743 }

```

\glsxtrLatinA

```

11744 \newcommand*{\glsxtrLatinA}{%
11745   a\string=\glshex 00AA\string=\glshex 2090,A
11746 }

```

\glsxtrLatinE

```

11747 \newcommand*{\glsxtrLatinE}{%
11748   e\string=\glshex 2091,E
11749 }

```

\glsxtrLatinH

```

11750 \newcommand*{\glsxtrLatinH}{%
11751   h\string=\glshex 2095,H
11752 }

```

\glsxtrLatinI

```

11753 \newcommand*{\glsxtrLatinI}{%
11754   i\string=\glshex 2071,I
11755 }

```

\glsxtrLatinK

```

11756 \newcommand*{\glsxtrLatinK}{%
11757   k\string=\glshex 2096,K
11758 }

```

\glsxtrLatinL

```

11759 \newcommand*{\glsxtrLatinL}{%
11760   l\string=\glshex 2097,L
11761 }

```

\glsxtrLatinM

```

11762 \newcommand*{\glsxtrLatinM}{%
11763   m\string=\glshex 2098,M
11764 }

```

\glsxtrLatinN

```

11765 \newcommand*{\glsxtrLatinN}{%
11766   n\string=\glshex 207F\string=\glshex 2099,N
11767 }

```



\glsxtrLatinO

```
11768 \newcommand*{\glsxtrLatinO}{%
11769   o\string=\glshex 00BA\string=\glshex 2092,0
11770 }
```

\glsxtrLatinP

```
11771 \newcommand*{\glsxtrLatinP}{%
11772   p\string=\glshex 209A,P
11773 }
```

\glsxtrLatinS

```
11774 \newcommand*{\glsxtrLatinS}{%
11775   s\string=\glshex 209B,S
11776 }
```

\glsxtrLatinT

```
11777 \newcommand*{\glsxtrLatinT}{%
11778   t\string=\glshex 209C,T
11779 }
```

\glsxtrLatinX

```
11780 \newcommand*{\glsxtrLatinX}{%
11781   x\string=\glshex 2093,X
11782 }
```

lsxtrLatinSchwa Latin schwa (lower case, subscript and upper case).

```
11783 \newcommand*{\glsxtrLatinSchwa}{%
11784   \glshex 0259\string=\glshex 2094,\glshex 018F
11785 }
```

trLatinEszettSs

```
11786 \newcommand*{\glsxtrLatinEszettSs}{%
11787   \glshex 00DF% eszett
11788   \string=\glshex 017Fs % long S s
11789 }
```

trLatinEszettSz

```
11790 \newcommand*{\glsxtrLatinEszettSz}{%
11791   \glshex 00DF% eszett
11792   \string= \glshex 017Fz % long S z
11793 }
```

\glsxtrLatinEth

```
11794 \newcommand*{\glsxtrLatinEth}{%
11795   \glshex 00F0,\glshex 00D0% eth
11796 }
```

lsxtrLatinThorn

```
11797 \newcommand*{\glxtrLatinThorn}{%
11798   \glshex 00FE,\glshex 00DE% thorn
11799 }
```

LatinAELigature

```
11800 \newcommand*{\glxtrLatinAELigature}{%
11801   \glshex 00E6,\glshex 00C6% AE-ligature
11802 }
```

LatinOELigature

```
11803 \newcommand*{\glxtrLatinOELigature}{%
11804   \glshex 0153,\glshex 0152% OE-ligature
11805 }
```

\glxtrLatinAA

```
11806 \newcommand*{\glxtrLatinAA}{%
11807   \glshex 00E5=a\glshex 030A,% \aa
11808   \glshex 00C5=A\glshex 030A% \AA
11809 }
```

glxtrLatinWynn

```
11810 \newcommand*{\glxtrLatinWynn}{%
11811   \glshex 01BF,\glshex 01F7% wynn
11812 }
```

trLatinInsularG

```
11813 \newcommand*{\glxtrLatinInsularG}{%
11814   \glshex 1D79,\glshex A77D% insular G
11815   \string; g, G
11816 }
```

lsxtrLatinOslash

```
11817 \newcommand*{\glxtrLatinOslash}{%
11818   \glshex 00F8,\glshex 00D8% \o, \O
11819 }
```

lsxtrLatinLslash

```
11820 \newcommand*{\glxtrLatinLslash}{%
11821   \glshex 0142,\glshex 0141% \l, \L
11822 }
```

thUpGreekIrules Includes digamma between epsilon and zeta.

```
11823 \newcommand*{\glxtrMathUpGreekIrules}{%
11824   \glxtrUpAlpha
11825   \string<\glxtrUpBeta
11826   \string<\glxtrUpGamma
11827   \string<\glxtrUpDelta
```

```

11828 \string<\glxtrUpEpsilon
11829 \string<\glxtrUpDigamma
11830 \string<\glxtrUpZeta
11831 \string<\glxtrUpEta
11832 \string<\glxtrUpTheta
11833 \string<\glxtrUpIota
11834 \string<\glxtrUpKappa
11835 \string<\glxtrUpLambda
11836 \string<\glxtrUpMu
11837 \string<\glxtrUpNu
11838 \string<\glxtrUpXi
11839 \string<\glxtrUpOmicron
11840 \string<\glxtrUpPi
11841 \string<\glxtrUpRho
11842 \string<\glxtrUpSigma
11843 \string<\glxtrUpTau
11844 \string<\glxtrUpUpsilon
11845 \string<\glxtrUpPhi
11846 \string<\glxtrUpChi
11847 \string<\glxtrUpPsi
11848 \string<\glxtrUpOmega
11849 }

```

hUpGreekIIrules Doesn't include digamma.

```

11850 \newcommand*{\glxtrMathUpGreekIIrules}{%
11851 \glxtrUpAlpha
11852 \string<\glxtrUpBeta
11853 \string<\glxtrUpGamma
11854 \string<\glxtrUpDelta
11855 \string<\glxtrUpEpsilon
11856 \string<\glxtrUpZeta
11857 \string<\glxtrUpEta
11858 \string<\glxtrUpTheta
11859 \string<\glxtrUpIota
11860 \string<\glxtrUpKappa
11861 \string<\glxtrUpLambda
11862 \string<\glxtrUpMu
11863 \string<\glxtrUpNu
11864 \string<\glxtrUpXi
11865 \string<\glxtrUpOmicron
11866 \string<\glxtrUpPi
11867 \string<\glxtrUpRho
11868 \string<\glxtrUpSigma
11869 \string<\glxtrUpTau
11870 \string<\glxtrUpUpsilon
11871 \string<\glxtrUpPhi
11872 \string<\glxtrUpChi
11873 \string<\glxtrUpPsi
11874 \string<\glxtrUpOmega

```

11875 }

**alicGreekIrules** Includes (upright) digamma between epsilon and zeta (there isn't an italic digamma), so don't mix with `\glxtrMathUpGreekIrules` or there may be unexpected results.

```
11876 \newcommand*{\glxtrMathItalicGreekIrules}{%
11877 \glxtrMathItalicAlpha
11878 \string<\glxtrMathItalicBeta
11879 \string<\glxtrMathItalicGamma
11880 \string<\glxtrMathItalicDelta
11881 \string<\glxtrMathItalicEpsilon
11882 \string<\glxtrUpDigamma
11883 \string<\glxtrMathItalicZeta
11884 \string<\glxtrMathItalicEta
11885 \string<\glxtrMathItalicTheta
11886 \string<\glxtrMathItalicIota
11887 \string<\glxtrMathItalicKappa
11888 \string<\glxtrMathItalicLambda
11889 \string<\glxtrMathItalicMu
11890 \string<\glxtrMathItalicNu
11891 \string<\glxtrMathItalicXi
11892 \string<\glxtrMathItalicOmicron
11893 \string<\glxtrMathItalicPi
11894 \string<\glxtrMathItalicRho
11895 \string<\glxtrMathItalicSigma
11896 \string<\glxtrMathItalicTau
11897 \string<\glxtrMathItalicUpsilon
11898 \string<\glxtrMathItalicPhi
11899 \string<\glxtrMathItalicChi
11900 \string<\glxtrMathItalicPsi
11901 \string<\glxtrMathItalicOmega
11902 }
```

**alicGreekIIrules** Doesn't include digamma.

```
11903 \newcommand*{\glxtrMathItalicGreekIIrules}{%
11904 \glxtrMathItalicAlpha
11905 \string<\glxtrMathItalicBeta
11906 \string<\glxtrMathItalicGamma
11907 \string<\glxtrMathItalicDelta
11908 \string<\glxtrMathItalicEpsilon
11909 \string<\glxtrMathItalicZeta
11910 \string<\glxtrMathItalicEta
11911 \string<\glxtrMathItalicTheta
11912 \string<\glxtrMathItalicIota
11913 \string<\glxtrMathItalicKappa
11914 \string<\glxtrMathItalicLambda
11915 \string<\glxtrMathItalicMu
11916 \string<\glxtrMathItalicNu
11917 \string<\glxtrMathItalicXi
11918 \string<\glxtrMathItalicOmicron
```

```

11919 \string<\glxtrMathItalicPi
11920 \string<\glxtrMathItalicRho
11921 \string<\glxtrMathItalicSigma
11922 \string<\glxtrMathItalicTau
11923 \string<\glxtrMathItalicUpsilon
11924 \string<\glxtrMathItalicPhi
11925 \string<\glxtrMathItalicChi
11926 \string<\glxtrMathItalicPsi
11927 \string<\glxtrMathItalicOmega
11928 }

```

perGreekIrules Upper case only (includes upright digamma).

```

11929 \newcommand*{\glxtrMathItalicUpperGreekIrules}{%
11930 \glshex 1D6E2% upper case alpha (maths italic)
11931 \string<\glshex 1D6E3% upper case beta (maths italic)
11932 \string<\glshex 1D6E4% upper case gamma (maths italic)
11933 \string<\glshex 1D6E5% upper case delta (maths italic)
11934 \string<\glshex 1D6E6% upper case epsilon (maths italic)
11935 \string<\glshex 03DC% upper case digamma
11936 \string<\glshex 1D6E7% upper case zeta (maths italic)
11937 \string<\glshex 1D6E8% upper case eta (maths italic)
11938 \string<\glshex 1D6E9% upper case theta (maths italic)
11939 \string<\glshex 1D6F3% upper case theta variant (maths italic)
11940 \string<\glshex 1D6EA% upper case iota (maths italic)
11941 \string<\glshex 1D6EB% upper case kappa (maths italic)
11942 \string<\glshex 1D6EC% upper case lambda (maths italic)
11943 \string<\glshex 1D6ED% upper case mu (maths italic)
11944 \string<\glshex 1D6EE% upper case nu (maths italic)
11945 \string<\glshex 1D6EF% upper case xi (maths italic)
11946 \string<\glshex 1D6F0% upper case omicron (maths italic)
11947 \string<\glshex 1D6F1% upper case pi (maths italic)
11948 \string<\glshex 1D6F2% upper case rho (maths italic)
11949 \string<\glshex 1D6F4% upper case sigma (maths italic)
11950 \string<\glshex 1D6F5% upper case tau (maths italic)
11951 \string<\glshex 1D6F6% upper case upsilon (maths italic)
11952 \string<\glshex 1D6F7% upper case phi (maths italic)
11953 \string<\glshex 1D6F8% upper case chi (maths italic)
11954 \string<\glshex 1D6F9% upper case psi (maths italic)
11955 \string<\glshex 1D6FA% upper case omega (maths italic)
11956 }

```

perGreekIIrules Upper case only (doesn't include upright digamma).

```

11957 \newcommand*{\glxtrMathItalicUpperGreekIIrules}{%
11958 \glshex 1D6E2% upper case alpha (maths italic)
11959 \string<\glshex 1D6E3% upper case beta (maths italic)
11960 \string<\glshex 1D6E4% upper case gamma (maths italic)
11961 \string<\glshex 1D6E5% upper case delta (maths italic)
11962 \string<\glshex 1D6E6% upper case epsilon (maths italic)
11963 \string<\glshex 1D6E7% upper case zeta (maths italic)

```

```

11964 \string<\glshex 1D6E8% upper case eta (maths italic)
11965 \string<\glshex 1D6E9% upper case theta (maths italic)
11966 \string=\glshex 1D6F3% upper case theta variant (maths italic)
11967 \string<\glshex 1D6EA% upper case iota (maths italic)
11968 \string<\glshex 1D6EB% upper case kappa (maths italic)
11969 \string<\glshex 1D6EC% upper case lambda (maths italic)
11970 \string<\glshex 1D6ED% upper case mu (maths italic)
11971 \string<\glshex 1D6EE% upper case nu (maths italic)
11972 \string<\glshex 1D6EF% upper case xi (maths italic)
11973 \string<\glshex 1D6F0% upper case omicron (maths italic)
11974 \string<\glshex 1D6F1% upper case pi (maths italic)
11975 \string<\glshex 1D6F2% upper case rho (maths italic)
11976 \string<\glshex 1D6F4% upper case sigma (maths italic)
11977 \string<\glshex 1D6F5% upper case tau (maths italic)
11978 \string<\glshex 1D6F6% upper case upsilon (maths italic)
11979 \string<\glshex 1D6F7% upper case phi (maths italic)
11980 \string<\glshex 1D6F8% upper case chi (maths italic)
11981 \string<\glshex 1D6F9% upper case psi (maths italic)
11982 \string<\glshex 1D6FA% upper case omega (maths italic)
11983 }

```

LowerGreekIrules Lower case only (includes upright digamma).

```

11984 \newcommand*{\glxtrMathItalicLowerGreekIrules}{%
11985 \glshex 1D6FC% lower case alpha (maths italic)
11986 \string<\glshex 1D6FD% lower case beta (maths italic)
11987 \string<\glshex 1D6FE% lower case gamma (maths italic)
11988 \string<\glshex 1D6FF% lower case delta (maths italic)
11989 \string<\glshex 1D700% lower case epsilon (maths italic)
11990 \string=\glshex 1D716% lower case epsilon variant (maths italic)
11991 \string<\glshex 03DD% lower case digamma
11992 \string<\glshex 1D701% lower case zeta (maths italic)
11993 \string<\glshex 1D702% lower case eta (maths italic)
11994 \string<\glshex 1D703% lower case theta (maths italic)
11995 \string=\glshex 1D717% lower case theta variant (maths italic)
11996 \string<\glshex 1D704% lower case iota (maths italic)
11997 \string<\glshex 1D705% lower case kappa (maths italic)
11998 \string=\glshex 1D718% lower case kappa variant (maths italic)
11999 \string<\glshex 1D706% lower case lambda (maths italic)
12000 \string<\glshex 1D707% lower case mu (maths italic)
12001 \string<\glshex 1D708% lower case nu (maths italic)
12002 \string<\glshex 1D709% lower case xi (maths italic)
12003 \string<\glshex 1D70A% lower case omicron (maths italic)
12004 \string<\glshex 1D70B% lower case pi (maths italic)
12005 \string=\glshex 1D71B% lower case pi variant (maths italic)
12006 \string<\glshex 1D70C% lower case rho (maths italic)
12007 \string=\glshex 1D71A% lower case rho variant (maths italic)
12008 \string<\glshex 1D70D% lower case final sigma (maths italic)
12009 \string=\glshex 1D70E% lower case sigma (maths italic)
12010 \string<\glshex 1D70F% lower case tau (maths italic)

```

```

12011 \string<\glshex 1D710% lower case upsilon (maths italic)
12012 \string<\glshex 1D711% lower case phi (maths italic)
12013 \string=\glshex 1D719% lower case phi variant (maths italic)
12014 \string<\glshex 1D712% lower case chi (maths italic)
12015 \string<\glshex 1D713% lower case psi (maths italic)
12016 \string<\glshex 1D714% lower case omega (maths italic)
12017 }

```

**LowerGreeklIrules** Lower case only (doesn't includes upright digamma).

```

12018 \newcommand*{\glxtrMathItalicLowerGreeklIrules}{%
12019 \glshex 1D6FC% lower case alpha (maths italic)
12020 \string<\glshex 1D6FD% lower case beta (maths italic)
12021 \string<\glshex 1D6FE% lower case gamma (maths italic)
12022 \string<\glshex 1D6FF% lower case delta (maths italic)
12023 \string<\glshex 1D700% lower case epsilon (maths italic)
12024 \string<\glshex 1D716% lower case epsilon variant (maths italic)
12025 \string<\glshex 1D701% lower case zeta (maths italic)
12026 \string<\glshex 1D702% lower case eta (maths italic)
12027 \string<\glshex 1D703% lower case theta (maths italic)
12028 \string=\glshex 1D717% lower case theta variant (maths italic)
12029 \string<\glshex 1D704% lower case iota (maths italic)
12030 \string<\glshex 1D705% lower case kappa (maths italic)
12031 \string=\glshex 1D718% lower case kappa variant (maths italic)
12032 \string<\glshex 1D706% lower case lambda (maths italic)
12033 \string<\glshex 1D707% lower case mu (maths italic)
12034 \string<\glshex 1D708% lower case nu (maths italic)
12035 \string<\glshex 1D709% lower case xi (maths italic)
12036 \string<\glshex 1D70A% lower case omicron (maths italic)
12037 \string<\glshex 1D70B% lower case pi (maths italic)
12038 \string=\glshex 1D71B% lower case pi variant (maths italic)
12039 \string<\glshex 1D70C% lower case rho (maths italic)
12040 \string=\glshex 1D71A% lower case rho variant (maths italic)
12041 \string<\glshex 1D70D% lower case final sigma (maths italic)
12042 \string=\glshex 1D70E% lower case sigma (maths italic)
12043 \string<\glshex 1D70F% lower case tau (maths italic)
12044 \string<\glshex 1D710% lower case upsilon (maths italic)
12045 \string<\glshex 1D711% lower case phi (maths italic)
12046 \string=\glshex 1D719% lower case phi variant (maths italic)
12047 \string<\glshex 1D712% lower case chi (maths italic)
12048 \string<\glshex 1D713% lower case psi (maths italic)
12049 \string<\glshex 1D714% lower case omega (maths italic)
12050 }

```

**MathGreeklIrules** Includes both upright and italic with digamma between epsilon and zeta.

```

12051 \newcommand*{\glxtrMathGreeklIrules}{%
12052 \glxtrMathItalicAlpha
12053 \string;\glxtrUpAlpha
12054 \string<\glxtrMathItalicBeta
12055 \string;\glxtrUpBeta

```

```

12056 \string<\glxtrMathItalicGamma
12057 \string;\glxtrUpGamma
12058 \string<\glxtrMathItalicDelta
12059 \string;\glxtrUpDelta
12060 \string<\glxtrMathItalicEpsilon
12061 \string;\glxtrUpEpsilon
12062 \string<\glxtrUpDigamma
12063 \string<\glxtrMathItalicZeta
12064 \string;\glxtrUpZeta
12065 \string<\glxtrMathItalicEta
12066 \string;\glxtrUpEta
12067 \string<\glxtrMathItalicTheta
12068 \string;\glxtrUpTheta
12069 \string<\glxtrMathItalicIota
12070 \string;\glxtrUpIota
12071 \string<\glxtrMathItalicKappa
12072 \string;\glxtrUpKappa
12073 \string<\glxtrMathItalicLambda
12074 \string;\glxtrUpLambda
12075 \string<\glxtrMathItalicMu
12076 \string;\glxtrUpMu
12077 \string<\glxtrMathItalicNu
12078 \string;\glxtrUpNu
12079 \string<\glxtrMathItalicXi
12080 \string;\glxtrUpXi
12081 \string<\glxtrMathItalicOmicron
12082 \string;\glxtrUpOmicron
12083 \string<\glxtrMathItalicPi
12084 \string;\glxtrUpPi
12085 \string<\glxtrMathItalicRho
12086 \string;\glxtrUpRho
12087 \string<\glxtrMathItalicSigma
12088 \string;\glxtrUpSigma
12089 \string<\glxtrMathItalicTau
12090 \string;\glxtrUpTau
12091 \string<\glxtrMathItalicUpsilon
12092 \string;\glxtrUpUpsilon
12093 \string<\glxtrMathItalicPhi
12094 \string;\glxtrUpPhi
12095 \string<\glxtrMathItalicChi
12096 \string;\glxtrUpChi
12097 \string<\glxtrMathItalicPsi
12098 \string;\glxtrUpPsi
12099 \string<\glxtrMathItalicOmega
12100 \string;\glxtrUpOmega
12101 }

```

**mathGreekIIrules** Includes both upright and italic (digamma not included).

```

12102 \newcommand*{\glxtrMathGreekIIrules}{%

```



```

12103 \glxtrMathItalicAlpha
12104 \string;\glxtrUpAlpha
12105 \string<\glxtrMathItalicBeta
12106 \string;\glxtrUpBeta
12107 \string<\glxtrMathItalicGamma
12108 \string;\glxtrUpGamma
12109 \string<\glxtrMathItalicDelta
12110 \string;\glxtrUpDelta
12111 \string<\glxtrMathItalicEpsilon
12112 \string;\glxtrUpEpsilon
12113 \string<\glxtrMathItalicZeta
12114 \string;\glxtrUpZeta
12115 \string<\glxtrMathItalicEta
12116 \string;\glxtrUpEta
12117 \string<\glxtrMathItalicTheta
12118 \string;\glxtrUpTheta
12119 \string<\glxtrMathItalicIota
12120 \string;\glxtrUpIota
12121 \string<\glxtrMathItalicKappa
12122 \string;\glxtrUpKappa
12123 \string<\glxtrMathItalicLambda
12124 \string;\glxtrUpLambda
12125 \string<\glxtrMathItalicMu
12126 \string;\glxtrUpMu
12127 \string<\glxtrMathItalicNu
12128 \string;\glxtrUpNu
12129 \string<\glxtrMathItalicXi
12130 \string;\glxtrUpXi
12131 \string<\glxtrMathItalicOmicron
12132 \string;\glxtrUpOmicron
12133 \string<\glxtrMathItalicPi
12134 \string;\glxtrUpPi
12135 \string<\glxtrMathItalicRho
12136 \string;\glxtrUpRho
12137 \string<\glxtrMathItalicSigma
12138 \string;\glxtrUpSigma
12139 \string<\glxtrMathItalicTau
12140 \string;\glxtrUpTau
12141 \string<\glxtrMathItalicUpsilon
12142 \string;\glxtrUpUpsilon
12143 \string<\glxtrMathItalicPhi
12144 \string;\glxtrUpPhi
12145 \string<\glxtrMathItalicChi
12146 \string;\glxtrUpChi
12147 \string<\glxtrMathItalicPsi
12148 \string;\glxtrUpPsi
12149 \string<\glxtrMathItalicOmega
12150 \string;\glxtrUpOmega
12151 }

```

\glxtrUpAlpha

```
12152 \newcommand*{\glxtrUpAlpha}{%  
12153 \glshex 03B1,% lower case alpha  
12154 \glshex 0391% upper case alpha  
12155 }
```

\glxtrUpBeta

```
12156 \newcommand*{\glxtrUpBeta}{%  
12157 \glshex 03B2,% lower case beta  
12158 \glshex 0392% upper case beta  
12159 }
```

\glxtrUpGamma

```
12160 \newcommand*{\glxtrUpGamma}{%  
12161 \glshex 03B3,% lower case gamma  
12162 \glshex 0393% upper case gamma  
12163 }
```

\glxtrUpDelta

```
12164 \newcommand*{\glxtrUpDelta}{%  
12165 \glshex 03B4,% lower case delta  
12166 \glshex 0394% upper case delta  
12167 }
```

glxtrUpEpsilon

```
12168 \newcommand*{\glxtrUpEpsilon}{%  
12169 \glshex 03B5% lower case epsilon  
12170 \string=\glshex 03F5,% lower case epsilon variant  
12171 \glshex 0395% upper case epsilon  
12172 }
```

glxtrUpDigamma

```
12173 \newcommand*{\glxtrUpDigamma}{%  
12174 \glshex 03DD,% lower case digamma  
12175 \glshex 03DC% upper case digamma  
12176 }
```

\glxtrUpZeta

```
12177 \newcommand*{\glxtrUpZeta}{%  
12178 \glshex 03B6,% lower case zeta  
12179 \glshex 0396% upper case zeta  
12180 }
```

\glxtrUpEta

```
12181 \newcommand*{\glxtrUpEta}{%  
12182 \glshex 03B7,% lower case eta  
12183 \glshex 0397% upper case eta  
12184 }
```

```

\glxtrUpTheta
12185 \newcommand*{\glxtrUpTheta}{%
12186   \glshex 03B8% lower case theta
12187   \string=\glshex 03D1,% lower case theta variant
12188   \glshex 0398% upper case theta
12189 }

\glxtrUpIota
12190 \newcommand*{\glxtrUpIota}{%
12191   \glshex 03B9,% lower case iota
12192   \glshex 0399% upper case iota
12193 }

\glxtrUpKappa
12194 \newcommand*{\glxtrUpKappa}{%
12195   \glshex 03BA% lower case kappa
12196   \string=\glshex 03F0,% lower case kappa variant
12197   \glshex 039A% upper case kappa
12198 }

\glxtrUpLambda
12199 \newcommand*{\glxtrUpLambda}{%
12200   \glshex 03BB,% lower lambda
12201   \glshex 039B% upper case lambda
12202 }

\glxtrUpMu
12203 \newcommand*{\glxtrUpMu}{%
12204   \glshex 03BC,% lower case mu
12205   \glshex 039C% upper case mu
12206 }

\glxtrUpNu
12207 \newcommand*{\glxtrUpNu}{%
12208   \glshex 03BD,% lower case nu
12209   \glshex 039D% upper case nu
12210 }

\glxtrUpXi
12211 \newcommand*{\glxtrUpXi}{%
12212   \glshex 03BE,% lower case xi
12213   \glshex 039E% upper case xi
12214 }

glxtrUpOmicron
12215 \newcommand*{\glxtrUpOmicron}{%
12216   \glshex 03BF,% lower case omicron
12217   \glshex 039F% upper case omicron
12218 }

```

$\backslash\mathrm{glxtrUpPi}$

```
12219 \newcommand*{\glxtrUpPi}{%
12220 \glshex 03C0% lower case pi
12221 \string=\glshex 03D6,% lower case pi variant
12222 \glshex 03A0% upper case pi
12223 }
```

$\backslash\mathrm{glxtrUpRho}$

```
12224 \newcommand*{\glxtrUpRho}{%
12225 \glshex 03C1% lower case rho
12226 \string=\glshex 03F1,% lower case rho variant
12227 \glshex 03A1% upper case rho
12228 }
```

$\backslash\mathrm{glxtrUpSigma}$

```
12229 \newcommand*{\glxtrUpSigma}{%
12230 \glshex 03C2% lower case sigma
12231 \string=\glshex 03C3,% lower case sigma
12232 \glshex 03A3% upper case sigma
12233 }
```

$\backslash\mathrm{glxtrUpTau}$

```
12234 \newcommand*{\glxtrUpTau}{%
12235 \glshex 03C4,% lower case tau
12236 \glshex 03A4% upper case tau
12237 }
```

$\mathrm{glxtrUpUpsilon}$

```
12238 \newcommand*{\glxtrUpUpsilon}{%
12239 \glshex 03C5,% lower case upsilon
12240 \glshex 03A5% upper case upsilon
12241 }
```

$\backslash\mathrm{glxtrUpPhi}$

```
12242 \newcommand*{\glxtrUpPhi}{%
12243 \glshex 03C6% lower case phi
12244 \string=\glshex 03D5,% lower case phi variant
12245 \glshex 03A6% upper case phi
12246 }
```

$\backslash\mathrm{glxtrUpChi}$

```
12247 \newcommand*{\glxtrUpChi}{%
12248 \glshex 03C7,% lower case chi
12249 \glshex 03A7% upper case chi
12250 }
```

$\backslash\mathrm{glxtrUpPsi}$

```
12251 \newcommand*{\glxtrUpPsi}{%
```

```

12252 \glshex 03C8,% lower case psi
12253 \glshex 03A8% upper case psi
12254 }

```

$\backslash\mathrm{glxtrUpOmega}$

```

12255 \newcommand*{\glxtrUpOmega}{%
12256 \glshex 03C9,% lower case omega
12257 \glshex 03A9% upper case omega
12258 }

```

$\mathrm{MathItalicAlpha}$

```

12259 \newcommand*{\glxtrMathItalicAlpha}{%
12260 \glshex 1D6FC,% lower case alpha (maths italic)
12261 \glshex 1D6E2% upper case alpha (maths italic)
12262 }

```

$\mathrm{rMathItalicBeta}$

```

12263 \newcommand*{\glxtrMathItalicBeta}{%
12264 \glshex 1D6FD,% lower case beta (maths italic)
12265 \glshex 1D6E3% upper case beta (maths italic)
12266 }

```

$\mathrm{MathItalicGamma}$

```

12267 \newcommand*{\glxtrMathItalicGamma}{%
12268 \glshex 1D6FE,% lower case gamma (maths italic)
12269 \glshex 1D6E4% upper case gamma (maths italic)
12270 }

```

$\mathrm{MathItalicDelta}$

```

12271 \newcommand*{\glxtrMathItalicDelta}{%
12272 \glshex 1D6FF,% lower case delta (maths italic)
12273 \glshex 1D6E5% upper case delta (maths italic)
12274 }

```

$\mathrm{thItalicEpsilon}$

```

12275 \newcommand*{\glxtrMathItalicEpsilon}{%
12276 \glshex 1D700% lower case epsilon (maths italic)
12277 \string=\glshex 1D716,% lower case epsilon variant (maths italic)
12278 \glshex 1D6E6% upper case epsilon (maths italic)
12279 }

```

$\mathrm{rMathItalicZeta}$

```

12280 \newcommand*{\glxtrMathItalicZeta}{%
12281 \glshex 1D701,% lower case zeta (maths italic)
12282 \glshex 1D6E7% upper case zeta (maths italic)
12283 }

```

trMathItalicEta

```
12284 \newcommand*{\glxtrMathItalicEta}{%
12285   \glshex 1D702,% lower case eta (maths italic)
12286   \glshex 1D6E8% upper case eta (maths italic)
12287 }
```

MathItalicTheta

```
12288 \newcommand*{\glxtrMathItalicTheta}{%
12289   \glshex 1D703% lower case theta (maths italic)
12290   \string=\glshex 1D717,% lower case theta variant (maths italic)
12291   \glshex 1D6E9% upper case theta (maths italic)
12292   \string=\glshex 1D6F3% upper case theta variant (maths italic)
12293 }
```

rMathItalicIota

```
12294 \newcommand*{\glxtrMathItalicIota}{%
12295   \glshex 1D704,% lower case iota (maths italic)
12296   \glshex 1D6EA% upper case iota (maths italic)
12297 }
```

MathItalicKappa

```
12298 \newcommand*{\glxtrMathItalicKappa}{%
12299   \glshex 1D705% lower case kappa (maths italic)
12300   \string=\glshex 1D718,% lower case kappa variant (maths italic)
12301   \glshex 1D6EB% upper case kappa (maths italic)
12302 }
```

athItalicLambda

```
12303 \newcommand*{\glxtrMathItalicLambda}{%
12304   \glshex 1D706,% lower case lambda (maths italic)
12305   \glshex 1D6EC% upper case lambda (maths italic)
12306 }
```

xtrMathItalicMu

```
12307 \newcommand*{\glxtrMathItalicMu}{%
12308   \glshex 1D707,% lower case mu (maths italic)
12309   \glshex 1D6ED% upper case mu (maths italic)
12310 }
```

xtrMathItalicNu

```
12311 \newcommand*{\glxtrMathItalicNu}{%
12312   \glshex 1D708,% lower case nu (maths italic)
12313   \glshex 1D6EE% upper case nu (maths italic)
12314 }
```

xtrMathItalicXi

```
12315 \newcommand*{\glxtrMathItalicXi}{%
12316   \glshex 1D709,% lower case xi (maths italic)
```

```

12317 \glshex 1D6EF% upper case xi (maths italic)
12318 }

```

thItalicOmicron

```

12319 \newcommand*{\glxtrMathItalicOmicron}{%
12320 \glshex 1D70A,% lower case omicron (maths italic)
12321 \glshex 1D6F0% upper case omicron (maths italic)
12322 }

```

xtrMathItalicPi

```

12323 \newcommand*{\glxtrMathItalicPi}{%
12324 \glshex 1D70B% lower case pi (maths italic)
12325 \string=\glshex 1D71B,% lower case pi variant (maths italic)
12326 \glshex 1D6F1% upper case pi (maths italic)
12327 }

```

trMathItalicRho

```

12328 \newcommand*{\glxtrMathItalicRho}{%
12329 \glshex 1D70C% lower case rho (maths italic)
12330 \string=\glshex 1D71A,% lower case rho variant (maths italic)
12331 \glshex 1D6F2% upper case rho (maths italic)
12332 }

```

MathItalicSigma

```

12333 \newcommand*{\glxtrMathItalicSigma}{%
12334 \glshex 1D70D% lower case final sigma (maths italic)
12335 \string=\glshex 1D70E,% lower case sigma (maths italic)
12336 \glshex 1D6F4% upper case sigma (maths italic)
12337 }

```

trMathItalicTau

```

12338 \newcommand*{\glxtrMathItalicTau}{%
12339 \glshex 1D70F,% lower case tau (maths italic)
12340 \glshex 1D6F5% upper case tau (maths italic)
12341 }

```

thItalicUpsilon

```

12342 \newcommand*{\glxtrMathItalicUpsilon}{%
12343 \glshex 1D710,% lower case upsilon (maths italic)
12344 \glshex 1D6F6% upper case upsilon (maths italic)
12345 }

```

trMathItalicPhi

```

12346 \newcommand*{\glxtrMathItalicPhi}{%
12347 \glshex 1D711% lower case phi (maths italic)
12348 \string=\glshex 1D719,% lower case phi variant (maths italic)
12349 \glshex 1D6F7% upper case phi (maths italic)
12350 }

```

trMathItalicChi

```
12351 \newcommand*{\glxtrMathItalicChi}{%
12352   \glshex 1D712,% lower case chi (maths italic)
12353   \glshex 1D6F8% upper case chi (maths italic)
12354 }
```

trMathItalicPsi

```
12355 \newcommand*{\glxtrMathItalicPsi}{%
12356   \glshex 1D713,% lower case psi (maths italic)
12357   \glshex 1D6F9% upper case psi (maths italic)
12358 }
```

MathItalicOmega

```
12359 \newcommand*{\glxtrMathItalicOmega}{%
12360   \glshex 1D714,% lower case omega (maths italic)
12361   \glshex 1D6FA% upper case omega (maths italic)
12362 }
```

thItalicPartial

```
12363 \newcommand*{\glxtrMathItalicPartial}{%
12364   \glshex 1D715% partial differential (maths italic)
12365 }
```

MathItalicNabla

```
12366 \newcommand*{\glxtrMathItalicNabla}{%
12367   \glshex 1D6FB% nabla (maths italic)
12368 }
```

lsxtrdigitrules Digits from the Basic Latin set and subscript and superscript digit rules.

```
12369 \newcommand*{\glxtrdigitrules}{%
12370   0\string=\glshex 2080\string=\glshex 2070
12371   \string<1\string=\glshex 2081\string=\glshex 00B9
12372   \string<2\string=\glshex 2082\string=\glshex 00B2
12373   \string<3\string=\glshex 2083\string=\glshex 00B3
12374   \string<4\string=\glshex 2084\string=\glshex 2074
12375   \string<5\string=\glshex 2085\string=\glshex 2075
12376   \string<6\string=\glshex 2086\string=\glshex 2076
12377   \string<7\string=\glshex 2087\string=\glshex 2077
12378   \string<8\string=\glshex 2088\string=\glshex 2078
12379   \string<9\string=\glshex 2089\string=\glshex 2079
12380 }
```

BasicDigitrules Digits from the Basic Latin set.

```
12381 \newcommand*{\glxtrBasicDigitrules}{%
12382   0\string<1\string<2\string<3\string<4%
12383   \string<5\string<6\string<7\string<8\string<9%
12384 }
```



criptDigitrules Subscript digits.

```
12385 \newcommand*{\glxtrSubScriptDigitrules}{%
12386 \glshex 2080% subscript 0
12387 \string<\glshex 2081% subscript 1
12388 \string<\glshex 2082% subscript 2
12389 \string<\glshex 2083% subscript 3
12390 \string<\glshex 2084% subscript 4
12391 \string<\glshex 2085% subscript 5
12392 \string<\glshex 2086% subscript 6
12393 \string<\glshex 2087% subscript 7
12394 \string<\glshex 2088% subscript 8
12395 \string<\glshex 2089% subscript 9
12396 }
```

criptDigitrules Superscript digits.

```
12397 \newcommand*{\glxtrSuperScriptDigitrules}{%
12398 \glshex 2070% superscript 0
12399 \string<\glshex 00B9% superscript 1
12400 \string<\glshex 00B2% superscript 2
12401 \string<\glshex 00B3% superscript 3
12402 \string<\glshex 2074% superscript 4
12403 \string<\glshex 2075% superscript 5
12404 \string<\glshex 2076% superscript 6
12405 \string<\glshex 2077% superscript 7
12406 \string<\glshex 2078% superscript 8
12407 \string<\glshex 2079% superscript 9
12408 }
```

trfractionrules Vulgar fractions.

```
12409 \newcommand*{\glxtrfractionrules}{%
12410 \glshex 215F% fraction numerator one (1/)
12411 \string<\glshex 2189% zero thirds (0/3 = 0)
12412 \string<\glshex 2152% one tenth (1/10 = 0.1)
12413 \string<\glshex 2151% one ninth (1/9 ~ 0.111)
12414 \string<\glshex 215B% one eighth (1/8 = 0.125)
12415 \string<\glshex 2150% one seventh (1/7 ~ 0.143)
12416 \string<\glshex 2159% one sixth (1/6 ~ 0.167)
12417 \string<\glshex 2155% one fifth (1/5 = 0.2)
12418 \string<\glshex 00BC% one quarter (1/4 = 0.25)
12419 \string<\glshex 2153% one third (1/3 ~ 0.333)
12420 \string<\glshex 215C% three eighths (3/8 = 0.375)
12421 \string<\glshex 2156% two fifths (2/5 = 0.4)
12422 \string<\glshex 00BD% one half (1/2 = 0.5)
12423 \string<\glshex 2157% three fifths (3/5 = 0.6)
12424 \string<\glshex 215D% five eighths (5/8 = 0.625)
12425 \string<\glshex 2154% two thirds (2/3 ~ 0.667)
12426 \string<\glshex 00BE% three quarters (3/4 = 0.75)
12427 \string<\glshex 2158% four fifths (4/5 = 0.8)
12428 \string<\glshex 215A% five sixths (5/6 ~ 0.833)
```

```

12429 \string<\glshex 215E% seven eighths (7/8 = 0.875)
12430 }

```

sxtrdialecthook Check for scripts associated with the document dialects.

```

12431 \renewcommand{\@glsxtrdialecthook}{%
12432   \ifundef\CurrentTrackedScript
12433   {%
12434     \TrackLangIfHasDefaultScript{\CurrentTrackedLanguage}%
12435     {%
12436       \edef\CurrentTrackedScript{%
12437         \TrackLangGetDefaultScript\CurrentTrackedLanguage}%
12438     }%
12439   }%
12440 }%
12441 {}%
12442 \ifdef\CurrentTrackedScript
12443 {%
12444   \let\gls@orgTrackLangRequireDialectPrefix\TrackLangRequireDialectPrefix
12445   \def\TrackLangRequireDialectPrefix{glossariesxtr-}%
12446   \let\CurrentTrackedTag\CurrentTrackedScript
12447   \IfFileExists{\TrackLangRequireDialectPrefix\CurrentTrackedTag.ldf}
12448   {\RequireGlossariesExtraLang{\CurrentTrackedTag}}%
12449   {}%
12450   \let\TrackLangRequireDialectPrefix\gls@orgTrackLangRequireDialectPrefix
12451 }%
12452 {}%
12453 }

```

If `\glsxtr@loaddialect` has been defined, then `glossaries-extra-bib2gls` has been loaded after `glossaries-extra`. (For example, through `\glossariesextrasetup`.) Not recommended, but if this has been done try to find the associated language resources.

```

12454 \ifdef\glsxtr@loaddialect
12455 {%
12456   \@ifpackageloaded{tracklang}
12457   {%
12458     \AnyTrackedLanguages
12459     {%
12460       \ForEachTrackedDialect{\this@dialect}{\glsxtr@loaddialect}%
12461     }%
12462   }%
12463 }
12464 {}
12465 }
12466 {}

```

## 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:

```
12467 \NeedsTeXFormat{LaTeX2e}
12468 \ProvidesPackage{glossaries-extra-stylemods}[2018/03/06 v1.28 (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`

```
12469 \newcommand*{\@glxtr@loadstyles}{}
```

`all` Provide all known styles.

```
12470 \DeclareOption{all}{%
12471   \appto\@glxtr@loadstyles{%
12472     \RequirePackage{glossary-inline}%
12473     \RequirePackage{glossary-list}%
12474     \RequirePackage{glossary-tree}%
12475     \RequirePackage{glossary-mcols}%
12476     \RequirePackage{glossary-long}%
12477     \RequirePackage{glossary-longragged}%
12478     \RequirePackage{glossary-longbooktabs}%
12479     \RequirePackage{glossary-super}%
12480     \RequirePackage{glossary-superragged}%
12481     \RequirePackage{glossary-bookindex}%
12482   }
12483 }

12484 \DeclareOption*{%
12485   \IfFileExists{glossary-\CurrentOption.sty}
12486   {\eappto\@glxtr@loadstyles{%
12487     \noexpand\RequirePackage{glossary-\CurrentOption}}}%
12488   }%
12489   {%
12490     \PackageError{glossaries-extra-styles}%

```

```

12491     {Unknown option ‘\CurrentOption’}{}%
12492   }%
12493 }

```

Process the package options:

```
12494 \ProcessOptions
```

Load the required packages:

```
12495 \@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`.

```
12496 \providecommand*{\glsxtrprelocation}{\space}
```

In case we have an old version of glossaries:

`ewglossarystyle`

```

12497 \providecommand{\renewglossarystyle}[2]{%
12498   \ifcsundef{@glsstyle@#1}%
12499   {%
12500     \PackageError{glossaries-extra}{Glossary style ‘#1’ isn’t already defined}{}%
12501   }%
12502   {%
12503     \csdef{@glsstyle@#1}{#2}%
12504   }%
12505 }

```

## 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.

```

12506 \ifdef{@glsstyle@listdotted}
12507 {%
12508   \renewglossarystyle{listdotted}{%
12509     \setglossarystyle{list}%
12510     \renewcommand*{\glossentry}[2]{%
12511       \item[]\makebox[\glslistdottedwidth][l]{%
12512         \glsentryitem{##1}%
12513         \glstarget{##1}{\glossentryname{##1}}%
12514         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
12515         \glossentrydesc{##1}\glspostdescription}%
12516     \renewcommand*{\subglossentry}[3]{%
12517       \item[]\makebox[\glslistdottedwidth][l]{%
12518         \glssubentryitem{##2}%
12519         \glstarget{##2}{\glossentryname{##2}}%
12520         \unskip\leaders\hbox to 2.9mm{\hss.}\hfill\strut}%
12521         \glossentrydesc{##2}\glspostdescription}%
12522   }

```

```
12523 }
12524 {%
```

Assume the style isn't required if it hasn't already been defined.

```
12525 }
```

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.

```
12526 \ifdef{\@glsstyle@list}
12527 {%
```

listprelocation Space before number list for top-level entries.

```
12528 \newcommand{\glslistprelocation}{\glsxtrprelocation}
```

childprelocation Space before number list for child entries.

```
12529 \newcommand{\glslistchildprelocation}{\glslistprelocation}
```

childpostlocation Full stop after number list.

```
12530 \newcommand{\glslistchildpostlocation}{.}
```

Redefine list to use these commands.

```
12531 \renewglossarystyle{list}{%
12532 \renewenvironment{theglossary}%
12533 {\begin{description}}{\end{description}}%
12534 \renewcommand*\{glossaryheader}{}%
12535 \renewcommand*\{glsgroupheading}[1]{}%
12536 \renewcommand*\{glossentry}[2]{%
12537 \item[\glsentryitem{##1}%
12538 \glstarget{##1}{\glossentryname{##1}}]
12539 \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
12540 \renewcommand*\{subglossentry}[3]{%
12541 \glssubentryitem{##2}%
12542 \glstarget{##2}{\strut}\space
12543 \glossentrydesc{##2}\glspostdescription
12544 \glslistchildprelocation ##3\glslistchildpostlocation}%
12545 \renewcommand*\{glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
12546 }
12547 }
12548 }
```

Similarly for altlist. Since it requires list, the new commands should have been defined above.

```
12549 \ifdef{\@glsstyle@altlist}
12550 {%
12551 \renewglossarystyle{altlist}{%
12552 \setglossarystyle{list}%
12553 \renewcommand*\{glossentry}[2]{%
12554 \item[\glsentryitem{##1}%
```

```

12555     \glstarget{##1}{\glossentryname{##1}}}%
12556     \mbox{}\par\nobreak\@afterheading
12557     \glossentrydesc{##1}\glspostdescription\glslistprelocation ##2}%
12558 \renewcommand{\subglossentry}[3]{%
12559     \par
12560     \glssubentryitem{##2}%
12561     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12562     \glslistchildprelocation ##3}%
12563 }
12564 }
12565 {}

```

Redefine listgroup so that it discourages a break after group headings.

```

12566 \ifdef{\@glsstyle@listgroup}
12567 {%
12568   \renewglossarystyle{listgroup}{%
12569     \setglossarystyle{list}%
12570     \renewcommand*\glsgroupheading[1]{%
12571       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
12572       \mbox{}\par\nobreak\@afterheading
12573     }%
12574   }
12575 }
12576 {}

```

Similarly for listhypergroup.

```

12577 \ifdef{\@glsstyle@listhypergroup}
12578 {%
12579   \renewglossarystyle{listhypergroup}{%
12580     \setglossarystyle{list}%
12581     \renewcommand*\glossaryheader{%
12582       \glslistnavigationitem{\glsnavigation}}%
12583     \renewcommand*\glsgroupheading[1]{%
12584       \item[\glslistgroupheaderfmt
12585         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
12586       \mbox{}\par\nobreak\@afterheading
12587     }%
12588   }
12589 }
12590 {}

```

Similarly for altlistgroup.

```

12591 \ifdef{\@glsstyle@altlistgroup}
12592 {%
12593   \renewglossarystyle{altlistgroup}{%
12594     \setglossarystyle{altlist}%
12595     \renewcommand*\glsgroupheading[1]{%
12596       \item[\glslistgroupheaderfmt{\glsgetgrouptitle{##1}}}%
12597       \mbox{}\par\nobreak\@afterheading
12598     }%
12599   }

```

```

12600 }
12601 {}

    Similarly for altlisthypergroup.
12602 \ifdef{\@glsstyle@altlisthypergroup}
12603 {%
12604   \renewglossarystyle{altlisthypergroup}{%
12605     \setglossarystyle{altlist}%
12606     \renewcommand*\{glossaryheader\}{%
12607       \glslistnavigationitem{\glsnavigation}}%
12608     \renewcommand*\{glsgroupheading\}[1]{%
12609       \item[\glslistgroupheaderfmt
12610         {\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}]%
12611       \mbox{}\par\nobreak\@afterheading
12612     }%
12613   }
12614 }
12615 {}

```

## 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`.

```

12616 \ifcsdef{@glsstyle@long}
12617 {%
12618   \renewglossarystyle{long}{%
12619     \renewenvironment{theglossary}%
12620       {\begin{longtable}{lp{\glsdescwidth}}}%
12621       {\end{longtable}}%
12622     \renewcommand*\{glossaryheader\}{}%
12623     \renewcommand*\{glsgroupheading\}[1]{}%
12624     \renewcommand{\glossentry}[2]{%
12625       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12626       \glossentrydesc{##1}\glspostdescription
12627       \glstrprelocation ##2\tabularnewline
12628     }%
12629     \renewcommand{\subglossentry}[3]{%
12630       &
12631       \glssubentryitem{##2}%
12632       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12633       \glstrprelocation ##3\tabularnewline
12634     }%
12635     \ifglsnogroupskip
12636       \renewcommand*\{glsgroupskip\}{}%
12637     \else
12638       \renewcommand*\{glsgroupskip\}{ & \tabularnewline}%
12639     \fi
12640   }

```

```
12641 }
12642 {}
```

Three column style:

```
12643 \ifcsdef{@glsstyle@long3col}
12644 {%
12645   \renewglossarystyle{long3col}{%
12646     \renewenvironment{theglossary}%
12647       {\begin{longtable}{lp{\glsdescwidth}p{\glspagelistwidth}}}%
12648       {\end{longtable}}}%
12649     \renewcommand*{\glossaryheader}{}%
12650     \renewcommand*{\glsgroupheading}[1]{}%
12651     \renewcommand{\glossentry}[2]{%
12652       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12653       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
12654     }%
12655     \renewcommand{\subglossentry}[3]{%
12656       &
12657       \glssubentryitem{##2}%
12658       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12659       ##3\tabularnewline
12660     }%
```

Conditional needs to be outside of \glsgroupskip otherwise it can cause “Incomplete \iftrue” errors.

```
12661   \ifglsgroupskip
12662     \renewcommand*{\glsgroupskip}{}%
12663   \else
12664     \renewcommand*{\glsgroupskip}{& &\tabularnewline}%
12665   \fi
12666 }
12667 }
12668 {}
```

Four column style:

```
12669 \ifcsdef{@glsstyle@long4col}
12670 {%
12671   \renewglossarystyle{long4col}{%
12672     \renewenvironment{theglossary}%
12673       {\begin{longtable}{llll}}}%
12674       {\end{longtable}}}%
12675     \renewcommand*{\glossaryheader}{}%
12676     \renewcommand*{\glsgroupheading}[1]{}%
12677     \renewcommand{\glossentry}[2]{%
12678       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12679       \glossentrydesc{##1}\glspostdescription &
12680       \glossentrysymbol{##1} &
12681       ##2\tabularnewline
12682     }%
12683     \renewcommand{\subglossentry}[3]{%
12684       &
```



```

12685     \glssubentryitem{##2}%
12686     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12687     \glossentrysymbol{##2} & ##3\tabularnewline
12688 }%

12689 \ifglsgroupskip
12690   \renewcommand*{\glsgroupskip}{}%
12691 \else
12692   \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
12693 \fi
12694 }
12695 }
12696 {}

```

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`.

```

12697 \ifcsdef{@glstyle@longragged}
12698 {%
12699   \renewglossarystyle{longragged}{%
12700     \renewenvironment{theglossary}%
12701       {\begin{longtable}[l>{\raggedright}p{\glsdescwidth}]}%
12702       {\end{longtable}}%
12703     \renewcommand*{\glossaryheader}{}%
12704     \renewcommand*{\glsgroupheading}[1]{}%
12705     \renewcommand{\glossentry}[2]{%
12706       \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12707       \glossentrydesc{##1}\glspostdescription\glxtrprelocation ##2%
12708       \tabularnewline
12709     }%
12710     \renewcommand{\subglossentry}[3]{%
12711       &
12712       \glssubentryitem{##2}%
12713       \glstarget{##2}{\strut}\glossentrydesc{##2}%
12714       \glspostdescription\glxtrprelocation ##3%
12715       \tabularnewline
12716     }%
12717     \ifglsgroupskip
12718       \renewcommand*{\glsgroupskip}{}%
12719     \else
12720       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12721     \fi
12722   }
12723 }

```

12724 {}

Three and four column styles don't use \glxstrprelocation since the number list is in its own column.

```
12725 \ifcsdef{@glsstyle@longragged3col}
12726 {%
12727   \renewglossarystyle{longragged3col}{%
12728     \renewenvironment{theglossary}%
12729       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}%
12730        >{\raggedright}p{\glspagelistwidth}}}%
12731       {\end{longtable}}}%
12732     \renewcommand*{\glossaryheader}{}%
12733     \renewcommand*{\glsgroupheading}[1]{}%
12734     \renewcommand{\glossentry}[2]{%
12735       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12736       \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
12737     }%
12738     \renewcommand{\subglossentry}[3]{%
12739       &
12740       \glssubentryitem{##2}%
12741       \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12742       ##3\tabularnewline
12743     }%
12744     \ifglsgroupskip
12745       \renewcommand*{\glsgroupskip}{}%
12746     \else
12747       \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12748     \fi
12749   }
12750 }
12751 {}
```

Four column style:

```
12752 \ifcsdef{@glsstyle@altlongragged4col}
12753 {%
12754   \renewglossarystyle{altlongragged4col}{%
12755     \renewenvironment{theglossary}%
12756       {\begin{longtable}{l>{\raggedright}p{\glsdescwidth}l%
12757        >{\raggedright}p{\glspagelistwidth}}}%
12758       {\end{longtable}}}%
12759     \renewcommand*{\glossaryheader}{}%
12760     \renewcommand*{\glsgroupheading}[1]{}%
12761     \renewcommand{\glossentry}[2]{%
12762       \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12763       \glossentrydesc{##1}\glspostdescription & \glossentrysymbol{##1} &
12764       ##2\tabularnewline
12765     }%
12766     \renewcommand{\subglossentry}[3]{%
12767       &
```

```

12768     \glssubentryitem{##2}%
12769     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12770     \glossentrysymbol{##2} & ##3\tabularnewline
12771 }%

12772 \ifglsnogroupskip
12773     \renewcommand*{\glsgroupskip}{}%
12774 \else
12775     \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
12776 \fi
12777 }
12778 }
12779 {}

```

## 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`.

```

12780 \ifcsdef{@glstyle@super}
12781 {%
12782     \renewglossarystyle{super}{%
12783         \renewenvironment{theglossary}%
12784             {\tablehead{}\tabletail}%
12785             \begin{supertabular}{lp{\glsdescwidth}}%
12786             {\end{supertabular}}%
12787     \renewcommand*{\glossaryheader}{}%
12788     \renewcommand*{\glsgroupheading}[1]{}%
12789     \renewcommand{\glossentry}[2]{%
12790         \glssubentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12791         \glossentrydesc{##1}\glspostdescription
12792         \glxtrprelocation ##2\tabularnewline
12793     }%
12794     \renewcommand{\subglossentry}[3]{%
12795         &
12796         \glssubentryitem{##2}%
12797         \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12798         \glxtrprelocation ##3\tabularnewline
12799     }%
12800     \ifglsnogroupskip
12801         \renewcommand*{\glsgroupskip}{}%
12802     \else
12803         \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12804     \fi
12805 }
12806 }
12807 {}

```

Three column style:

```

12808 \ifcsdef{@glstyle@super3col}

```

```

12809 {%
12810   \renewglossarystyle{super3col}{%
12811     \renewenvironment{theglossary}%
12812       {\tablehead{}\tabletail{}}%
12813       \begin{supertabular}{lp{\glsgdescwidth}p{\glspagelistwidth}}}%
12814       {\end{supertabular}}}%
12815   \renewcommand*{\glossaryheader}{}%
12816   \renewcommand*{\glsgroupheading}[1]{}%
12817   \renewcommand{\glossentry}[2]{%
12818     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12819     \glossentrydesc{##1}\glspostdescription & ##2\tabularnewline
12820   }%
12821   \renewcommand{\subglossentry}[3]{%
12822     &
12823     \glssubentryitem{##2}%
12824     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12825     ##3\tabularnewline
12826   }%

12827   \ifglsgnogroupskip
12828     \renewcommand*{\glsgroupskip}{}%
12829   \else
12830     \renewcommand*{\glsgroupskip}{ & &\tabularnewline}%
12831   \fi
12832 }
12833 }
12834 {}

```

Four column styles:

```

12835 \ifcsdef{@glstyle@super4col}
12836 {%
12837   \renewglossarystyle{super4col}{%
12838     \renewenvironment{theglossary}%
12839       {\tablehead{}\tabletail{}}%
12840       \begin{supertabular}{llll}}}%
12841       \end{supertabular}}}%
12842   \renewcommand*{\glossaryheader}{}%
12843   \renewcommand*{\glsgroupheading}[1]{}%
12844   \renewcommand{\glossentry}[2]{%
12845     \glentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12846     \glossentrydesc{##1}\glspostdescription &
12847     \glossentrysymbol{##1} & ##2\tabularnewline
12848   }%
12849   \renewcommand{\subglossentry}[3]{%
12850     &
12851     \glssubentryitem{##2}%
12852     \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12853     \glossentrysymbol{##2} & ##3\tabularnewline
12854   }%

```

```

12855 \ifglsgroupskip
12856 \renewcommand*{\glsgroupskip}{}%
12857 \else
12858 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
12859 \fi
12860 }
12861 }
12862 {}

```

## 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`.

```

12863 \ifcsdef{@glstyle@superragged}
12864 {%
12865 \renewglossarystyle{superragged}{%
12866 \renewenvironment{theglossary}%
12867 {\tablehead{}}\tabletail{}}%
12868 \begin{supertabular}{1>{\raggedright}p{\glsgdescwidth}}%
12869 {\end{supertabular}}%
12870 \renewcommand*{\glossaryheader}{}%
12871 \renewcommand*{\glsgroupheading}[1]{}%
12872 \renewcommand{\glossentry}[2]{%
12873 \glstryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12874 \glossentrydesc{##1}\glspostdescription\glstrprelocation ##2%
12875 \tabularnewline
12876 }%
12877 \renewcommand{\subglossentry}[3]{%
12878 &
12879 \glssubentryitem{##2}%
12880 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription
12881 \glstrprelocation ##3%
12882 \tabularnewline
12883 }%
12884 \ifglsgroupskip
12885 \renewcommand*{\glsgroupskip}{}%
12886 \else
12887 \renewcommand*{\glsgroupskip}{& \tabularnewline}%
12888 \fi
12889 }
12890 }
12891 {}

```

Three column style:

```

12892 \ifcsdef{@glstyle@superragged3col}
12893 {%
12894 \renewglossarystyle{superragged3col}{%

```

```

12895 \renewenvironment{theglossary}%
12896 {\tablehead{}\tabletail{}}%
12897 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}%
12898 >{\raggedright}p{\glspagelistwidth}}}%
12899 {\end{supertabular}}%
12900 \renewcommand*{\glossaryheader}{}%
12901 \renewcommand*{\glsgroupheading}[1]{}%
12902 \renewcommand{\glossentry}[2]{%
12903 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12904 \glossentrydesc{##1}\glspostdescription &
12905 ##2\tabularnewline
12906 }%
12907 \renewcommand{\subglossentry}[3]{%
12908 &
12909 \glssubentryitem{##2}%
12910 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12911 ##3\tabularnewline
12912 }%
12913 \ifglsgroupskip
12914 \renewcommand*{\glsgroupskip}{}%
12915 \else
12916 \renewcommand*{\glsgroupskip}{ & \tabularnewline}%
12917 \fi
12918 }
12919 }
12920 {}

```

Four columns:

```

12921 \ifcsdef{@glsstyle@altsuperragged4col}
12922 {%
12923 \renewglossarystyle{altsuperragged4col}{%
12924 \renewenvironment{theglossary}%
12925 {\tablehead{}\tabletail{}}%
12926 \begin{supertabular}{l>{\raggedright}p{\glsdescwidth}l%
12927 >{\raggedright}p{\glspagelistwidth}}}%
12928 {\end{supertabular}}%
12929 \renewcommand*{\glossaryheader}{}%
12930 \renewcommand{\glossentry}[2]{%
12931 \glsentryitem{##1}\glstarget{##1}{\glossentryname{##1}} &
12932 \glossentrydesc{##1}\glspostdescription &
12933 \glossentrysymbol{##1} & ##2\tabularnewline
12934 }%
12935 \renewcommand{\subglossentry}[3]{%
12936 &
12937 \glssubentryitem{##2}%
12938 \glstarget{##2}{\strut}\glossentrydesc{##2}\glspostdescription &
12939 \glossentrysymbol{##2} & ##3\tabularnewline
12940 }%

```

```

12941 \ifglsnogroupskip
12942 \renewcommand*{\glsgroupskip}{}%
12943 \else
12944 \renewcommand*{\glsgroupskip}{& & \tabularnewline}%
12945 \fi
12946 }
12947 }
12948 {}

```

## 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`.

```

12949 \ifdef{\@glsstyle@inline}
12950 {%
12951 \renewcommand*{\glspostinline}{.\spacefactor\sfcode'\.}
12952 \renewcommand*{\glsinlinedescformat}[3]{%
12953 \space#1\glxtrpostdescription}
12954 \renewcommand*{\glsinlinesubdescformat}[3]{%
12955 #1\glxtrpostdescription}

```

Just use `\glxtrpostdescription` instead of `\glspostdescription`.

```

12956 }
12957 {}

```

## 2.8 Tree Styles

The index style is redefined so that the space before the number list isn't hard coded.

```

12958 \ifdef{\@glsstyle@index}
12959 {

```

`treeprelocation` The space before the number list for top-level entries. This is shared by the other tree styles.

```

12960 \newcommand*{\glstreeprelocation}{\glxtrprelocation}

```

`childprelocation` The space before the number list for child entries. This is shared by the other tree styles.

```

12961 \newcommand*{\glstreechildprelocation}{\glstreeprelocation}

12962 \renewglossarystyle{index}{%
12963 \renewenvironment{theglossary}%
12964 {\setlength{\parindent}{0pt}%
12965 \setlength{\parskip}{0pt plus 0.3pt}%
12966 \let\item\glstreeitem
12967 \let\subitem\glstreesubitem

```

```

12968     \let\subsubitem\glstreesubsubitem
12969   }%
12970 {\par}%
12971   \renewcommand*\{\glossaryheader}\{%
12972   \renewcommand*\{\glsgroupheading}[1]\{%
12973   \renewcommand*\{\glossentry}[2]\{%
12974     \item\glstentryitem{##1}%
12975     \glstreenamefmt{\glstarget{##1}\glossentryname{##1}}%
12976     \ifglshassymbol{##1}\space(\glossentrysymbol{##1})\{%
12977     \glstreepredesc \glossentrydesc{##1}\glspostdescription
12978     \glstreeprelocation ##2%
12979   }%
12980   \renewcommand{\subglossentry}[3]\{%
12981     \ifcase##1\relax
12982       \item
12983     \or
12984       \subitem
12985       \glssubentryitem{##2}%
12986     \else
12987       \subsubitem
12988     \fi
12989     \glstreenamefmt{\glstarget{##2}\glossentryname{##2}}%
12990     \ifglshassymbol{##2}\space(\glossentrysymbol{##2})\{%
12991     \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
12992     \glstreechildprelocation ##3%
12993   }%
12994   \renewcommand*\{\glsgroupskip}\{\ifglsnogroupskip\else\indexspace\fi}%
12995 }
12996 }
12997 {}

```

The indexgroup style is redefined to discourage a page break after the heading.

```

12998 \ifdef{\@glsstyle@indexgroup}
12999 {%
13000   \renewglossarystyle{indexgroup}\{%
13001     \setglossarystyle{index}%
13002     \renewcommand*\{\glsgroupheading}[1]\{%
13003       \item\glstreegroupheaderfmt{\glsggetgrouptitle{##1}}%
13004       \nopagebreak\indexspace
13005       \nobreak\@afterheading
13006     }%
13007   }
13008 }
13009 {}

```

Similarly for indexhypergroup.

```

13010 \ifdef{\@glsstyle@indexhypergroup}
13011 {%
13012   \renewglossarystyle{indexhypergroup}\{%
13013     \setglossarystyle{index}%

```



```

13014 \renewcommand*{\glossaryheader}{%
13015     \item\glstreenavigationfmt{\glsnavigation}%
13016     \nobreak\@afterheading\indexspace}%
13017 \renewcommand*{\glsgroupheading}[1]{%
13018     \item\glstreegroupheaderfmt
13019     {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13020     \nopagebreak\indexspace
13021     \nobreak\@afterheading}%
13022 }%
13023 }
13024 {}

```

Adjust tree style to remove hard coded space before number list.

```

13025 \ifdef{\@glsstyle@tree}
13026 {%
13027 \renewglossarystyle{tree}{%
13028     \renewenvironment{theglossary}%
13029     {\setlength{\parindent}{0pt}%
13030     \setlength{\parskip}{0pt plus 0.3pt}}%
13031     {}%
13032     \renewcommand*{\glossaryheader}{}%
13033     \renewcommand*{\glsgroupheading}[1]{}%
13034     \renewcommand{\glossentry}[2]{%
13035         \hangindent0pt\relax
13036         \parindent0pt\relax
13037         \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13038         \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}%
13039         \glstreepredesc\glossentrydesc{##1}\glspostdescription
13040         \glstreeprelocation##2\par
13041     }%
13042     \renewcommand{\subglossentry}[3]{%
13043         \hangindent##1\glstreeindent\relax
13044         \parindent##1\glstreeindent\relax
13045         \ifnum##1=1\relax
13046             \glssubentryitem{##2}%
13047         \fi
13048         \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}%
13049         \ifglshassymbol{##2}{\space(\glossentrysymbol{##2})}%
13050         \glstreechildpredesc\glossentrydesc{##2}\glspostdescription
13051         \glstreechildprelocation ##3\par
13052     }%
13053     \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13054 }%
13055 }
13056 {}

```

The treegroup style is redefined to discourage a page break after the heading.

```

13057 \ifdef{\@glsstyle@treegroup}
13058 {%
13059 \renewglossarystyle{treegroup}{%

```

```

13060 \setglossarystyle{tree}%
13061 \renewcommand{\glsgroupheading}[1]{\par
13062 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}\par
13063 \nopagebreak\indexspace\nobreak\@afterheading}%
13064 }
13065 }
13066 {}

```

Similarly for treehypergroup

```

13067 \ifdef{\@glsstyle@treehypergroup}
13068 {%
13069 \renewglossarystyle{treehypergroup}{%
13070 \setglossarystyle{tree}%
13071 \renewcommand*\glossaryheader{%
13072 \par\noindent\glstreenavigationfmt{\glsnavigation}\par
13073 \nobreak\@afterheading\indexspace}%
13074 \renewcommand*\glsgroupheading}[1]{%
13075 \par\noindent
13076 \glstreegroupheaderfmt
13077 {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
13078 \nopagebreak\indexspace\nobreak\@afterheading}%
13079 }
13080 }
13081 {}

```

Adjust treenoname style to remove hard coded space before number list.

```

13082 \ifdef{\@glsstyle@treenoname}
13083 {%
13084 \renewglossarystyle{treenoname}{%
13085 \renewenvironment{theglossary}%
13086 {\setlength{\parindent}{0pt}%
13087 \setlength{\parskip}{0pt plus 0.3pt}}%
13088 {}%
13089 \renewcommand*\glossaryheader{}%
13090 \renewcommand*\glsgroupheading}[1]{}%
13091 \renewcommand{\glossentry}[2]{%
13092 \hangindent0pt\relax
13093 \parindent0pt\relax
13094 \glstryitem{##1}\glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13095 \ifglshassymbol{##1}{\space(\glossentrysymbol{##1})}%
13096 \glstreepredesc\glossentrydesc{##1}\glspostdescription
13097 \glstreeprelocation##2\par
13098 }%
13099 \renewcommand{\subglossentry}[3]{%
13100 \hangindent##1\glstreeindent\relax
13101 \parindent##1\glstreeindent\relax
13102 \ifnum##1=1\relax
13103 \glssubentryitem{##2}%
13104 \fi
13105 \glstarget{##2}{\strut}%

```

```

13106      \glossentrydesc{##2}\glspostdescription\glstreechildprelocation##3\par
13107    }%
13108    \renewcommand*{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13109  }
13110 }
13111 {}

```

The `treenonamegroup` style is redefined to discourage a page break after the heading.

```

13112 \ifdef{\@glsstyle@treenonamegroup}
13113 {%
13114   \renewglossarystyle{treenonamegroup}{%
13115     \setglossarystyle{treenoname}%
13116     \renewcommand{\glsgroupheading}[1]{\par
13117       \noindent\glstreegroupheaderfmt
13118       {\glsggetgrouptitle{##1}}}%
13119     \nopagebreak\indexspace\nobreak\@afterheading
13120   }%
13121 }
13122 }
13123 {}

```

Similarly for `treenonamehypergroup`

```

13124 \ifdef{\@glsstyle@treenonamehypergroup}
13125 {%
13126   \renewglossarystyle{treenonamehypergroup}{%
13127     \setglossarystyle{treenoname}%
13128     \renewcommand*{\glossaryheader}{%
13129       \par\noindent\glstreenavigationfmt{\glsnavigation}\par
13130       \nobreak\@afterheading\indexspace}%
13131     \renewcommand*{\glsgroupheading}[1]{%
13132       \par\noindent
13133       \glstreegroupheaderfmt
13134       {\glsnavhypertarget{##1}{\glsggetgrouptitle{##1}}}%
13135     \nopagebreak\indexspace\nobreak\@afterheading}%
13136   }
13137 }
13138 {}

```

The `almtree` style is redefined to make it easier to made minor adjustments.

```

13139 \ifdef{\@glsstyle@almtree}
13140 {%

```

Only redefine this style if it's already been defined.

mbolDescLocation

```
\glxtralmtreeSymbolDescLocation{\langle label \rangle}{\langle location list \rangle}
```

Layout the symbol, description and location for top-level entries.

```

13141   \newcommand{\glxtralmtreeSymbolDescLocation}[2]{%

```

```

13142   {%
13143       \let\par\glxstrAltTreePar
13144       \ifglshassymbol{#1}{(\glossentrysymbol{#1})\space}{}%
13145       \glossentrydesc{#1}\glspostdescription\glstreeprelocation #2\par
13146   }%
13147 }

```

**trAltTreeIndent** Paragraph indent for subsequent paragraphs in multi-paragraph descriptions.

```
13148 \newlength\glxstrAltTreeIndent
```

**lsxtrAltTreePar** Multi-paragraph descriptions need to keep the hanging indent.

```

13149 \newcommand{\glxstrAltTreePar}{%
13150     @@par
13151     \glxstrAltTreeSetHangIndent
13152     \setlength{\parindent}{\dimexpr\hangindent+\glxstrAltTreeIndent}%
13153 }

```

**symbolDescLocation** `\glxstralttreeSubSymbolDescLocation{<level>}{<label>}{<location list>}`

Layout the symbol, description and location for sub-entries. Defaults to the same as the top-level.

```

13154 \newcommand{\glxstralttreeSubSymbolDescLocation}[3]{%
13155     \glxstralttreeSymbolDescLocation{#2}{#3}%
13156 }

```

**trtreetopindent** The original style has to keep computing the width of the name at each entry. This register allows the style to compute it once for the top-level at the start of the glossary.

```
13157 \newlength\glxstrtreetopindent
```

**lxstralttreeInit** User-level initialisation for the alttree style.

```

13158 \newcommand*{\glxstralttreeInit}{%
13159     \settowidth{\glxstrtreetopindent}{\glstreenamefmt{\glsgetwidestname\space}}%
13160     \glxstrAltTreeIndent=\parindent
13161 }

```

**\glsglsetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\gdef`.

```

13162 \newcommand*{\glsglsetwidest}[2][0]{%
13163     \csgdef{@glswidestname\romannumeral#1}{#2}%
13164 }

```

**\eglssetwidest** The original `\glsglsetwidest` only uses `\def`. This uses `\protected@csedef`.

```

13165 \newcommand*{\eglssetwidest}[2][0]{%
13166     \protected@csedef{@glswidestname\romannumeral#1}{#2}%
13167 }

```

`\xglsetwidest` Like the above but uses `\protected@csxdef`.

```
13168 \newcommand*\xglsetwidest}[2][0]{%
13169   \protected@csxdef{#1}{#2}%
13170 }
```

`glsupdatewidest` Only sets if new value is wider than old value.

```
13171 \newcommand*\glsupdatewidest}[2][0]{%
13172   \ifcsundef{#1}%
13173   {\csdef{#1}{#2}}%
13174   {%
13175     \settowidth{#1}{\csuse{#1}}%
13176     \settowidth{#2}%
13177     \ifdim#2>#1%
13178     \csdef{#1}{#2}%
13179     \fi
13180   }%
13181 }
```

`glsupdatewidest` As above but global definition.

```
13182 \newcommand*\glsupdatewidest}[2][0]{%
13183   \ifcsundef{#1}%
13184   {\csgdef{#1}{#2}}%
13185   {%
13186     \settowidth{#1}{\csuse{#1}}%
13187     \settowidth{#2}%
13188     \ifdim#2>#1%
13189     \csgdef{#1}{#2}%
13190     \fi
13191   }%
13192 }
```

`glsupdatewidest` As `\glsupdatewidest` but expands value.

```
13193 \newcommand*\eglsupdatewidest}[2][0]{%
13194   \ifcsundef{#1}%
13195   {\protected@csedef{#1}{#2}}%
13196   {%
13197     \settowidth{#1}{\csuse{#1}}%
13198     \settowidth{#2}%
13199     \ifdim#2>#1%
13200     \protected@csedef{#1}{#2}%
13201     \fi
13202   }%
13203 }
```

`glsupdatewidest` As above but global.

```
13204 \newcommand*\xglsupdatewidest}[2][0]{%
13205   \ifcsundef{#1}%
13206   {\protected@csxdef{#1}{#2}}%
13207   {%
```

```

13208      \settowidth{\dimen@}{\csuse{@glswidestname\romannumeral#1}}%
13209      \settowidth{\dimen@ii}{#2}%
13210      \ifdim\dimen@ii>\dimen@
13211        \protected@csxdef{@glswidestname\romannumeral#1}{#2}%
13212      \fi
13213    }%
13214  }

```

**glswidestname** Provide a user-level macro to obtain the widest top-level name.

```

13215  \newcommand*{\glswidestname}{\@glswidestname}

```

**glswidestsubname** Provide a user-level macro to obtain the widest sub-entry name.

```

13216  \newcommand*{\glswidestsubname}[1]{%
13217    \ifcsundef{@glswidestname\romannumeral#1}%
13218    {\@glswidestname}%
13219    {\csuse{@glswidestname\romannumeral#1}}%
13220  }

```

**glswidestTopLevelName** CamelCase is easier for long command names. Provide a CamelCase synonym of `\glswidestTopLevelName`.

```

13221  \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.

```

13222  \newrobustcmd*{\glswidestUsedTopLevelName}[1][\@glo@types]{%
13223    \dimen@=0pt\relax
13224    \gls@tmplen=0pt\relax
13225    \forallglossaries[#1]{\@gls@type}%
13226    {%
13227      \forglsentries[\@gls@type]{\@glo@label}%
13228      {%
13229        \ifglswidest{\@glo@label}%
13230        {%
13231          \ifglshasparent{\@glo@label}%
13232          {}%
13233          {%
13234            \settowidth{\dimen@}{%
13235              {\glstreenamfmt{\glstryname{\@glo@label}}}%
13236            \ifdim\dimen@>\gls@tmplen
13237              \gls@tmplen=\dimen@
13238            \eglssetwidest{\glstryname{\@glo@label}}%
13239            \fi
13240          }%
13241        }%
13242      }%
13243    }%
13244  }%
13245  }

```

`destUsedAnyName` Like the above but doesn't check the parent key. Useful if all levels should have the same width for the name.

```

13246 \newrobustcmd*{\glsFindWidestUsedAnyName}[1][\@glo@types]{%
13247   \dimen@=0pt\relax
13248   \gls@tmplen=0pt\relax
13249   \forallglossaries[#1]{\@gls@type}%
13250   {%
13251     \forallgsentries[\@gls@type]{\@glo@label}%
13252     {%
13253       \ifglsused{\@glo@label}%
13254       {%
13255         \settowidth{\dimen@}%
13256           {\glstreenamefmt{\glssentryname{\@glo@label}}}%
13257         \ifdim\dimen@>\gls@tmplen
13258           \gls@tmplen=\dimen@
13259           \eglssetwidest{\glssentryname{\@glo@label}}%
13260         \fi
13261       }%
13262     }%
13263   }%
13264 }%
13265 }
```

`ndWidestAnyName` Like the above but doesn't check if the entry has been used.

```

13266 \newrobustcmd*{\glsFindWidestAnyName}[1][\@glo@types]{%
13267   \dimen@=0pt\relax
13268   \gls@tmplen=0pt\relax
13269   \forallglossaries[#1]{\@gls@type}%
13270   {%
13271     \forallgsentries[\@gls@type]{\@glo@label}%
13272     {%
13273       \settowidth{\dimen@}%
13274         {\glstreenamefmt{\glssentryname{\@glo@label}}}%
13275       \ifdim\dimen@>\gls@tmplen
13276         \gls@tmplen=\dimen@
13277         \eglssetwidest{\glssentryname{\@glo@label}}%
13278       \fi
13279     }%
13280   }%
13281 }
```

`estUsedLevelTwo` This is like `\glsFindWidestUsedTopLevelName` but also sets the first two sub-levels as well. Any entry that has a great-grandparent is ignored.

```

13282 \newrobustcmd*{\glsFindWidestUsedLevelTwo}[1][\@glo@types]{%
13283   \dimen@=0pt\relax
13284   \dimen@i=0pt\relax
13285   \dimen@ii=0pt\relax
13286   \forallglossaries[#1]{\@gls@type}%
13287   {%
```

```

13288 \forglsentries[\@gls@type]{\@glo@label}%
13289 {%
13290 \ifglsused{\@glo@label}%
13291 {%
13292 \ifglsashasparent{\@glo@label}%
13293 {%
13294 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@label}@parent}}}%
13295 \ifglsashasparent{\@glo@parent}%
13296 {%
13297 \edef\@glo@parent{\csuse{glo@\glsdetoklabel{\@glo@parent}@parent}}}%
13298 \ifglsashasparent{\@glo@parent}%
13299 {}%
13300 {%
13301 \settowidth{\gls@tmplen}%
13302 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13303 \ifdim\gls@tmplen>\dimen@ii
13304 \dimen@ii=\gls@tmplen
13305 \eglssetwidest[2]{\glsentryname{\@glo@label}}%
13306 \fi
13307 }%
13308 }%
13309 {%
13310 \settowidth{\gls@tmplen}%
13311 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13312 \ifdim\gls@tmplen>\dimen@i
13313 \dimen@i=\gls@tmplen
13314 \eglssetwidest[1]{\glsentryname{\@glo@label}}%
13315 \fi
13316 }%
13317 }%
13318 {%
13319 \settowidth{\gls@tmplen}%
13320 {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13321 \ifdim\gls@tmplen>\dimen@
13322 \dimen@=\gls@tmplen
13323 \eglssetwidest{\glsentryname{\@glo@label}}%
13324 \fi
13325 }%
13326 }%
13327 {}%
13328 }%
13329 }%
13330 }

```

`dWidestLevelTwo` This is like `\glsFindWidestUsedLevelTwo` but doesn't check if the entry has been used.

```

13331 \newrobustcmd*{\glsFindWidestLevelTwo}[1][\@glo@types]{%
13332 \dimen@=0pt\relax
13333 \dimen@i=0pt\relax
13334 \dimen@ii=0pt\relax

```



```

13335 \forall glossaries[#1]{\@gls@type}%
13336 {%
13337   \forall sentries[\@gls@type]{\@glo@label}%
13338   {%
13339     \ifglshasparent{\@glo@label}%
13340     {%
13341       \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@label}@parent}}%
13342       \ifglshasparent{\@glo@parent}%
13343       {%
13344         \edef\@glo@parent{\csuse{glo@\glstetoklabel{\@glo@parent}@parent}}%
13345         \ifglshasparent{\@glo@parent}%
13346         {}%
13347         {%
13348           \settowidth{\gls@tmplen}%
13349             {\glstreenamfmt{\glstentryname{\@glo@label}}}%
13350           \ifdim\gls@tmplen>\dimen@ii
13351             \dimen@ii=\gls@tmplen
13352             \eglssetwidest[2]{\glstentryname{\@glo@label}}%
13353           \fi
13354         }%
13355       }%
13356     {%
13357       \settowidth{\gls@tmplen}%
13358         {\glstreenamfmt{\glstentryname{\@glo@label}}}%
13359       \ifdim\gls@tmplen>\dimen@i
13360         \dimen@i=\gls@tmplen
13361         \eglssetwidest[1]{\glstentryname{\@glo@label}}%
13362       \fi
13363     }%
13364   }%
13365   {%
13366     \settowidth{\gls@tmplen}%
13367       {\glstreenamfmt{\glstentryname{\@glo@label}}}%
13368     \ifdim\gls@tmplen>\dimen@
13369       \dimen@=\gls@tmplen
13370       \eglssetwidest{\glstentryname{\@glo@label}}%
13371     \fi
13372   }%
13373 }%
13374 }%
13375 }

```

**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.

```

13376 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbol}[2][\@glo@types]{%
13377   \dimen@=0pt\relax
13378   \gls@tmplen=0pt\relax
13379   #2=0pt\relax
13380   \forall glossaries[#1]{\@gls@type}%

```

```

13381  {%
13382    \forglsentries[\@gls@type]{\@glo@label}%
13383    {%
13384      \ifglsused{\@glo@label}%
13385      {%
13386        \settowidth{\dimen@}%
13387          {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13388        \ifdim\dimen@>\gls@tmplen
13389          \gls@tmplen=\dimen@
13390          \eglssetwidest{\glsentryname{\@glo@label}}%
13391        \fi
13392        \settowidth{\dimen@}%
13393          {\glsentrysymbol{\@glo@label}}%
13394        \ifdim\dimen@>#2\relax
13395          #2=\dimen@
13396        \fi
13397      }%
13398    }%
13399  }%
13400 }%
13401 }

```

**stAnyNameSymbol** Like the above but doesn't check if the entry has been used.

```

13402 \newrobustcmd*{\glsFindWidestAnyNameSymbol}[2][\@glo@types]{%
13403   \dimen@=0pt\relax
13404   \gls@tmplen=0pt\relax
13405   #2=0pt\relax
13406   \forallglossaries[#1]{\@gls@type}%
13407   {%
13408     \forglsentries[\@gls@type]{\@glo@label}%
13409     {%
13410       \settowidth{\dimen@}%
13411         {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13412       \ifdim\dimen@>\gls@tmplen
13413         \gls@tmplen=\dimen@
13414         \eglssetwidest{\glsentryname{\@glo@label}}%
13415       \fi
13416       \settowidth{\dimen@}%
13417         {\glsentrysymbol{\@glo@label}}%
13418       \ifdim\dimen@>#2\relax
13419         #2=\dimen@
13420       \fi
13421     }%
13422   }%
13423 }

```

**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.

```

13424 \newrobustcmd*{\glsFindWidestUsedAnyNameSymbolLocation}[3][\@glo@types]{%
13425   \dimen@=0pt\relax
13426   \gls@tmplen=0pt\relax
13427   #2=0pt\relax
13428   #3=0pt\relax
13429   \forallglossaries[#1]{\@gls@type}%
13430   {%
13431     \forglsentries[\@gls@type]{\@glo@label}%
13432     {%
13433       \ifglsused{\@glo@label}%
13434       {%
13435         \settowidth{\dimen@}%
13436         {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13437         \ifdim\dimen@>\gls@tmplen
13438           \gls@tmplen=\dimen@
13439           \eglssetwidest{\glsentryname{\@glo@label}}%
13440         \fi
13441         \settowidth{\dimen@}%
13442         {\glsentrysymbol{\@glo@label}}%
13443         \ifdim\dimen@>#2\relax
13444           #2=\dimen@
13445         \fi
13446         \settowidth{\dimen@}%
13447         {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13448         \ifdim\dimen@>#3\relax
13449           #3=\dimen@
13450         \fi
13451       }%
13452     }%
13453   }%
13454 }%
13455 }
```

`eSymbolLocation` Like the `\glsFindWidestUsedAnyNameSymbol` but doesn't check if the entry has been used.

```

13456 \newrobustcmd*{\glsFindWidestAnyNameSymbolLocation}[3][\@glo@types]{%
13457   \dimen@=0pt\relax
13458   \gls@tmplen=0pt\relax
13459   #2=0pt\relax
13460   #3=0pt\relax
13461   \forallglossaries[#1]{\@gls@type}%
13462   {%
13463     \forglsentries[\@gls@type]{\@glo@label}%
13464     {%
13465       \settowidth{\dimen@}%
13466       {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13467       \ifdim\dimen@>\gls@tmplen
13468         \gls@tmplen=\dimen@
13469         \eglssetwidest{\glsentryname{\@glo@label}}%
```

```

13470      \fi
13471      \settowidth{\dimen@}%
13472        {\glsentrysymbol{\@glo@label}}}%
13473      \ifdim\dimen@>#2\relax
13474        #2=\dimen@
13475      \fi
13476      \settowidth{\dimen@}%
13477        {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13478      \ifdim\dimen@>#3\relax
13479        #3=\dimen@
13480      \fi
13481    }%
13482  }%
13483 }

```

**AnyNameLocation** Like the `\glsFindWidestUsedAnyNameSymbolLocation` 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.

```

13484 \newrobustcmd*{\glsFindWidestUsedAnyNameLocation}[2][\@glo@types]{%
13485   \dimen@=0pt\relax
13486   \gls@tmplen=0pt\relax
13487   #2=0pt\relax
13488   \forallglossaries[#1]{\@gls@type}%
13489   {%
13490     \forallglsentries[\@gls@type]{\@glo@label}%
13491     {%
13492       \ifglsused{\@glo@label}%
13493       {%
13494         \settowidth{\dimen@}%
13495           {\glstreenamfmt{\glsentryname{\@glo@label}}}%
13496         \ifdim\dimen@>\gls@tmplen
13497           \gls@tmplen=\dimen@
13498           \eglssetwidest{\glsentryname{\@glo@label}}%
13499         \fi
13500         \settowidth{\dimen@}%
13501           {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13502         \ifdim\dimen@>#2\relax
13503           #2=\dimen@
13504         \fi
13505       }%
13506     }%
13507   }%
13508 }%
13509 }

```

**AnyNameLocation** Like the `\glsFindWidestAnyNameLocation` but doesn't check the **first use** flag.

```

13510 \newrobustcmd*{\glsFindWidestAnyNameLocation}[2][\@glo@types]{%
13511   \dimen@=0pt\relax
13512   \gls@tmplen=0pt\relax

```

```

13513      #2=Opt\relax
13514      \forallglossaries[#1]{\@gls@type}%
13515      {%
13516        \forglsentries[\@gls@type]{\@glo@label}%
13517        {%
13518          \settowidth{\dimen@}%
13519            {\glstreenamefmt{\glsentryname{\@glo@label}}}%
13520          \ifdim\dimen@>\gls@tmplen
13521            \gls@tmplen=\dimen@
13522            \eglssetwidest{\glsentryname{\@glo@label}}%
13523          \fi
13524          \settowidth{\dimen@}%
13525            {\GlsXtrFormatLocationList{\glsentrynumberlist{\@glo@label}}}%
13526          \ifdim\dimen@>#2\relax
13527            #2=\dimen@
13528          \fi
13529        }%
13530      }%
13531    }

```

**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`.

```

13532    \newcommand*{\glsxtrComputeTreeIndent}[1]{%
13533      \glstreeindent=\glsxtrtreetopindent\relax
13534    }

```

**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.

```

13535    \newcommand*{\glsxtrComputeTreeSubIndent}[3]{%
13536      \ifcsundef{@glswidestname\romannumeral#1}%
13537      {%
13538        \settowidth{#3}{\glstreenamefmt{\@glswidestname\space}}%
13539      }%
13540      {%
13541        \settowidth{#3}{\glstreenamefmt{%
13542          \csname @glswidestname\romannumeral#1\endcsname\space}}%
13543      }%
13544    }

```

**treeSetHangIndent** Set `\hangindent` for top-level entries:

```

13545    \newcommand*{\glsxtrAltTreeSetHangIndent}{\hangindent\glstreeindent}

```

etSubHangIndent Set \hangindent for sub-entries:

```
13546 \newcommand*{\glxtrAltTreeSetSubHangIndent}[1]{\hangindent\glstreeindent}
```

Redefine alttree:

```
13547 \renewglossarystyle{alttree}{%
13548 \renewenvironment{theglossary}%
13549 {%
13550 \glxtralttreeInit
13551 \def\@gls@prevlevel{-1}%
13552 \mbox{}\par}%
13553 {\par}%
13554 \renewcommand*{\glossaryheader}{}%
13555 \renewcommand*{\glsgroupheading}[1]{}%
13556 \renewcommand{\glossentry}[2]{%
13557 \ifnum\@gls@prevlevel=0\relax
13558 \else
13559 \glxtrComputeTreeIndent{##1}%
13560 \fi
13561 \parindent\glstreeindent
13562 \glxtrAltTreeSetHangIndent
13563 \makebox[0pt][r]%
13564 {%
13565 \glstreenamebox{\glstreeindent}%
13566 {%
13567 \glssentryitem{##1}%
13568 \glstreenamefmt{\glstarget{##1}{\glossentryname{##1}}}%
13569 }%
13570 }%
13571 \glxtralttreeSymbolDescLocation{##1}{##2}%
13572 \def\@gls@prevlevel{0}%
13573 }
13574 \renewcommand{\subglossentry}[3]{%
13575 \ifnum##1=1\relax
13576 \glssubentryitem{##2}%
13577 \fi
13578 \ifnum\@gls@prevlevel=##1\relax
13579 \else
13580 \glxtrComputeTreeSubIndent{##1}{##2}{\gls@tmplen}%
13581 \ifnum\@gls@prevlevel<##1\relax
13582 \setlength\glstreeindent\gls@tmplen
13583 \addtolength\glstreeindent\parindent
13584 \parindent\glstreeindent
13585 \else
13586 \ifnum\@gls@prevlevel=0\relax
13587 \glxtrComputeTreeIndent{##2}%
13588 \else
13589 \glxtrComputeTreeSubIndent{\@gls@prevlevel}{##2}{\glstreeindent}%
13590 \fi
13591 \addtolength\parindent{-\glstreeindent}%

```

```

13592         \setlength\glstreeindent\parindent
13593     \fi
13594 \fi
13595 \glstrAltTreeSetSubHangIndent{##1}%
13596 \makebox[Opt][r]{\glstreenamebox{\gls@tmplen}{%
13597     \glstreenamefmt{\glstarget{##2}{\glossentryname{##2}}}}}%
13598 \glstralttreeSubSymbolDescLocation{##1}{##2}{##3}%
13599 \def\@gls@prevlevel{##1}%
13600 }%
13601 \renewcommand*\{\glsgroupskip}{\ifglsnogroupskip\else\indexspace\fi}%
13602 }
13603 }%
13604 {%
13605 }

```

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.

```

13606 \ifdef{\@glsstyle@alttreegroup}
13607 {%
13608     \renewglossarystyle{alttreegroup}{%
13609         \setglossarystyle{alttree}%
13610         \renewcommand{\glsgroupheading}[1]{\par
13611             \def\@gls@prevlevel{-1}%
13612             \hangindentOpt\relax
13613             \parindentOpt\relax
13614             \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}%
13615             \nopagebreak\indexspace\nopagebreak
13616         }%
13617     }%
13618 }%
13619 {%
13620 }

```

Similarly for `alttreehypergroup`.

```

13621 \ifdef{\@glsstyle@alttreehypergroup}
13622 {%
13623     \renewglossarystyle{alttreehypergroup}{%
13624         \setglossarystyle{alttree}%
13625         \renewcommand*\{\glossaryheader}{%
13626             \par
13627             \def\@gls@prevlevel{-1}%
13628             \hangindentOpt\relax
13629             \parindentOpt\relax
13630             \glstreenavigationfmt{\glsnavigation}\par\indexspace
13631         }%
13632         \renewcommand*\{\glsgroupheading}[1]{%
13633             \par
13634             \def\@gls@prevlevel{-1}%
13635             \hangindentOpt\relax
13636             \parindentOpt\relax

```

```

13637     \glstreegroupheaderfmt
13638     {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}\par
13639     \nopagebreak\indexspace\nopagebreak
13640 }%
13641 }
13642 }%
13643 {%
13644 }

```

## 2.9 Multicolumn Styles

Adjust mcolindexgroup to discourage page breaks after the group headings.

```

13645 \ifdef{\@glsstyle@mcolindexgroup}
13646 {%
13647   \renewglossarystyle{mcolindexgroup}{%
13648     \setglossarystyle{mcolindex}%
13649     \renewcommand*{\glsgroupheading}[1]{%
13650       \item\glstreegroupheaderfmt{\glsgrouptitle{##1}}%
13651       \nopagebreak\indexspace\nobreak\@afterheading
13652     }%
13653   }
13654 }%
13655 {%
13656 }

```

Similarly for mcolindexhypergroup.

```

13657 \ifdef{\@glsstyle@mcolindexhypergroup}
13658 {%
13659   \renewglossarystyle{mcolindexhypergroup}{%
13660     \setglossarystyle{mcolindex}%
13661     \renewcommand*{\glossaryheader}{%
13662       \item\glstreenavigationfmt{\glsnavigation}%
13663       \indexspace
13664     }%
13665     \renewcommand*{\glsgroupheading}[1]{%
13666       \item\glstreegroupheaderfmt
13667         {\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
13668       \nopagebreak\indexspace\nobreak\@afterheading
13669     }%
13670   }
13671 }%
13672 {%
13673 }

```

Similarly for mcolindexspannav.

```

13674 \ifdef{\@glsstyle@mcolindexspannav}
13675 {%
13676   \renewglossarystyle{mcolindexspannav}{%
13677     \setglossarystyle{index}%

```



```

13678 \renewenvironment{theglossary}%
13679 {%
13680 \begin{multicols}{\glsmcols}[\noindent\glstreenavigationfmt{\glsnavigation}]]%
13681 \setlength{\parindent}{0pt}%
13682 \setlength{\parskip}{0pt plus 0.3pt}%
13683 \let\item\glstreeitem}%
13684 {\end{multicols}}}%
13685 \renewcommand*{\glsgroupheading}[1]{%
13686 \item\glstreegroupheaderfmt
13687 {\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13688 \nopagebreak\indexspace\nobreak\@afterheading
13689 }%
13690 }
13691 }%
13692 {%
13693 }

```

Similarly for mcoltreegroup.

```

13694 \ifdef{\@glstyle@mcoltreegroup}
13695 {%
13696 \renewglossarystyle{mcoltreegroup}{%
13697 \setglossarystyle{mcoltree}%
13698 \renewcommand{\glsgroupheading}[1]{\par
13699 \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
13700 \nopagebreak\indexspace\nobreak\@afterheading
13701 }%
13702 }
13703 }%
13704 {%
13705 }

```

Similarly for mcoltreehypergroup.

```

13706 \ifdef{\@glstyle@mcoltreehypergroup}
13707 {%
13708 \renewglossarystyle{mcoltreehypergroup}{%
13709 \setglossarystyle{mcoltree}%
13710 \renewcommand*{\glossaryheader}{%
13711 \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace
13712 }%
13713 \renewcommand*{\glsgroupheading}[1]{%
13714 \par\noindent
13715 \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13716 \nopagebreak\indexspace\nobreak\@afterheading
13717 }%
13718 }
13719 }%
13720 {%
13721 }

```

Similarly for mcoltreespannav.

```

13722 \ifdef{\@glstyle@mcoltreespannav}

```

```

13723 {%
13724   \renewglossarystyle{mcoltreespannav}{%
13725     \setglossarystyle{tree}%
13726     \renewenvironment{theglossary}%
13727     {%
13728       \begin{multicols}{\glsmcols}%
13729       [\noindent\glstreenavigationfmt{\glsnavigation}]%
13730       \setlength{\parindent}{0pt}%
13731       \setlength{\parskip}{0pt plus 0.3pt}%
13732     }%
13733   {\end{multicols}}}%
13734   \renewcommand*{\glsgroupheading}[1]{%
13735     \par\noindent
13736     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13737     \nopagebreak\indexspace\nobreak\@afterheading
13738   }%
13739 }
13740 }%
13741 {%
13742 }

```

Similarly for mcoltreenonamegroup.

```

13743 \ifdef{\@glsstyle@mcoltreenonamegroup}
13744 {%
13745   \renewglossarystyle{mcoltreenonamegroup}{%
13746     \setglossarystyle{mcoltreenoname}%
13747     \renewcommand*{\glsgroupheading}[1]{\par
13748       \noindent\glstreegroupheaderfmt{\glsgrouptitle{##1}}}%
13749     \nopagebreak\indexspace\nobreak\@afterheading
13750   }%
13751 }
13752 }%
13753 {%
13754 }

```

Similarly for mcoltreenonamehypergroup.

```

13755 \ifdef{\@glsstyle@mcoltreenonamehypergroup}
13756 {%
13757   \renewglossarystyle{mcoltreenonamehypergroup}{%
13758     \setglossarystyle{mcoltreenoname}%
13759     \renewcommand*{\glossaryheader}{%
13760       \par\noindent\glstreenavigationfmt{\glsnavigation}\par\indexspace}%
13761     \renewcommand*{\glsgroupheading}[1]{%
13762       \par\noindent
13763       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgrouptitle{##1}}}%
13764       \nopagebreak\indexspace\nobreak\@afterheading}%
13765   }
13766 }%
13767 {%
13768 }

```

Similarly for mcoltreenonamespannav.

```

13769 \ifdef{\@glsstyle@mcoltreenonamespannav}
13770 {%
13771   \renewglossarystyle{mcoltreenonamespannav}{%
13772     \setglossarystyle{treenoname}%
13773     \renewenvironment{theglossary}%
13774     {%
13775       \begin{multicols}{\glsmcols}%
13776       [\noindent\glstreenavigationfmt{\glsnavigation}}%
13777       \setlength{\parindent}{0pt}%
13778       \setlength{\parskip}{0pt plus 0.3pt}%
13779     }%
13780     {\end{multicols}}}%
13781   \renewcommand*{\glsgroupheading}[1]{%
13782     \par\noindent
13783     \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
13784     \nopagebreak\indexspace\nobreak\@afterheading}%
13785   }
13786 }%
13787 {%
13788 }

```

mcolalmtree needs adjusting so that it uses \glxtralttreeInit This doesn't use \mbox{\par which would unbalance the top of the columns.

```

13789 \ifdef{\@glsstyle@mcolalmtree}
13790 {%
13791   \renewglossarystyle{mcolalmtree}{%
13792     \setglossarystyle{almtree}%
13793     \renewenvironment{theglossary}%
13794     {%
13795       \glxtralttreeInit
13796       \def\@gls@prevlevel{-1}%
13797       \begin{multicols}{\glsmcols}%
13798     }%
13799     {\par\end{multicols}}}%
13800   }
13801 }%
13802 {%
13803 }

```

Redefine mcolalmtreegroup to discourage page breaks after the group headings.

```

13804 \ifdef{\@glsstyle@mcolalmtreegroup}
13805 {%
13806   \renewglossarystyle{mcolalmtreegroup}{%
13807     \setglossarystyle{mcolalmtree}%
13808     \renewcommand{\glsgroupheading}[1]{\par
13809       \def\@gls@prevlevel{-1}%
13810       \hangindent0pt\relax
13811       \parindent0pt\relax
13812       \glstreegroupheaderfmt{\glsgetgrouptitle{##1}}}%

```

```

13813      \nopagebreak\indexspace\nopagebreak
13814    }%
13815  }
13816 }%
13817 {%
13818 }

```

Similarly for mcolalmtreehypergroup.

```

13819 \ifdef{\@glsstyle@mcolalmtreehypergroup}
13820 {%
13821   \renewglossarystyle{mcolalmtreehypergroup}{%
13822     \setglossarystyle{mcolalmtree}%
13823     \renewcommand*{\glossaryheader}{%
13824       \par
13825       \def\@gls@prevlevel{-1}%
13826       \hangindent0pt\relax
13827       \parindent0pt\relax
13828       \glstreenavigationfmt{\glsnavigation}%
13829       \par\indexspace
13830     }%
13831     \renewcommand*{\glsgroupheading}[1]{%
13832       \par
13833       \def\@gls@prevlevel{-1}%
13834       \hangindent0pt\relax
13835       \parindent0pt\relax
13836       \glstreegroupheaderfmt{\glsnavhypertarget{##1}{\glsgetgrouptitle{##1}}}%
13837       \nopagebreak\indexspace\nopagebreak
13838     }%
13839   }
13840 }%
13841 {%
13842 }

```

Similarly for mcolalttreespannav.

```

13843 \ifdef{\@glsstyle@mcolalttreespannav}
13844 {%
13845   \renewglossarystyle{mcolalttreespannav}{%
13846     \setglossarystyle{alttree}%
13847     \renewenvironment{theglossary}%
13848     {%
13849       \glsxtralmtreeInit
13850       \def\@gls@prevlevel{-1}%
13851       \begin{multicols}{\glsncols}%
13852         [\noindent\glstreenavigationfmt{\glsnavigation}]%
13853     }%
13854     {\par\end{multicols}}%
13855     \renewcommand*{\glsgroupheading}[1]{%
13856       \par
13857       \def\@gls@prevlevel{-1}%
13858       \hangindent0pt\relax

```

```

13859      \parindent0pt\relax
13860      \glstreegroupheaderfmt{\glsnahypertarget{##1}{\glsgrouptitle{##1}}}%
13861      \nopagebreak\indexspace\nopagebreak
13862    }%
13863  }
13864 }%
13865 {%
13866 }

Reset the default style
13867 \ifx\@glossary@default@style\relax
13868 \else
13869   \setglossarystyle{\@glxtr@current@style}
13870 \fi

```

# 3 bookindex style (glossary-bookindex.sty)

## 3.1 Package Initialisation and Options

```
13871 \NeedsTeXFormat{LaTeX2e}
13872 \ProvidesPackage{glossary-bookindex}[2018/03/06 v1.28 (NLCT)]
```

Load required packages.

```
13873 \RequirePackage{multicol}
13874 \RequirePackage{glossary-tree}
```

`trbookindexcols` Number of columns.

```
13875 \newcommand{\glstrbookindexcols}{2}
```

`trbookindexname` Format used for top-level entries. (Argument is the label.)

```
13876 \newcommand*{\glstrbookindexname}[1]{\glossentryname{#1}}
```

`ookindexsubname` Format used for sub entries.

```
13877 \newcommand*{\glstrbookindexsubname}[1]{\glstrbookindexname{#1}}
```

`strprelocation` Provide in case glossaries-stylemods isn't loaded.

```
13878 \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.

```
13879 \newcommand*{\glstrbookindexprelocation}[1]{%
13880   \glstrifhasfield{location}{#1}%
13881   {,\glstrprelocation}%
13882   {\glstrprelocation}%
13883 }
```

`xsubprelocation` Separator used before location list for sub-entries.

```
13884 \newcommand*{\glstrbookindexsubprelocation}[1]{%
13885   \glstrbookindexprelocation{#1}%
13886 }
```

`xparentchildsep` Separator used between top-level parent and child entry.

```
13887 \newcommand{\glstrbookindexparentchildsep}{\nopagebreak}
```

`rentsubchildsep` Separator used between sub-level parent and child entry.

```
13888 \newcommand{\glstrbookindexparentsubchildsep}{\glstrbookindexparentchildsep}
```

`bookindexbetween` Between two top-level entries identified by the labels in the arguments.

```
13889 \newcommand{\glxstrbookindexbetween}[2]{}
```

`indexsubbetween` Between two level 1 entries identified by the labels in the arguments.

```
13890 \newcommand{\glxstrbookindexsubbetween}[2]{}
```

`exsubsubbetween` Between two level 2 entries identified by the labels in the arguments.

```
13891 \newcommand{\glxstrbookindexsubsubbetween}[2]{}
```

`indexatendgroup` At the end of a letter group. The argument is the index of the last top-level entry.

```
13892 \newcommand{\glxstrbookindexatendgroup}[1]{}
```

`exsubatendgroup` At the end of a letter group. The argument is the index of the last level 1 entry.

```
13893 \newcommand{\glxstrbookindexsubatendgroup}[1]{}
```

`subsubatendgroup` At the end of a letter group. The argument is the index of the last level 2 entry.

```
13894 \newcommand{\glxstrbookindexsubsubatendgroup}[1]{}
```

`kindexgroupskip` Group separator.

```
13895 \newcommand{\glxstrbookindexgroupskip}{\ifglsnogroupskip\else\indexspace\fi}
```

Format group title.

`indexformatheader` Group separator.

```
13896 \newcommand*{\glxstrbookindexformatheader}[1]{%
13897 \par{\centering\glstreegroupheaderfmt{#1}\par}%
13898 }
```

`bookindexbookmark` Book mark group heading if supported.

```
13899 \ifdef\pdfbookmark
13900 {%
13901 \newcommand*{\glxstrbookindexbookmark}[2]{%
13902 \ifdefstring{\@@glossarysec}{chapter}%
13903 {\pdfbookmark[1]{#1}{#2}}%
13904 {\pdfbookmark[2]{#1}{#2}}%
13905 }
13906 }
13907 {%
13908 \newcommand*{\glxstrbookindexbookmark}[2]{%
13909 }
```

`kindexcolspread`

```
13910 \newcommand*{\glxstrbookindexcolspread}{}%
```

`indexmulticolenv`

```
13911 \newcommand*{\glxstrbookindexmulticolenv}{multicols}
```

Define the style.

```

13912 \newglossarystyle{bookindex}{%
13913   \setglossarystyle{index}%
13914   \renewenvironment{theglossary}%
13915   {%
13916     \ifdefempty{glxstrbookindexcolspread}
13917     {%
13918       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
13919       {\glxstrbookindexcols}%
13920     }%
13921     {%
13922       \expandafter\begin\expandafter{\glxstrbookindexmulticolseenv}%
13923       {\glxstrbookindexcols}[\glxstrbookindexcolspread]%
13924     }%
13925     \setlength{\parindent}{0pt}%
13926     \setlength{\parskip}{0pt plus 0.3pt}%
13927     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
13928     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
13929     \let\@glxstr@bookindex@between\@gobble
13930     \let\@glxstr@bookindex@subbetween\@gobble
13931     \let\@glxstr@bookindex@subsubbetween\@gobble
13932     \let\@glxstr@bookindex@atendgroup\relax
13933     \let\@glxstr@bookindex@subatendgroup\relax
13934     \let\@glxstr@bookindex@subsubatendgroup\relax
13935     \let\@glxstr@bookindex@groupskip\relax
13936   }%
13937   {%

```

Do end group hooks.

```

13938     \@glxstr@bookindex@subsubatendgroup
13939     \@glxstr@bookindex@subatendgroup
13940     \@glxstr@bookindex@atendgroup

```

End multicol environment.

```

13941     \expandafter\end\expandafter{\glxstrbookindexmulticolseenv}%
13942   }%

```

Use ragged right as columns are likely to be narrow and indexes tend not to be fully justified.

```

13943   \renewcommand*{\glossaryheader}{\raggedright}%

```

Top level entry format.

```

13944   \renewcommand*{\glossentry}[2]{%

```

Do separator.

```

13945     \@glxstr@bookindex@between{##1}%

```

Update separators.

```

13946     \let\@glxstr@bookindex@sep\glxstrbookindexparentchildsep
13947     \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
13948     \let\@glxstr@bookindex@subbetween\@gobble
13949     \let\@glxstr@bookindex@subsubbetween\@gobble
13950     \edef\@glxstr@bookindex@between{%

```



```

13951      \noexpand\glxstrbookindexbetween{##1}%
13952  }%
13953  \edef\@glxstr@bookindex@atendgroup{%
13954      \noexpand\glxstrbookindexatendgroup{##1}%
13955  }%
13956  \let\@glxstr@bookindex@subatendgroup\relax
13957  \let\@glxstr@bookindex@subsubatendgroup\relax

```

Format entry.

```

13958  \glstreeitem
13959      \glstryitem{##1}%
13960      \glstarget{##1}{\glxstrbookindexname{##1}}%
13961  \glxstrbookindexprelocation{##1}##2%
13962  }%
13963  \renewcommand{\subglossentry}[3]{%
13964      \ifcase##1\relax

```

Level 0 (shouldn't happen as that's formatted with \glossentry).

```

13965      \glstreeitem
13966      \or

```

Level 1.

```

13967      \@glxstr@bookindex@sep
13968      \@glxstr@bookindex@subbetween{##2}%
13969      \let\@glxstr@bookindex@sep\relax

```

Update separators.

```

13970      \let\@glxstr@bookindex@subsubbetween\@gobble
13971      \let\@glxstr@bookindex@subsep\glxstrbookindexparentsubchildsep
13972      \edef\@glxstr@bookindex@subbetween{%
13973          \noexpand\glxstrbookindexsubbetween{##2}%
13974      }%
13975      \edef\@glxstr@bookindex@atsubendgroup{%
13976          \noexpand\glxstrbookindexatsubendgroup{##1}%
13977      }%

```

Start sub-item.

```

13978      \glstreesubitem
13979      \glssubentryitem{##2}%
13980      \else

```

All other levels.

```

13981      \@glxstr@bookindex@subsep
13982      \@glxstr@bookindex@subsubbetween{##2}%

```

Update separators.

```

13983      \let\@glxstr@bookindex@subsep\relax
13984      \edef\@glxstr@bookindex@subsubbetween{%
13985          \noexpand\glxstrbookindexsubsubbetween{##2}%
13986      }%
13987      \edef\@glxstr@bookindex@atsubsubendgroup{%
13988          \noexpand\glxstrbookindexatsubsubendgroup{##1}%
13989      }%

```

Start sub-sub-item.

```
13990 \glstreesubsubitem
13991 \fi
```

Format entry.

```
13992 \glstarget{##2}{\glxtrbookindexsubname{##2}}%
13993 \glxtrbookindexsubprelocation{##2}##3%
13994 }%
```

The group skip is moved to the group heading to avoid interfering with the end letter group hooks.

```
13995 \renewcommand*{\glsgroupskip}{}%
```

Group heading format.

```
13996 \renewcommand*{\glsgroupheading}[1]{%
```

Do end group hooks.

```
13997 \@glxtr@bookindex@subsubatendgroup
13998 \@glxtr@bookindex@subatendgroup
13999 \@glxtr@bookindex@atendgroup
14000 \@glxtr@bookindexgroupskip
```

Update separators.

```
14001 \let\@glxtr@bookindexgroupskip\glxtrbookindexgroupskip
14002 \let\@glxtr@bookindex@between\@gobble
14003 \let\@glxtr@bookindex@atendgroup\relax
14004 \let\@glxtr@bookindex@subatendgroup\relax
14005 \let\@glxtr@bookindex@subsubatendgroup\relax
```

Fetch the group title from the label supplied in #1.

```
14006 \glxtrgetgrouptitle{##1}{\thisgrptitle}%
```

Do the PDF bookmark if supported.

```
14007 \glxtrbookindexbookmark{\thisgrptitle}{index.##1}%
```

Format the group title.

```
14008 \glxtrbookindexformatheader{\thisgrptitle}%
14009 \nopagebreak\indexspace\nopagebreak\@afterheading
14010 }%
14011 }
```

Some supplementary commands that may be useful. These store the entry label for the current page. Since the page number is needed in the control sequence, this uses `\glxtrbookindexthepage` instead of `\thepage` in case the page numbering has been set to something that contains formatting commands.

`\glxtrbookindexthepage` The `\@printglossary` sets `\currentglossary` to the current glossary label. This is used as a prefix in case the page number is reset.

```
14012 \newcommand{\glxtrbookindexthepage}{%
14013 \ifdef\currentglossary{\currentglossary.\arabic{page}}{\arabic{page}}%
14014 }
```

bookindexmarkentry Writes entry information to the .aux file. The argument is the entry label.

```
14015 \newcommand*{\glxtrbookindexmarkentry}[1]{%
14016   \protected@write\@auxout
14017   {\let\glxtrbookindexthepage\relax}%
14018   {\string\glxtr@setbookindexmark{\glxtrbookindexthepage}{#1}}%
14019 }
```

etbookindexmark

```
14020 \newcommand*{\glxtr@setbookindexmark}[2]{%
14021   \ifcsundef{glxtr@idxfirstmark@#1}%
14022   {\csgdef{glxtr@idxfirstmark@#1}{#2}}%
14023   {}%
14024   \csgdef{glxtr@idxlastmark@#1}{#2}%
14025 }
```

indexfirstmarkfmt

```
14026 \newcommand*{\glxtrbookindexfirstmarkfmt}[1]{%
14027   \glstryname{#1}%
14028 }
```

indexfirstmark

```
14029 \newcommand*{\glxtrbookindexfirstmark}{%
14030   \letcs{\glxtr@label}{glxtr@idxfirstmark@\glxtrbookindexthepage}%
14031   \ifdef\glxtr@label
14032   {\glxtrbookindexfirstmarkfmt{\glxtr@label}}%
14033   {}%
14034 }
```

indexlastmarkfmt

```
14035 \newcommand*{\glxtrbookindexlastmarkfmt}[1]{%
14036   \glstryname{#1}%
14037 }
```

okindexlastmark

```
14038 \newcommand*{\glxtrbookindexlastmark}{%
14039   \letcs{\glxtr@label}{glxtr@idxlastmark@\glxtrbookindexthepage}%
14040   \ifdef\glxtr@label
14041   {\glxtrbookindexlastmarkfmt{\glxtr@label}}%
14042   {}%
14043 }
```

# 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 . . . . .	66
General: Initial experimental release . . . . .	5	\@Glsymbolplural@: added	
0.2 (2015-11-30)		redefinition . . . . .	66
\Glsfmtshort: new . . . . .	307	\@Glstext@: added redefinition . . . . .	61
\glsfmtshort: new . . . . .	306	\@Glsuseri@: added redefinition . . . . .	66
\Glsfmtshortpl: new . . . . .	307	\@Glsuserii@: added redefinition . . . . .	67
\glsfmtshortpl: new . . . . .	307	\@Glsuseriii@: added redefinition . . . . .	67
short: switched inline full form to short		\@Glsuseriv@: added redefinition . . . . .	67
(long) . . . . .	210	\@Glsuserv@: added redefinition . . . . .	68
0.3 (2015-12-02)		\@Glsuservi@: added redefinition . . . . .	68
\@ACRlong: added redefinition . . . . .	71	\@acrlong: added redefinition . . . . .	70
\@ACRlongpl: added redefinition . . . . .	72	\@acrlongpl: added redefinition . . . . .	71
\@ACRshort: added redefinition . . . . .	69	\@acrshort: added redefinition . . . . .	68
\@ACRshortpl: added redefinition . . . . .	70	\@acrshortpl: added redefinition . . . . .	69
\@Acrlong: added redefinition . . . . .	71	\@gls@field@link: added optional	
\@Acrlongpl: added redefinition . . . . .	72	argument . . . . .	55
\@Acrshort: added redefinition . . . . .	69	\@glsdescplural@: added redefinition . . . . .	65
\@Acrshortpl: added redefinition . . . . .	70	\@glsfirst@: added redefinition . . . . .	62
\@GLSdesc: added redefinition . . . . .	65	\@glsfirstplural@: added redefinition . . . . .	63
\@GLSdescplural@: added redefinition . . . . .	65	\@glsplural@: added redefinition . . . . .	63
\@GLSfirst@: added redefinition . . . . .	62	\@glssymbolplural@: added	
\@GLSfirstplural@: added redefinition . . . . .	64	redefinition . . . . .	66
\@GLSname@: added redefinition . . . . .	64	\@glsxtr@defaultnoglossarywarning:	
\@GLSplural@: added redefinition . . . . .	63	new . . . . .	122
\@GLSsymbol@: added redefinition . . . . .	66	\@glsxtr@field@linkdefs: new . . . . .	60
\@GLSsymbolplural@: added		\@glsxtr@insertdots: new . . . . .	179
redefinition . . . . .	66	\@print@glossary: added redefinition . . . . .	119
\@GLStext@: added redefinition . . . . .	61	\@glsabbrvdefaultfont: renamed from	
\@GLSuseri@: added redefinition . . . . .	67	\abbrvdefaultfont . . . . .	185
\@GLSuserii@: added redefinition . . . . .	67	\@glsaccessdesc: new . . . . .	147
\@GLSuseriii@: added redefinition . . . . .	67	\@glsaccessdescplural: new . . . . .	147
\@GLSuseriv@: added redefinition . . . . .	68	\@glsaccessfirst: new . . . . .	144
\@GLSuserv@: added redefinition . . . . .	68	\@glsaccessfirstplural: new . . . . .	145
\@GLSuservi@: added redefinition . . . . .	68	\@glsaccesslong: new . . . . .	149
\@Glsdesc@: added redefinition . . . . .	65	\@glsaccesslong: new . . . . .	149
\@Glsdescplural@: added redefinition . . . . .	65	\@glsaccessname: new . . . . .	143
\@Glsfirst@: added redefinition . . . . .	62	\@glsaccessplural: new . . . . .	144
\@Glsfirstplural@: added redefinition . . . . .	63	\@glsaccessshort: new . . . . .	148
\@Glsname@: added redefinition . . . . .	64	\@glsaccessshort: new . . . . .	148
\@Glsplural@: added redefinition . . . . .	63	\@glsaccessshortpl: new . . . . .	149

\glsaccesssshortpl: new .....	149	\cGLSpl: new .....	96
\glsaccessssymbol: new .....	146	\cGLSpl@: new .....	96
\glsaccessssymbolplural: new .....	146	\glsxtr@setentrycountunsetattr:	
\glsaccesstext: new .....	143	new .....	91
\glstentryfmt: added check for short ..	54	\cGLS: new .....	95
\glslongpltok: new .....	179	\cGLSformat: new .....	96
\glsshortpltok: new .....	179	\cGLSpl: new .....	96
\glsxtr@newabbreviation: fixed family		\cGLSplformat: new .....	96
name in \setkeys .....	181	\GlossariesExtraWarningNoLine:	
\glsxtrdiscardperiod: added check		new .....	15
for plural .....	176	\glsenableentrycount: new .....	91
\GLSxtrlongpl: new .....	194	\glsfirstabbrvdefaultfont: new ..	185
\Glsxtrlongpl: new .....	194	\glsfirstlongdefaultfont: new ...	185
\glsxtrlongpl: new .....	193	\Glsfmtfirst: new .....	310
\glsxtrNoGlossaryWarning: new ....	20	\glsfmtfirst: new .....	309
\glsxtrpostlinkAddDescOnFirstUse:		\Glsfmtfirstpl: new .....	310
new .....	176	\glsfmtfirstpl: new .....	310
\glsxtrpostlinkAddSymbolOnFirstUse:		\Glsfmtplural: new .....	309
new .....	176	\glsfmtplural: new .....	309
\glsxtrpostlinkendsentence: new ..	176	\Glsfmtshort: changed to use	
\GLSxtrshortpl: new .....	193	\Glsxtrtitleshort .....	307
\Glsxtrshortpl: new .....	192	renamed from \glstentryfmtshort ..	307
\glsxtrshortpl: new .....	192	\glsfmtshort: changed to use	
short-long-desc: fixed name to use		\glsxtrtitleshort .....	306
\glslabeltok .....	205	renamed from \glstentryfmtshort ..	306
long-short-desc: fixed name to use		\Glsfmtshortpl: changed to use	
\glslabeltok .....	203	\Glsxtrtitleshortpl .....	307
0.4 (2015-12-03)		renamed from	
\@glsxtr@doabbreviationsdef: added		\Glstentryfmtshortpl .....	307
redefinition of \acronymtype .....	17	\glsfmtshortpl: changed to use	
\Glsfmtshort: changed to use		\glsxtrtitleshortpl .....	307
\Glsxtrshort .....	307	renamed from	
\glsfmtshort: changed to use		\glstentryfmtshortpl .....	307
\glsxtrshort .....	306	\Glsfmttext: new .....	308
\Glsfmtshortpl: changed to use		\glsfmttext: new .....	308
\glsxtrshortpl .....	307	\glshasattribute: new .....	155
\glsfmtshortpl: changed to use		\glshascategoryattribute: new ...	154
\glsxtrshortpl .....	307	\glsxtremsuffix: new .....	246
\glsxtrifemptyglossary: new .....	26	\GlsXtrEnableEntryCounting: new ..	91
\glsxtrnewnumber: added extra		\glsxtrifcounttrigger: new .....	93
argument .....	158	\glsxtrscfont: new .....	218
\glsxtrnewsymbol: added extra		\glsxtrscsuffix: new .....	218
argument .....	158	\glsxtrsmfont: new .....	232
\MakeAcronymsAbbreviations: set the		\glsxtrsmsuffix: new .....	232
default type to \acronymtype ....	104	short-em: new .....	253
\newterm: fixed name argument .....	157	short-em-desc: new .....	255
0.5 (2015-12-07)		short-em-footnote: new .....	264
\@cGLS: new .....	96	short-em-long: new .....	250
\@cGLS@: new .....	96	short-em-long-desc: new .....	251

short-em-postfootnote: new .....	266	\glxtrheadshortpl: now uses headuc	
short-sc-footnote: new .....	228	attribute .....	298
short-sc-postfootnote: new .....	230	\Glsxtrheadtext: now uses headuc	
short-sm: new .....	236	attribute .....	300
short-sm-desc: new .....	237	\glxtrheadtext: now uses headuc	
short-sm-footnote: new .....	243	attribute .....	300
short-sm-long: new .....	234	short-em-footnote: switch off regular	
short-sm-long-desc: new .....	235	attribute if set .....	264
short-sm-postfootnote: new .....	244	short-long: switch off regular attribute	
long-noshort-em: new .....	257	if set .....	204
long-noshort-em-desc: new .....	260	short-long-desc: switch off regular	
long-noshort-sm: new .....	239	attribute if set .....	205
long-noshort-sm-desc: new .....	241	short-sc-footnote: switch off regular	
long-short-em: new .....	246	attribute if set .....	229
long-short-em-desc: new .....	248	short-sm-footnote: switch off regular	
long-short-sm: new .....	232	attribute if set .....	243
long-short-sm-desc: new .....	233	long-short: switch off regular attribute	
0.5.1 (2015-12-02)		if set .....	202
\Glsaccessstext: new .....	144	long-short-desc: switch off regular	
0.5.1 (2015-12-07)		attribute if set .....	203
\@glxtr@doaccsupp: new .....	20	long-short-sc-desc: switch off regular	
General: removed \ifglxtruseuchead	297	attribute if set .....	219
\Glsaccessdesc: new .....	147	footnote: switch off regular attribute if	
\Glsaccessdescplural: new .....	148	set .....	206
\Glsaccessfirst: new .....	145	postfootnote: switch off regular	
\Glsaccessfirstplural: new .....	145	attribute if set .....	208
\Glsaccessname: new .....	143	0.5.2 (2015-12-08)	
\Glsaccessplural: new .....	144	\@GLSdesc@: added accessibility support	65
\Glsaccesssymbol: new .....	146	\@GLSdescplural@: added accessibility	
\Glsaccesssymbolplural: new .....	146	support .....	65
\glxtrheadfirst: now uses headuc		\@GLSfirst@: added accessibility	
attribute .....	302	support .....	62
\glxtrheadfirst: now uses headuc		\@GLSfirstplural@: added accessibility	
attribute .....	301	support .....	64
\Glsxtrheadfirstplural: now uses		\@GLSname@: added accessibility support	64
headuc attribute .....	302	\@GLSplural@: added accessibility	
\glxtrheadfirstplural: now uses		support .....	63
headuc attribute .....	302	\@GLSsymbol@: added accessibility	
\Glsxtrheadplural: now uses headuc		support .....	66
attribute .....	301	\@GLSsymbolplural@: added	
\glxtrheadplural: now uses headuc		accessibility support .....	66
attribute .....	300	\@GLStext@: added accessibility support	61
\Glsxtrheadshort: now uses headuc		\@GLSdesc@: added accessibility support	65
attribute .....	298	\@GLSdescplural@: added accessibility	
\glxtrheadshort: now uses headuc		support .....	65
attribute .....	297	\@GLSfirst@: added accessibility	
\Glsxtrheadshortpl: now uses headuc		support .....	62
attribute .....	298	\@GLSfirstplural@: added accessibility	
		support .....	63

\@Glsname@: add accessibility support ..	64	\GLSaccesssymbolplural: new ..	147, 152
\@Glsplural@: added accessibility support .....	63	\GLSaccessstext: new .....	144, 151
\@Glsymbol@: added accessibility support .....	66	\glsentryfmt: moved	
\@Glsymbolplural@: added accessibility support .....	66	\glssetabbrvfmt from	
\@Glstext@: added accessibility support	61	\glsxtrabbrvfmt to here .....	54
\@glsdesc@: added accessibility support	64	\GlsXtrEnableInitialTagging: new	172
\@glsdescplural@: added accessibility support .....	65	\glsxtrfieldtitlecase: new .....	159
\@glsfirst@: added accessibility support .....	62	\GlsXtrFormatLocationList: new ...	52
\@glsfirstplural@: added accessibility support .....	63	\glsxtrnewabbrevpresetkeyhook:	
\@glsname@: added accessibility support	64	new .....	183
\@glsplural@: added accessibility support .....	63	\glsxtrtagfont: new .....	174
\@glsymbol@: added accessibility support .....	65	\KV@printgloss@nonumberlist: added	54
\@glsymbolplural@: added accessibility support .....	66	\mfu@checkword@do: added .....	173
\@Glstext@: added accessibility support	61	\setabbreviationstyle: added check	
\@glsxtr@activate@initialtagging:		for post-definition style switch ....	198
new .....	174	0.5.3 (2015-12-09)	
\@glsxtr@do@titlecaps@warn: new ..	174	\@glsxtr@autoindex@at: new .....	169
\@glsxtr@tag: new .....	174	\@glsxtr@autoindex@encap: new ...	170
General: fixed typo in glossaries-accsupp		\@glsxtr@autoindex@esc: new .....	170
and tidied up code to use just one		\@glsxtr@autoindex@level: new ...	170
\@ifpackageloaded .....	143	\@glsxtr@autoindex@setname: new ..	168
removed \glsxtrabbrvfmt .....	195	\@glsxtr@doabbreviationsdef: new ..	17
\glossaryentrynumbers: added .....	52	General: removed	
\Glossentrydesc: added .....	172	\GlsXtrNoGlsWarningNoAutoMakeMain	
\Glossentryname: added .....	163	.....	121
\Glossentrysymbol: added .....	172	\glsdescwidth: added .....	51
\glossentrysymbol: added .....	172	\glspagelistwidth: added .....	51
\GLSaccessdesc: new .....	147, 152	\glsxtrdoautoindexname: new .....	167
\GLSaccessdescplural: new ...	148, 152	\glsxtrpostnamehook: new .....	165
\GLSaccessfirst: new .....	145, 151	\if@glsxtr@format@override: new ..	167
\GLSaccessfirstplural: new ..	145, 151	\ProvidesGlossariesExtraLang: new	313
\GLSaccesslong: new .....	149, 153	\RequireGlossariesExtraLang: new	313
\GLSaccesslongpl: new .....	150, 153	0.5.4 (2015-12-15)	
\Glsaccesslongpl: new .....	150	\@newglossaryentry@defunitcounters:	
\glsaccesslongpl: new .....	150	new .....	97
\GLSaccessname: new .....	143, 150	\@GLSxtr@p@acrlong@: new .....	84
\GLSaccessplural: new .....	144, 151	\@GLSxtr@p@acrlongpl@: new .....	84
\GLSaccessshort: new .....	148, 152	\@GLSxtr@p@acrshort@: new .....	84
\GLSaccessshortpl: new .....	149, 153	\@GLSxtr@p@acrshortpl@: new .....	84
\GLSaccesssymbol: new .....	146, 151	\@GLSxtr@p@long@: new .....	83
		\@GLSxtr@p@longpl@: new .....	84
		\@GLSxtr@p@plural@: new .....	82
		\@GLSxtr@p@short@: new .....	83
		\@GLSxtr@p@shortpl@: new .....	83
		\@GLSxtr@p@text@: new .....	82
		\@GlsXtrEnableOnTheFly: new .....	47
		\@Glsxtr: new .....	48
		\@Glsxtr@p@acrlong@: new .....	84
		\@Glsxtr@p@acrlongpl@: new .....	84



\@Glsxtr@p@acrshort@: new	84	\glsenableentryunitcount: new	99
\@Glsxtr@p@acrshortpl@: new	84	\glshasattribute: added check for	
\@Glsxtr@p@long@: new	83	entry's existence	155
\@Glsxtr@p@longpl@: new	83	\glusifattribute: added check for	
\@Glsxtr@p@plural@: new	82	entry's existence	155
\@Glsxtr@p@short@: new	82	\glspostlinkhook: added existence	
\@Glsxtr@p@shortpl@: new	83	check	175
\@Glsxtr@p@text@: new	82	\Glsxtr: new	48
\@Glsxtrpl: new	49	\glxtr: new	48
\@alt@glshyp@opt: new	78	\glxtrcat: new	48
\@glscalt@hyp@opt: new	78	\glxtrdowrglossaryhook: new	78
\@glscalt@hyp@opt@char: new	78	\GlsXtrEnableEntryUnitCounting:	
\@glscalt@hyp@opt@keys: new	78	new	102
\@glsc@increment@currunitcount:		\GlsXtrEnableOnTheFly: new	47
new	98	\Glsxtrpl: new	49
\@glsc@local@increment@currunitcount:		\glxtrpl: new	48
new	98	\glxtrpostlocalreset: new	90
\@glsc@setdefault@glslink@opts:		\glxtrpostlocalunset: new	90
new	75	\glxtrpostreset: new	90
\@glsxtr: new	48	\glxtrpostunset: new	90
\@glsxtr@addunitcounter: new	97	\glxtrprotectlinks: new	81
\@glsxtr@currunitcount: new	99	\GlsXtrSetAltModifier: new	78
\@glsxtr@ifunitcounter: new	97	\GlsXtrSetDefaultGlsOpts: new	77
\@glsxtr@p@acrlong@: new	84	\glxtrstarflywarn: new	47
\@glsxtr@p@acrlongpl@: new	84	\GlsXtrWarning: new	49
\@glsxtr@p@acrshort@: new	84	\MakeAcronymsAbbreviations: now	
\@glsxtr@p@acrshortpl@: new	84	disables \setacronymstyle	104
\@glsxtr@p@long@: new	83	1.0 (2016-01-24)	
\@glsxtr@p@longpl@: new	83	\@glsxtr@autoindexcrossrefs: new	15
\@glsxtr@p@plural@: new	82	\@glsxtr@idx@displaynumberlist:	
\@glsxtr@p@short@: new	82	new	113
\@glsxtr@p@shortpl@: new	83	\@glsxtr@idx@entrynumberlist: new	114
\@glsxtr@p@text@: new	82	\@glsxtr@noidx@displaynumberlist:	
\@glsxtr@prevunitcount: new	99	new	113
\@glsxtr@setentryunitcountunsetattr:		\@glsxtr@noidx@entrynumberlist:	
new	103	new	114
\@glsxtr@unitcountlist: new	97	\@glsxtr@noidx@numberlistloop:	
\@glsxtrpl: new	48	new	113
\@newglossaryentryposthook: added		\@glsxtr@reg@glosslist: new	106
empty see value if not set and added		\makeglossaries: new	106
'see' to field key map	39	1.01 (2016-02-02)	
\@sGlsXtrEnableOnTheFly: new	47	\glxtrdiscardperiod: added check	
\cGlsformat: added	96	for first use	176
\cGlsformat: added	96	short-desc: fixed typo in	
\cGlsplformat: added	97	\glxtrinlinfullformat and	
\cGlsplformat: added	97	added missing second argument	212
\glstdisablehyper: added	80	1.02 (2016-04-25)	
\glstdohyperlink: added	79	\@glsxtr@current@style: new	50
\glstdonohyperlink: added	81	\Glsfmtfull: new	312

\glsfmtfull: new .....	312	\@GLSdescplural@: set abbreviation and regular format .....	65
\Glsfmtfullpl: new .....	312	\@GLSfirst@: set abbreviation format ..	62
\glsfmtfullpl: new .....	312	\@GLSfirstplural@: set abbreviation and regular format .....	64
\Glsfmtlong: new .....	311	\@GLSname@: set abbreviation and regular format .....	64
\glsfmtlong: new .....	310	\@GLSplural@: set abbreviation and regular format .....	63
\Glsfmtlongpl: new .....	311	\@GLSsymbol@: set regular format .....	66
\glsfmtlongpl: new .....	311	\@GLSsymbolplural@: set regular format	66
\Glsxtrheadfull: new .....	305	\@GLStext@: set abbreviation and regular format .....	61
\glsxtrheadfull: new .....	305	\@GLSuseri@: set regular format .....	67
\Glsxtrheadfullpl: new .....	306	\@GLSuserii@: set regular format .....	67
\glsxtrheadfullpl: new .....	305	\@GLSuseriii@: set regular format .....	67
\Glsxtrheadlong: new .....	304	\@GLSuseriv@: set regular format .....	68
\glsxtrheadlong: new .....	303	\@GLSuseriv@: set regular format .....	68
\Glsxtrheadlongpl: new .....	304	\@GLSdesc@: set abbreviation and regular format .....	65
\glsxtrheadlongpl: new .....	303	\@GLSdescplural@: set abbreviation and regular format .....	65
\Glsxtrtitlefull: new .....	306	\@GLSfirst@: set abbreviation and regular format .....	62
\glsxtrtitlefull: new .....	305	\@GLSfirstplural@: set abbreviation and regular format .....	63
\Glsxtrtitlefullpl: new .....	306	\@GLSname@: set abbreviation and regular format .....	64
\glsxtrtitlefullpl: new .....	305	\@GLSplural@: set abbreviation and regular format .....	63
\Glsxtrtitlelong: new .....	304	\@GLSsymbol@: set regular format .....	66
\glsxtrtitlelong: new .....	303	\@GLSsymbolplural@: set regular format	66
\Glsxtrtitlelongpl: new .....	304	\@GLStext@: set abbreviation and regular format .....	61
\glsxtrtitlelongpl: new .....	304	\@GLSuseri@: set regular format .....	66
\ifglsxtrinsertinside: new .....	201	\@GLSuserii@: set regular format .....	67
postfootnote: added redef of		\@GLSuseriii@: set regular format .....	67
\glsxtrsetupfulldefs .....	208	\@GLSuseriv@: set regular format .....	67
stylemods: new .....	21	\@GLSuseriv@: set regular format .....	68
1.03 (2016-04-27)		\@GLSuseriv@: set regular format .....	68
\@GLSfirstplural@: bug fix: misspelt cs name .....	64	\@GLSdesc@: set abbreviation and regular format .....	65
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	63	\@GLSdescplural@: set abbreviation and regular format .....	65
\@GLSfirstplural@: bug fix: misspelt cs name .....	63	\@GLSfirst@: set abbreviation and regular format .....	62
\@GLSplural@: fixed bug \@GLSplural@ should be redefined not \@GLSplural	63	\@GLSfirstplural@: set abbreviation and regular format .....	63
\@glsplural@: fixed bug \@glsplural@ should be redefined not \@glsplural	63	\@GLSname@: set abbreviation and regular format .....	64
\glsxtrtitlelongpl: bug fix: changed \glsxtrlong to \glsxtrlongpl ..	304	\@GLSplural@: set abbreviation and regular format .....	63
\glsxtrtitleshortpl: bug fix: changed \glsxtrshort to \glsxtrshortpl	298	\@GLSsymbol@: set regular format .....	66
1.04 (2015-04-30)		\@GLSsymbolplural@: set regular format	66
short-em-footnote: renamed from “footnote-em” .....	264	\@GLStext@: set abbreviation and regular format .....	61
1.04 (2016-05-02)		\@GLSuseri@: set regular format .....	66
\@glsxtrpostloctag: new .....	53	\@GLSuserii@: set regular format .....	67
\@GLSdesc@: set abbreviation and regular format .....	65	\@GLSuseriii@: set regular format .....	67
		\@GLSuseriv@: set regular format .....	67
		\@GLSuseriv@: set regular format .....	68
		\@GLSdesc@: set abbreviation and regular format .....	64
		\@GLSdescplural@: set abbreviation and regular format .....	65
		\@GLSfirst@: set abbreviation and regular format .....	62

\@glsfirstplural@: set abbreviation and regular format .....	63	\glxtruserparen: new .....	268
\@glsname@: set abbreviation and regular format .....	64	\glxtrusersuffix: new .....	268
\@glsplural@: set abbreviation and regular format .....	63	\GlsXtrWarnDeprecatedAbbrStyle: new .....	200
\@glssymbol@: set regular format .....	65	short-em-long-em: new .....	252
\@glssymbolplural@: set regular format .....	66	short-em-long-em-desc: new .....	253
\@glstext@: set abbreviation and regular format .....	61	short-em-nolong: new .....	255
\@glxtr@deprecated@abbrstyle: new .....	200	short-em-nolong-desc: new .....	256
\@glxtr@do@style: new .....	21	short-em-postfootnote: renamed from “postfootnote-em” .....	266
\@glxtr@doloctag: new .....	54	short-footnote: new .....	207
\@glxtr@idx@entrynumberlist: switched from \let to \newcommand .....	114	short-long-user: new .....	275
\@glxtr@pagetag: new .....	53	short-long-user-desc: new .....	276
\@glxtr@pagetag: new .....	53	short-nolong: new .....	211
\@glxtr@preloctag: new .....	54	short-nolong-desc: new .....	213
\@glxtr@postloctag: new .....	53	short-postfootnote: new .....	209
\@glxtr@preloctag: new .....	53	short-sc-footnote: renamed from “footnote-sc” .....	228
\glossentrydesc: added glossdescfont attribute check .....	159	short-sc-nolong: new .....	223
\Glossentryname: added glossnamefont attribute check .....	163	short-sc-nolong-desc: new .....	224
\glossentryname: added glossnamefont attribute check .....	161	short-sc-postfootnote: renamed from “postfootnote-sc” .....	230
moved post name hook inside condition .....	163	short-sm-footnote: renamed from “footnote-sm” .....	243
\glsabbrvmfont: new .....	246	short-sm-nolong: new .....	237
\glsabbrvuserfont: new .....	268	short-sm-nolong-desc: new .....	238
\glsfirstabbrvmfont: new .....	246	short-sm-postfootnote: renamed from “postfootnote-sm” .....	244
\glsfirstabbrvuserfont: new .....	268	\letabbreviationstyle: new .....	200
\glsfirstlongemfont: new .....	246	\newabbreviationstyle: bug fix: corrected test for existence .....	199
\glsfirstlonguserfont: new .....	268	long-em-noshort-em: new .....	258
\glsifnotregularcategory: new ...	156	long-em-noshort-em-desc: new ....	262
\glslongdefaultfont: new .....	185	long-em-short-em: new .....	248
\glslongemfont: new .....	246	long-em-short-em-desc: new .....	249
\glslongfont: new .....	185	long-noshort: new .....	217
\glslonguserfont: new .....	268	long-noshort-desc: new .....	216
\glxtrassignfieldfont: new .....	61	long-noshort-em: renamed from “long-em” .....	257
\GlsXtrEnablePreLocationTag: new .	52	long-noshort-em-desc: renamed from “long-desc-em” .....	260
\glxtrfirstscfont: new .....	218	long-noshort-sc: renamed from “long-sc” .....	225
\glxtrfirstsmfont: new .....	232	long-noshort-sc-desc: renamed from “long-desc-sc” .....	227
\glxtrlongshortdescsort: new ...	202	long-noshort-sm: renamed from “long-sm” .....	239
\glxtrpostnamehook: added category check .....	165	long-noshort-sm-desc: renamed from \long-desc-sm .....	241
\glxtrregularfont: new .....	54		
\glxtruserfield: new .....	268		

long-short-user: new .....	269	docdef option changed to choice .....	14
long-short-user-desc: new .....	274	\glxtr@usesee: new .....	40
\renewabbreviationstyle: new ....	199	\glxtrusesee: new .....	39
style: new .....	22	\glxtruseseeformat: new .....	40
1.05 (2016-06-10)		\if@glxtrdocdefrestricted: new ..	14
\eglssetwidest: new .....	364	1.07 (2016-08-15)	
\glsFindWidestAnyName: new .....	367	\@@glxtrp: new .....	85
\glsFindWidestAnyNameLocation:		\@GLSfirst@: added check for	
new .....	372	nohyperfirst attribute .....	62
\glsFindWidestAnyNameSymbol: new	370	\@GLSfirstplural@: added check for	
\glsFindWidestAnyNameSymbolLocation:		nohyperfirst attribute .....	64
new .....	371	\@GLSxtrp: new .....	86
\glsFindWidestLevelTwo: new ....	368	\@Glsfirst@: added check for	
\glsFindWidestUsedAnyName: new ..	367	nohyperfirst attribute .....	62
\glsFindWidestUsedAnyNameLocation:		\@Glsfirstplural@: added check for	
new .....	372	nohyperfirst attribute .....	63
\glsFindWidestUsedAnyNameSymbol:		\@Glsxtrp: new .....	85
new .....	369	\@gls@preglossaryhook: added	
\glsFindWidestUsedAnyNameSymbolLocation:		\glossxtrsetpopts .....	174
new .....	370	\@glsfirst@: added check for	
\glsFindWidestUsedLevelTwo: new .	367	nohyperfirst attribute .....	62
\glsFindWidestUsedTopLevelName:		\@glsfirstplural@: added check for	
new .....	366	nohyperfirst attribute .....	63
\glsfirstlongfootnotefont: new ..	205	\@glxtrinmark: new .....	295
\glsgetwidestname: new .....	366	\@glxtrnotinmark: new .....	295
\glsgetwidestsubname: new .....	366	\@glxtrp: new .....	85
\glslongfootnotefont: new .....	205	\@glxtrp@opt: new .....	84
\glxtrAltTreeIndent: new .....	364	\glossxtrsetpopts: new .....	85
\glxtralttreeInit: new .....	364	\glsp: new .....	87
\glxtrAltTreePar: new .....	364	\glsp: new .....	87
\glxtrAltTreeSetHangIndent: new	373	\glxtr@entry@p: new .....	86
\glxtrAltTreeSetSubHangIndent:		\glxtrabbrvfootnote: new .....	206
new .....	374	\glxtrchecknohyperfirst: new ....	62
\glxtralttreeSubSymbolDescLocation:		\glxtrfieldtitlecasecs: new ....	159
new .....	364	\glxtrifinmark: new .....	295
\glxtralttreeSymbolDescLocation:		\GLSxtrp: new .....	88
new .....	363	\Glsxtrp: new .....	87
\glxtrComputeTreeIndent: new ...	373	\glxtrp: new .....	86
\glxtrComputeTreeSubIndent: new	373	\glxtrsetpopts: new .....	85
\glxtrtreetopindent: new .....	364	short-long-desc: added text key ....	205
short-em-long: fixed incorrect font used		fixed misspelling of \glsabbrvfont in	
by long form .....	250	plural key .....	205
\xglsetwidest: new .....	365	long-short-desc: added missing text	
1.06 (2016-06-18)		key .....	203
\@glsdoifexistsorwarn: new .....	14	fixed misspelling of \glsabbrvfont .	203
\@glxtr@docdefval: new .....	14	footnote: changed first forms to use	
\@glxtr@usesee: new .....	40	\glsfirstlongfootnotefont ...	206
General: disabled docdef key at the start		postfootnote: removed \footnote	
of the document .....	25	from first keys .....	208

switched from \glsfirstlongfont to \glsfirstlongfootnotefont ...	209	1.10 (2016-12-17)	\@GLSp1@: fixed bug caused by typo in command name .....	56
\RestoreAcronyms: modified \@gls@link@checkfirsthyper to set \glsxtrifwasfirstuse .....	105	1.11 (2017-01-19)	\@glsxtr@do@redef@for@gl@sentries: new .....	6
1.08 (2016-12-13)			\@glsxtr@noidx@do: new .....	132
\@glsxtr@record: new .....	8		\@glsxtr@redef@for@gl@sentries: new .	6
\@GLS@: added \@glsxtr@record .....	56		\@glsxtr@shortcutsval: new .....	19
\@GLSp1@: added \@glsxtr@record ...	56		\@glsxtr@unsrt@getgroup@title: new	131
\@Gls@: added \@glsxtr@record .....	56		\@print@noidx@glossary: added redefinition .....	116
\@Glspl@: added \@glsxtr@record ...	56		\glsxtr@addloc@listfield: added group key .....	12
\@gls@: added \@glsxtr@record .....	55		added location key .....	12
\@gls@@link@: added \@glsxtr@record .....	57		\glsxtr@fields: new .....	124
\@gls@field@link: added \@glsxtr@record .....	55		\glsxtr@linkprefix: new .....	124
\@gls@saveentrycounter: new .....	25		\glsxtr@org@new@ignored@glossary: new .....	34
\@glsdisp: added \@glsxtr@record ..	56		\glsxtr@s@new@ignored@glossary: new	35
\@glspl@: added \@glsxtr@record ...	55		\glsxtr@shortcutsval: new .....	124
\@glsxtr@dorecord: new .....	10		\glsxtr@texencoding: new .....	124
\@glsxtr@err@undefaction: new .....	6		\glsxtr@writefields: new .....	124
\@glsxtr@record: new .....	7		\GlsXtrLoadResources: new .....	124
\@glsxtr@warn@onexists@do: new ...	6		\glsxtr@resourcefile: changed extension to .glstex .....	123
\@glsxtr@warn@undefaction: new ....	6		\new@ignored@glossary: added starred version .....	34
\@print@unsrt@glossary: new .....	129	1.12 (2017-02-03)		
General: added record package option ...	12		\@glsxtr@recordcounter: new .....	11
\glsadd: added \@glsxtr@record ....	60		\@gls@preglossaryhook: check for definition .....	174
\glsdoifexists: now defines \glslabel .....	38		\@glsxtr@counter@recordhook: new .	126
\glsxtr@do@wrglossary: new .....	25		\@glsxtr@display@loc: new .....	117
\glsxtr@addloc@listfield: new .....	11		\@glsxtr@docounter@record: new ...	126
\glsxtr@indexonly@saveentrycounter: new .....	11		\@glsxtr@longnew@glossaryentry: new .....	34
\glsxtr@record: new .....	126		\@glsxtr@noop@recordcounter: new .	11
\glsxtr@resource: new .....	124		\@glsxtr@op@recordcounter: new ...	11
\glsxtr@saveentrycounter: new ....	25		\@glsxtr@provide@storagekey: new .	26
\glsxtr@setup@record: new .....	11		\@glsxtr@s@longnew@glossaryentry: new .....	33
\glsxtr@assignfieldfont: added check for existence .....	61		\@glsxtrentryfmt: new .....	28
\glsxtr@resourcefile: new .....	123		\@glsxtr@index@aliased: new .....	76
\printunsrt@glossaries: new .....	129		\@glsxtr@set@alias@noindex: new ....	76
\printunsrt@glossary: new .....	128		\@new@glossaryentry@posthook: added check for alias key .....	43
1.09 (2016-12-16)			\@no@glsxtr@index@aliased: new .....	76
\@glsxtr@gettype: new .....	112		\@printunsrt@glossary: new .....	128
\@glsxtr@mixed@assign@sortkey: new .....	112			
\@print@glossary: redefined to save options .....	112			
\glsxtr@make@glossaries: new .....	112			

General: added target key to printgloss		\GlsXtrLoadResources: removed	
family .....	112	restriction on only one per document	124
\apptoglossarypreamble: new .....	32	\glxtrlocrangefmt: new .....	118
\csGlsXtrLetField: new .....	31	\glxtrpostlongdescription: new ..	34
\eGlsXtrSetField: new .....	32	\glxtrprovidestoragekey: new ....	26
\gGlsXtrSetField: new .....	32	\GlsXtrRecordCounter: new .....	126
\glsdohyperlink: added check for alias		\glxtrresourcecount: new .....	124
field .....	79	\glxtrresourcefile: added catcode	
\glsnoidxdisplayloc: added		change for @ .....	123
redefinition .....	117	\glxtrsetaliasnoindex: new .....	76
\glissettoctitle: added patch .....	35	\GlsXtrSetField: new .....	31
\glxtr@counterrecord: new .....	126	\glxtrsetfieldifexists: new ....	31
\glxtr@langtag: new .....	124	\glxtrunsrtdo: new .....	131
\glxtr@newabbreviation: new ....	181	\Glsxtrusefield: new .....	31
\glxtr@org@newignoredglossary:		\glxtrusefield: new .....	31
Added check for existence .....	34	short-postlong-user: new .....	272
\glxtr@pluralsuffixes: new .....	124	short-postlong-user-desc: new ...	274
\glxtr@provideignoredglossary:		\longnewglossaryentry: added starred	
new .....	36	version .....	33
\glxtr@s@newignoredglossary:		long-postshort-user: new .....	270
Added check for existence .....	35	long-postshort-user-desc: new ...	271
\glxtr@s@provideignoredglossary:		postdot: new .....	15
new .....	37	\pretoglossarypreamble: new .....	33
\glxtrabbrvpluralsuffix: new ...	185	\print@noop@unsrtglossaryunit:	
\glxtralias: new .....	43	new .....	131
\glxtrcopytoglossary: new .....	37	\print@op@unsrtglossaryunit: new	131
\glxtrdeffield: new .....	31	\printunsrtglossary: added starred	
\glxtrdisplayendloc: new .....	117	form .....	128
\glxtrdisplayendlochook: new ...	118	\printunsrtglossaryhandler: new .	130
\glxtrdisplaysingleloc: new ....	117	\printunsrtglossaryunit: new ....	11
\glxtrdisplaystartloc: new ....	117	\printunsrtglossaryunitsetup: new	131
\glxtrdeffield: new .....	31	\provideignoredglossary: new ....	36
\glxtrentryfmt: new .....	28	\s@glxtr@provide@storagekey: new	27
\glxtrfielddolistloop: new .....	29	\s@printunsrtglossary: new .....	128
\glxtrfieldforlistloop: new ....	29	\xGlsXtrSetField: new .....	32
\glxtrfieldifinlist: new .....	30	1.13 (2017-02-07)	
\glxtrfieldlistadd: new .....	29	\@glsdisp: removed	
\glxtrfieldlistead: new .....	29	\@glxtr@org@glsdisp .....	56
\glxtrfieldlistgadd: new .....	29	\glxtrsetaliasnoindex: switched to	
\glxtrfieldlistxadd: new .....	29	\providecommand .....	76
\glxtrfieldxifinlist: new .....	30	1.14 (2017-04-18)	
\glxtrfmt: new .....	27	\@gls@link: added redefinition .....	58
\GlsXtrFmtDefaultOptions: new ....	27	\@gls@noidx@getgrouptitle: new ..	114
\GlsXtrFmtField: new .....	27	\@gls@removespaces: new .....	118
\glxtrifkeydefined: new .....	26	\@glxtr@do@automake@err: new ...	126
\glxtrindexaliased: new .....	76	\@glxtr@org@gloautosee: new ....	24
\GlsXtrLetField: new .....	31	\@glxtr@record: added third arg ....	7
\GlsXtrLetFieldToField: new .....	32	\@glxtr@recordsee: new .....	11

General: added \glsadd option		1.16 (2017-06-15)	
theHvalue .....	60	\@glo@autosee: added redefinition ....	24
added \glsadd option thevalue ....	60	\@gls@noidx@getgrouptitle: fixed	
\glsdisablehyper: added redefinition .	80	bug .....	114
\glsenableentrycount: fixed		\@glsxtr@addunusedxrefs: added	
assignment of \@cGls@ .....	92	check for seealso field .....	44
\glsenableentryunitcount: fixed		\@glsxtr@checkgroup: use \csuse	
assignment of \@cGls@ .....	101	instead of \csname .....	132
\glsnavigation: new .....	116	\@glsxtr@dorecordnodefer: new ....	10
\glsxtr@org@getgrouptitle: new ..	115	\@print@unsrt@glossary: corrected	
\glsxtr@recordsee: new .....	7	misspelt command .....	129
\glsxtr@writefields: added check for		\@printunsrt@glossary@handler:	
automake .....	125	new .....	130
\glsxtrdisplayendloc: added check		General: added check for	
for empty format .....	117	\@gls@setupsort@none .....	13
\glsxtrgetgrouptitle: new .....	115	\@gls@checkseeallowed: added	
\glsxtrinitwrgloss: new .....	57	redefinition .....	24
\glsxtrlocationhyperlink: new ...	118	\@glsxtr@writefields: added	
\glsxtrsetgrouptitle: new .....	115	\providecommand lines .....	125
\glsxtrsupphypernumber: new ....	118	\@glsxtrautoindex: new .....	168
\ifglsxtrwrglossbefore: new .....	57	\@glsxtrautoindexassignsort: new .	168
1.15 (2017-05-10)		\@glsxtrautoindexentry: new .....	168
\@glsxtr@dorecord: corrected		\@glsxtrindexseealso: new .....	41
premature expansion of \@glslocref	10	\@glsxtrseealsolabels: new .....	43
short-em-long-em: fixed spelling of		\@glsxtrseelist: new .....	40
\glsabbrvfont .....	252	\@glsxtruseseealso: new .....	40
short-long: fixed spelling of		\@glsxtruseseealsoformat: new ....	40
\glsabbrvfont .....	204	\@seealso: new .....	41
short-long-user: fixed spelling of		autoseeindex: new .....	15
\glsabbrvfont .....	275	1.17 (2017-08-09)	
short-postlong-user: fixed spelling of		\@glsxtr@mark@wordseps: new ....	180
\glsabbrvfont .....	272	\@glsxtr@markwordseps: new .....	180
short-postlong-user-desc: fixed		\@glsxtr@noidx@displaynumberlist:	
spelling of \glsabbrvfont .....	274	replace hard-coded ?? with	
long-em-short-em: fixed spelling of		\glsxtrundeftag .....	113
\glsabbrvfont .....	248	\@glsxtr@noidx@entrynumberlist:	
long-postshort-user: fixed spelling of		replace hard-coded ?? with	
\glsabbrvfont .....	270	\glsxtrundeftag .....	114
long-postshort-user-desc: fixed		\@glsxtr@noidx@numberlistloop:	
spelling of \glsabbrvfont .....	272	replace hard-coded ?? with	
long-short: fixed spelling of		\glsxtrundeftag .....	114
\glsabbrvfont .....	201	\@glsxtrifhyphenstart: new .....	277
long-short-user: fixed spelling of		General: removed some inconsistencies	
\glsabbrvfont .....	269	in the abbreviation styles .....	201
footnote: fixed spelling of		\glsabbrvhyphenfont: new .....	278
\glsabbrvfont .....	206	\glsabbrvonlyfont: new .....	291
postfootnote: fixed spelling of		\glsabbrvscfont: new .....	218
\glsabbrvfont .....	208	\glsabbrvsmfont: new .....	232

\glsabbrvuserfont: initialised to default font .....	268	short-long-user-desc: corrected first forms .....	276
\glsfirstabbrvhyphenfont: new ...	278	short-nolong-desc-noreg: new ...	213
\glsfirstabbrvonlyfont: new .....	291	short-nolong-noreg: new .....	211
\glsfirstabbrvscfont: new .....	218	long-em-noshort-em-desc-noreg: new .....	264
\glsfirstabbrvsmfont: new .....	232	long-em-noshort-em-noreg: new ...	260
\glsfirstlonghyphenfont: new ...	278	long-hyphen-noshort-desc-noreg: new .....	280
\glsfirstlongonlyfont: new .....	291	long-hyphen-postshort-hyphen: new	283
\glslonghyphenfont: new .....	278	long-hyphen-postshort-hyphen-desc: new .....	285
\glslongonlyfont: new .....	291	long-hyphen-short-hyphen: new ...	278
\glslonguserfont: initialised to default font .....	268	long-hyphen-short-hyphen-desc: new .....	279
\glsxtr@newabbreviation: added \glsxtrorgshort and \glsxtrorglong .....	181	long-noshort-desc-noreg: new ...	216
\GlsXtrDefineAcShortcuts: new ....	18	long-noshort-noreg: new .....	217
\glsxtrgenabbrvfmt: added check for \ifglsxtrininsertinside .....	195	long-only-short-only: new .....	291
\glsxtrhyphensuffix: new .....	278	long-only-short-only-desc: new ..	293
\glsxtrifhyphenstart: new .....	277	long-short-user-desc: corrected first forms .....	274
\glsxtrlonghyphen: new .....	282	1.18 (2017-08-10)	
\glsxtrlonghyphennoshort: new ...	280	stylemods: changed default value to "default" .....	21
\glsxtrlonghyphenshort: new ....	277	1.19 (2017-09-09)	
\glsxtrlongshortdescname: new ...	203	\@glsxtr@defaultnumberformat: new .	7
\glsxtronlydescname: new .....	293	\@glsxtr@dorecord: Use \@glsrecordlocref instead of \@glslocref .....	10
\glsxtronlydescsort: new .....	293	\@glsxtr@dorecordnodefer: Use \theglentrycounter for the location rather than \@glslocref ..	10
\glsxtronlysuffix: new .....	291	\@glsxtr@record@setting: new .....	12
\glsxtrparen: new .....	183	\@glsxtr@record@setting@alsoindex: new .....	12
\glsxtrposthyphenlong: new .....	288	\@glsxtr@record@setting@alsoindex: new .....	30
\glsxtrposthyphenshort: new .....	282	General: added \glslink option theHvalue .....	57
\glsxtrposthyphensubsequent: new	283	added \glslink option thevalue ...	57
\glsxtrshortdescname: new .....	211	\glsxtr@writefields: removed double-quotes around \jobname ..	125
\glsxtrshorthyphen: new .....	288	\glsxtrdoautoindexname: changed format test .....	168
\glsxtrshorthyphenlong: new ....	286	\glsxtrhyperlink: new .....	80
\glsxtrshortlongdescname: new ...	205	\glsxtrifhasfield: new .....	30
\glsxtrshortlongdescsort: new ...	205	\GlsXtrSetDefaultNumberFormat: new .....	7
\Glsxtrsubsequentfmt: new .....	197	\s@glsxtrifhasfield: new .....	30
\glsxtrsubsequentfmt: new .....	197		
\Glsxtrsubsequentplfmt: new ....	198		
\glsxtrsubsequentplfmt: new ....	197		
\glsxtrword: new .....	180		
\glsxtrwordsep: new .....	180		
short-hyphen-long-hyphen: new ...	286		
short-hyphen-long-hyphen-desc: new .....	287		
short-hyphen-postlong-hyphen: new	288		
short-hyphen-postlong-hyphen-desc: new .....	290		





headings .....	378	\glxtrbookindexbetween: new ....	383
redefined		\glxtrbookindexbookmark: new ...	383
mcoltreenonamehypergroup to		\glxtrbookindexcols: new .....	382
discourage breaks after group		\glxtrbookindexcolspread: new ..	383
headings .....	378	\glxtrbookindexfirstmark: new ..	387
redefined mcoltreenonamespannav to		\glxtrbookindexfirstmarkfmt: new	387
discourage breaks after group		\glxtrbookindexformatheader: new	383
headings .....	379	\glxtrbookindexgroupskip: new ..	383
redefined mcoltreepspannav to		\glxtrbookindexlastmark: new ...	387
discourage breaks after group		\glxtrbookindexlastmarkfmt: new	387
headings .....	377	\glxtrbookindexmarkentry: new ..	387
redefined treegroup to discourage		\glxtrbookindexname: new .....	382
breaks after group headings .....	361	\glxtrbookindexparentchildsep:	
redefined treehypergroup to		new .....	382
discourage breaks after group		\glxtrbookindexparentschildsep:	
headings .....	362	new .....	382
redefined treenonamegroup to		\glxtrbookindexprelocation: new	382
discourage breaks after group		\glxtrbookindexsubatendgroup:	
headings .....	363	new .....	383
redefined treenonamehypergroup to		\glxtrbookindexsubbetween: new ..	383
discourage breaks after group		\glxtrbookindexsubname: new ....	382
headings .....	363	\glxtrbookindexsubprelocation:	
debug: new .....	22	new .....	382
\gglsssetwidest: new .....	364	\glxtrbookindexsubsubatendgroup:	
\glsdisablehyper: added check for		new .....	383
existence .....	80	\glxtrbookindexsubsubbetween:	
changed to use \def rather than \let .	80	new .....	383
\glsenablehyper: changed to use \def		\glxtrbookindexthepage: new ....	386
rather than \let .....	80	\glxtrdetoklocation: new .....	136
\Glsfmtname: new .....	308	\glxtrenablerecordcount: new ...	136
\glsfmtname: new .....	308	\glxtrglossentry: new .....	127
\glshex: new .....	314	\glxtrgroupfield: new .....	132
\glslstchildpostlocation: new ..	349	\Glsxtrheadname: new .....	299
\glslstchildprelocation: new ...	349	\glxtrheadname: new .....	299
\glslstprelocation: new .....	349	\GlsXtrIfFieldEqStr: new .....	32
\glsnahyperlink: patched .....	78	\glxtriflabelinlist: new .....	130
\glssseeitemformat: new .....	40	\glxtrifrecordtrigger: new ....	137
\glsshows target: new .....	23	\glxtrindexseealso: added check	
\glstreechildprelocation: new ...	359	that the entry exists .....	41
\glstreeprelocation: new .....	359	\glxtrtrinityhyperoutside: new ....	58
\glstriggerrecordformat: new ....	138	\GlsXtrLocationRecordCount: new ..	136
\glseuseabbrvfont: new .....	195	\glxtrnewgls: new .....	133, 134
\glseuselongfont: new .....	195	\glxtrnewGLSlike: new .....	135
\glxtr@do@alsoindex@wrglossary:		\glxtrnewglslike: new .....	135
new .....	8	\glxtrnewrgls: new .....	135
\glxtr@org@@@do@wrglossary: new ..	25	\glxtrnewrGLSlike: new .....	135
\glxtr@org@dohyperlink: new ....	78	\glxtrnewrglslike: new .....	135
\glxtr@setbookindexmark: new ...	387	\glxtrprelocation: new ....	348, 382
\glxtrbookindexatendgroup: new ..	383	\GlsXtrRecordCount: new .....	135

\glxtrrecordtriggervalue: new ..	136	\glseeitemformat: switched check	
\glxtrresourcefile: now disables		from regular to short .....	40
record key .....	123	\glxtr@setaccessdisplay: new ...	165
\glxtrresourceinit: new .....	124	\glxtr@writefields: provide	
\GlsXtrSetRecordCountAttribute:		\glxtr@record in aux file .....	125
new .....	136	\glxtractivatenopost: new .....	111
\glxtrtitlename: new .....	299	\glxtrbookindexprelocation:	
\glxtrtitleorpdforheading: new .	295	removed check for no post dot ....	382
\GlsXtrTotalRecordCount: new ....	135	\glxtrglossentryother: new .....	127
\glxtrwrglossmark: new .....	22	\glxtrnopostpunc: new .....	111
short-em: new .....	254	1.23 (2017-11-12)	
short-sc: corrected first letter		\@@glxtrfmt: added check for indexing	28
uppercasing .....	222	added grouping .....	28
short-sm: corrected first letter		new .....	28
uppercasing .....	237	\@glxtr@nopostpunc@postdesc: new	111
\ifglxtr@hyperoutside: new .....	57	\@glxtr@restore@postpunc: new ..	111
all: new .....	347	\@glxtrentryfmt: fixed missing label	
nolong-short: new .....	213	argument .....	28
nolong-short-em: new .....	256	\@glxtrfmt: new .....	27
nolong-short-noreg: new .....	214	\eglupdatewidest: new .....	365
nolong-short-sc: new .....	224	\gglupdatewidest: new .....	365
nolong-short-sm: new .....	239	\glupdatewidest: new .....	365
nopostdot: new .....	16	\GlsXtrDefineAbbreviationShortcuts:	
postpunc: new .....	16	changed \newabbr definition to use	
\printunsrtglossaryentryprocesshook:		\providecommand .....	18
new .....	130	\GlsXtrDefineAcShortcuts: changed	
\printunsrtglossarypredoglossary:		\newabbr definition to use	
new .....	130	\providecommand .....	18
\rGLS: new .....	139	\glxtrfmtdisplay: new .....	28
\rGls: new .....	138	\glxtrifcustomdiscardperiod: new	175
\rgls: new .....	138	\GlsXtrIfFieldUndef: new .....	31
\rGLSformat: new .....	141	\glxtrrestorepostpunc: new .....	112
\rGlsformat: new .....	140	\s@glxtrfmt: new .....	27
\rglsformat: new .....	140	\s@glxtrfmt: new .....	27
\rGLSpl: new .....	140	\xglupdatewidest: new .....	365
\rGlspl: new .....	139	1.24 (2017-11-14)	
\rglspl: new .....	138	\glsadd: added \@gls@setsort .....	60
\rGLSplformat: new .....	141	\glxtrforcsvfield: new .....	30
\rGlsplformat: new .....	140	\glxtrlocalsetgrouptitle: new ..	115
\rglsplformat: new .....	140	1.25 (2017-11-14)	
\s@glxtrifhasfield: switched from		\glxtrbookindexmulticolseenv: new	383
\ifdef to \ifundef .....	30	1.25 (2017-11-24)	
1.22 (2017-11-08)		\glsextrapostnamehook: new .....	165
\@glxtr@nopostpunc: new .....	111	\glxtrfootnotename: new .....	206
\@glxtr@orgprintglossary: changed		\glxtrlongnoshortdescname: new .	214
explicit \let for \nopostdesc to		\glxtrlongnoshortname: new .....	217
\glxtractivatenopost .....	110	\glxtrlongshortname: new .....	201
\@glxtrglossentryother: new ....	128	\glxtrlongshortuserdescname: new	271
\glossentrynameother: new .....	166	\glxtronlyname: new .....	291

\glxtrpostlinkAddDescOnFirstUse: changed to use \glxtrparen ....	176	\glxtrGeneralLatinVIrules: new ..	326
\glxtrpostlinkAddSymbolOnFirstUse: changed to use \glxtrparen ....	176	\glxtrGeneralLatinVrules: new ..	325
\glxtrshortlongname: new .....	203	\glxtrgeneralpuncIIrules: new ..	322
\glxtrshortlonguserdesname: new	274	\glxtrgeneralpuncIrules: new ...	321
\glxtrshortnolongname: new .....	209	\glxtrgeneralpuncrules: new ....	320
1.26 (2018-01-05)		\glxtrhyphenrules: new .....	320
\@glxtr@do@inc@linkcount: new ..	141	\glxtrLatinA: new .....	328
\glslinkpresetkeys: new .....	58	\glxtrLatinAA: new .....	330
\glxtr@inc@linkcount: new .....	58	\glxtrLatinAELigature: new .....	330
\GlsXtrEnableLinkCounting: new ..	142	\glxtrLatinE: new .....	328
\GlsXtrIfLinkCounterDef: new ....	142	\glxtrLatinEszettSs: new .....	329
\glxtrinclinkcounter: new .....	142	\glxtrLatinEszettSz: new .....	329
\GlsXtrLinkCounterName: new .....	142	\glxtrLatinEth: new .....	329
\GlsXtrLinkCounterValue: new ....	142	\glxtrLatinH: new .....	328
\GlsXtrTheLinkCounter: new .....	142	\glxtrLatinI: new .....	328
1.27 (2018-02-26)		\glxtrLatinInsularG: new .....	330
\@glxtrdialecthook: new .....	25	\glxtrLatinK: new .....	328
General: added		\glxtrLatinL: new .....	328
glossaries-extra-bib2gls.sty .....	314	\glxtrLatinLslash: new .....	330
\Alpha: new .....	314	\glxtrLatinM: new .....	328
\Beta: new .....	314	\glxtrLatinN: new .....	328
\Chi: new .....	315	\glxtrLatinO: new .....	329
\Digamma: new .....	315	\glxtrLatinOELigature: new .....	330
\Epsilon: new .....	314	\glxtrLatinOslash: new .....	330
\Eta: new .....	314	\glxtrLatinP: new .....	329
\glxtr@loaddialect: new .....	313	\glxtrLatinS: new .....	329
\glxtrBasicDigitrules: new .....	344	\glxtrLatinSchwa: new .....	329
\glxtrcombiningdiacriticIIIrules: new .....	319	\glxtrLatinT: new .....	329
\glxtrcombiningdiacriticIIrules: new .....	318	\glxtrLatinThorn: new .....	330
\glxtrcombiningdiacriticIrules: new .....	318	\glxtrLatinWynn: new .....	330
\glxtrcombiningdiacriticIVrules: new .....	320	\glxtrLatinX: new .....	329
\glxtrcombiningdiacriticrules: new .....	318	\glxtrMathGreekIIrules: new ....	336
\glxtrcontrolrules: new .....	317	\glxtrMathGreekIrules: new .....	335
\glxtrcurrencyrules: new .....	321	\glxtrMathItalicAlpha: new .....	341
\glxtrdigitrules: new .....	344	\glxtrMathItalicBeta: new .....	341
\glxtrfractionrules: new .....	345	\glxtrMathItalicChi: new .....	344
\glxtrGeneralLatinIIIrules: new	323	\glxtrMathItalicDelta: new .....	341
\glxtrGeneralLatinIIrules: new ..	323	\glxtrMathItalicEpsilon: new ...	341
\glxtrGeneralLatinIrules: new ..	322	\glxtrMathItalicEta: new .....	342
\glxtrGeneralLatinIVrules: new ..	324	\glxtrMathItalicGamma: new ....	341
\glxtrGeneralLatinVIIrules: new	327	\glxtrMathItalicGreekIIrules: new .....	332
\glxtrGeneralLatinVIIrules: new	326	\glxtrMathItalicGreekIrules: new	332
		\glxtrMathItalicIota: new .....	342
		\glxtrMathItalicKappa: new .....	342
		\glxtrMathItalicLambda: new ....	342
		\glxtrMathItalicLowerGreekIIrules: new .....	335

<code>\glxtrMathItalicLowerGreekIrules:</code>	<code>\glxtrUpPi: new</code>	340
new	<code>\glxtrUpPsi: new</code>	340
<code>\glxtrMathItalicMu: new</code>	<code>\glxtrUpRho: new</code>	340
new	<code>\glxtrUpSigma: new</code>	340
<code>\glxtrMathItalicNabla: new</code>	<code>\glxtrUpTau: new</code>	340
new	<code>\glxtrUpTheta: new</code>	339
<code>\glxtrMathItalicNu: new</code>	<code>\glxtrUpUpsilon: new</code>	340
new	<code>\glxtrUpXi: new</code>	339
<code>\glxtrMathItalicOmega: new</code>	<code>\glxtrUpZeta: new</code>	338
new	<code>\Iota: new</code>	315
<code>\glxtrMathItalicOmicron: new</code>	<code>\Kappa: new</code>	315
new	<code>\Mu: new</code>	315
<code>\glxtrMathItalicPartial: new</code>	<code>\Nu: new</code>	315
new	<code>\Omicron: new</code>	315
<code>\glxtrMathItalicPhi: new</code>	<code>\omicron: new</code>	315
new	<code>\Rho: new</code>	315
<code>\glxtrMathItalicPi: new</code>	<code>\Tau: new</code>	315
new	<code>\Upalpha: new</code>	315
<code>\glxtrMathItalicPsi: new</code>	<code>\Upbeta: new</code>	315
new	<code>\Upchi: new</code>	316
<code>\glxtrMathItalicRho: new</code>	<code>\Upepsilon: new</code>	315
new	<code>\Upeta: new</code>	316
<code>\glxtrMathItalicSigma: new</code>	<code>\Upiota: new</code>	316
new	<code>\Upkappa: new</code>	316
<code>\glxtrMathItalicTau: new</code>	<code>\Upmu: new</code>	316
new	<code>\Upnu: new</code>	316
<code>\glxtrMathItalicTheta: new</code>	<code>\Upomicron: new</code>	316
new	<code>\upomicron: new</code>	316
<code>\glxtrMathItalicUpperGreekIrules:</code>	<code>\Uprho: new</code>	316
new	<code>\Uptau: new</code>	316
<code>\glxtrMathItalicUpperGreekIrules:</code>	<code>\Upzeta: new</code>	315
new	<code>\Zeta: new</code>	314
<code>\glxtrMathItalicUpsilon: new</code>		
new		
<code>\glxtrMathItalicXi: new</code>		
new		
<code>\glxtrMathItalicZeta: new</code>		
new		
<code>\glxtrMathUpGreekIrules: new</code>		
new		
<code>\glxtrMathUpGreekIrules: new</code>		
new		
<code>\glxtrnonprintablerules: new</code>		
new		
<code>\glxtrprovidecommand: new</code>		
new		
<code>\glxtrspacerules: new</code>		
new		
<code>\glxtrSubScriptDigitrules: new</code>		
new		
<code>\glxtrSuperScriptDigitrules: new</code>		
new		
<code>\glxtrUpAlpha: new</code>		
new		
<code>\glxtrUpBeta: new</code>		
new		
<code>\glxtrUpChi: new</code>		
new		
<code>\glxtrUpDelta: new</code>		
new		
<code>\glxtrUpDigamma: new</code>		
new		
<code>\glxtrUpEpsilon: new</code>		
new		
<code>\glxtrUpEta: new</code>		
new		
<code>\glxtrUpGamma: new</code>		
new		
<code>\glxtrUpIota: new</code>		
new		
<code>\glxtrUpKappa: new</code>		
new		
<code>\glxtrUpLambda: new</code>		
new		
<code>\glxtrUpMu: new</code>		
new		
<code>\glxtrUpNu: new</code>		
new		
<code>\glxtrUpOmega: new</code>		
new		
<code>\glxtrUpOmicron: new</code>		
new		
<code>\glxtrUpPhi: new</code>		
new		

1.28 (2018-03-06)	
<code>\@glxtr@docdefval: changed from</code>	
count register to macro	14
<code>\@glxtrdialecthook: save and restore</code>	
<code>\TrackLangRequireDialectPrefix</code>	
.....	346
<code>\glxtrdeffield: changed \csedef to</code>	
<code>\protected@csedef</code>	31
<code>\glxtrlocalsetgrouptitle: changed</code>	
<code>\csedef \protected@csedef</code>	115
<code>\glxtrsetgrouptitle: changed</code>	
<code>\csxdef \protected@csxdef</code>	115

# 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, 176, 359
\@	123
\@cGLS@	92, 101
\@cGLSpl@	92, 101
\@cGls@	92, 101
\@cGlspl@	92, 101
\@cgls@	92, 101
\@cglspl@	92, 101
\@do@wrglossary	8–10, 107
\@do@wrglossary	11, 13, 25, 60, 76
\@glo@assign@sortkey	113
\@glo@list	6
\@glo@type	129
\@glossarysec	383
\@gls@expand@field	27
\@glslocalreset	90
\@glslocalunset	90
\@glsreset	90
\@glsunset	90
\@glsxtr@autoindex@escspch	170, 171
\@glsxtr@checkspch	168, 169, 171
\@glsxtr@disabledflycommand	50
\@glsxtr@org@postdescription	111
\@glsxtr@record	13
\@glsxtr@recordcounter	13, 14, 126
\@glsxtrfmt	27
\@glsxtrp	85, 86
\@glsxtrpostloctag	52
\@glsxtrpreloctag	52, 53
\@glsxtrwrglossmark	8, 9, 11, 23, 25, 41, 45, 107
\@newglossaryentry@defcounters	91
\@newglossaryentry@defunitcounters	99
\@par	364
\@ACRlong	82
\@ACRlongpl	82
\@ACRshort	81
\@ACRshortpl	82
\@Acrlong	82
\@Acrlongpl	82
\@Acrshort	81
\@Acrshortpl	82
\@GLS@	81, 95, 96, 140
\@GLSdesc@	65
\@GLSpl@	81, 95, 96, 140
\@GLSplural@	82
\@GLSSymbol@	66
\@GLStext@	82
\@GLSxtr@full	186
\@GLSxtr@fullpl	188
\@GLSxtr@p@acrlong@	82
\@GLSxtr@p@acrlongpl@	82
\@GLSxtr@p@acrshort@	81
\@GLSxtr@p@acrshortpl@	82
\@GLSxtr@p@long@	81
\@GLSxtr@p@longpl@	81
\@GLSxtr@p@plural@	81
\@GLSxtr@p@short@	81
\@GLSxtr@p@shortpl@	81
\@GLSxtr@p@text@	81
\@GLSxtrlong	81, 191
\@GLSxtrlongpl	81, 194
\@GLSxtrp	89
\@GLSxtrshort	81, 189
\@GLSxtrshortpl	81, 193
\@Gls@	81, 94, 95, 139
\@Gls@acronymname	103
\@Gls@entry@field	73, 88, 166
\@Gls@entryname	103
\@GlsXtrEnableOnTheFly	47
\@Glspl@	81, 95, 139
\@Glsplural@	82
\@Glstext@	82

\@Glsxtr .....	48, 49	\@end@glxstrifhyphenstart .....	277
\@Glsxtr@full .....	186	\@endfortrue .....	30, 165, 198
\@Glsxtr@fullpl .....	187	\@firstofone .....	61, 129, 160, 167, 173
\@Glsxtr@p@acrlong@ .....	82	\@firstofthree .....	56,
\@Glsxtr@p@acrlongpl@ .....	82		61, 69–72, 78, 185, 187, 189, 190, 192, 193
\@Glsxtr@p@acrshort@ .....	81	\@firstoftwo .....	62–66, 70, 72, 75, 78, 105, 165,
\@Glsxtr@p@acrshortpl@ .....	82		166, 177, 178, 186–188, 192–194, 295, 296
\@Glsxtr@p@long@ .....	81	\@for .....	6, 21, 30, 45, 91, 103,
\@Glsxtr@p@longpl@ .....	81		106, 109, 116, 129, 136, 142, 158, 165, 172
\@Glsxtr@p@plural@ .....	81	\@glo@alias .....	42, 43
\@Glsxtr@p@short@ .....	81	\@glo@assign@sortkey .....	109
\@Glsxtr@p@shortpl@ .....	81	\@glo@autosee .....	24
\@Glsxtr@p@text@ .....	81	\@glo@autoseehook .....	43
\@Glsxtrlong .....	81, 191	\@glo@category .....	97
\@Glsxtrlongpl .....	81, 194	\@glo@check@sortallowed .....	109
\@Glsxtrp .....	88	\@glo@counterprefix .....	10, 118
\@Glsxtrpl .....	49	\@glo@countunit .....	97
\@Glsxtrshort .....	81, 189	\@glo@default@sorttype .....	109
\@Glsxtrshortpl .....	81, 192	\@glo@desc .....	33, 34
\@acrlong .....	82	\@glo@descplural .....	34
\@acrlongpl .....	82	\@glo@group .....	12
\@acrshort .....	81	\@glo@label .....	12, 26, 39, 42–44, 73, 80, 366–373
\@acrshortpl .....	82	\@glo@location .....	12
\@addtoreset .....	141	\@glo@loclist .....	11, 12
\@afterheading .....		\@glo@name .....	168
	350, 351, 360–363, 376–379, 386	\@glo@no@assign@sortkey .....	113
\@alt@glshyp@opt .....	78	\@glo@parent .....	368, 369
\@auxout .....	10, 11, 45, 53, 54, 93,	\@glo@see .....	39, 40, 43, 44
	101, 102, 106, 107, 119, 123, 125, 126, 387	\@glo@seealso .....	42, 43
\@bibgl@restoreat .....	123	\@glo@sort .....	168
\@cGLS .....	95	\@glo@sorttype .....	109, 116
\@cGLS@ .....	92, 96, 101	\@glo@text .....	56
\@cGLSpl .....	96	\@glo@thislettergrp .....	132
\@cGLSpl@ .....	92, 96, 101	\@glo@thisvalue .....	268
\@cGls@ .....	92, 101	\@glo@tmp .....	26, 41, 73
\@cGlspl@ .....	92, 101	\@glo@type .....	44, 79, 103,
\@cgls@ .....	92, 101		106, 110, 112, 116, 117, 119, 122, 123, 129
\@cglspl@ .....	92, 101	\@glo@types .....	156, 157, 366–372
\@disable@onlypremakeg .....	107	\@glossary@default@style .....	50, 51, 110, 381
\@do@auxoutstuff .....	119	\@glossarystyle .....	110
\@do@gl@getcounterprefix .....	10	\@gls@ .....	81, 94, 95, 138
\@do@gl@see .....	43, 44	\@gls@@link .....	57
\@do@newglossaryentry .....	103, 104, 183	\@gls@ReturnAfterFi .....	118
\@do@seeglossary .....	13, 23, 45, 107	\@gls@actualchar .....	169
\@do@wrglossary .....	59, 60, 137	\@gls@adjustmode .....	60
\@empty .....	61, 69–72, 111, 169, 185–195	\@gls@alt@hyp@opt .....	78
\@end@glxtr@addunused .....	44, 45	\@gls@alt@hyp@opt@char .....	78
\@end@glxtr@gettype .....	109, 112	\@gls@alt@hyp@opt@keys .....	78
\@end@glxtr@usese .....	39, 40	\@gls@automake .....	109

<code>\@gls@between</code> .....	116	<code>\@gls@noidxloclist@prev</code> .....	113
<code>\@gls@checkedmkidx</code> .....	168, 169, 171	<code>\@gls@noidxloclist@sep</code> .....	113
<code>\@gls@checkmkidxchars</code> .....	41, 168	<code>\@gls@noref@warn</code> .....	108, 117
<code>\@gls@codepage</code> .....	119	<code>\@gls@org@glsnoidxdisplayloc</code> ..	113, 114
<code>\@gls@counter</code> .....	9, 10, 58, 60, 76, 137	<code>\@gls@org@glssseeformat</code> .....	114
<code>\@gls@currentlettergroup</code> ...	116, 129, 132	<code>\@gls@preglossaryhook</code> .....	110, 173
<code>\@gls@declareoption</code> .....	5	<code>\@gls@prevlevel</code> .....	374, 375, 379, 380
<code>\@gls@default@longpl</code> .....	181, 182	<code>\@gls@quotechar</code> .....	168
<code>\@gls@doautomake</code> .....	109, 126	<code>\@gls@reference</code> .....	45, 46, 106, 107
<code>\@gls@doautomake@err</code> .....	126	<code>\@gls@saveentrycounter</code> 13, 14, 25, 59, 60, 137	
<code>\@gls@encapchar</code> .....	169	<code>\@gls@see@noindex</code> .....	24, 123, 124
<code>\@gls@entry@count</code> .....	92, 93	<code>\@gls@setdefault@glslink@opts</code> .....	9, 28, 58, 77
<code>\@gls@entry@field</code> .....	26, 27, 31, 42, 43, 73, 86–89, 92, 127, 128	<code>\@gls@setsort</code> .....	59, 60
<code>\@gls@entry@unitcount</code> .....	101, 102	<code>\@gls@setupsort@none</code> .....	13
<code>\@gls@field@font</code> .....	61–68	<code>\@gls@short</code> .....	181, 182
<code>\@gls@field@link</code> .....	61–68, 73, 74	<code>\@gls@shortpl</code> .....	179, 182
<code>\@gls@getcounterprefix</code> .....	10	<code>\@gls@sort</code> .....	132
<code>\@gls@getgrouptitle</code> .....	115, 129	<code>\@gls@thisval</code> .....	165
<code>\@gls@grptitle</code> .....	79, 116	<code>\@gls@tmp</code> .....	116
<code>\@gls@hyp@opt</code> .....	73, 74, 78, 95, 96, 134, 138–140, 185–194	<code>\@gls@tmpb</code> .....	171
<code>\@gls@hyp@opt@cs</code> .....	78	<code>\@gls@type</code> .....	107–109, 198, 366–373
<code>\@gls@ifinlist</code> .....	131	<code>\@gls@write@entrycounts</code> .....	92
<code>\@gls@increment@currcount</code> .....	92	<code>\@gls@write@entryunitcounts</code> .....	101
<code>\@gls@increment@currunitcount</code> ....	100	<code>\@gls@write@entryunitcounts@do</code> ....	102
<code>\@gls@keymap</code> ..	11, 12, 26, 39, 42, 73, 125, 165	<code>\@gls@xref</code> .....	11, 41
<code>\@gls@label</code> ...	8–10, 45, 77, 78, 107, 126, 198	<code>\@gls@abbrv@current@abbreviation</code> 181, 195	
<code>\@gls@levelchar</code> .....	169	<code>\@gls@acronymlists</code> .....	103
<code>\@gls@link</code> .....	28, 55–57, 69–72, 186–195	<code>\@gls@doifexistsorwarn</code> ...	14, 161–164, 166
<code>\@gls@link@checkfirsthyper</code> .....	56, 105	<code>\@gls@entry</code> .....	93, 101, 102
<code>\@gls@link@label</code> .....	58, 137	<code>\@gls@link</code> .....	59, 79–81
<code>\@gls@link@nocheckfirsthyper</code> .....	55, 68–72, 185–194	<code>\@gls@nextpages</code> .....	110
<code>\@gls@link@opts</code> .....	58	<code>\@gls@nonextpages</code> .....	110
<code>\@gls@list</code> .....	116	<code>\@gls@numberformat</code> .....	9, 10, 58, 60, 76, 137, 165, 168
<code>\@gls@local@increment@currcount</code> ....	92	<code>\@gls@order</code> .....	106
<code>\@gls@local@increment@currunitcount</code> 100		<code>\@gls@pl@</code> .....	81, 94, 95, 138
<code>\@gls@location</code> .....	132, 133	<code>\@gls@plural@</code> .....	82
<code>\@gls@loclist</code> .....	113, 114, 132, 133	<code>\@gls@punc@token</code> .....	178
<code>\@gls@long</code> .....	181	<code>\@gls@recordlocref</code> .....	10
<code>\@gls@longpl</code> .....	179, 181, 182	<code>\@gls@showtarget</code> .....	80
<code>\@gls@map</code> .....	165	<code>\@gls@style@altlist</code> .....	349
<code>\@gls@nohyperlist</code> .....	35–37	<code>\@gls@style@altlistgroup</code> .....	350
<code>\@gls@noidx@do</code> .....	116	<code>\@gls@style@altlisthypergroup</code> .....	351
<code>\@gls@noidx@getgrouptitle</code> .....	129	<code>\@gls@style@alttree</code> .....	363
<code>\@gls@noidx@nosanitizesort</code> .....	109	<code>\@gls@style@alttreegroup</code> .....	375
<code>\@gls@noidx@sanitizesort</code> .....	109	<code>\@gls@style@alttreehypergroup</code> .....	375
<code>\@gls@noidxloclist@finalsep</code> .....	113	<code>\@gls@style@index</code> .....	359
		<code>\@gls@style@indexgroup</code> .....	360



<code>\@glsstyle@indexhypergroup</code>	360	<code>\@glsxtr@autoseeindexfalse</code>	13
<code>\@glsstyle@inline</code>	359	<code>\@glsxtr@autoseeindextrue</code>	15
<code>\@glsstyle@list</code>	349	<code>\@glsxtr@bookindex@atendgroup</code>	384–386
<code>\@glsstyle@listdotted</code>	348	<code>\@glsxtr@bookindex@atsubendgroup</code>	385
<code>\@glsstyle@listgroup</code>	350	<code>\@glsxtr@bookindex@atsubsubendgroup</code>	385
<code>\@glsstyle@listhypergroup</code>	350	<code>\@glsxtr@bookindex@between</code>	384, 386
<code>\@glsstyle@mcolalttree</code>	379	<code>\@glsxtr@bookindex@sep</code>	384, 385
<code>\@glsstyle@mcolalttreegroup</code>	379	<code>\@glsxtr@bookindex@subatendgroup</code>	..
<code>\@glsstyle@mcolalttreehypergroup</code>	380		384–386
<code>\@glsstyle@mcolalttreespannav</code>	380	<code>\@glsxtr@bookindex@subbetween</code>	384, 385
<code>\@glsstyle@mcolindexgroup</code>	376	<code>\@glsxtr@bookindex@subsep</code>	384, 385
<code>\@glsstyle@mcolindexhypergroup</code>	376	<code>\@glsxtr@bookindex@subsubatendgroup</code>	..
<code>\@glsstyle@mcolindexspannav</code>	376		384–386
<code>\@glsstyle@mcoltreegroup</code>	377	<code>\@glsxtr@bookindex@subsubbetween</code>	..
<code>\@glsstyle@mcoltreehypergroup</code>	377		384, 385
<code>\@glsstyle@mcoltreenamegroup</code>	378	<code>\@glsxtr@bookindex@groupskip</code>	384, 386
<code>\@glsstyle@mcoltreenamehypergroup</code>	378	<code>\@glsxtr@cat</code>	91, 103, 136, 172, 173
<code>\@glsstyle@mcoltreenamepannav</code>	379	<code>\@glsxtr@checkgroup</code>	130
<code>\@glsstyle@mcoltreespannav</code>	377	<code>\@glsxtr@counterrecordhook</code>	10, 11
<code>\@glsstyle@tree</code>	361	<code>\@glsxtr@csname</code>	98–100
<code>\@glsstyle@treegroup</code>	361	<code>\@glsxtr@current@style</code>	50, 381
<code>\@glsstyle@treehypergroup</code>	362	<code>\@glsxtr@currentunitcount</code>	98, 100
<code>\@glsstyle@treename</code>	362	<code>\@glsxtr@currunitcount</code>	99, 101
<code>\@glsstyle@treenamegroup</code>	363	<code>\@glsxtr@debugnr</code>	22
<code>\@glsstyle@treenamehypergroup</code>	363	<code>\@glsxtr@debugval</code>	22
<code>\@glstarget</code>	80, 81, 112	<code>\@glsxtr@declareoption</code>	5, 15, 17, 20
<code>\@glstext@</code>	82	<code>\@glsxtr@defaultnoglossarywarning</code>	20, 21
<code>\@glswidestname</code>	366, 373	<code>\@glsxtr@defaultnumberformat</code>	..
<code>\@glsxtr</code>	48, 49		7, 9, 58, 60, 76, 165, 168
<code>\@glsxtr@@do@wrglossary</code>	107	<code>\@glsxtr@defpostpunc</code>	15, 16, 23
<code>\@glsxtr@abbreviationsdef</code>	17, 24, 25	<code>\@glsxtr@deprecated@abbrstyle</code>	..
<code>\@glsxtr@accessdisplay</code>	165–167		227, 228, 230,
<code>\@glsxtr@activate@initialtagging</code>	..		232, 241, 242, 244, 246, 258, 262, 266, 267
	173, 174	<code>\@glsxtr@disabledflycommand</code>	49
<code>\@glsxtr@addunitcounter</code>	97	<code>\@glsxtr@display@loc</code>	117
<code>\@glsxtr@addunused</code>	45	<code>\@glsxtr@do@@wrgindex</code>	77
<code>\@glsxtr@addunusedxrefs</code>	44, 45	<code>\@glsxtr@do@gl:disablehyperinlist</code>	75
<code>\@glsxtr@attrval</code>	59, 159–164, 166, 168	<code>\@glsxtr@do@inc@linkcount</code>	142
<code>\@glsxtr@autoindex@at</code>	168–170	<code>\@glsxtr@do@record@wrglossary</code>	8, 13
<code>\@glsxtr@autoindex@doextra@esc</code>	168	<code>\@glsxtr@do@redef@for@gl@sentries</code>	7
<code>\@glsxtr@autoindex@encap</code>	168–170	<code>\@glsxtr@do@style</code>	22, 314
<code>\@glsxtr@autoindex@esc</code>	168–171	<code>\@glsxtr@do@titlecaps@warn</code>	..
<code>\@glsxtr@autoindex@escat</code>	169, 170		160–163, 166, 173
<code>\@glsxtr@autoindex@escencap</code>	169, 170	<code>\@glsxtr@doabbreviationsdef</code>	17
<code>\@glsxtr@autoindex@esclevel</code>	169, 170	<code>\@glsxtr@doaccsupp</code>	20, 23
<code>\@glsxtr@autoindex@escquote</code>	169, 170	<code>\@glsxtr@docdefsetting</code>	14
<code>\@glsxtr@autoindex@level</code>	169, 170	<code>\@glsxtr@docdefval</code>	14, 46
<code>\@glsxtr@autoindex@setname</code>	168	<code>\@glsxtr@docounterrecord</code>	11
<code>\@glsxtr@autoindexcrossrefs</code>	13, 15, 39, 42	<code>\@glsxtr@doglossary</code>	129, 130

<code>\@glsxtr@doiflabelinlist</code> .....	131	<code>\@glsxtr@noidx@entrynumberlist</code> ....	108
<code>\@glsxtr@doloctag</code> .....	52, 53	<code>\@glsxtr@noidx@numberlistloop</code> ....	108
<code>\@glsxtr@dorecord</code> .....	8, 9	<code>\@glsxtr@nomissingglsxtrtextnr</code> .....	20
<code>\@glsxtr@dorecordnodefer</code> .....	8, 9	<code>\@glsxtr@nomissingglsxtrtextval</code> .....	20
<code>\@glsxtr@dosee@alsoindex@glossary</code> ..	13	<code>\@glsxtr@noop@recordcounter</code> ....	11, 13
<code>\@glsxtr@doseeglossary</code> .....	13, 23	<code>\@glsxtr@nopostpunc</code> .....	111
<code>\@glsxtr@dostylewarn</code> .....	198	<code>\@glsxtr@nopostpunc@postdesc</code> ....	111
<code>\@glsxtr@enabletagging</code> .....	172	<code>\@glsxtr@notfoundinlist</code> .....	178
<code>\@glsxtr@end@</code> .....	47	<code>\@glsxtr@op@recordcounter</code> ....	13, 14
<code>\@glsxtr@endescspch</code> .....	169–171	<code>\@glsxtr@optlist</code> .....	49
<code>\@glsxtr@entrycount@org@localreset</code> .	92	<code>\@glsxtr@org@@starttoc</code> .....	294, 295
<code>\@glsxtr@entrycount@org@localunset</code> .	92	<code>\@glsxtr@org@GLS@</code> .....	56
<code>\@glsxtr@entrycount@org@reset</code> .....	92	<code>\@glsxtr@org@GLSpl@</code> .....	56
<code>\@glsxtr@entrycount@org@unset</code> .....	92	<code>\@glsxtr@org@Gls@</code> .....	56
<code>\@glsxtr@entryunitcount@org@localreset</code>		<code>\@glsxtr@org@Glspl@</code> .....	56
.....	100	<code>\@glsxtr@org@Glsxtrtitlefirst</code> .	296, 297
<code>\@glsxtr@entryunitcount@org@localunset</code>		<code>\@glsxtr@org@Glsxtrtitlefirstplural</code>	
.....	100	.....	296, 297
<code>\@glsxtr@entryunitcount@org@reset</code> .	100	<code>\@glsxtr@org@Glsxtrtitlefull</code> ..	296, 297
<code>\@glsxtr@entryunitcount@org@unset</code> .	100	<code>\@glsxtr@org@Glsxtrtitlefullpl</code>	296, 297
<code>\@glsxtr@err@undefaction</code> .....	7, 13	<code>\@glsxtr@org@Glsxtrtitlelong</code> ..	296, 297
<code>\@glsxtr@field@linkdefs</code> .....	55	<code>\@glsxtr@org@Glsxtrtitlelongpl</code>	296, 297
<code>\@glsxtr@format@overridefalse</code> ....	167	<code>\@glsxtr@org@Glsxtrtitlename</code> ..	296, 297
<code>\@glsxtr@format@overridefalse</code> ....	167	<code>\@glsxtr@org@Glsxtrtitleplural</code>	296, 297
<code>\@glsxtr@foundinlist</code> .....	178	<code>\@glsxtr@org@Glsxtrtitleshort</code> .	295, 297
<code>\@glsxtr@full</code> .....	185	<code>\@glsxtr@org@Glsxtrtitleshortpl</code>	295, 297
<code>\@glsxtr@fullpl</code> .....	187	<code>\@glsxtr@org@Glsxtrtitletext</code> ..	296, 297
<code>\@glsxtr@gettype</code> .....	109	<code>\@glsxtr@org@MakeUppercase</code> ....	295, 297
<code>\@glsxtr@glossdescfont</code> .....	159–161	<code>\@glsxtr@org@checkfirsthyper</code> ...	74, 105
<code>\@glsxtr@glossnamefont</code> .	161–164, 166, 167	<code>\@glsxtr@org@delimN</code> .....	53
<code>\@glsxtr@gobbleto@endescspch</code> .....	171	<code>\@glsxtr@org@delimR</code> .....	53
<code>\@glsxtr@groupheading</code> .....	130, 132	<code>\@glsxtr@org@doseeglossary</code> ....	23, 107
<code>\@glsxtr@idx@displaynumberlist</code> ....	108	<code>\@glsxtr@org@gloautoosee</code> .....	24
<code>\@glsxtr@idx@entrynumberlist</code> .....	108	<code>\@glsxtr@org@gls@</code> .....	55
<code>\@glsxtr@ifcsstart</code> .....	47	<code>\@glsxtr@org@glsdohypertarget</code> ....	112
<code>\@glsxtr@ifpunctoken</code> .....	178	<code>\@glsxtr@org@glsignore</code> .....	53
<code>\@glsxtr@ifunitcounter</code> .....	97	<code>\@glsxtr@org@glspl@</code> .....	55
<code>\@glsxtr@insert@dots</code> .....	180	<code>\@glsxtr@org@glsxtrtitlefirst</code> .	296, 297
<code>\@glsxtr@insert@dots@next</code> .....	180	<code>\@glsxtr@org@glsxtrtitlefirstplural</code>	
<code>\@glsxtr@insertdots</code> .....	181	.....	296, 297
<code>\@glsxtr@label</code> .....	30, 45, 142, 158, 159	<code>\@glsxtr@org@glsxtrtitlefull</code> ..	296, 297
<code>\@glsxtr@loadstyles</code> .....	347, 348	<code>\@glsxtr@org@glsxtrtitlefullpl</code>	296, 297
<code>\@glsxtr@longnewglossaryentry</code> .....	33	<code>\@glsxtr@org@glsxtrtitlelong</code> ..	296, 297
<code>\@glsxtr@mark@wordseps</code> .....	180	<code>\@glsxtr@org@glsxtrtitlelongpl</code>	296, 297
<code>\@glsxtr@mark@wordseps@next</code> .....	180	<code>\@glsxtr@org@glsxtrtitlename</code> ..	296, 297
<code>\@glsxtr@markwordseps</code> .....	181, 182	<code>\@glsxtr@org@glsxtrtitleorpdforheading</code>	
<code>\@glsxtr@mixed@assign@sortkey</code> ....	109	.....	295, 297
<code>\@glsxtr@noidx@displaynumberlist</code> ..	108	<code>\@glsxtr@org@glsxtrtitleplural</code>	296, 297
<code>\@glsxtr@noidx@do</code> .....	131	<code>\@glsxtr@org@glsxtrtitleshort</code> .	295, 297

<code>\@glsxtr@org@glsxtrtitleshortpl</code>	295, 297	<code>\@glsxtr@setentryunitcountunsetattr</code>	
<code>\@glsxtr@org@glsxtrtitletext</code>	296, 297		102, 103
<code>\@glsxtr@org@makeglossaries</code>	106	<code>\@glsxtr@setupshortcuts</code>	19, 20, 24, 25
<code>\@glsxtr@org@markboth</code>	294, 295	<code>\@glsxtr@shortcutsnr</code>	19
<code>\@glsxtr@org@markright</code>	294, 295	<code>\@glsxtr@shortcutsval</code>	19, 125
<code>\@glsxtr@org@newacronymstyle</code>	104, 105	<code>\@glsxtr@swaptwo</code>	178
<code>\@glsxtr@org@postdescription</code>	111, 174	<code>\@glsxtr@tag</code>	173
<code>\@glsxtr@org@see@noindex</code>	123, 124	<code>\@glsxtr@taggingcs</code>	173
<code>\@glsxtr@org@setacronymstyle</code>	104, 105	<code>\@glsxtr@textformat</code>	59, 60
<code>\@glsxtr@org@theHvalue</code>	8, 9	<code>\@glsxtr@theHvalue</code>	8, 9, 57–60, 137
<code>\@glsxtr@org@prefix</code>	10	<code>\@glsxtr@thevalue</code>	8, 9, 57–60, 137
<code>\@glsxtr@org@printglossary</code>	49, 112	<code>\@glsxtr@thisloctag</code>	53
<code>\@glsxtr@org@warndep</code>	179	<code>\@glsxtr@titlelabel</code>	114, 115, 131
<code>\@glsxtr@p@acrlong@</code>	82	<code>\@glsxtr@tmp</code>	21, 117, 118
<code>\@glsxtr@p@acrlongpl@</code>	82	<code>\@glsxtr@type</code>	159
<code>\@glsxtr@p@acrshort@</code>	81	<code>\@glsxtr@unitcountlist</code>	97
<code>\@glsxtr@p@acrshortpl@</code>	82	<code>\@glsxtr@unsrt@getgrouptitle</code>	129
<code>\@glsxtr@p@long@</code>	81	<code>\@glsxtr@usesee</code>	40
<code>\@glsxtr@p@longpl@</code>	81	<code>\@glsxtr@warn@onexistsordo</code>	7, 13, 14
<code>\@glsxtr@p@plural@</code>	81	<code>\@glsxtr@warn@undefaction</code>	7, 13, 14
<code>\@glsxtr@p@short@</code>	81	<code>\@glsxtr@wrglossnr</code>	57
<code>\@glsxtr@p@shortpl@</code>	81	<code>\@glsxtr@wrglossval</code>	57
<code>\@glsxtr@p@text@</code>	81	<code>\@glsxtr@dialecthook</code>	313
<code>\@glsxtr@pagetag</code>	52, 53	<code>\@glsxtr@docdeffalse</code>	46
<code>\@glsxtr@pagetag</code>	52, 53	<code>\@glsxtr@entryfmt</code>	28
<code>\@glsxtr@prevunitcount</code>	99	<code>\@glsxtr@fmt</code>	27
<code>\@glsxtr@printglossnr</code>	112	<code>\@glsxtr@glossentry</code>	127
<code>\@glsxtr@printglossopts</code>	49, 109, 112	<code>\@glsxtr@glossentryother</code>	127, 128
<code>\@glsxtr@printglossval</code>	112	<code>\@glsxtr@hypernameprefix</code>	112, 131
<code>\@glsxtr@printunsrtglossaryskipentry</code>		<code>\@glsxtr@trifhasfield</code>	30
	129, 130	<code>\@glsxtr@trifhyphenstart</code>	277
<code>\@glsxtr@provide@addstoragekey</code>	27	<code>\@glsxtr@indexaliased</code>	76
<code>\@glsxtr@provide@storagekey</code>	26	<code>\@glsxtr@indexcrossrefsfalse</code>	15
<code>\@glsxtr@record</code>	13, 55–57, 60	<code>\@glsxtr@indexcrossrefstrue</code>	15
<code>\@glsxtr@record@setting</code>		<code>\@glsxtr@inmark</code>	294, 295
	8, 9, 12, 41, 45, 46, 106	<code>\@glsxtr@long</code>	81, 190
<code>\@glsxtr@record@setting@alsoindex</code>		<code>\@glsxtr@longpl</code>	81, 193
	8, 9, 41, 106	<code>\@glsxtr@newgls</code>	134, 135
<code>\@glsxtr@record@setting@off</code>	45	<code>\@glsxtr@newgls@inner</code>	133, 134
<code>\@glsxtr@record@setting@only</code>	106	<code>\@glsxtr@newgls@innercsname</code>	134
<code>\@glsxtr@recordsee</code>	13, 23, 41	<code>\@glsxtr@notinmark</code>	294, 295
<code>\@glsxtr@redef@forglsentries</code>	7, 24, 25	<code>\@glsxtr@p</code>	87
<code>\@glsxtr@redef@styles</code>	21, 314	<code>\@glsxtr@p@opt</code>	85
<code>\@glsxtr@reg@glosslist</code>	106–109, 112	<code>\@glsxtr@pl</code>	48, 49
<code>\@glsxtr@restore@postpunc</code>	111	<code>\@glsxtr@postloctag</code>	52, 54
<code>\@glsxtr@rglstrigger@record</code>	138–140	<code>\@glsxtr@preloctag</code>	52, 54
<code>\@glsxtr@s@longnewglossaryentry</code>	33	<code>\@glsxtr@setaliasnoindex</code>	75–77
<code>\@glsxtr@save@preloctag</code>	52–54	<code>\@glsxtr@short</code>	81, 188
<code>\@glsxtr@setentrycountunsetattr</code>	91	<code>\@glsxtr@shortpl</code>	81, 192

\@glxstrundeftag .....	6, 25	\@rglspl@ .....	138
\@glxstrwrglossmark .....	22	\@sGlsXtrEnableOnTheFly .....	47
\@gobble .. 7, 13, 15, 61, 130, 131, 180, 384–386		\@secondofthree .....	61–
\@gobbletwo .....	179	63, 69–72, 74, 186, 188, 189, 191, 192, 194	
\@ifnextchar .....	78	\@secondoftwo .....	56, 61,
\@ifpackageloaded .....	5, 17, 125,	64–72, 75, 80, 105, 112, 165, 178, 185–	
143, 159, 161, 163, 165, 167, 313, 315, 346		187, 189–194, 208, 231, 245, 266, 295, 297	
\@ifstar .. 26, 27, 30, 33, 34, 36, 47, 78, 128, 172		\@sglsxtr@provide@storagekey .....	26
\@ifundefined .....	313	\@starttoc .....	295
\@ignored@glossaries .....	35–37	\@thirdofthree .....	61–64,
\@input .....	123	69–72, 74, 187, 188, 190, 191, 193, 194, 296	
\@input@ .....	119	\@thirdoftwo .....	64–68
\@istfilename .....	106	\@this@key .....	165
\@makeglossary .....	106, 107	\@warn@nomakeglossaries .....	120
\@mfu@domakefirstuc .....	173	\@xdy@main@language .....	119
\@mfu@nocaplist .....	173	\@xdycrossrefhook .....	41
\@one .....	93, 102, 134	\@xdy@language .....	119
\@newglossaryentry@defcounters .. 91, 99		\@xdy@locationclassorder .....	41
\@newglossaryentryposthook .....		\\ .....	118
11, 12, 26, 42, 73			
\@newglossaryentryprehook .....			
11, 12, 26, 33, 34, 42, 73		\_ .....	46, 121, 122
\@nil .....	118, 132		
\@nnil .....	169, 171, 178, 180		
\@no@glxstrindexaliased .....	76		
\@no@makeglossaries .....	123		
\@nocounterr .....	142		
\@nopostdesc .....	111		
\@onelevel@sanitize ... 11, 41, 49, 115, 131			
\@onlypreamble .....			
50, 53, 101, 124, 126, 142, 167, 170–172			
\@org@glossaryentrynumbers .... 110, 111			
\@org@newglossaryentryprehook ... 33, 34			
\@print@unsrt@glossary .....	128, 129		
\@printgloss@setsort .....	109, 110		
\@printglossary .....	49, 128, 129		
\@printunsrt@glossary@handler .... 130			
\@printunsrtglossary .....	128		
\@rGLS .....	139		
\@rGLS@ .....	139		
\@rGLSpl .....	140		
\@rGLSpl@ .....	140		
\@rGls .....	138		
\@rGls@ .....	139		
\@rGlspl .....	139		
\@rGlspl@ .....	139		
\@rgls .....	138		
\@rgls@ .....	138		
\@rglspl .....	138		

## A

\AA .....	330
\aa .....	330
\AB .....	17
\Ab .....	17
\ab .....	17
abbreviation styles:	
long-hyphen-postshort-hyphen	282, 283, 285
long-hyphen-short-hyphen .....	279, 283
long-postshort-user .....	271
long-short-user .....	270
nolong-short .....	214
short .....	212
short-hyphen-long-hyphen .....	287, 288
short-hyphen-postlong-hyphen ...	288, 290
short-long-user .....	272
short-nolong .....	211, 213
short-nolong-desc .....	213
short-postlong-user .....	274
\abbreviationsname .....	17
\abbrvpluralsuffix .....	
125, 182, 202, 204, 206, 208,	
210, 212, 215, 219, 220, 222, 223, 225,	
227, 229, 231, 233, 235, 236, 238, 240,	
241, 243, 245, 247, 249, 250, 252, 254,	
255, 257, 259, 261, 262, 265, 266, 269,	
270, 273, 276, 279, 280, 284, 287, 289, 292	



234, 236–238, 241–244, 248, 250–254, 256, 258, 260, 261, 263, 264, 266, 269, 275, 276, 278–280, 282, 286, 287, 292, 293	
textformat .....	59
\cdot .....	22
\centering .....	383
\cGLS .....	17, 18, 91, 102
\cGls .....	17, 18, 91, 102
\cglS .....	17, 18, 91, 102
\cGLSformat .....	95
\cGlsformat .....	94
\cglSformat .....	94, 96
\cGLSpl .....	17, 18, 91, 102
\cGlspl .....	17, 18, 91, 102
\cglSpl .....	17, 18, 91, 102
\cGLSplformat .....	95
\cGlsplformat .....	94
\cglSplformat .....	94, 96
\changes .....	300
\char .....	115
\columnwidth .....	51
\count@ .....	93, 102
\csappto .....	33
\csdef .....	26, 31, 33, 73, 74, 92, 97, 98, 100, 154, 198–200, 208, 230, 244, 266, 270, 272, 274, 283, 285, 289, 290, 348, 365
\cseappto .....	37
\csedef .....	99
\csgdef .....	32, 35–37, 46, 52, 92, 98, 100, 101, 364, 365, 387
\cslet .....	31, 34, 110
\csletcs .....	31, 32, 200
\csname .....	6, 27, 36, 41, 45, 50, 54, 56, 58, 60, 69–74, 76, 85, 98, 99, 107, 116, 119, 122, 123, 129, 134– 137, 142, 159, 179, 186–195, 200, 201, 373
\cspreto .....	33
\csuse .....	28, 29, 36, 43, 44, 53, 73, 74, 86–88, 97–101, 110, 114, 116, 117, 126, 127, 131, 132, 154, 165, 175, 176, 198–200, 365, 366, 368, 369
\csxdef .....	39, 42, 98, 101
\currentglossary .....	110, 127, 128, 386
\CurrentOption .....	22, 347, 348
\CurrentTrackedLanguage .....	346
\CurrentTrackedLanguageTag .....	125
\CurrentTrackedScript .....	346
\CurrentTrackedTag .....	313, 346
\CustomAbbreviationFields .....	183, 201, 203, 205, 206, 208, 210, 212, 215, 217–223, 225, 228, 230, 232, 234–237, 239, 243, 244, 246, 248–253, 255, 257, 259, 262, 264, 266, 269–272, 274–276, 278–280, 282, 283, 285–288, 290, 291, 293
<b>D</b>	
\DeclareAcronymList .....	103
\DeclareOption .....	5, 347
\DeclareOptionX .....	5, 22
\def .....	9–14, 23, 25, 27, 28, 33, 34, 36, 40–42, 45, 47–49, 52, 54–72, 78, 80– 84, 94–96, 103, 107, 109–113, 115–119, 129, 131, 132, 134, 137–140, 168–171, 173, 177–182, 185–195, 198, 277, 280, 282, 283, 286, 288, 346, 374, 375, 379, 380
\defglSentryfmt .....	35–37
\define@boolkey .....	15, 57, 75
\define@choicekey .....	7, 12, 14, 16, 19, 20, 22, 57, 112
\define@key .....	11, 12, 16, 21, 22, 26, 42, 57, 60, 73, 112, 179
\DefineAcronymSynonyms .....	19
\delimN .....	53
\delimR .....	53
\detokenize .....	47
\dimen@ .....	105, 365–373
\dimen@i .....	367–369
\dimen@ii .....	365–369
\dimexpr .....	51, 364
\disable@keys .....	17, 25, 46, 123
\do .....	6, 21, 30, 45, 91, 103, 106, 109, 116, 129, 136, 142, 158, 165, 172
\do@glS@link@checkfirsthyper .....	28, 55–58, 68–72, 185–194
\do@glSdisablehyperinlist .....	58, 76
doc package .....	171
\dolistcsloop .....	29
\DTLifinlist .....	107, 108, 112
\DTLifint .....	115
<b>E</b>	
\eappto .....	11, 21, 35–37, 130, 132, 168, 347
\edef .....	6, 8–10, 35–38, 41, 43–45, 58–60, 75, 76, 79, 80, 97, 98, 100, 106–108, 112, 115, 117–119, 123, 127, 128, 137, 141, 159–162, 164–166, 168, 169, 171, 179, 346, 368, 369, 384, 385
\eglssetwidest .....	366–373

\egroup	34, 111	fontspec package	125
\else	8–11, 15–17, 19, 20, 23, 24, 28, 32, 46, 47, 52, 54, 56, 59, 60, 76, 77, 93, 105, 106, 109–112, 115, 117, 118, 121, 122, 124, 137, 167–169, 171, 178, 180, 182, 189–195, 197, 198, 202, 204, 207, 209–216, 219–233, 235–247, 249, 251–267, 269–271, 273, 276–278, 280, 283, 285, 286, 288, 290, 292, 293, 349, 351–361, 363, 374, 375, 381, 383, 385	\footnote	206
\emph	246	\forall glossaries	44, 129, 157, 159, 366, 367, 369–373
\empty	117, 118	\forall gls entries	93, 102
\encapchar	171	\ForEachTrackedDialect	314, 346
\end	116, 121, 122, 130, 349, 351–358, 377–380, 384	\forgls entries	6, 44, 157, 159, 366–373
\end@glxtr@display@loc	117	\forlistcsloop	29, 102, 116
\endcsname	6, 27, 36, 41, 45, 50, 54, 56, 58, 60, 69–74, 76, 85, 98, 99, 107, 116, 119, 122, 123, 129, 134– 137, 142, 159, 179, 186–195, 200, 201, 373	\forlistloop	113, 114, 173
\endgroup	8, 10, 28, 76, 127–129, 141	\futurelet	178
\ensuremath	22		
entry categories:		G	
abbreviation	195	\gdef	53, 170
general	153, 155	\Genacrfullformat	104
index	157	\genacrfullformat	104
\epreto	168	\GenericAcronymFields	104
\equal	122, 123	\Genplacrfullformat	104
etoolbox package	5	\genplacrfullformat	104
\expandafter	22, 27, 28, 30, 39, 40, 44, 45, 47–49, 73, 74, 78, 85, 96, 97, 107–109, 112, 116–118, 129, 130, 132, 134, 141, 159, 162, 163, 165, 167, 168, 171, 178, 181, 182, 277, 384	\glo@grabfirst	132
\expandonce	104, 132, 169, 202, 280	\glo@name	162, 163, 166, 167
F		\gloaliaslabel	80
\fi	7–11, 14–17, 19–24, 28, 32, 39, 41, 42, 44, 46, 47, 50–52, 54, 56, 57, 59, 60, 76, 77, 93, 94, 101, 102, 105, 109–112, 115, 117–119, 121–124, 126, 137, 167– 169, 171, 178, 180, 182, 189–195, 197, 198, 202, 204, 207, 209–216, 219–233, 235–247, 249, 251–267, 269–271, 273, 276–278, 280, 283–286, 288, 290, 292, 293, 349, 351–363, 365–375, 381, 383, 386	\global	10, 34, 111, 132
first use	388	\glolinkprefix	59, 60, 80, 125
flag	388	glossaries package	13, 23, 24, 39, 41–43, 109, 348
text	388	glossaries-accsupp package	20, 23, 143
\firstacronymfont	104, 105	glossaries-extra package	2, 346
		glossaries-extra-bib2gls package	13, 25, 313, 346
		glossaries-extra-stylemods package	21, 175, 314
		glossaries-stylemods package	382
		glossaries.sty package	34
		\GlossariesExtraWarning	6, 15, 33, 47, 49, 59, 105, 108, 117, 121, 124, 129, 159–162, 164, 166, 173, 200
		\GlossariesExtraWarningNoLine	15, 93, 102
		\GlossariesWarning	52, 108, 110, 113, 114, 198
		\GlossariesWarningNoLine	107, 120
		glossary styles:	
		altlist	349
		altlistgroup	350
		altlisthypergroup	351
		almtree	363, 364, 374
		almtreegroup	375
		almtreehypergroup	375
		index	359
		indexgroup	360
		indexhypergroup	360
		inline	359
		list	349
		listdotted	348



listdottedstyle	349	\gls@checkseeallowed	46, 47, 107
listgroup	350	\gls@codepage	119
listhypergroup	350	\gls@defdocnewglossaryentry	91, 99
mcolalmtree	379	\gls@defglossaryentry	34, 48, 49
mcolalmtreegroup	379	\gls@dotocitle	110
mcolalmtreehypergroup	380	\gls@glossary	41
mcolalmtreespannav	380	\gls@grplabel	79
mcolindexgroup	376	\gls@level	132, 133
mcolindexhypergroup	376	\gls@noidxglossary	107
mcolindexspannav	376	\gls@org@glossaryentryfield	110
mcoltreegroup	377	\gls@org@glossarysubentryfield	110
mcoltreehypergroup	377	\gls@orgTrackLangRequireDialectPrefix	346
mcoltreenonamegroup	378	\gls@save@numberlist	52, 54
mcoltreenonamehypergroup	378	\gls@set@xr@key	42
mcoltreenonamespannav	379	\gls@tmplen	366–375
mcoltreespannav	377	\gls@type	107
sublistdotted	349	\glsabbrvdefaultfont	185, 202, 204, 206, 208, 210, 212, 215, 268, 278, 281, 291
tree	361	\glsabbrvemfont	246–255, 257, 259, 261, 262, 264–266
treegroup	361	\glsabbrvfont	82, 83, 104, 185, 189, 190, 192, 193, 195, 197, 198, 201–206, 208–210, 212, 215, 217, 219, 220, 222, 223, 225, 227, 229, 231, 233, 235, 236, 238, 240, 241, 243, 245, 247, 249, 250, 252, 254, 255, 257, 259, 261, 262, 265, 266, 269, 270, 273, 275, 276, 279, 281, 283, 284, 287, 289, 292
treehypergroup	362	\glsabbrvhyphenfont	278, 279, 283–290
treenoname	362	\glsabbrvonlyfont	291–293
treenonamegroup	363	\glsabbrvscfont	218–223, 225, 227, 229–231
treenonamehypergroup	363	\glsabbrvsmfont	232–238, 240, 241, 243–245
glossary-bookindex package	348	\glsabbrvuserfont	268–276
glossary-hypernav package	79	\GLSaccessdesc	65
glossary-long package	353	\Glsaccessdesc	65, 160, 172
glossary-longbooktabs package	353	\glsaccessdesc	64, 160, 176
\glossaryentrynumbers	54, 110, 111, 133	\GLSaccessdescplural	65
\glossaryheader	116, 129, 349–358, 360–363, 374–378, 380, 384	\Glsaccessdescplural	65
\glossaryname	110	\glsaccessdescplural	65
\glossarypostamble	116, 130, 131	\GLSaccessfirst	62
\glossarypreamble	116, 129	\Glsaccessfirst	62
\glossarysection	116, 122, 129, 131	\glsaccessfirst	62
\glossarytitle	36, 110, 116, 122, 129	\GLSaccessfirstplural	64
\glossarytoctitle	36, 110, 116, 122, 129	\Glsaccessfirstplural	64
\glossentry	110, 133, 348, 349, 351–358, 360–362, 374, 384	\glsaccessfirstplural	63
\glossentrydesc	348–358, 360–364	\Glsaccesslong	71, 183, 191, 202, 210, 215, 216, 219, 225–228,
\glossentryname	127, 348–358, 360–362, 374, 375, 382		
\glossentrynameother	128		
\glossentrysymbol	352–356, 358, 360–362, 364		
\glossxtrsetpopts	174		
\GLS	91, 102, 136		
\Gls	48, 91, 102, 136		
\gls	32, 48, 50, 91, 102, 108, 120, 136		
\gls@assign@desc	34		
\gls@assign@field	12, 26, 73		



233, 239–242, 247, 249, 256–259, 261,	\Glsaccessstext .....	62
263, 269, 271, 279, 281, 284, 285, 287, 292	\glsaccessstext .....	40, 61
\glsaccessslong .....	\glsacrshortcutstrue .....	19, 20
190, 191, 202, 204, 207, 209, 210, 212,	\glsacspacemax .....	105
214–216, 219, 221, 222, 224–233, 235–	\glsadd .....	28, 45, 120
244, 246, 247, 249, 251–263, 265, 267,	\glsadd options	
269–271, 273, 276, 279, 281, 284, 287, 292	theHvalue .....	9
\Glsaccessslongpl .....	thevalue .....	9
72, 184, 194, 202,	\glsaddstoragekey .....	43, 15
211, 215, 216, 219, 225–228, 233, 239,	\glsbackslash .....	47
240, 242, 247, 249, 257–261, 263, 270,	\glsacpscase .....	56, 61–72, 74, 185–197
271, 279, 281, 282, 284, 285, 287, 292, 293	\glsacategory ..	54, 61, 75, 82, 83, 154–156,
\glsaccessslongpl .....	159–166, 172, 175, 176, 185–190, 192, 193	
72, 184, 194,	\glsacategorylabel .....	
195, 202, 204, 207, 209, 210, 212, 214–	75, 179, 181, 182, 208, 230,	
216, 219, 221, 222, 224–230, 232, 233,	244, 266, 270, 272, 274, 283, 285, 289, 290	
235–244, 246, 247, 249, 251–267, 269,	\glsaclosebrace .....	41, 121–123
271, 273, 276, 279, 281, 284, 285, 287, 292	\glsacurrententrylabel .....	
\GlsAccesssname .....	52, 53, 110, 118, 127–130, 174, 175	
64	\glsacurrentfieldvalue .....	28–30, 32, 268
\Glsaccesssname .....	\glsacustomtext ..	55, 56, 69–72, 185–195, 197
64	\glsadefaulttype .....	6,
\glsaccesssname .....	16, 32, 33, 109, 110, 120, 128, 129, 131	
40, 64	\glsadescriptionaccesssdisplay ..	147, 160
\GlsAccesssplural .....	\glsadescriptionpluralaccesssdisplay	
63	147, 148	
\Glsaccesssplural .....	\glsadescwidth .....	351–358
63	\glsadetoklabel .....	8, 9, 29–32,
\glsaccesssplural .....	34, 38–40, 44–47, 58, 60, 76, 80, 92, 98–	
63	102, 107, 110, 113, 114, 127, 128, 132,	
\Glsaccesssshort .....	135–137, 159, 162, 163, 166, 167, 368, 369	
69, 189, 197,	\glsadisplaynumberlist .....	108, 113
204, 207, 209, 212, 214, 221, 222, 224,	\glsadohyperlink .....	78, 79, 81
229–232, 235, 237, 238, 243–246, 251,	\glsadohypertarget .....	81, 112
253–255, 265, 267, 273, 276, 284, 289, 290	\glsadoifexists .....	
\glsaccesssshort .....	14, 23, 31, 37, 39–41, 55, 56, 60,	
.. 69, 183, 189, 190, 197, 202, 204, 207,	68–72, 80, 107, 113, 114, 127, 128, 185–194	
209–216, 219, 220, 222–226, 228–231,	\glsadoifexistsordo .....	28, 29, 57
233, 235–240, 242–245, 247, 249, 251,	\glsadoifexistsorwarn .....	14, 159, 160, 172
252, 254–261, 263, 265, 267, 269, 271,	\glsadoifnoexists .....	33, 34
273, 276, 279, 281, 284, 287, 289, 290, 292	\glsadonohyperlink .....	59, 60, 80, 81
\GlsAccesssshortpl .....	\glsadosanitizesort .....	108
70, 192, 198, 204,	\glsasenableentrycount .....	91, 93, 101
207, 209, 212, 214, 221, 222, 224, 229–	\glsasenableentryunitcount .....	93, 102
232, 235, 237, 238, 244–246, 251, 253–	\glsasentrycounter .....	118
255, 265, 267, 273, 276, 284, 289, 290, 293	\glsasentrycurrcount .....	92, 93, 99
\glsaccesssshortpl .....	\Glsasentrydesc .....	147, 152, 160
.. 70, 184, 192, 193, 197, 202, 204, 207,	\glsasentrydesc .....	147, 152, 161
209–214, 216, 219, 221–226, 228–231,	\Glsasentrydescplural .....	148, 152
233, 235–240, 242–247, 249, 251, 252,		
254–261, 263, 265, 267, 269–271, 273,		
276, 279, 281, 284, 287, 289, 290, 292, 293		
\GlsAccessssymbol .....		
66		
\Glsaccessssymbol .....		
66, 172		
\glsaccessssymbol .....		
65, 172, 176		
\GlsAccessssymbolplural .....		
66		
\Glsaccessssymbolplural .....		
66		
\glsaccessssymbolplural .....		
66		
\GlsAccessstext .....		
61		

<code>\glsentrydescplural</code> .....	147, 148, 152	<code>\Glsentryuserv</code> .....	68
<code>\Glsentryfirst</code> .....	96, 140, 145, 151	<code>\glsentryuserv</code> .....	68
<code>\glsentryfirst</code> ...	96, 140, 145, 151, 309, 310	<code>\Glsentryuservi</code> .....	68
<code>\Glsentryfirstplural</code> ....	97, 141, 145, 151	<code>\glsentryuservi</code> .....	68
<code>\glsentryfirstplural</code> .....		<code>\glsextrapostnamehook</code> .....	165
.....	97, 140, 145, 146, 151, 310	<code>\glsfieldfetch</code> .....	80
<code>\glsentryfmt</code> .....	35–37	<code>\glsfieldxdef</code> .....	158, 159
<code>\Glsentryfull</code> .....	104	<code>\glsfindwidesttoplevelname</code> .....	366
<code>\glsentryfull</code> .....	104	<code>\GLSfirst</code> .....	301, 302
<code>\Glsentryfullpl</code> .....	104	<code>\Glsfirst</code> .....	302
<code>\glsentryfullpl</code> .....	104	<code>\glsfirst</code> .....	301, 302
<code>\glsentryitem</code> .....	127,	<code>\glsfirstabbrvdefaultfont</code> .....	
128, 348, 349, 351–358, 360–362, 374, 385		184, 202, 204, 206, 208, 210, 212, 215, 281	
<code>\Glsentrylong</code> .....	83, 84, 96, 140, 149, 153	<code>\glsfirstabbrvmfont</code> .....	247–267
<code>\glsentrylong</code> .....	83, 84, 96, 140, 149,	<code>\glsfirstabbrvfont</code> .....	104, 183,
153, 208, 230, 245, 266, 272, 274, 288, 311		184, 201–216, 219, 220, 222, 223, 225,	
<code>\Glsentrylongpl</code> .....	83, 84, 97, 150, 153	227, 229, 231, 233, 235, 236, 238, 240,	
<code>\glsentrylongpl</code> ....	83, 84, 97, 150, 153, 311	241, 243, 245, 247, 249, 250, 252, 254,	
<code>\Glsentrylongplural</code> .....	141	255, 257, 259, 261, 262, 265, 266, 269,	
<code>\glsentrylongplural</code> .....	140	270, 273, 276, 279, 281, 284, 287, 289, 292	
<code>\Glsentryname</code> .....	143, 150, 161, 163, 164	<code>\glsfirstabbrvhyphenfont</code> .....	
<code>\glsentryname</code> .....		.....	278, 279, 283, 284, 286–290
....	127, 143, 150, 168, 308, 366–373, 387	<code>\glsfirstabbrvonlyfont</code> .....	292, 293
<code>\glsentrynumberlist</code> ....	108, 114, 371–373	<code>\glsfirstabbrvscfont</code> .....	218–232
<code>\Glsentryplural</code> .....	144, 151	<code>\glsfirstabbrvsmfont</code> .....	233–246
<code>\glsentryplural</code> .....	144, 151, 309	<code>\glsfirstabbrvuserfont</code> .	269–273, 275, 276
<code>\glsentryprevcount</code> .....	92, 93, 99	<code>\glsfirstaccessdisplay</code> .....	145
<code>\glsentryprevmaxcount</code> .....	100	<code>\glsfirstlongdefaultfont</code> .....	
<code>\glsentryprevtotalcount</code> .....	100	202, 204, 210, 212, 215, 218–228, 232–	
<code>\Glsentryshort</code> .....	82, 84, 148, 152	242, 247, 248, 250, 251, 254–258, 261, 262	
<code>\glsentryshort</code> .....	82–	<code>\glsfirstlongemfont</code> .....	
84, 105, 148, 152, 270, 272, 283, 306, 307		....	248, 249, 252, 253, 259, 260, 262–264
<code>\Glsentryshortpl</code> .....	83, 84, 149, 153	<code>\glsfirstlongfont</code> 183, 184, 201–206, 209–	
<code>\glsentryshortpl</code> ..	83, 84, 149, 152, 153, 307	212, 214–217, 219, 220, 222, 223, 225,	
<code>\Glsentrysymbol</code> .....	146, 151	227, 229, 231, 233, 235, 236, 238, 240,	
<code>\glsentrysymbol</code> .....	146, 151, 370–372	241, 243, 245, 247, 249, 250, 252, 254,	
<code>\Glsentrysymbolplural</code> .....	147, 152	255, 257, 259, 261, 262, 265, 267, 269,	
<code>\glsentrysymbolplural</code> .....	146, 147, 152	270, 273, 276, 279, 281, 284, 287, 289, 292	
<code>\Glsentrytext</code> .....	144, 150	<code>\glsfirstlongfootnotefont</code> .....	
<code>\glsentrytext</code> 80, 143, 144, 150, 151, 308, 309		....	206–209, 229–232, 243–246, 264–267
<code>\glsentrytype</code> .....	127, 128	<code>\glsfirstlonghyphenfont</code> .....	278–289
<code>\Glsentryuseri</code> .....	67	<code>\glsfirstlongonlyfont</code> .....	291–293
<code>\glsentryuseri</code> .....	67	<code>\glsfirstlonguserfont</code> .....	269–276
<code>\Glsentryuserii</code> .....	67	<code>\GLSfirstplural</code> .....	302, 303
<code>\glsentryuserii</code> .....	67	<code>\Glsfirstplural</code> .....	303
<code>\Glsentryuseriii</code> .....	67	<code>\glsfirstplural</code> .....	302
<code>\glsentryuseriii</code> .....	67	<code>\glsfirstpluralaccessdisplay</code> ..	145, 146
<code>\Glsentryuseriv</code> .....	67	<code>\glsforeachincategory</code> .....	198
<code>\glsentryuseriv</code> .....	68	<code>\glsenentryfmt</code> .....	54

<code>\glsgetattribute</code> .....	59, 79, 93, 98, 99, 118, 137, 141, 159–162, 164, 166, 168
<code>\glsgetcategoryattribute</code> .....	154
<code>\glsgetgrouptitle</code> .....	350, 351, 360–363, 375–381
<code>\glsgetwidestname</code> .....	364
<code>\glsgroupheading</code> .....	..... 132, 349–358, 360–363, 374–380, 386
<code>\glsgroupskip</code> .....	132, 349, 351–361, 363, 375, 386
<code>\glshasattribute</code> .....	..... 59, 79, 93, 98, 100, 102, 118, 137, 141, 159–164, 166, 167, 202–206, 208, 211, 213, 214, 216–218, 220, 221, 229, 231, 233, 234, 236, 243, 245, 247– 253, 260, 264, 266, 269, 270, 272, 274– 276, 278–280, 282, 284–287, 289, 291–293
<code>\glshascategoryattribute</code> .....	155
<code>\glshex</code> ..	317–322, 325–330, 333–335, 338–346
<code>\glshyperlink</code> .....	80
<code>\glshypernavsep</code> .....	116
<code>\glshypernumber</code> .....	119, 167
<code>\glsifattribute</code> .....	57, 58, 62, 75, 77, 86, 141, 157, 160–163, 166, 174, 176, 177, 297–306
<code>\glsifcategory</code> .....	157
<code>\glsifcategoryattribute</code> .....	..... 75, 155, 156, 181, 182
<code>\glsifnotregular</code> .....	61
<code>\glsifnotregularcategory</code> .....	156
<code>\glsifplural</code> .....	.. 56, 61, 63–66, 68–72, 176, 177, 185–196
<code>\glsifregular</code> .....	54, 61, 96, 97, 140, 141
<code>\glsifregularcategory</code> .....	156
<code>\glsifusetranslator</code> .....	36
<code>\glsignore</code> .....	53
<code>\glsinlinedescformat</code> .....	359
<code>\glsinlinesubdescformat</code> .....	359
<code>\glsinsert</code> .....	56, 61, 69–72, 185–197, 277, 283–285, 289–291
<code>\glskeylisttok</code> .....	103, 104, 181, 183
<code>\glslabel</code> .....	8, 9, 28, 38, 40, 54, 57–60, 75, 76, 79, 80, 105, 137, 141, 142, 175, 176, 195–197, 208, 230, 245, 266, 270, 272, 274, 283–285, 289–291
<code>\glslabeltok</code> .....	. 103, 181, 183, 201–206, 208, 210–223, 225, 229, 231–237, 240, 243, 245, 247– 255, 257, 259, 260, 262, 264, 266, 269– 272, 274–280, 282, 284–287, 289, 291–293
<code>\glsletentryfield</code> .....	168
<code>\glslink</code> .....	104
<code>\glslink options</code>	
counter .....	9
format .....	167
hyper .....	294
hyperoutside .....	57, 58
noindex .....	8, 75, 294
theHvalue .....	59
thevalue .....	59, 134
wrgloss .....	8, 57
<code>\glslinkcheckfirsthyperhook</code> .....	75
<code>\glslinkpostsetkeys</code> .....	59, 137
<code>\glslinkpresetkeys</code> .....	58, 137
<code>\glslinkvar</code> .....	78
<code>\glslistchildpostlocation</code> .....	349
<code>\glslistchildprelocation</code> .....	349, 350
<code>\glslistdottedwidth</code> .....	348
<code>\glslistgroupheaderfmt</code> .....	350, 351
<code>\glslistnavigationitem</code> .....	350, 351
<code>\glslistprelocation</code> .....	349, 350
<code>\glslocalunset</code> .....	56, 137
<code>\glslongaccessdisplay</code> .....	149
<code>\glslongdefaultfont</code> .....	185, 202, 204, 205, 210, 212, 215, 219, 220, 222, 223, 225–227, 233, 235, 236, 238–242, 247, 250, 254, 255, 257, 261, 268, 278, 291
<code>\glslongemfont</code> .....	246, 248, 249, 252, 259, 262, 263
<code>\glslongfont</code> .....	83, 84, 185, 190, 191, 194, 195, 202–206, 209, 210, 212, 214, 215, 217, 219, 220, 222, 223, 225, 227, 229, 231, 233, 235, 236, 238, 240, 241, 243, 245, 247, 249, 250, 252, 254, 255, 257, 259, 261, 262, 265, 267, 269, 270, 273, 276, 279, 281, 284, 287, 289, 292, 293
<code>\glslongfootnotefont</code> .....	205, 206, 209, 229, 231, 243, 245, 265, 267
<code>\glslonghyphenfont</code> ..	278–284, 286, 287, 289
<code>\glslongonlyfont</code> .....	291, 292
<code>\glslongpltok</code> .....	..... 182, 183, 201, 203–206, 215, 217– 221, 225, 229, 233–235, 239, 243, 247– 253, 257, 259, 262, 264, 269–272, 274– 276, 278–280, 282, 283, 285–287, 291, 293
<code>\glslongpluralaccessdisplay</code> .....	150
<code>\glslongtok</code> .....	103, 104, 181, 183, 201, 203–206, 208, 210, 212, 214, 215, 217–223, 225, 228–230, 232– 237, 239, 240, 243, 244, 247–255, 257, 259, 262, 264, 266, 269–272, 274–276, 278–280, 282, 283, 285–287, 289, 291, 293

<code>\glslonguserfont</code> .....	268–276	<code>\glssetattribute</code> .....	202–206, 208, 210–218, 220–223, 225, 229, 231, 233, 234, 236, 237, 240, 243, 245, 247–255, 257, 259, 260, 262, 264, 266, 269, 270, 272, 274, 275, 277–280, 282, 284–287, 289, 291–293
<code>\glsmcols</code> .....	377–380	<code>\glssetcategoryattribute</code> .....	91, 103, 105, 136, 142, 154, 155, 157, 158, 173
<code>\GLSname</code> .....	299	<code>\glssetnoexpandfield</code> .....	12
<code>\Glsname</code> .....	299, 300	<code>\glssettoctitle</code> .....	110
<code>\glsname</code> .....	299	<code>\glsshortaccessdisplay</code> .....	148
<code>\glsnameaccessdisplay</code> ..	143, 161, 162, 164	<code>\glsshortpltok</code> .....	182, 183, 201, 203–206, 208, 210, 212, 218– 223, 229, 230, 233–237, 243, 244, 247– 253, 255, 264, 266, 269, 270, 272, 274– 276, 278, 279, 283, 285–288, 290, 291, 293
<code>\glsnamefont</code> .....	161, 163, 164, 166	<code>\glsshortpluralaccessdisplay</code> .....	149
<code>\glsnavhyperlink</code> .....	116	<code>\glsshorttok</code> .....	.... 103, 104, 181–183, 201, 203–206, 208–210, 212, 217–223, 225, 228, 230, 232–237, 239, 243, 244, 247–253, 255, 257, 259, 264, 266, 269–272, 274–276, 278, 279, 282, 283, 285–288, 290, 291, 293
<code>\glsnavhyperlinkname</code> .....	79	<code>\glssubentryitem</code> .....	.... 127, 128, 348–358, 360–362, 374, 385
<code>\glsnavhypertarget</code> .....	350, 351, 361–363, 376–381	<code>\glssymbolaccessdisplay</code> .....	146
<code>\glsnavigation</code> ..	350, 351, 361–363, 375–380	<code>\glssymbolpluralaccessdisplay</code> ..	146, 147
<code>\glsnextpages</code> .....	110	<code>\glstarget</code> .....	127, 128, 348–358, 360–362, 374, 375, 385, 386
<code>\glsnoidxdisplayloc</code> .....	113, 114	<code>\GLStext</code> .....	300
<code>\glsnoidxdisplayloclisthandler</code> ....	113	<code>\Glstext</code> .....	300
<code>\glsnoidxloclist</code> .....	114, 133	<code>\glstext</code> .....	300
<code>\glsnoidxnumberlistloophandler</code> ....	114	<code>\glstextaccessdisplay</code> .....	143, 144
<code>\glsnonnextpages</code> .....	110	<code>\glstextformat</code> .....	57, 59
<code>\glsnonnumberlistfalse</code> .....	52	<code>\glstextup</code> .....	218
<code>\glsnonnumberlisttrue</code> .....	52	<code>\glstreechildpredesc</code> .....	360, 361
<code>\glsnopostdotfalse</code> .....	111	<code>\glstreechildprelocation</code> ...	360, 361, 363
<code>\glsnopostdottrue</code> .....	111	<code>\glstreegroupheaderfmt</code> .....	360–363, 375–381, 383
<code>\glsnumberlistloop</code> .....	108	<code>\glstreeindent</code> .....	361, 362, 373–375
<code>\glsnumlistlastsep</code> .....	113	<code>\glstreeitem</code> .....	359, 377, 385
<code>\glsnumlistsep</code> .....	113	<code>\glstreenamebox</code> .....	374, 375
<code>\glsopenbrace</code> .....	41, 121–123	<code>\glstreenamefmt</code> ....	360–362, 364, 366–375
<code>\glsorder</code> .....	106	<code>\glstreenavigationfmt</code> ..	361–363, 375–380
<code>\glspagelistwidth</code> .....	352, 354, 356, 358	<code>\glstreepredesc</code> .....	360–362
<code>\glspar</code> .....	131	<code>\glstreeprelocation</code> .....	359–362, 364
<code>\GLSpl</code> .....	91, 102, 136	<code>\glstreesubitem</code> .....	359, 385
<code>\Glspl</code> .....	49, 91, 102, 136	<code>\glstreesubsubitem</code> .....	360, 386
<code>\glspl</code> .....	49, 91, 102, 136	<code>\GlsrLetField</code> .....	31
<code>\GLSplural</code> .....	301	<code>\glstype</code> .....	56, 58, 69–72, 137, 186–195
<code>\Glsplural</code> .....	301		
<code>\glsplural</code> .....	301		
<code>\glspluralaccessdisplay</code> .....	144		
<code>\glspluralsuffix</code> .....	125, 181, 185		
<code>\glspostdescription</code> .....	15, 16, 111, 174, 348–358, 360–364		
<code>\glspostinline</code> .....	359		
<code>\glspostlinkhook</code> ..	55–57, 69–72, 85, 186–195		
<code>\glsprestandardsort</code> .....	108		
<code>\glsresetentrylist</code> .....	116, 129		
<code>\glssee</code> .....	42–44		
<code>\glsseeformat</code> .....	40, 45, 107, 114		
<code>\glsseelist</code> .....	41		
<code>\glssetabbrvfmt</code> .....	54, 61, 82, 83, 159–164, 166, 172, 185–190, 192, 193, 195		

<code>\glsunset</code> .....	45, 56, 94, 95, 137	<code>\glsxtr@shortcutsval</code> .....	125
<code>\glswrite</code> .....	41, 106	<code>\glsxtr@texencoding</code> .....	125
<code>\glswriteentry</code> .....	8, 9	<code>\glsxtr@undefaction@nr</code> .....	7
<code>\Glsxtr</code> .....	49	<code>\glsxtr@undefaction@val</code> .....	7
<code>\glsxtr</code> .....	49	<code>\glsxtr@usesee</code> .....	39
<code>\glsxtr@@do@wrglossary</code> .....	8, 9, 11, 13	<code>\glsxtr@warnonexistsordo</code> .	7, 13, 14, 38, 39
<code>\glsxtr@addlocloclistfield</code> .....	13, 14	<code>\glsxtr@writefields</code> .....	123
<code>\glsxtr@addunused</code> .....	44	<code>\glsxtrabbrvfootnote</code> .....	
<code>\glsxtr@applyabbrvfmt</code> .....	195	....	206–208, 229, 230, 243–245, 264–266
<code>\glsxtr@applyabbrvstyle</code> ....	179, 181, 198	<code>\glsxtrabbrvpluralsuffix</code> .....	125,
<code>\glsxtr@counterrecord</code> .....	126	185, 202, 204, 206, 208, 210, 212, 215,	
<code>\glsxtr@do@alsoindex@wrglossary</code> ....	13	218, 232, 246, 268, 278, 280, 284, 289, 291	
<code>\glsxtr@doooption</code> .....	5, 15, 16, 22, 24	<code>\glsxtrabbrvtype</code> .....	17, 183
<code>\glsxtr@fields</code> .....	125	<code>\glsxtractivatenopost</code> .....	110
<code>\glsxtr@headentry@p</code> .....	86, 87	<code>\glsxtraddallcrossrefs</code> .....	44
<code>\glsxtr@hyperoutsidetrue</code> .....	58	<code>\glsxtralias</code> .....	76
<code>\glsxtr@hyperoutsidetrue</code> .....	57, 58	<code>\glsxtrAltTreeIndent</code> .....	364
<code>\glsxtr@ifnextpunc</code> .....	178	<code>\glsxtraltrtreeInit</code> .....	374, 379, 380
<code>\glsxtr@ifpunctoken</code> .....	178	<code>\glsxtrAltTreePar</code> .....	364
<code>\glsxtr@inc@linkcount</code> .....	58, 142	<code>\glsxtrAltTreeSetHangIndent</code> ...	364, 374
<code>\glsxtr@indexonly@saveentrycounter</code> .....	13, 14, 25	<code>\glsxtrAltTreeSetSubHangIndent</code> ....	375
<code>\glsxtr@keylist</code> .....	48, 49	<code>\glsxtraltrtreeSubSymbolDescLocation</code> ..	375
<code>\glsxtr@label</code> .....	387	<code>\glsxtraltrtreeSymbolDescLocation</code> ..	
<code>\glsxtr@langtag</code> .....	125	.....	364, 374
<code>\glsxtr@linkprefix</code> .....	125	<code>\glsxtrassignfieldfont</code> .....	61–68
<code>\glsxtr@loaddialect</code> .....	314, 346	<code>\glsxtrautoindex</code> .....	168
<code>\glsxtr@makeglossaries</code> .....	106	<code>\glsxtrautoindexassignsort</code> .....	168
<code>\glsxtr@newabbreviation</code> .....	104, 181	<code>\glsxtrautoindexentry</code> .....	168
<code>\glsxtr@next</code> .....	178	<code>\glsxtrbookindexatendgroup</code> .....	385
<code>\glsxtr@org@@do@wrglossary</code> .....	25	<code>\glsxtrbookindexatsubendgroup</code> ....	385
<code>\glsxtr@org@dohyperlink</code> .....	79	<code>\glsxtrbookindexatsubsubendgroup</code> ..	385
<code>\glsxtr@org@getgrouptitle</code> .....	115	<code>\glsxtrbookindexbetween</code> .....	385
<code>\glsxtr@org@newignoredglossary</code> ....	34	<code>\glsxtrbookindexbookmark</code> .....	386
<code>\glsxtr@org@makenoidxglossaries</code> ....	45	<code>\glsxtrbookindexcols</code> .....	384
<code>\glsxtr@pluralsuffixes</code> .....	125	<code>\glsxtrbookindexcolspread</code> .....	384
<code>\glsxtr@process</code> .....	129, 130	<code>\glsxtrbookindexfirstmarkfmt</code> .....	387
<code>\glsxtr@provideignoredglossary</code> ....	36	<code>\glsxtrbookindexformatheader</code> .....	386
<code>\glsxtr@punclist</code> .....	177, 178	<code>\glsxtrbookindexgroupskip</code> .....	386
<code>\glsxtr@record</code> .....	10, 125	<code>\glsxtrbookindexlastmarkfmt</code> .....	387
<code>\glsxtr@record@nr</code> .....	12, 13	<code>\glsxtrbookindexmulticolseenv</code> .....	384
<code>\glsxtr@recordsee</code> .....	11	<code>\glsxtrbookindexname</code> .....	382, 385
<code>\glsxtr@resource</code> .....	123, 125	<code>\glsxtrbookindexparentchildsep</code> ..	382, 384
<code>\glsxtr@s@newignoredglossary</code> .....	34	<code>\glsxtrbookindexparentschildsep</code> .	
<code>\glsxtr@s@provideignoredglossary</code> ...	36	.....	384, 385
<code>\glsxtr@saveentrycounter</code> ....	8, 9, 11, 76	<code>\glsxtrbookindexprelocation</code> ...	382, 385
<code>\glsxtr@setaccessdisplay</code> .....	166	<code>\glsxtrbookindexsubbetween</code> .....	385
<code>\glsxtr@setbookindexmark</code> .....	387	<code>\glsxtrbookindexsubname</code> .....	386
<code>\glsxtr@setup@record</code> .....	13, 24, 25	<code>\glsxtrbookindexsubprelocation</code> ....	386
		<code>\glsxtrbookindexsubsubbetween</code> ....	385

\glxtrbookindexthepage .....	387	\Glsxtrfullformat .....	184, 197, 199, 200, 202, 204,
\glxtrcat .....	48, 49		207, 209, 211, 213, 216, 219, 221, 223,
\glxtrchecknohyperfirst .....	62–64		224, 227–229, 231, 233, 235, 237, 238,
\glxtrcombiningdiacriticIIrules .	318		241–243, 245, 247, 249, 251, 252, 254,
\glxtrcombiningdiacriticIrules ..	318		256, 258, 260, 262, 263, 265, 267, 269,
\glxtrcombiningdiacriticIVrules ..	318		271, 273, 276, 279, 281, 284, 287, 289, 292
\glxtrComputeTreeIndent .....	374	\glxtrfullformat .....	184, 197, 199, 200, 202, 204, 207,
\glxtrComputeTreeSubIndent .....	374		209, 211, 213, 216, 219, 220, 223, 224,
\glxtrcurrencyrules .....	320		226, 228, 229, 231, 233, 235, 237, 238,
\Glsxtrdefaultsubsequentfmt ...	197, 199		241–243, 245, 247, 249, 251, 252, 254,
\glxtrdefaultsubsequentfmt ...	197, 199		256, 258, 260, 262, 263, 265, 267, 269,
\Glsxtrdefaultsubsequentplfmt .	198, 199		270, 273, 276, 279, 281, 284, 287, 289, 292
\glxtrdefaultsubsequentplfmt .	197, 199	\GLSxtrfullpl .....	18, 305, 306
\GlsXtrDefineAbbreviationShortcuts		\Glsxtrfullpl .....	17, 18, 306
.....	19, 20	\glxtrfullpl .....	17, 18, 305
\GlsXtrDefineAcShortcuts .....	20	\Glsxtrfullplformat .....	184, 196, 199, 200, 202, 204, 207,
\GlsXtrDefineOtherShortcuts .....	19, 20		209, 211, 213, 216, 219, 221, 223, 224,
\glxtrdetoklocation .....	136		227–229, 231, 233, 235, 237, 238, 241,
\glxtrdiscardperiod .....	175		242, 244, 245, 247, 249, 251, 253, 255,
\glxtrdisplayendloc .....	117		256, 258, 260, 262, 263, 265, 267, 270,
\glxtrdisplayendloohook .....	118		271, 273, 276, 279, 282, 284, 287, 290, 292
\glxtrdisplaysingleloc .....	117, 118	\glxtrfullplformat .....	196, 199, 200, 202, 204, 207,
\glxtrdisplaystartloc .....	117		209, 211, 213, 216, 219, 221, 223, 224,
\glxtrdoautoindexname .....	77, 78, 165		226, 228, 229, 231, 233, 235, 237, 238,
\glxtrdopostpunc .....	208, 230, 245, 266		241–243, 245, 247, 249, 251, 252, 254,
\glxtrdowrglossaryhook .....	77		256, 258, 260, 262, 263, 265, 267, 269,
\glxtremsuffix .....	247, 249, 250,		270, 273, 276, 279, 281, 284, 287, 289, 292
	252, 254, 255, 257, 259, 261, 262, 265, 266	\glxtrfullsep .....	183, 184, 201–
\GlsXtrEnableEntryCounting .....	103		205, 207, 209–212, 214–216, 218–228,
\GlsXtrEnableEntryUnitCounting .....	91		230–240, 242, 244, 246–261, 263, 265–
\GlsXtrEnableOnTheFly .....	47, 50		268, 278, 279, 281, 283, 286–288, 292, 293
\glxtrendfor .....	30	\glxtrgenabbrvfmt .....	54
\glxtrfieldlistgadd .....	126	\glxtrgeneralpuncIIrules .....	320
\glxtrfieldtitlecase .....	160–163, 166	\glxtrgeneralpuncIrules .....	320
\glxtrfieldtitlecasecs .....	159	\glxtrgetgroupitle .....	116, 386
\glxtrfieldxifinlist .....	131	\glxtrgroupfield .....	132
\glxtrfirstscfont .....	218	\Glsxtrheadfirst .....	296
\glxtrfirstsmfont .....	232	\glxtrheadfirst .....	296
\GlsXtrFmtDefaultOptions .....	28	\Glsxtrheadfirstplural .....	296
\glxtrfmtdisplay .....	28	\glxtrheadfirstplural .....	296
\GlsXtrFmtField .....	28, 29	\Glsxtrheadfull .....	296
\glxtrfootnotename .....		\glxtrheadfull .....	296
.....	206, 208, 228, 230, 243, 244, 264, 266	\Glsxtrheadfullpl .....	296
\GlsXtrFormatLocationList	52, 54, 371–373	\glxtrheadfullpl .....	296
\GLSxtrfull .....	18, 305, 306	\Glsxtrheadlong .....	296
\Glsxtrfull .....	17, 18, 306		
\glxtrfull .....	17, 18, 305		



<code>\glxtrheadlong</code> .....	296	207, 209, 210, 212, 214, 216, 222, 224–
<code>\Glsxtrheadlongpl</code> .....	296	226, 228, 230, 232, 237–240, 242, 244,
<code>\glxtrheadlongpl</code> .....	296	246, 254, 255, 257, 258, 260, 261, 263,
<code>\Glsxtrheadname</code> .....	296	265, 267, 271, 273, 281, 285, 290, 293, 313
<code>\glxtrheadname</code> .....	127, 296	<code>\glxtrinilinefullplformat</code> .....
<code>\Glsxtrheadplural</code> .....	296	..... 184, 187, 188, 199, 200,
<code>\glxtrheadplural</code> .....	296	207, 209, 210, 212, 214, 215, 222, 224–
<code>\Glsxtrheadshort</code> .....	296	226, 228, 230, 231, 236, 238–240, 242,
<code>\glxtrheadshort</code> .....	296	244, 246, 254–256, 258, 259, 261, 263,
<code>\Glsxtrheadshortpl</code> .....	296	265, 267, 271, 273, 281, 285, 290, 292, 312
<code>\glxtrheadshortpl</code> .....	296	<code>\glxtrininsertinsidefalse</code> .....
<code>\Glsxtrheadtext</code> .....	296	..... 201
<code>\glxtrheadtext</code> .....	296	<code>\glxstrLatinA</code> .....
<code>\glxtrhyperlink</code> .....	80, 118	..... 322–327
<code>\glxtrhyphensuffix</code> .....	279, 287	<code>\glxstrLatinAELigature</code> .....
<code>\glxtrifcounttrigger</code> .....	94, 95	..... 324, 326, 327
<code>\glxtrifcustomdiscardperiod</code> .....	175	<code>\glxstrLatinE</code> .....
<code>\glxtrifemptyglossary</code> .....	116, 122, 129	..... 322–327
<code>\glxtrifhasfield</code> .....	32, 76, 382	<code>\glxstrLatinEszettSs</code> .....
<code>\glxtrifhyphenstart</code> .....	..... 277, 280, 282, 283, 286, 288	..... 323, 325, 327
<code>\glxtrifindexing</code> .....	77	<code>\glxstrLatinEszettSz</code> .....
<code>\glxtrifinmark</code> .....	60, 86–89, 295–297	..... 324, 326
<code>\glxtrifnextpunc</code> .....	178	<code>\glxstrLatinEth</code> .....
<code>\glxtrifperiod</code> .....	175, 177	..... 323–326
<code>\glxtrifrecordtrigger</code> .....	138–140	<code>\glxstrLatinH</code> .....
<code>\glxtrifwasfirstuse</code> .	61–64, 68–72, 75,	..... 322–327
	105, 176, 186, 189–194, 208, 230, 231,	<code>\glxstrLatinI</code> .....
	245, 266, 270, 272, 274, 283, 285, 289, 290	..... 322–327
<code>\glxtrinclinkcounter</code> .....	142	<code>\glxstrLatinInsularG</code> .....
<code>\glxtrindexaliased</code> .....	76	..... 326
<code>\glxtrindexseealso</code> .....	43	<code>\glxstrLatinK</code> .....
<code>\glxtrinithyperoutside</code> .....	58	..... 322–327
<code>\glxtrinitwrgloss</code> .....	58, 137	<code>\glxstrLatinL</code> .....
<code>\glxtrinitwrglossbeforefalse</code> .....	57	..... 322–327
<code>\glxtrinitwrglossbeforetrue</code> .....	57	<code>\glxstrLatinM</code> .....
<code>\Glsxtrinilinefullformat</code> .....	184, 186, 199,	..... 322–327
	200, 207, 209, 210, 212, 214, 216, 222,	<code>\glxstrLatinN</code> .....
	224–226, 228, 230, 232, 237–240, 242,	..... 323–327
	244, 246, 254–256, 258, 259, 261, 263,	<code>\glxstrLatinO</code> .....
	265, 267, 271, 273, 281, 285, 290, 292, 312	..... 323–327
<code>\glxtrinilinefullformat</code> .....	..... 184, 185, 187, 199, 200,	<code>\glxstrLatinOELigature</code> .....
	207, 209, 210, 212, 214, 215, 222, 223,	..... 325, 327
	225–227, 230, 231, 236, 238–240, 242,	<code>\glxstrLatinP</code> .....
	244, 245, 254–256, 258, 259, 261, 263,	..... 323–327
	265, 267, 271, 273, 281, 284, 290, 292, 312	<code>\glxstrLatinS</code> .....
<code>\Glsxtrinilinefullplformat</code> .....	..... 184, 188, 199, 200,	..... 323–327
	..... 203, 219, 234, 248, 249, 279, 285	<code>\glxstrLatinT</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 323–327
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrLatinThorn</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 327
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrLatinX</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 323–328
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlocationhyperlink</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 118
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlocrangefmt</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 117, 118
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\GLSxtrlong</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 17, 18, 303, 304
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\Glsxtrlong</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 17, 18, 304
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlong</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 17, 18, 303
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlonghyphen</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 284
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlonghyphennoshort</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 281, 282
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlonghyphenshort</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 279
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlongnoshortdescname</code> .
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 215, 262, 280
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlongnoshortname</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 217, 225, 239, 257, 259, 282
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\GLSxtrlongpl</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 18, 303, 304
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\Glsxtrlongpl</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 17, 18, 304, 305
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlongpl</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 17, 18, 304
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlongshortdescname</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 203, 219, 234, 248, 249, 279, 285
	..... 203, 219, 234, 248, 249, 274, 279, 285	<code>\glxstrlongshortdescsort</code> .....
	..... 203, 219, 234, 248, 249, 274, 279, 285	..... 203, 219, 234, 248, 249, 274, 279, 285

<code>\glxtrlongshortname</code> .....	<code>\glxtrorgshort</code> .....
201, 218, 232, 247, 248, 269, 270, 278, 283	181, 202
<code>\glxtrlongshortuserdescname</code> ..	<code>\GLSxtrp</code> .....
271, 274	86
<code>\glxtrmarkhook</code> .....	<code>\Glsxtrp</code> .....
294, 295	86
<code>\glxtrMathItalicAlpha</code> .....	<code>\glxtrp</code> .....
332, 335, 337	85, 87
<code>\glxtrMathItalicBeta</code> .....	<code>\glxtrparen</code> .....
332, 335, 337	.... 176, 183, 184, 201–205, 207, 209–
<code>\glxtrMathItalicChi</code> ...	212, 214–216, 218–222, 224–226, 228,
332, 333, 336, 337	230–240, 242, 244, 246–261, 263, 265–
<code>\glxtrMathItalicDelta</code> .....	268, 278, 279, 281, 283, 286–288, 292, 293
332, 336, 337	<code>\Glsxtrpl</code> .....
<code>\glxtrMathItalicEpsilon</code> ...	49
332, 336, 337	<code>\glxtrpl</code> .....
<code>\glxtrMathItalicEta</code> .....	49
332, 336, 337	<code>\glxtrpostdescription</code> .
<code>\glxtrMathItalicGamma</code> .....	111, 157, 174, 359
332, 336, 337	<code>\glxtrposthyphenlong</code> .....
<code>\glxtrMathItalicIota</code> .....	289, 290
332, 336, 337	<code>\glxtrposthyphenshort</code> .....
<code>\glxtrMathItalicKappa</code> .....	283, 285
332, 336, 337	<code>\glxtrposthyphensubsequent</code> .....
<code>\glxtrMathItalicLambda</code> ....	..... 284, 285, 289, 291
332, 336, 337	<code>\glxtrpostlink</code> .....
<code>\glxtrMathItalicMu</code> .....	175
332, 336, 337	<code>\glxtrpostlinkendsentence</code> .....
<code>\glxtrMathItalicNu</code> .....	175
332, 336, 337	<code>\glxtrpostlinkhook</code> .....
<code>\glxtrMathItalicOmega</code> .	175
332, 333, 336, 337	<code>\glxtrpostlocalreset</code> .....
<code>\glxtrMathItalicOmicron</code> ...	90, 92, 100
332, 336, 337	<code>\glxtrpostlocalunset</code> .....
<code>\glxtrMathItalicPhi</code> ...	90, 92, 100
332, 333, 336, 337	<code>\glxtrpostlongdescription</code> .....
<code>\glxtrMathItalicPi</code> ....	34
332, 333, 336, 337	<code>\glxtrpostnamehook</code> .....
<code>\glxtrMathItalicPsi</code> ...	162–164, 167
332, 333, 336, 337	<code>\GlsXtrPostNewAbbreviation</code> .....
<code>\glxtrMathItalicSigma</code> .	..... 183, 199, 202–
332, 333, 336, 337	206, 208, 210–218, 220–223, 225, 229,
<code>\glxtrMathItalicTau</code> ...	230, 233, 234, 236, 237, 240, 243, 244,
332, 333, 336, 337	247, 248, 250–255, 257, 259, 260, 262,
<code>\glxtrMathItalicTheta</code> .....	264, 266, 269, 270, 272, 274–276, 278–
332, 336, 337	280, 282, 283, 285–287, 289, 290, 292, 293
<code>\glxtrMathItalicUpsilon</code> 332, 333, 336, 337	<code>\glxtrpostreset</code> .....
<code>\glxtrMathItalicXi</code> .....	90, 92, 100
332, 336, 337	<code>\glxtrpostunset</code> .....
<code>\glxtrMathItalicZeta</code> .....	90, 92, 100
332, 336, 337	<code>\glxtrprelocation</code> .....
<code>\glxtrnewabbrevpresetkeyhook</code> .....	..... 349, 351, 353, 355, 357, 359, 382
182	<code>\glxtrprotectlinks</code> .....
<code>\glxtrnewnumber</code> .....	79–81
19	<code>\GlsXtrRecordCounter</code> .....
<code>\glxtrnewsymbol</code> .....	11
19	<code>\glxtrrecordtriggervalue</code> .....
<code>\glxtrNoGlossaryWarning</code> .....	137
20, 21, 119	<code>\glxtrregularfont</code> .....
<code>\GlsXtrNoGlsWarningAutoMake</code> .....	54, 61
122	<code>\glxtrresourcecount</code> .....
<code>\GlsXtrNoGlsWarningBuildInfo</code> .....	124
123	<code>\glxtrresourcefile</code> .....
<code>\GlsXtrNoGlsWarningCheckFile</code> .....	124
122	<code>\glxtrresourceinit</code> .....
<code>\GlsXtrNoGlsWarningEmptyMain</code> ..	123
122, 123	<code>\glxtrrestoremarkhook</code> .....
<code>\GlsXtrNoGlsWarningEmptyNotMain</code> ...	294, 295
122	<code>\glxtrrestorepostpunc</code> .....
<code>\GlsXtrNoGlsWarningEmptyStart</code> .....	111
122	<code>\glxtrtrscfont</code> .....
<code>\GlsXtrNoGlsWarningEmptyTail</code> .....	218
122	<code>\glxtrtrscsuffix</code> .....
<code>\GlsXtrNoGlsWarningHead</code> .....	.... 219, 220, 222, 223, 225, 227, 229, 231
122	<code>\GlsXtrSetActualChar</code> .....
<code>\GlsXtrNoGlsWarningMisMatch</code> .....	171
123	<code>\glxtrsetaliasnoindex</code> .....
<code>\GlsXtrNoGlsWarningNoOut</code> .....	13, 14, 76
123	<code>\GlsXtrSetEncapChar</code> .....
<code>\GlsXtrNoGlsWarningTail</code> .....	171
123	
<code>\glxtrnopostpunc</code> .....	
111	
<code>\glxtronlydescname</code> .....	
293	
<code>\glxtronlydescsort</code> .....	
293	
<code>\glxtronlyname</code> .....	
291	
<code>\glxtronlysuffix</code> .....	
292	
<code>\glxtrorg@ifKV@glslink@hyper</code> .....	
55	
<code>\glxtrorglong</code> .....	
181, 202, 280	



<code>\GlsXtrSetEscChar</code> .....	171	<code>\glxtrrtitlelong</code> .....	296, 297, 311
<code>\glxtrsetfieldifexists</code> .....	31, 32	<code>\Glsxtrrtitlelongpl</code> .....	296, 297, 311
<code>\GlsXtrSetLevelChar</code> .....	171	<code>\glxtrrtitlelongpl</code> .....	296, 297, 311
<code>\glxtrsetpopts</code> .....	85	<code>\Glsxtrrtitlename</code> .....	296, 297, 308
<code>\glxtrsetupfulldefs</code> .....		<code>\glxtrrtitlename</code> .....	296, 297, 308
.....	186–188, 208, 231, 245, 266	<code>\glxtrrtitleorpdforheading</code> .....	
<code>\GLSxtrshort</code> .....	17, 18, 89, 297, 298	.....	23, 127, 128, 295–297
<code>\Glsxtrshort</code> .....	17, 18, 298	<code>\Glsxtrrtitleplural</code> .....	296, 297, 309
<code>\glxtrshort</code> .....	17, 18, 297	<code>\glxtrrtitleplural</code> .....	296, 297, 309
<code>\glxtrshortdescname</code> ...	212, 223, 237, 255	<code>\Glsxtrrtitleshort</code> .....	295–297, 307
<code>\glxtrshorthyphen</code> .....	289, 290	<code>\glxtrrtitleshort</code> .....	295–297, 306, 307
<code>\glxtrshorthyphenlong</code> .....	287	<code>\Glsxtrrtitleshortpl</code> .....	295–297, 307
<code>\glxtrshortlongdescname</code> .....		<code>\glxtrrtitleshortpl</code> .....	295–297, 307
.....	205, 221, 235, 251, 253, 287, 290	<code>\Glsxtrrtitletext</code> .....	296, 297, 309
<code>\glxtrshortlongdescsort</code> .....		<code>\glxtrrtitletext</code> .....	296, 297, 308
....	205, 221, 235, 251, 253, 276, 287, 290	<code>\GlsXtrTotalRecordCount</code> .....	136
<code>\glxtrshortlongname</code> .....		<code>\glxtrtreetopindent</code> .....	364, 373
.....	203, 220, 234, 250, 252, 272, 275, 286, 288	<code>\glxtrundefaction</code> .....	
<code>\glxtrshortlonguserdescname</code> ..	274, 276	.....	7, 13, 14, 26, 34, 35, 37–39
<code>\glxtrshortnolongname</code> .	210, 222, 236, 253	<code>\glxtrundeftag</code> .....	25, 113, 114
<code>\GLSxtrshortpl</code> .....	17, 18, 298, 299	<code>\glxtrunsrtdo</code> .....	130, 131
<code>\Glsxtrshortpl</code> .....	17, 18, 299	<code>\glxtrUpAlpha</code> .....	330, 331, 335, 337
<code>\glxtrshortpl</code> .....	17, 18, 298	<code>\glxtrUpBeta</code> .....	330, 331, 335, 337
<code>\glxtrsmfont</code> .....	232	<code>\glxtrUpChi</code> .....	331, 336, 337
<code>\glxtrsmsuffix</code> .....		<code>\glxtrUpDelta</code> .....	330, 331, 336, 337
....	233, 235, 236, 238, 240, 241, 243, 245	<code>\glxtrUpDigamma</code> .....	331, 332, 336
<code>\Glsxtrsubsequentfmt</code> .....		<code>\glxtrUpEpsilon</code> .....	331, 336, 337
.....	196, 199, 215, 226, 227,	<code>\glxtrUpEta</code> .....	331, 336, 337
.....	240, 241, 257, 259, 261, 263, 281, 284, 289	<code>\glxtrUpGamma</code> .....	330, 331, 336, 337
<code>\glxtrsubsequentfmt</code> .....		<code>\glxtrUpIota</code> .....	331, 336, 337
.....	196, 199, 215, 226, 227,	<code>\glxtrUpKappa</code> .....	331, 336, 337
.....	240, 241, 257, 259, 261, 262, 281, 284, 289	<code>\glxtrUpLambda</code> .....	331, 336, 337
<code>\Glsxtrsubsequentplfmt</code> .....		<code>\glxtrUpMu</code> .....	331, 336, 337
.....	195, 199, 215, 226, 227,	<code>\glxtrUpNu</code> .....	331, 336, 337
.....	240, 242, 257, 259, 261, 263, 281, 284, 289	<code>\glxtrUpOmega</code> .....	331, 336, 337
<code>\glxtrsubsequentplfmt</code> .....		<code>\glxtrUpOmicron</code> .....	331, 336, 337
.....	195, 196, 199, 215, 226, 227,	<code>\glxtrUpPhi</code> .....	331, 336, 337
.....	240, 241, 257, 259, 261, 263, 281, 284, 289	<code>\glxtrUpPi</code> .....	331, 336, 337
<code>\glxtrsupplocationurl</code> .....	118, 119	<code>\glxtrUpPsi</code> .....	331, 336, 337
<code>\glxtrtagfont</code> .....	174	<code>\glxtrUpRho</code> .....	331, 336, 337
<code>\Glsxtrrtitlefirst</code> .....	296, 297, 310	<code>\glxtrUpSigma</code> .....	331, 336, 337
<code>\glxtrrtitlefirst</code> .....	296, 297, 309	<code>\glxtrUpTau</code> .....	331, 336, 337
<code>\Glsxtrrtitlefirstplural</code> ....	296, 297, 310	<code>\glxtrUpTheta</code> .....	331, 336, 337
<code>\glxtrrtitlefirstplural</code> ....	296, 297, 310	<code>\glxtrUpUpsilon</code> .....	331, 336, 337
<code>\Glsxtrrtitlefull</code> .....	296, 297, 312	<code>\glxtrUpXi</code> .....	331, 336, 337
<code>\glxtrrtitlefull</code> .....	296, 297, 312	<code>\glxtrUpZeta</code> .....	331, 336, 337
<code>\Glsxtrrtitlefullpl</code> .....	296, 297, 313	<code>\GlsXtrUseAbbrStyleFmts</code> .....	
<code>\glxtrrtitlefullpl</code> .....	296, 297, 312	.....	203, 205, 211, 213, 214,
<code>\Glsxtrrtitlelong</code> .....	296, 297, 311	.....	217, 218, 220, 221, 224, 234, 236, 239,

248, 250, 252, 253, 256, 260, 264, 272, 274, 275, 277, 280, 282, 286, 288, 291, 293	\ifdefempty .....	7–9, 30, 35–37, 39, 40, 59, 60, 91, 103, 106, 109, 117, 129, 132, 136, 137, 173, 180, 195, 384
\GlsXtrUseAbbrStyleSetup .....	\ifdefequal .....	45, 123, 132, 165
..... 211, 213, 214, 216, 217, 224, 227, 239, 241, 256, 260, 261, 264	\ifdefstring .....	6, 32, 168, 173, 383
\glsxtruserfield .....	\ifdefvoid .....	.....
\glsxtruserparen .....	..... 39, 42–44, 80, 97, 114, 115, 118, 132, 133	
\glsxtrusersuffix .....	\ifdim .....	51, 105, 365–373
\glsxtruseseealsoformat .....	\IfFileExists .....	21, 119, 122, 123, 126, 346, 347
\glsxtruseseeformat .....	\ifglossaryexists .....	38, 39
\GlsXtrWarnDeprecatedAbbrStyle .....	\ifglssacronym .....	17, 122
\GlsXtrWarning .....	\ifglssacrshortcuts .....	19
\glsxtrword .....	\ifglssautomake .....	109, 122, 126
\glsxtrwordsep .....	\ifglssentrycounter .....	32
\glsxtrwrglossmark .....	\ifglssentryexists .....	9, 37, 38, 48, 49, 52, 61, 132, 155, 174, 175
	\ifglssfieldeq .....	153
	\ifglsshasfield .....	28, 29, 268
	\ifglsshaslong .....	96, 97, 140, 141
	\ifglsshasparent .....	127–129, 132, 366, 368, 369
	\ifglsshasshort .....	40, 54, 61
	\ifglsshasymbol .....	176, 360–362, 364
	\ifglssindexonlyfirst .....	77
	\ifglssnogroupskip .....	349, 351–361, 363, 375, 383
	\ifglssnonumberlist .....	54
	\ifglssnopostdot .....	15, 16, 111
	\ifglsssanitizesort .....	109
	\ifglsssubentrycounter .....	32
	\ifglssused .....	44, 45, 75, 77, 93, 102, 105, 195, 366–368, 370–372
	\ifglssxindy .....	119, 121
	\ifglssxtr@hyperoutside .....	59
	\ifglssxtrtrinitwrglossbefore .....	57, 59, 60, 137
	\ifglssxtrininsertinside .....	189–195, 197, 198, 202, 204, 207, 209–216, 219–233, 235–247, 249, 251–267, 269–271, 273, 276, 278, 280, 283–286, 288, 290, 292, 293
	\ifHy@hyperindex .....	167
	\ifinlistcs .....	30, 46
	\ifinner .....	23
	\ifKV@glsslink@hyper .....	55, 58–60
	\ifKV@glsslink@local .....	56, 137
	\ifKV@glsslink@noindex .....	8, 9, 11, 28, 76, 77
	\ifmmode .....	23
	\ifnum .....	14, 93, 101, 102, 115, 124, 137, 361, 362, 374
	\ifstreempty .....	127, 134, 142
	\ifstrequal .....	16, 21
	\ifthenelse .....	122, 123

## H

\hangindent .....	361, 362, 364, 373–375, 379, 380
\hbox .....	348
\hfill .....	348
\href .....	79
\hsize .....	51
\hss .....	348
\hyperlink .....	80
\hyperpage .....	167
\hyperref .....	79, 118
hyperref package .....	81, 167, 294, 306

## I

\if .....	47
\if@glssxtr@autoseeindex .....	24, 39, 42
\if@glssxtr@format@override .....	168
\if@glssxtr@docdefrestricted .....	45
\if@glssxtr@index@crossrefs .....	15, 44
\ifblank .....	26, 27, 48, 49, 106
\ifcase .....	7, 13, 19, 20, 22, 46, 57, 112, 360, 385
\ifcsdef .....	26, 33–37, 59, 73, 74, 85–89, 97, 110, 115, 116, 127, 131, 134–136, 141, 159– 162, 164–166, 176, 179, 195, 199, 351–358
\ifcsstring .....	26, 155, 198
\ifcsundef .....	31, 33, 35–37, 46, 50, 52, 81, 92, 97–101, 115, 116, 119, 131, 142, 155, 198–201, 348, 365, 366, 373, 387
\ifcsvoid .....	43, 154
\ifdef .....	13, 18, 19, 24, 28, 38, 41, 42, 50, 51, 75, 79, 80, 86–88, 109, 113, 114, 125, 134, 157, 158, 171, 174, 268, 295, 306–312, 346, 348–351, 359–363, 375–380, 383, 386, 387

<code>\IfTrackedLanguageFileExists</code> .....	313	<code>\makeatletter</code> .....	119, 123, 170
<code>\ifundef</code> .....	30, 106, 173, 174, 346	<code>\makeatother</code> .....	170
<code>\ifx</code> ...	8–10, 41, 50, 52, 106, 110, 117, 118, 126, 168, 169, 171, 178, 180, 182, 277, 381	<code>\makebox</code> .....	348, 374, 375
<code>\immediate</code> .....	93, 101, 102, 119, 126	<code>\makefirstuc</code> .....	173
<code>\index</code> .....	168	<code>makeglossaries</code> .....	112
<code>\indexspace</code> .	349, 360–363, 375–381, 383, 386	<code>\makeglossaries</code> .....	106, 120–123, 126
<code>\input</code> .....	313	<code>\makeglossary</code> .....	107
<code>\inputencodingname</code> .....	125	<code>makeindex</code> .....	388
<code>\istfilename</code> .....	106	<code>makeindex</code> .....	13, 106
<code>\item</code> ....	121, 122, 348–351, 359–361, 376, 377	<code>\makenoidxglossaries</code> .....	121
<b>J</b>			
<code>\jobname</code> .....	119, 121–124, 126	<code>\MakeTextUppercase</code> .....	296
<b>K</b>			
<code>\key@ifundefined</code> .	11, 12, 26, 27, 73, 129, 132	<code>\MakeUppercase</code> .....	295–297
<code>\KV@glslink@hyperfalse</code> .....	62, 75, 80, 81	<code>\marginpar</code> .....	23
<code>\KV@glslink@hypertrue</code> .....	81	<code>\markboth</code> .....	295
<code>\KV@glslink@noindexfalse</code> .....	75, 76	<code>\markright</code> .....	295
<code>\KV@glslink@noindextrue</code> .....	76, 81	<code>\mathit</code> .....	315
<b>L</b>			
<code>\L</code> .....	327, 330	<code>\mathrm</code> .....	314–316
<code>\l</code> .....	330	<code>\maxdimen</code> .....	51
<code>\LaTeX</code> .....	121, 122	<code>\mbox</code> .....	350, 351, 374
<code>\leaders</code> .....	348	<code>\medskip</code> .....	122, 123, 131
<code>\leavevmode</code> .....	34, 58	<code>\MessageBreak</code> 46, 50, 93, 102, 106, 109, 110, 198	
<code>\let</code> 5, 7–11, 13–15, 17–19, 23–25, 28, 30, 33, 34, 45–47, 50–53, 55–72, 74–76, 78–82, 85, 91–93, 100, 101, 103–107, 109–116, 123–126, 129, 130, 132, 137, 142, 160, 161, 163–169, 173, 174, 178–182, 185– 195, 197–199, 208, 231, 245, 266, 294– 297, 346, 359, 360, 364, 366, 377, 384–387		<code>mfirstuc package</code> .....	173
<code>\letabbreviationstyle</code> ..	208, 209, 211, 213, 216, 217, 223, 224, 237, 239, 255, 256	<code>\mfirstucMakeUppercase</code> .	61–72, 74, 83, 84, 86, 89, 96, 104, 141, 143–153, 162, 163, 166, 187, 188, 190, 191, 193, 195–197
<code>\letcs</code> .....	26, 30, 39, 40, 44, 59, 73, 113–115, 131, 132, 159–167, 387	<code>\mfu@checkword@arg</code> .....	173
<code>\levelchar</code> .....	171	<code>\mfu@checkword@do</code> .....	173
<code>\listadd</code> .....	97	<b>N</b>	
<code>\listbreak</code> .....	173	<code>\NeedsTeXFormat</code> .....	5, 314, 347, 382
<code>\listcsadd</code> .....	29	<code>\new@glossaryentry</code> .....	46, 109
<code>\listcseadd</code> .....	29, 98	<code>\new@ifnextchar</code> .....	27, 73, 74, 96, 134, 138–140, 177, 185–194
<code>\listcsgadd</code> .....	29, 46	<code>\newabbr</code> .....	18
<code>\listcsxadd</code> .....	29, 98	<code>\newabbreviation</code> .....	18
<code>\loadglsentries</code> .....	46, 120	<code>\newabbreviationhook</code> .....	182
<code>\long</code> .....	33, 34	<code>\newabbreviationstyle</code> .....	201, 203, 205, 206, 208, 210–225, 227, 228, 230, 232–237, 239, 241, 243, 244, 246, 248–253, 255–258, 260, 262, 264, 266, 269–272, 274–276, 278–280, 282, 283, 285–288, 290, 291, 293
<b>M</b>			
<code>\MakeAcronymsAbbreviations</code> .....	105	<code>\newacronym</code> .....	103, 104
		<code>\newacronymhook</code> .....	103
		<code>\newacronymstyle</code> .....	104, 105
		<code>\newcommand</code> .....	5–8, 10– 12, 14–40, 42–45, 47–50, 52–55, 57, 58, 61, 62, 73, 74, 76–78, 80, 81, 84–93, 96– 106, 110–115, 117, 118, 120–124, 126– 132, 134–159, 165, 167–181, 183–195,

197–203, 205, 206, 209, 212, 214, 217, 218, 232, 246, 268, 271, 274, 277, 278, 280, 282, 283, 286, 288, 291, 293, 295, 297–314, 317–336, 338–345, 347, 349, 359, 363–366, 373, 374, 382, 383, 386, 387	\nopostdesc ..... 34, 48, 49, 111, 157
\newcount ..... 124, 133	\ns@GLSxtrfull ..... 186
\newcounter ..... 141	\ns@Glsxtrfull ..... 186
\newentry ..... 18	\ns@glxtrfull ..... 185
\newglossary ..... 17, 107	\ns@GLSxtrfullpl ..... 188
\newglossaryentry ..... ..... 18, 46, 91, 99, 103, 157, 158, 183	\ns@Glsxtrfullpl ..... 187
\newglossaryentry options	\ns@glxtrfullpl ..... 187
alias ..... 15, 39, 42–44	\ns@GLSxtrlong ..... 191
desc ..... 147, 152	\ns@Glsxtrlong ..... 190
descplural ..... 147, 148, 152	\ns@glxtrlong ..... 190
first 79, 144, 145, 151, 201, 301–303, 309, 388	\ns@GLSxtrlongpl ..... 194
firstplural ..... 145, 151, 201, 302, 310, 388	\ns@Glsxtrlongpl ..... 194
group ..... 132	\ns@glxtrlongpl ..... 193
loclist ..... 29	\ns@GLSxtrshort ..... 189
long ..... 149, 153, 310	\ns@Glsxtrshort ..... 189
longplural ..... 150, 153, 311	\ns@glxtrshort ..... 188
name ..... 40, 143, 150, 168, 299, 300, 308	\ns@GLSxtrshortpl ..... 193
plural ..... 144, 151, 201, 300, 301, 309	\ns@Glsxtrshortpl ..... 192
see ..... 15, 24, 39, 42, 44, 46, 107	\ns@glxtrshortpl ..... 192
seealso ..... 15, 39, 40, 42–44, 399	\null ..... 20
short ..... 148, 152, 179	\number ..... 98–101, 123, 134
shortplural ..... 149, 153, 179	\numexpr ..... 98, 99, 101
symbol ..... 146, 151	
symbolplural ..... 146, 147, 152	<b>O</b>
text 79, 143, 144, 150, 151, 201, 203, 300, 308	\O ..... 327, 330
\newglossarystyle ..... 384	\o ..... 330
\newif ..... 57, 167, 201	\or ..... 7, 13, 19, 20, 22, 46, 57, 360, 385
\newlength ..... 364	\org@glossaryentrynumbers ..... 52, 110
\newnum ..... 19	\org@glossarytitle ..... 110
\newrobustcmd ..... 27, 28, 30–32, 41, 42, 73, 74, 85, 86, 95, 96, 111, 115, 127, 128, 130, 131, 134, 135, 138–140, 166, 173, 174, 185–195, 277, 295, 297–306, 366–372	\org@ifKV@glslink@hyper ..... 58, 60
\newsym ..... 19	
\newterm ..... 157	<b>P</b>
\newtoks ..... 179	\p@glS@hyp@opt ..... 78
\newwrite ..... 106	package options:
\nobreak ..... 350, 351, 360–363, 376–379	abbreviations ..... 16, 17
\NoCaseChange ..... 86–89, 128, 297–306	accsupp ..... 20, 143
\noexpand .. 10, 11, 21, 41, 43, 44, 103, 104, 119, 123, 130–132, 141, 169, 183, 347, 385	acronym ..... 17
\nofiles ..... 122	automake ..... 109, 121, 125
\noindent ..... 122, 123, 362, 363, 377–380	true ..... 126
\nopagebreak ..... 360–363, 375–382, 386	autoseeindex ..... 24
	false ..... 24
	debug
	showtargets ..... 80
	docdef ..... 14, 45, 46, 91, 99
	false ..... 46
	restricted ..... 14
	true ..... 46
	docdefs
	restricted ..... 45
	nonumberlist ..... 52



<code>\rGLS</code> .....	136	<code>\subitem</code> .....	359, 360
<code>\rGls</code> .....	136	<code>\subsubitem</code> .....	360
<code>\rgls</code> .....	136		
<code>\rGLSformat</code> .....	139		
<code>\rGlsformat</code> .....	139		
<code>\rglsformat</code> .....	138, 141		
<code>\rGLSpl</code> .....	136		
<code>\rGlspl</code> .....	136		
<code>\rglspl</code> .....	136		
<code>\rGLSplformat</code> .....	140		
<code>\rGlsplformat</code> .....	139		
<code>\rglsplformat</code> .....	138, 141		
<code>\romannumeral</code> .....	364–366, 373		
	<b>S</b>		
<code>\s@glxtrfmt</code> .....	27		
<code>\s@glshyp@opt</code> .....	78		
<code>\s@glxtr@enabletagging</code> .....	172		
<code>\s@glxtrfmt</code> .....	27		
<code>\s@glxtrifhasfield</code> .....	30		
<code>\s@printunsrtglossary</code> .....	128, 131		
<code>\seealsoname</code> .....	40, 42		
<code>\seename</code> .....	40		
<code>\setabbreviationstyle</code> .....	105, 202, 211		
<code>\setacronymstyle</code> .....	104, 105		
<code>\setentrycounter</code> .....	117		
<code>\SetGenericNewAcronym</code> .....	105		
<code>\setglossarystyle</code> .....	22,		
	110, 348–351, 360, 362, 363, 375–381, 384		
<code>\setkeys</code> .....	9,		
	22, 25, 28, 59, 60, 77, 103, 110, 137, 181, 182		
<code>\setlength</code> .....	51,		
	359, 361, 362, 364, 374, 375, 377–379, 384		
<code>\settowidth</code> .....	105, 364–373		
<code>\setupglossaries</code> .....	5, 24		
<code>\sfcode</code> .....	15, 16, 176, 359		
<code>\small</code> .....	23		
<code>\space</code> .....	6, 11, 41, 46, 47, 50,		
	76, 91, 93, 99, 101–103, 105–108, 110,		
	120, 122, 126, 130, 131, 134, 176, 180,		
	184, 202, 348, 349, 359–362, 364, 373, 382		
<code>\spacefactor</code> .....	15, 16, 176, 182, 359		
<code>\stepcounter</code> .....	142		
<code>\string</code> 6, 10, 11, 41, 45–47, 50, 53, 54, 59, 73,			
	74, 76, 85, 86, 91, 93, 99, 101–103, 105–		
	110, 119–123, 125, 126, 130, 131, 134,		
	161, 163, 164, 166, 168, 314, 317–346, 387		
<code>\strut</code> .....	348–358, 362		
<code>\subglossentry</code> .....			
	.... 110, 133, 348–358, 360–362, 374, 385		
		<b>T</b>	
		<code>\tablehead</code> .....	355–358
		<code>\tabletail</code> .....	355–358
		<code>\tabularnewline</code> .....	351–359
		<code>\TeX</code> .....	121
		<code>\texorpdfstring</code> ....	28, 86–89, 295, 306–312
		textcase package .....	294
		<code>\textsc</code> .....	218
		<code>\textsmaller</code> .....	232
		<code>\texttt</code> .....	23, 120–123
		<code>\the</code> .....	103, 104, 118, 124, 171, 183,
			201–206, 208–223, 225, 228–237, 239,
			240, 243–245, 247–255, 257, 259, 260,
			262, 264, 266, 269–272, 274–280, 282–293
		<code>\theglentrycounter</code> ....	8–10, 59, 60, 137
		<code>\theHglentrycounter</code> ....	8–10, 59, 60, 137
		<code>\theindex</code> .....	167
		<code>\this@dialect</code> .....	313, 314, 346
		<code>\thisgrptitle</code> .....	386
		<code>\toks@</code> .....	118, 171
		tracklang package .....	125, 316
		<code>\TrackLangGetDefaultScript</code> .....	346
		<code>\TrackLangIfHasDefaultScript</code> .....	346
		<code>\TrackLangRequireDialectPrefix</code> ....	346
		<b>U</b>	
		<code>\u</code> .....	314
		<code>\undef</code> .....	13, 14, 172
		<code>\underline</code> .....	174
		<code>\unskip</code> .....	34, 45, 348
		upgreek package .....	315
		<code>\usepackage</code> .....	121–123
		<b>W</b>	
		<code>\warn@nomakeglossaries</code> .....	107
		<code>\warn@noprintglossary</code> .....	107, 111
		<code>\write</code> .....	41, 93, 101, 102, 106, 119, 126
		<b>X</b>	
		<code>\x</code> .....	118, 141
		<code>\xcapitalisewords</code> .....	159
		<code>\xdef</code> .....	110, 130
		<code>\xifinlist</code> .....	97
		<code>\xifinlistcs</code> .....	30
		xindy .....	388
		xindy .....	13, 106
		xkeyval package .....	5

\XKV@checkchoice .....	54	\XKV@resa .....	54
\XKV@plfalse .....	54	\XKV@sttrue .....	54